



runZero

User Guide

v5.1.260908.1

Updated on 2026-09-10

Contents

- [User Guide](#)
- [runZero](#)
 - [Data Sources](#)
 - [Live Inventory](#)
 - [Reports](#)
 - [Monitoring and Alerts](#)
- [Quickstart guide](#)
 - [1. Set up an Explorer](#)
 - [2. Run your first scan](#)
 - [3. Configure integrations](#)
 - [4. Review scan results](#)
- [Full deployment plan](#)
 - [1. Plan your deployment](#)
 - [2. Initial setup](#)
 - [3. Run discovery and assess](#)
 - [4. Automate and optimize](#)
 - [5. Rollout and monitor](#)
- [Creating an account](#)
 - [Activating your account](#)
 - [Changing your password](#)
 - [Updating your profile picture](#)
- [Installing an Explorer](#)
 - [Installation](#)
 - [System requirements](#)
 - [Configuration](#)
 - [Web screenshots](#)
 - [Network communication](#)
 - [Restarting an Explorer](#)
 - [Removing an Explorer](#)
 - [Log management](#)
 - [Certificate Authorities \(CAs\)](#)
 - [Manual mode](#)
 - [Storage locations](#)
 - [Container installations](#)
 - [Automated installations](#)
 - [AWS EC2 installations](#)
 - [Automated MSI deployments](#)
 - [Installing on a Raspberry Pi](#)
 - [Managing Explorers](#)
 - [Verifying binaries](#)
- [Full-scale deployment](#)
 - [1. Identify key success outcomes](#)
 - [2. Planning your deployment](#)
 - [3. Initial configuration](#)
 - [4. Review Assets and Exposures](#)
 - [5. Optimization](#)

- 6. Automation
- 7. Rollout
- Additional Resources
- Getting help
- Types of networks
- Organizations
 - Use cases for organizations
 - Creating organizations
 - Organization configurations
 - Projects
 - Organization and project details
- Sites
 - Use cases for sites
 - Creating sites
 - Site configurations
 - Subnet tagging
 - Importing subnets
 - Importing and exporting sites
 - Sites and Explorers
- Self-hosting runZero
 - Background
 - Requirements
 - Offline mode
 - Installation steps
 - Installation with your own PostgreSQL database
 - runZero updates
 - Managing users
 - CLI service management
 - CLI update management
 - CLI user management
 - CLI organization management
 - Advanced configuration
 - Permissions
 - Backup and restoration
 - Support and debugging
 - Manual migrations
 - Offline mode configuration
 - High-availability configuration
 - Self-hosted troubleshooting
- Data retention
 - Stale asset expiration
 - Offline asset expiration
 - Stale integration attribute expiration
 - Stale vulnerability expiration
 - Scan data expiration
 - Event records
 - AI threads and artifacts
 - Custom integration script versions
 - Data deletion after account termination
- Available roles

- Superuser
- Administrator
- User
- Billing
- Annotator
- Viewer
- No Access
- Custom roles
- Inviting users
- User details
- Account settings
 - Single-sign on (SSO)
 - Multi-factor authentication (MFA)
 - Disabling support access
 - Idle times and sign in duration
 - Account API keys
 - Roles
 - API clients
 - License information
 - Entity information
 - Audit log
- Custom roles and permissions
 - How permissions work
 - What the built-in roles grant
 - Creating a custom role
 - Assigning custom roles
 - How multiple roles combine
 - Auditing
- Managing user groups
 - Creating user groups
 - Adding users to user groups
 - Setting an expiration date for a user group
 - Viewing users in a user group
 - Viewing user groups assigned to a user
 - Removing users from a user group
 - Deleting user groups
 - Searching for users and user groups
- Bulk importing users
 - Creating the CSV file for importing users
 - Importing users into runZero
 - Verifying users have registered
- Managing external users
 - Inviting an external user into your account
 - Switching clients and organizations as an external user
 - Removing an external user from your account
- Implementing SSO
 - Specific SSO providers
 - Identity provider settings
 - SSO walkthrough
 - Service provider information

- Common problems
- Managing SSO group mappings
- Setting up Microsoft Entra SSO
- Setting up Okta SSO
- Managing licenses
 - How do I view my license?
 - What count as recent assets?
 - When does my subscription expire?
 - How do I renew my subscription?
 - How do I convert to the Community Edition?
 - How do I find my invoices?
 - How do I change or cancel my subscription?
- Active scanning
- Traffic sampling
- Inbound integrations
- Active scanning
 - Site
 - Explorer
 - Hosted External Explorer
 - Discovery scope
 - Scan name
 - Scan speed
 - Schedule
 - Scheduling grace period
 - Scan duration limit
 - Advanced scan options
 - Autonomous discovery
 - Initial network scans
 - Identifying gaps in scanning
 - Managing scan templates
 - Scanning with credentials
 - Scanning with SNMP
 - Using custom fingerprints
- Passive sampling
 - Configuring passive sampling
- Protocol support
 - Additional protocols
- Default ports
 - What TCP ports does runZero scan?
 - What UDP ports does runZero scan?
- Scanning IoT and OT
 - Can I safely scan my IoT and OT environments?
 - What makes the Explorer safe for sensitive environments?
 - Where should I deploy the Explorer?
 - OT/ICS vendors runZero fingerprints
 - IoT and OT protocols
- Inbound integrations
 - Enriching runZero results with data from other tools
 - Supported integrations
 - Scan probes or connector tasks

- Importing integration data
- Automatic asset merge
- Removing an integration data source
- Excluding integration attributes
- Source names and IDs
- Amazon Web Services
- Censys Search & Data
- Censys Platform
- Cisco Meraki Dashboard
- CrowdStrike Falcon
- Custom Integration Scripts
- Dragos
- Google Cloud Platform
- Google Workspace
- Microsoft 365 Defender
- Microsoft Active Directory
- Microsoft Azure
- Microsoft Endpoint Configuration Manager (MECM)
- Microsoft Entra ID
- Microsoft Intune
- Miradore MDM
- NetBox CMDB
- Step 5: Enable Unknown Assets in Recurring Sync
 - Qualys VMDR
 - Palo Alto Networks Firewall
 - Palo Alto Prisma Cloud
 - Rapid7
 - SentinelOne
 - Shodan
 - Tailscale
 - Tanium API Gateway
 - Tenable
 - VMware
 - Wiz
- Outbound integrations
 - Using runZero data to enrich other tools
 - Atlassian Insight & Jira Service Management
 - Jira
 - Panther
 - SecurityGate.io
 - ServiceNow Service Graph
 - Splunk Search
 - Sumo Logic
 - Tines
 - Thinkst Canary
- Comprehensive asset inventory
- Total attack surface management
 - External & Cloud
 - Internal
 - IoT & OT

- Inside-Out
- Vulnerability detection
- Risk prioritization
- Continuous monitoring
- Rapid responses
- Query library
- Templates
- Findings
- Issues
 - What is an issue
 - Creating an issue
 - Working with an issue
 - Automatic remediation tracking
 - Linking a Jira ticket
 - Permissions
- Reviewing results
 - Task details
 - Dashboard & inventory views
 - Insights from queries
 - Reports
 - Alerts
- Using dashboards
 - Dashboard selection
 - Creating dashboards
 - Customizing dashboards
 - Sharing dashboards
 - Duplicating dashboards
 - Default dashboards
 - Deleting dashboards
 - Widget library
 - Widget types
 - Drill down
 - Printing and exporting
 - Recalculate metrics
 - Themes
 - Display mode
- AI Threads
 - Requirements
 - Threads
 - Tools
 - Artifacts
 - Reporting
 - Limits
 - Retention
- Using the inventory
 - Understanding assets
 - Loading assets
 - Connecting to other systems
 - Viewing services
 - Viewing screenshots

- Viewing software
- Viewing vulnerabilities
- Viewing certificates
- Viewing wireless networks
- Viewing users and groups
- Understanding assets
 - Asset fields
- Understanding findings
 - What are findings?
 - Finding categories
 - Findings list
 - Finding details
 - How are Findings different from Vulnerabilities?
- Asset risk and criticality
 - Defining risk and criticality
 - Assigning asset risk and criticality
 - Asset risk report
- Certificates
 - TLS certificates
- Geolocation
 - Where locations come from
 - GeoIP lookups against MaxMind
 - Cloud region codes
 - Discrete country and city attributes
 - Free-form location strings
 - Manually setting an asset location
 - Searching by location in the asset inventory
 - When locations are refreshed
- Managing ownership
 - Ownership types
 - Default asset owner
 - Assigning owners to assets and vulnerabilities
- Managing tasks
 - Task status values
 - Tabs
 - Task details
 - Scheduled tasks
 - System tasks
 - Dismissing failed tasks
 - Reprocessing tasks
- Tracking goal progress
 - Goal creation
 - Creating saved query goals
 - Creating baseline goals
 - Goal progress calculation
 - Goal events and notifications
- Understanding suppression
 - How suppression works
 - Suppressing and unsuppressing objects
 - Searching for suppressed objects

- Dashboard widget
- Suppression events
- Understanding network segmentation
 - runZero multi-homed asset detection
 - Using the bridge report
 - Using the asset route pathing report
- Protocol gateways
 - How it works
 - Supported protocols
 - Enabling enumeration
 - Finding gateways and sub-assets
 - Where sub-assets show up
 - Safety
 - See also
- Understanding fingerprints
 - How fingerprints work
 - Refingerprinting and submitting improvements
 - Fingerprint categories
 - Common fingerprint attributes
 - Using fingerprints in queries
 - Troubleshooting fingerprints
- Managing alerts
 - Using the rules engine
 - Creating alert templates
 - runZero events
- Querying your data
 - Filtering and searching data
 - System and custom queries
 - Creating and editing queries
 - Search query syntax
 - Query examples
 - Inventory keywords
 - Vulnerability Count
 - Has vulnerability
 - Interface keywords
 - Automating queries
- Exporting asset data
- Exporting HP iLO data
 - How to export HP iLO CSV data
 - HP iLO CSV export data
- Network Map
 - Accessing the map
 - 2D and 3D modes
 - Subnet layout
 - Network Intel card
 - Search
 - Tracing paths
 - The Trace panel and per-hop filters
 - Choke points
 - Exporting

- [Sharing a view](#)
- [World Map](#)
 - [Prerequisites](#)
 - [Accessing the map](#)
 - [2D and 3D modes](#)
 - [Searching the map](#)
 - [Searching the asset inventory by location](#)
 - [Exporting](#)
 - [Sharing a view](#)
- [Organization Overview](#)
 - [Generating the Organization Overview Report](#)
 - [Email notifications](#)
- [RFC 1918 Report](#)
 - [RFC1918 coverage report](#)
- [External Assets](#)
 - [Generating the External Asset Report](#)
 - [Email notifications](#)
- [Site Comparison](#)
 - [Generate a site comparison](#)
 - [View how assets change over time](#)
 - [View how exposure differs between networks](#)
 - [Analyze the results in the site comparison report](#)
 - [Search the site comparison report](#)
- [External reporting](#)
 - [What you can share](#)
 - [Sharing a report](#)
 - [Share dialog options](#)
 - [How recipients view a report](#)
 - [Managing reports and snapshots](#)
 - [Security and access](#)
- [Switch Topology](#)
 - [Generating the switch topology report](#)
 - [Filtering the switch topology report](#)
 - [Limitations of the switch topology report](#)
- [Network Bridge Report](#)
 - [Accessing the report](#)
 - [How bridges are detected](#)
 - [Reading the report](#)
 - [Querying multi-homed assets directly](#)
 - [See also](#)
- [Asset Route Pathing](#)
 - [Terminology](#)
 - [Generating the asset route pathing report](#)
 - [Analyzing the report](#)
 - [Sharing the report](#)
 - [Exporting a dotfile](#)
 - [FAQs](#)
- [Cryptographic bill of materials](#)
 - [Post-quantum status](#)
 - [Migration deadlines](#)

- Interpreting the results
- Leveraging the API
 - API keys and tokens
 - Authentication
 - API rate limiting
 - IP Address Allowlist
 - runZero API usage examples
 - Additional documentation
- API clients
 - Grants
 - Expiration
 - Creating a user-bound API client
 - Creating an account-bound API client
 - Role locks
 - Managing existing clients
 - Using an API client
 - Credentials created by AI clients
- Using the CLI
 - Scanner
 - runZero Command Line Interface (CLI)
- Model Context Protocol
 - MCP Terms
 - Authentication
 - Connecting through OAuth
 - Presenting a credential directly
 - Clients
 - Available tools
 - Prompts and resources
 - Exploring MCP tools, resources, and prompts
- Glossary
 - Terms
- Frequently asked questions
 - Issues and FAQs
 - Identical assets in inventory
 - Scanning routers
 - Scanning VMWare virtual machines
 - Scanning on Windows with Intel AMX
 - Scanning on Windows with EDR or antivirus
 - Explorer not capturing screenshots
 - Explorer security model
 - Browsers supported by the runZero Console
 - Common sign-in issues
 - runZero data formats
 - runZero data dictionary

What is runZero?

runZero

runZero is an attack surface and exposure management platform that helps you see everything on your network, from managed IT systems to unmanaged OT, IoT, cloud, mobile, and remote assets. It combines high-speed active scanning, passive discovery, and API integrations to build a unified, continuously updated inventory you can trust.

You can run runZero as a hosted service (SaaS) or [on-premise](#). The platform includes one or more Consoles, distributed [Explorers](#) that run as lightweight services at your points of presence, and a [command-line tool](#) for offline collection and automation. Teams can work through the web interface, automate through the API, and for self-hosted deployments, manage operations from the command line.

Data Sources

- **Active Scans:** runZero's best-in-class [active scan engine](#) is fast, accurate, and safe for all environments, with support for a massive number of [protocols](#) and [applications](#).
- **Passive Traffic Sampling:** runZero's [passive traffic sampling engine](#) scales with available resources and works with broadcast traffic, SPAN ports, and encapsulated streams. Any runZero Explorer can be used for passive traffic sampling, regardless of location, configuration, or resources.
- **API Integrations:** runZero supports [inbound](#) and [outbound](#) integrations with major Cloud, Endpoint, CMDB, and Endpoint providers. In addition to the native options, customers can create their own integrations using the [Custom Integration API](#) and [Custom Integration Scripts](#).

Live Inventory

- **Assets:** runZero tracks all [assets](#) across the environment; including cloud, mobile, endpoint, server, OT, IoT, and everything else in between. Assets are correlated and merged across data sources to provide a multi-perspective snapshot of all organization resources. The asset inventory supports deep search, configurable columns, and simple export.
- **Services:** runZero tracks all identified network services, via active scans, passive discovery, and integrations (where applicable). The services inventory simplifies exposure management tasks and enables deep search and exports.
- **Screenshots:** runZero takes a snapshot of each exposed web service included in active scans. The screenshot inventory allows security teams to visually inspect unknown devices and services.

- **Software:** runZero identifies network-exposed software and imports software records from API integrations. The result is a software inventory that can be used to quickly identify specific packages and versions across the environment.
- **Vulnerabilities:** runZero reports vulnerabilities based on identified exposures and imports vulnerability data from API integrations. The vulnerability inventory is provided as both a detailed, per-asset inventory, as well as a grouped view that simplifies investigation into specific issues.
- **Certificates:** runZero identifies SSL/TLS certificates, including those that are expired or soon-to-expire. The certificate inventory includes all relevant metadata, such as issuer, expiration date, and certificate chain.
- **Wireless:** runZero active scans also enumerate wireless access points within range of the Explorer running the task. This inventory includes the BSSID, SSID, encryption settings, and signal strength.
- **Users and Groups:** runZero imports user and group information from directory services, including Active Directory, Azure Active Directory (Entra ID), and Google Workspace. The user and group inventories can be used to identify accounts with specific attributes, such as expired passwords and excessive group permissions.

Reports

runZero includes a comprehensive set of reports that cover everything from [layer-2 topology maps](#) to outliers and asset risk. In addition to pre-defined reports, most attributes within an asset or service can be used to create a grouped report with a single click. For deep customization, the runZero [Export API](#) provides CSV and JSON(L) formats with arbitrary search filters, which can be used to drive analytics platforms like Tableau and PowerBI.

Monitoring and Alerts

runZero provides monitoring and [alert](#) capabilities that can trigger based on changes to the inventory, new results for custom search [queries](#), and any system-level event (of dozens). These alerts can be delivered either in-product, by email, or to a webhook destination, including Slack channels. In addition to alerts, rules, and custom queries, [goals](#) and custom dashboard widgets can be defined to track progress towards a specific outcome.

Getting started

To get started, you'll need to sign up for a runZero account. The default account is a trial of the full runZero Platform. After the trial expires, you will have the option to convert to the free Community Edition or purchase a subscription.

- [Sign up for a runZero account](#)
- Read up on [creating an account](#) for help activating your account, changing your password, and adding a profile picture.
- Once your account is set up, there are a couple of paths you can take to deploy runZero.
 - [Quickstart](#)
 - [Full deployment plan](#)

Quickstart guide

The quickstart path is ideal for those who want to jump into using runZero and explore its core functionalities. This section covers the initial setup, running basic scans, and configuring integrations.

1. Set up an Explorer

To begin asset discovery, you need to deploy an Explorer:

- Navigate to [Deploy > Deploy Explorers](#) in the runZero Console.
- Review the system requirements and choose the appropriate binary for your platform.
- Ensure the Explorer connects to your organization. Each download link is organization-specific.

2. Run your first scan

Once the Explorer is installed:

- Go to [Scan](#) in the console.
- Click [Start standard scan](#) and configure the discovery scope:
 - Choose an Explorer.
 - Set the target subnets or CIDR ranges.
 - Run the scan and monitor progress from the console.



We recommend starting with a `/24` or handful of small CIDR blocks to verify connectivity in the network and then scaling out from there.

3. Configure integrations

Enhance your data with third-party integrations:

- Navigate to [Integrate](#).
- Select and configure supported integrations for EDR, Vulnerability Management, Cloud, etc.
 - [Documentation for each inbound integration](#) is available.
- Ensure appropriate permissions are set in the integrated systems.
- Run sync tasks to import data and merge with existing asset information.

4. Review scan results

After the scan completes:

- Check results under [Inventory > Assets](#).
- Use the [Query library](#) to better understand misconfigurations and other notable findings in the inventory.
- Review [reports](#) and customize dashboards for deeper insights.
- Here are a few sample reports to check out:
 - [Network Map](#) — interactive 2D/3D topology with attack-path tracing and choke-point ranking.
 - [World Map](#) — geolocate assets on a globe by GeoIP, cloud region, and manual geo* tags.
 - [SNMP default communities](#)
 - [SSH authentication methods](#)
 - [TLS supported versions](#)



If you have no results with the ``matches:>0`` search, you may need to wait for the metrics to be calculated. You can see the status of the metrics calculations [here](#).

Full deployment plan

For organizations planning a production deployment, careful preparation ensures a smooth and scalable setup. Follow these steps:

1. Plan your deployment

- Determine the goals for asset visibility, network segmentation, or compliance.
- Ensure adequate resources for Explorers and network access points.
- Familiarize yourself with runZero-specific terms in the glossary.

2. Initial setup

- Deploy Explorers across key network segments.
- Configure credentials for access to devices and services (e.g., SNMP, EDR, MDM, etc).
- Use network communication guidelines to verify connectivity.

3. Run discovery and assess

- Perform discovery scans on all required segments.
- Validate data integrity and optimize scan settings as needed.
- Identify any scanning gaps and adjust configurations.

4. Automate and optimize

- Implement scan schedules for continuous monitoring.
- Use integrations for enriched asset data.
- Set up alerting and reporting tailored to your organization's needs.

5. Rollout and monitor

- Deploy across remaining network segments based on your plan.
- Monitor scan results and network health regularly.
- Document configurations and processes for ongoing management.

For detailed instructions, refer to the [production deployment documentation](#). For assistance, please contact [support](#).

Creating an account

To get started, you'll need to sign up for a runZero account. The default account is a trial of the full runZero Platform. After the trial expires, you will have the option to convert to the free Community Edition or purchase a subscription.

[Sign up for a runZero account](#)

Activating your account

After you sign up for an account, we'll email you a link to activate your account. If you don't see an email from us, check your spam folder.

Open the link in the email to go to the **Activation** page. Follow the instructions on the page to activate your account. You'll need to provide your name, set up a password, specify your location, and accept our [privacy policy](#) and [terms of service](#).

After activating your account, you'll be taken directly to the runZero Console. Your new account has administrative access, so you will be able to manage [sites](#), [organizations](#), [users](#), and [Explorers](#).

If you have any trouble creating your account, please contact [support](#).

Changing your password

To change your password, go to your [account settings](#). You'll need to provide your current password before you can enter a new one.

All passwords must contain:

- At least 8 characters
- At least 1 uppercase character
- At least 1 lowercase character
- At least 1 number

Updating your profile picture

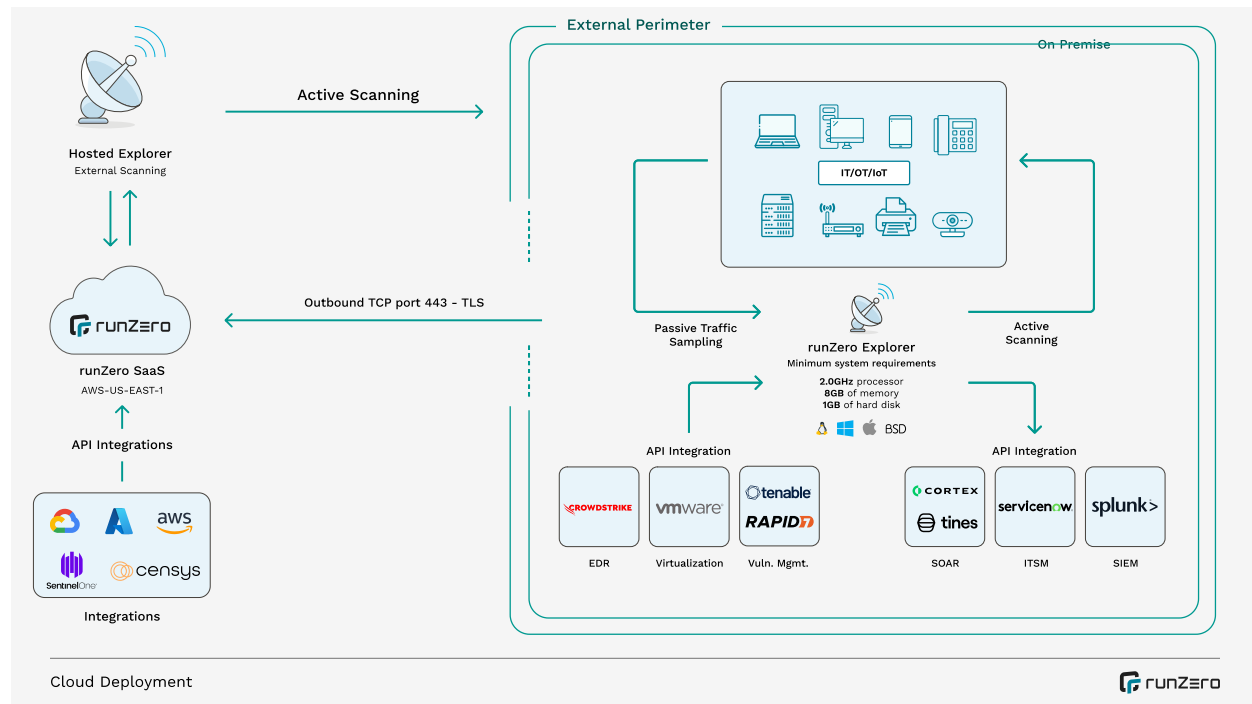
User profile images are managed through [Gravatar](#) and associated with your email address. If you don't have an account, [sign up for one](#).

Installing an Explorer

runZero requires the use of at least one Explorer within your environment to enable active and passive network discovery. The Explorer should be installed on a system with reliable connectivity to the network you want to discover. For internal networks, runZero works best when installed on a system with a wired (vs wireless) connection.

For external network discovery, nearly any cloud provider with a reliable connection should do. If the runZero Explorer is installed in a container or virtualized system, ensure that it has direct access to the network (host networking in Docker, bridged networking in VMware, etc). SaaS customers with a Platform license can use the runZero hosted Explorers at no additional cost.

[View an interactive version of this diagram](#)



Installation

To install the runZero Explorer, sign in to the runZero Console and switch to the Organization that should be associated with the Explorer. Explorer downloads are then available by selecting Deploy in the left navigator and choosing the Deploy Explorers sub-menu.

Note: The Explorer download link is specific to your active organization and **using the wrong link can result in a new Explorer being associated with the wrong organization.**

Download the correct binary for your system from the [Explorer download page](#). For most systems, select the 64-bit (x86_64) architecture. For macOS, you will need to select 64-bit

Intel (x86_64) or ARM (Apple M*), depending on your hardware. For embedded devices, such as the Raspberry Pi 3+, choose the ARM7 architecture. Windows binaries are signed with a valid Authenticode signature, which should be validated before the executable is launched.

The Explorer installation process requires administrative privileges. On Windows, a UAC prompt may be displayed. On Linux and macOS the downloaded binary should be made executable (`chmod u+x runzero-agent.bin`) and then executed with root privileges (`sudo` or from root shell). In either case, the Explorer should install itself as a system service and start immediately, displaying a new entry in the [Explorers page](#).

System requirements

Windows

- Operating System
 - Windows Server 2016 or newer
 - Windows 10 Build 1604 or newer
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space

Limitations

- The Trellix agent for Windows appears to interfere with network scans and traffic capture. Please use a system without the Trellix agent for active scans or passive network sampling. Alternatively, switching the Trellix HIDS agent from “block” to “monitor” may allow successful scans.
- EDR, antivirus, and endpoint protection products on Windows may also interfere with Explorer scans, packet capture, or process execution. If scan reliability issues occur, add allowlist exceptions for runZero and npcap/winpcap components, or use monitor-only policy modes while validating scans. In some environments, these controls may need to be disabled on the host for stable operation.
- Windows systems with Intel Advanced Matrix Extensions (AMX) enabled may experience Explorer instability or crashes. This can affect both physical hardware and virtual machines. Disable AMX at the BIOS or host level before installing or running the Explorer. For Windows guests on Hyper-V, enabling **Processor Compatibility** can mitigate this issue.
- Earlier Windows versions may be able to run the Explorer, but are not officially supported.
- Windows Explorers are limited to a single concurrent scan or passive task due to performance limitations of the raw packet driver.

Linux

- Kernel version 3.2 or later
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space

Linux ARM devices with limited processing power and memory, such as the Raspberry Pi, can run the runZero Explorer, but may have trouble scanning larger networks, or running integrations with large amounts of data.

MacOS

- macOS 11 (Big Sur) or newer
- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space

Due to Notary requirements introduced in macOS Catalina (10.15), using the **curl** download method provided on the deployment page in the Console is required.

BSD variants

- Processor running at 2.0 GHz or faster
- At least 16GiB of memory (8GiB for small environments)
- At least 1GB of free storage space

Requires root access to a system running a recent version of the operating system. FreeBSD 11.2 or newer, recent versions of NetBSD/DragonFly/OpenBSD.

Configuration



Existing installations may refer to `Rumble` in the directory name, service name, and binary names. `Rumble, Inc`` was the previous name of our business and new installations will only refer to `runZero`, our new name. going forward.

The Explorer can be configured by setting variables in a `config.txt` file located in the same directory as the executable. On Windows this file should be created in `C:\Program Files\runZero\config.txt`, while other platforms should use `/opt/runzero/bin/config.txt`. In addition to `config.txt`, the legacy `.env` name is also supported. If the file `/etc/runzero/config` is present, this will be preferred as the configuration settings for the Explorer.

The format of this file is `VAR=VAL` with one variable per line.

Configuration file locations

Windows: C:\Program Files\runZero\config.txt

Other Platforms: /opt/runzero/bin/config.txt

Web screenshots

The scanner will try to work with any Chrome, Chromium, or Edge version available and indicate the browser path and version within the Explorers list in the web console. If no supported browser is found, we recommend installing the official [Google Chrome](#) package for your operating system in order to enable [web screenshots](#).

To install the latest Chrome package on Debian-based Linux installations (including Ubuntu):

```
curl -o chrome.deb https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb && \
sudo apt install ./chrome.deb
```

To install the latest Chrome package on RedHat-based Linux installations (including Fedora, CentOS, Rocky, and Alma):

```
curl -o chrome.rpm https://dl.google.com/linux/direct/google-chrome-stable_current_x86_64.rpm && \
sudo dnf install ./chrome.rpm
```

The following variables in the environment or configuration file to change web screenshot behavior:

The `RUNZERO_CHROMIUM_IGNORE_SYSTEM` parameter can be set to `true` to ignore system Chromium paths for screenshots.

The `RUNZERO_CHROMIUM_AUTOMATIC_INSTALL` parameter can be set to `true` to enable automatic Chromium installation if necessary. This pulls directly from the Google and Puppeteer CDNs.

The `RUNZERO_CHROMIUM_DEBUG` parameter can be set to `true` to enable detailed Chromium process logging while screenshots are taken.

The `RUNZERO_CHROMIUM_USER` parameter can be set to a valid system username on non-Windows systems. When the scanner is run as root, the Chromium process will be run as this user account (and default GID).

The `RUNZERO_CHROMIUM_CACHE_DIR` parameter can be set to a directory path to use for automatic Chromium installations (or installations managed through content updates). This defaults to `/opt/runzero/chromium` on non-Windows and the binary directory plus `\chromium` on Windows. This directory contains a file called `LATEST.txt` that contains a numeric revision, which refers to a sub-directory named `chromium-[revision]`, which includes a copy of the Chromium installation.

The `RUNZERO_CHROMIUM_DISABLE_CHROMEDP` parameter can be set to `true` to disable the debug protocol capture mode (now the default).

The `RUNZERO_CHROMIUM_DISABLE_CHROMECL` parameter can be set to `true` to disable the command-line headless `--screenshot` capture mode (used as a fallback otherwise).

Network communication

The Explorer connects to the runZero Console on TCP port 443 using TLS. This connection is used for Explorer registration, job scheduling, status messages, and submission of completed scan jobs. For completely offline environments, the [runZero CLI](#) can be used to create scan data files that can be uploaded later via the [Inventory Import](#) action. The console is used for automatic updates of the Explorer executable. The specific IP addresses used depend on your deployment model and region. For customers using the SaaS console, the following static IPs are associated with the runZero consoles.

United States

The console hostname is `console.runzero.com`.

IPv4

- 13.248.161.247
- 76.223.34.198

IPv6

- 2600:9000:a415:cd87:fbe5:476a:3533:69f2
- 2600:9000:a716:ee91:85f9:3c9:48c9:59b9

Germany

The console hostname is `console-eu.runzero.com`.

IPv4

- 15.197.131.232
- 3.33.248.90

IPv6

- 2600:9000:a603:e925:542d:6d40:6897:bc3a
- 2600:9000:a70e:635f:71bd:bb0a:8e43:9466

NAT

The Explorer should be run on a system with a full bidirectional network connection. It should **not** be run on a system behind a NAT gateway. This includes virtual machines, which should be set up with bridged network adapters.

Juniper DDoS protection is known to break runZero scans.

Proxy support

Please note that certain web proxies that perform TLS inspection do not handle Websocket communication properly and TLS inspection will need to be disabled for the runZero Explorer to successfully connect. The most popular product with this problem is the Sophos (previously Cyberoam) security appliance. Websense users may need to add a [bypass rule](#) for `console.runzero.com`.

Proxy support is handled automatically in most cases. On the Windows platform, proxy information is read from the registry keys (used by Chrome, Edge, and IE).

The proxy can also be configured by setting the `HTTPS_PROXY` environment variable. The value of the `HTTPS_PROXY` environment variable should be a hostname and port (`proxy.example.com:8080`) or just a hostname (`proxy`). Environment variables are read from your configuration file. Please view the [Configuration](#) section to see how to set environment variables. The common option is to create a file named `config.txt` in the same directory as the Explorer binary and set the environment variables in the format described below.

It's also possible to use a SOCKS proxy by setting `HTTPS_PROXY` to a socks5 URL, for example `HTTPS_PROXY=socks5://socks.example.com:1080`.

It's also possible to set a specific proxy override for AWS connections, by specifying `HTTPS_PROXY_AWS`. This setting takes priority over any base `HTTPS_PROXY` setting for the Explorer.

The Explorer will attempt to use the configured proxy for each probe. If it doesn't succeed, it will try making a direct connection. This means both proxy and non-proxy connection attempts may appear in logs. Using only proxies to try to hide or anonymize Explorer connections is not supported.

TLS configuration

The minimum and maximum version of TLS used by the Explorer for outbound communication to the console can be configured through environment variable and the configuration file. The `TLS_VERSION_MIN` and `TLS_VERSION_MAX` variables can be set to any of 1.0, 1.1, 1.2, and 1.3. The default configuration is to use a minimum version of TLS 1.2 and a maximum of TLS 1.3. If a maximum version is set to a value lower than the minimum value, the maximum will be set to the minimum value. runZero does not recommend using TLS versions prior to 1.2.

The following example will configure the Explorer to only speak TLS version 1.3.

```
TLS_VERSION_MIN=1.3
TLS_VERSION_MAX=1.3
```

Restarting an Explorer

The easiest way to restart an Explorer is to force a software update from the cloud console. Otherwise, you can find the service on the host machine and restart it by hand.

On Linux or Mac, you can run `/opt/runzero/bin/runzero-agent-[uuid] restart` where `[uuid]` is the ID of the organization the Explorer belongs to.

On Linux systems you can also use `systemd` to restart Explorers. First obtain the name of the Explorer (`runzero-agent-[uuid]`) service:

```
systemctl | grep runzero-agent
```

Then restart the service using this name:

```
systemctl restart runzero-agent-[uuid]
```

A `kill -9` of the Explorer pid should cause a restart as well.

On macOS, you can use `launchctl` to restart the Explorer:

```
launchctl kickstart -k runzero-agent-[uuid]
```

As with Linux, the `[uuid]` is the organization UUID, which you can find by looking at the `runzero-agent-*` filename in `/opt/runzero/bin`

On Windows, you can use the Services console to restart the Explorer like any other background service.

Removing an Explorer

The easiest way to remove an Explorer is to use the [Explorers](#) page and locate the Explorer you want to remove, then click the *Delete Explorer* trash bin icon under the *Actions* column. This will remove the service and terminate the current Explorer process. In addition, you can remove all online Explorers by clicking the *Manage All Explorers* menu and choose the *Uninstall All Online Explorers* option. If you would like to remove an Explorer without using the runZero Console, there are a couple of options.

On the Windows platform, each Explorer will be listed in Programs and Features (as the runZero Agent), and can be uninstalled like any other application.

On all platforms, including Windows, the Explorer can uninstall itself if run with the `uninstall` argument from a root or Administrator shell:

Removal on Windows

```
c:\Program Files\runZero\runzero-agent-[uuid].exe uninstall
```

Removal on Other Platforms

```
/opt/runzero/bin/runzero-agent-[uuid] uninstall
```

Log management

The Explorer logs to a file and to standard output by default. On Windows the default log file location is the installation directory (C:\Program Files\runZero Or C:\Program Files\Rumble for old installations) while other platforms log to the files /var/log/runzero.log and /var/log/runzero.err. The default configuration limits log files to 100MiB, creates three backups, and expires logs after 90 days. These defaults can be changed by setting the following values in the \$BIN/config.txt file:

- The RUNZERO_AGENT_LOG_MAX_SIZE setting controls the maximum log size in mibibytes. The default is **100**.
- The RUNZERO_AGENT_LOG_MAX_BACKUPS setting controls the number of backup files created by log rotation. The default is **3**.
- The RUNZERO_AGENT_LOG_MAX_AGE setting controls the maximum age in days, this applies to all files, including backups. The default is **90**.
- The RUNZERO_AGENT_LOG_COMPRESS setting determines whether to gzip compress the backups. The default is **false**.
- The RUNZERO_AGENT_LOG_STDOUT setting determines whether to write logs to standard output. The default is **true**. On Linux this results in logs being written to the system log when the Explorer is started by systemd or upstart. On macOS this results in separate logs viewable in the Console application under "Log Reports" when the Explorer is started by launchd.
- The RUNZERO_AGENT_LOG_FILE setting determines whether to write logs to a log file as described above. The default is **true**, set to `false` to disable log file writing.

The Explorer must be restarted for these settings to take effect.

Certificate Authorities (CAs)

The runZero Explorer uses the system-installed certificate authorities to validate TLS connections in addition to an internal CA certificate bundle. By default, both the system certificate roots, and the bundled roots are considered for all secure TLS connections, including Starlark-based custom integrations. This behavior can be controlled via environment variables (set in the \$BIN/config.txt file or at the system level):

- The RUNZERO_TLS_IGNORE_SYSTEM_ROOTCA setting can be set to **true** to ignore the system CA roots.
- The RUNZERO_TLS_IGNORE_EMBEDDED_ROOTCA setting can be set to **true** to ignore the bundled CA roots.
- The RUNZERO_TLS_ADDITIONAL_ROOTCA setting can be set to a file path containing additional CA roots in PEM format.

Manual mode

If a supported system service manager, such as systemd or upstart, is not detected, the runZero Explorer will switch to manual mode, running in the foreground, and replacing and re-executing its own binary as new updates become available. For temporary Explorer

installations or to run the Explorer in a container environment, the argument “manual” can be specified:

```
$ sudo ./runzero-agent.bin manual
```

Storage locations

The runZero Explorer installs into %PROGRAMFILES%\runZero on Windows and /opt/runzero/bin on all other platforms. Temporary files are stored in the default operating system locations. These locations can be overridden using the config.txt file (see the above [Configuration](#) section). Note that the Explorer service needs to be restarted (or force updated) for these changes to take effect. Older installations may still refer to the Rumble versions of the previously mentioned directories and .env instead of config.txt for environment overrides.

On Windows, the temporary file location is chosen from the first non-empty environment value of TMP, TEMP, or USERPROFILE, falling back to the Windows directory. To override this location, set an entry in config.txt like the following:

```
TMP=D:\Storage\runZero
```

On all other platforms, the temporary file location is chosen based on the value of TMPDIR, falling back to /tmp otherwise. To override this location, set an entry in config.txt like the following:

```
TMPDIR=/home/storage/runzero
```

Any scans that fail to upload are stored in the runZero Explorer installation directory and can be imported into the platform manually or using the runZero CLI's scan --import and scan --upload options.

Container installations

The runZero Explorer can run in standard container environments, with three important caveats:

1. runZero Explorer binaries are updated frequently and these updates include time-sensitive security coverage. If the container filesystem is read-write, the Explorer will automatically update, but revert to an older state on container restart. We recommend rebuilding the container image frequently or having the entry point of the container retrieve and run the latest binary for your organization on startup. If the container is read-only, no updates will be applied, and each task will be delayed by up to five minutes as the Console tries to update anyways.
2. The container must run as root with full privileges in order to access the raw network device. The specific network parameters differ by container runtime, but typically this means using a bridged, macvlan, or host (not host-only) networking option when deploying the container. Without raw network access many of runZero's features will not work, including ARP scans, IPv6 neighbor discovery, TCP SYN scans, and passive network sampling via SPAN ports.

3. The container must be configured with a persistent `RUNZERO_AGENT_HOST_ID` environment variable and only one container with this ID should be run at a given time. Without this setting, the Explorer will re-register on every container restart, and if more than one container share the same identifier, task processing will not work correctly as the container fight over the task status. This host ID environment should be set to a unique 32-character hexadecimal string.

To generate a suitable identifier, the `openssl` tool may be used:

```
openssl rand -hex 16
01b0283809b24511929d0b062bd36109
```

Here is a sample Containerfile you can edit and use:

```
#
# Sample Containerfile for running the runZero Explorer in a container, with
# screenshot support.
#
FROM debian:stable

WORKDIR /opt/runzero

# Ensure curl is available and install tools for wireless scanning.
#
RUN apt update && apt install -y curl wireless-tools

# Install Chrome for screenshots.
#
RUN curl -o chrome.deb https://dl.google.com/linux/direct/google-chrome-stable_current_amd64.deb && \
    apt install -y ./chrome.deb

# Set AGENT_URL to be the download URL for your Linux runZero Explorer. To
# find your URL, go to https://console.runzero.com/deploy/download/explorers
# and click on the first URL box to copy it to the clipboard.
#
ENV AGENT_URL=https://console.runzero.com/download/explorer/DT[uniqueToken]/[versionID]/runzero-agent-linux-amd64.bin

# This ID is used to track the Explorer even if the container is rebuilt.
# Set it to a unique 32 character hex ID. You can generate one via:
#
# $ openssl rand -hex 16
#
ENV RUNZERO_AGENT_HOST_ID=[UNIQUE-ID]

# If you need to set environment variables to change the Explorer behavior,
# you can do so via the ENV directive. Example:
#
# ENV RUNZERO_AGENT_LOG_DEBUG=true

ADD ${AGENT_URL} runzero-agent.bin

RUN chmod +x runzero-agent.bin
```

```
# For full functionality the runZero CLI needs to send and receive raw
# packets, which requires elevated privileges.
USER root

# The argument `manual` tells runZero not to look for SystemD or upstart.
ENTRYPOINT [ "/opt/runzero/runzero-agent.bin", "manual" ]
```

This containerfile works with podman as well as Docker. Note that because of the requirement for root privileges, you should start the container as root. For the best results, run the container with the `--privileged` option to allow the Explorer to listen to network traffic.

Automated installations

The Explorer will automatically install when executed if root or administrative privileges are available.

On Linux and BSD systems, automatic installation depends on the presence of a supported init service like `systemd` or `upstart`. If no supported init service is found, the Explorer will instead run in manual mode, automatically overwriting and re-executing itself with each update. To automatically deploy an Explorer on systems without a supported init service, the Explorer should be executed in the background and with the `nohup` wrapper.

On Windows systems, the Explorer will automatically install when run interactively or when the `updater` parameter is passed to the binary. For environments where MSIs are required, the [Explorer MSI](#) wrapper can be used to deploy an Explorer from the runZero Console or a local mirror.

AWS EC2 installations

The runZero Explorer can be run in an AWS EC2 instance. However, there are a number of configuration changes required to avoid packet loss when scanning.

- DNS resolution from EC2 to the AWS DNS server has [a fixed cap of 1024 packets per second](#) which cannot be increased. To avoid this, set a custom list of non-AWS nameservers in the scan configuration advanced section.
- Any Security Group without a `0/0` rule results in [connection tracking](#), which has an undocumented limit on connections per instance type. Avoid this by adding `0/0 allow` for ICMP/UDP inbound and outbound for the Explorer instance. Also add a `0/0 allow` for outbound TCP connections. (Inbound TCP is not currently required for runZero scans, but may be needed in the future for callback protocols.)
- Overall packet rates have [undocumented limits which depend on instance type](#). You will need to experiment with sizing your Explorer instances until scans are consistent for a given scan rate. We hope to gather and share data on appropriate instance sizes soon.

The Explorer should work well deployed to a memory optimized, compute optimized, or general compute instance. Since the Explorer can make full use of multi-core systems, you may want to target the number of cores to the number of simultaneous scans. You probably won't want to run larger scans on an instance with less than 32GiB of RAM.

Automated MSI deployments

runZero uses dynamically generated binaries for the runZero Explorer downloads and this doesn't always play well with MSI-based installation methods.

To work around this issue, we have provided a shim MSI package that can be used with automated installers. This package has a valid Authenticode signature and can also be verified using the [runZero Verifier](#).



Some components of the application still reference the name "Rumble" for backwards compatibility. All new installations will use runZero for directory, file, and user names.



Windows hosts with Intel Advanced Matrix Extensions (AMX) enabled may experience Explorer instability or crashes. This can affect physical hardware and virtual machines. Disable AMX at the BIOS or host level before deployment. For Windows guests on Hyper-V, enabling **Processor Compatibility** can mitigate this issue.



EDR, antivirus, and other endpoint protection software on Windows may interfere with Explorer installation or scan behavior. If problems occur, add allowlist exceptions for runZero and npcap/winpcap components or use monitor-only policy modes while validating deployments. In some environments, endpoint protections may need to be disabled for stable operation.

To use this package, deploy it with the URL parameter specified as the organization-specific download URL from the runZero Console Explorers section.

```
msiexec /i runzero-explorer-installer-amd64.msi URL=https://console.runzero.com/download/explorer/DT[uniqueToken]/[versionID]/runzero-explorer-windows-amd64.exe
```



The above command should be entered as one long line with a single URL parameter, but is shown wrapped here.

The MSI shim will verify that the URL contains a valid runZero Explorer binary and install it normally.

Warning: Note that installing with the `/a` parameter will not work and `/i` must be used instead.

Binary downloads

Build	SHA256
runZero Explorer Installer MSI x86 64-bit	sha-256
runZero Explorer Installer MSI x86 32-bit	sha-256

Installing on a Raspberry Pi

The runZero Explorer enables discovery scanning. In most cases, you can deploy an Explorer on an existing system that has connectivity to the network you want to discover. However, there may be times when the traditional deployment model may not work for you. Some locations, like retail stores or customer sites, may not have staff or hardware available to install the Explorer, making remote deployment a bit tricky.

In these types of scenarios, you can install a runZero Explorer on a Raspberry Pi and send the device to the location for them to plug into their network.

What you'll need

- Raspberry Pi 4 Model B (4GB or 8GB), Raspberry Pi Compute Module 4, or Raspberry Pi 400
- At least 1GB free of storage space on your Raspberry Pi's MicroSD card after installing the operating system

Connecting to your Raspberry Pi via SSH

In this section, we're going to show you how to SSH to your Raspberry Pi and install the Explorer from your terminal.

Step 1. Enable SSH on your Raspberry Pi

Before you can connect to your Raspberry Pi, make sure to [enable SSH](#) on the device.

You can add a file called `ssh.txt` at the root of the SD card and reboot the Raspberry Pi. The contents of the file can be empty. On reboot, the Pi looks for the SSH file and enables SSH if it finds the file.

Step 2. Update the password for your Raspberry Pi

If you enable SSH on your Raspberry Pi, you must [update your credentials](#).

Log in as the `pi` user and use the `passwd` command to change the default password. Entering the `passwd` command will prompt you for your current password to authenticate before you can change it.

Step 3. Copy the Explorer instructions

- Go to your [console](#). Verify you are in the right organization. runZero keys your Explorer's download link to the organization you are currently viewing to associate them together.
- Go to the [Explorer deployment page](#) and select **Linux Distributions**, and then choose Linux ARM 32-bit V7. Note that Raspbian uses a 32-bit kernel by default, even on 64-bit Raspberry Pi hardware.
- Click the instructions at the bottom of the Linux installation page to place the commands into your clipboard.

Step 4. Install the Explorer



Some components of the application still reference the name "Rumble" for backwards compatibility. The documentation will be updated as these are changed.

Paste the installation command into the terminal after connecting to the Raspberry Pi via SSH. The URL in the command links the installation to your active organization and will be different from the example below.

```
me@mac ~ % ssh pi@hostIP
pi@host's password:
pi@raspberrypi:~ $ sudo bash
root@raspberrypi:/home/pi# cd Downloads
root@raspberrypi:/home/pi# curl -o runzero-explorer.bin https://console.runzero.com/download/explorer/DT[uniqueToken]/
```

The Explorer automatically installs itself in `/opt/runzero` and sets up a systemd service with the name `runzero-agent-`. The service automatically starts on boot.

Next steps

Your Raspberry Pi is now set up to use as a runZero Explorer box and ready to be sent wherever you need.

You can always view and manage your [registered Explorers](#) from your console.

Managing Explorers

The runZero Explorer is a lightweight scan engine that enables network and asset discovery. You should have at least one Explorer deployed. After deployment, you can manage your Explorers from the [Deploy page](#) in your runZero web console.

Viewing all Explorers

For each Explorer, you can see:

- The Explorer status (whether it is communicating with runZero)
- The OS it is running on
- Its name
- Any site it is associated with
- Its IP addresses
- The software version it is running
- Whether the version of nmap installed is up-to-date, if the OS is Windows (see [upgrading nmap](#) below)
- The CPU architecture of the host machine
- Any tags associated with the Explorer
- The status of its last scan
- Its capabilities, like Chrome support

Screenshot capabilities

To capture screenshots, Chrome must be installed. You can check if an Explorer has screenshot capabilities by looking for the Chrome icon in the *Capabilities* column.

Here's what each icon means:

- Green icon - The Explorer has access to a Google Chrome binary and can take screenshots.
- Red icon - No suitable Chrome binary was found.

Searching for Explorers

You can use the search bar to find Explorers. The query syntax is similar to other search bars in runZero, with keywords to filter by specific fields:

Keyword	Search by	Example
arch:	CPU architecture	arch:amd64
name:	assigned name	name:scanner.local
address:	IP address	address:10.0.1.200
capability:	capabilities	capability:screenshot OR capability:aws
tag:	assigned tag	tag:dev
nmap_version:	nmap version	nmap_version:1.31

Explorer actions

Each Explorer has a set of action buttons that allow you to:

- **Reinstall an Explorer** - Performs a reinstall or upgrade of the Explorer. The current Explorer will download the latest Explorer code from runZero, and then run the install process.
- **Configure an Explorer** - You can associate the Explorer with a specific site, enable recurring speed tests, and add tags to it. You can also set the maximum number of concurrent scans allowed. A single Explorer can be configured to run multiple tasks at once.
- **Reassign an Explorer** - You can reassign an Explorer to a different organization within your account or even to a different runZero client account entirely.
- **Remove an Explorer** - If the Explorer is running, the Explorer will be asked to uninstall itself from the host machine. If the Explorer is not running, you can still tell runZero to forget about it. This is useful if you have decommissioned the machine the Explorer was running on or uninstalled the Explorer manually. If the Explorer runs again after runZero has been told to forget it, it will be readded to the registered Explorers list.

Bulk management operations

Bulk operations allow you to perform a set of actions to multiple Explorers at one time. Bulk actions are available from the *Manage all Explorers* menu.

You can bulk:

- **Update all online Explorers** - Tells all Explorers—that are up and communicating with runZero—to upgrade their software.
- **Forget all offline Explorers** - Clears all Explorers currently offline, and makes runZero forget them. No data will be lost. If any of the Explorers are reactivated, they will be added back to the active list.
- **Uninstall all online Explorers** - Tells all online Explorers to uninstall themselves from their host systems.
- **Automatically assign sites** - Runs through all of the Explorers that are not currently assigned to a specific site. It checks their IP address against the CIDR IP ranges of the registered subnets of all sites in the current organization. If the Explorer's IP address only matches a single site, the Explorer is assigned to that site.

Viewing Explorer details

Clicking on an Explorer's name takes you to a page showing the diagnostic information for that Explorer, including its software version, available memory, and network interfaces.

At the bottom of the page is a diagnostics text area. Clicking the *Update Diagnostics* button will fetch an updated list of all sub-processes active within the Explorer. This is useful to send to runZero support if you are having problems with a particular Explorer.

Internet speed tests

Explorers can be configured to run internet speed tests on recurring hourly intervals, or run on-demand as desired. The most recent speed test results are shown in the “Internet speed tests” table, which can be useful in monitoring the network performance of the Explorer’s host machine over time.

Speed tests can be initiated on-demand by clicking the “Start speed test” button, which will run a single speed test and update the table with the results. To configure recurring speed tests:

1. From the **Registered Explorers** page, select the Explorer you wish to run speed tests.
2. In the “Explorer details” card, click the “Edit” button and enter a number into the “Speed tests” input for how often you want the speed test to run, in hours.
3. Click **Save** to confirm your configuration and initiate recurring speed tests at the selected interval.

Note: your Explorer version must be up-to-date to run speed tests. If needed, click the “Update Explorer” button to update it to the latest version and then configure it as desired.

The table of speed test results can also be downloaded in CSV, JSON or JSONL format by clicking the “Download speed test results” button and selecting the desired file type. Additionally, any time a speed test is run, a `speedtest-completed` event will be generated in the runZero audit log with its results for tracking purposes.

Traffic sampling

The Explorer details page is also where users can configure traffic sampling.

1. From the **Registered Explorers** page, select the Explorer you wish to configure to perform traffic sampling.
2. In the traffic sampling card, configure the following options:
 - **Site:** Specify the site the assets discovered as a result of Traffic Sampling will be added to.
 - **Discovery scope:** List the IP addresses or CIDR networks that traffic sampling will observe on this Explorer.
 - **Asset tags** (optional): List the tags you want applied to assets discovered through traffic sampling.
 - **Excluded hosts** (optional): List the IP addresses or CIDR networks that traffic sampling will exclude from the results.
 - **Interfaces:** Toggle the switches for the interfaces you want this Explorer to listen on.
3. Click **Save** to save your configuration and initiate the traffic sampling task.

Once configured, traffic sampling can be disabled by returning to this page and toggling off the selected interfaces. Upon saving, the traffic sampling tasks will automatically stop.

Explorer Groups

With runZero Explorer groups, a user can configure a logical grouping of Explorers that will intelligently distribute scheduled tasks amongst themselves. Using Explorer groups makes it easier to schedule a large number of tasks, without having to manage which Explorer they run on.

Creating an Explorer group:

Explorer groups can be created and managed from the [Explorer groups](#) page. When creating a new Explorer group, configure a name, description, site, and select the Explorers you want in the group. It should be noted that all Explorers in the group must be set to the same site as the group. This is due to the fact that there is no guarantee of which Explorer in the group will run a given task. If this was not enforced, assets could be merged into the wrong site.

Using an Explorer group:

When configuring a scan or integration task, select an Explorer group and the group will elect the best fit Explorer for the tasks.

Upgrading npcap

On Windows, runZero uses a licensed third-party library called [npcap](#) for access to raw network traffic. Other software installed on the Explorer's host machine may also use npcap, and sometimes will have installed obsolete versions of the software. This can cause reliability problems.

runZero will alert you to obsolete versions of npcap by displaying a warning icon in the list of Explorers.

However, runZero cannot yet reliably upgrade npcap for you. runZero can't automatically upgrade npcap/winpcap, as it tends to be shared between applications, and forcing an upgrade from the runZero side can break other services (EDRs, Wireshark, etc).

To upgrade npcap manually:

1. Stop any running runZero services. This can be done using the Windows Services app. You'll need to look for "runZero Network Discovery Explorer".
2. Stop any other running software which uses npcap.
3. Uninstall Winpcap and any npcap installations via the Windows Control Panel.
4. Reboot the computer.

runZero will restart automatically, and install the latest npcap.

Verifying binaries

runZero uses dynamically generated binaries for the runZero Console, CLI, and Explorer downloads. Although Windows binaries have a valid Authenticode signature, all binaries also

contain a secondary, internal signature. Dynamic binaries make it easy to deploy Explorers that connect back to the right organization, but present a challenge for independent integrity validation. To enable verification of the internal signature, we offer the **runZero Verifier**. This verification tool can confirm whether a given binary contains a valid internal signature, in addition to any existing Authenticode signatures.

To get started, download the latest version of the verifier from the bottom of this page along with the PGP signature file for the selected architecture.

The runZero Verifier is always signed by PGP Key ID [60EBAAE9AEF08C6D](#).

To validate the signature of the runZero Verifier for Windows 64-bit, you will need a [GPG](#) client and to run the following commands:

```
C:\> curl -s https://www.runzero.com/.well-known/security.pub.asc | gpg --import
C:\> gpg --verify runzero-verifier-3.1.0-windows-amd64.exe.asc
```

Successful validation will show a valid signature by key ID 9B5DAFF7D43349298A3039BD60EBAAE9AEF08C6D.

```
gpg: Signature made Sun 07 Aug 2022 11:33:15 AM CDT
gpg:          using RSA key 9B5DAFF7D43349298A3039BD60EBAAE9AEF08C6D
gpg:          issuer "security@runzero.com"
gpg: Good signature from "runZero Security <security@runzero.com>" [unknown]
```

The warning below is expected and does not indicate a problem with the signature:

```
gpg: WARNING: This key is not certified with a trusted signature!
gpg: There is no indication that the signature belongs to the owner.
```

Once the runZero Verifier itself has been validated, it can be used to check the signature of any runZero binary:

```
C:\> runzero-verifier-3.1.0-windows-amd64.exe runzero-explorer-3.1.0-windows-amd64.exe
runzero-explorer-3.1.0-windows-amd64.exe: VALID SIGNATURE
```

A failed validation will show the error Invalid or missing signature and the verifier will set exit status to 1.

Binary downloads

Windows

Build	PGP sig	SHA hash
runZero Verifier x86 64-bit	pgp signature	sha-256
runZero Verifier x86 32-bit	pgp signature	sha-256

Linux

Build	PGP sig	SHA hash
runZero Verifier x86 64-bit	pgp signature	sha-256
runZero Verifier x86 32-bit	pgp signature	sha-256

Additional Linux builds

Build	PGP sig	SHA hash
runZero Verifier ARM v5 32-bit	pgp signature	sha-256
runZero Verifier ARM v6 32-bit	pgp signature	sha-256
runZero Verifier ARM v7 32-bit	pgp signature	sha-256
runZero Verifier ARM 64-bit (aarch64)	pgp signature	sha-256
runZero Verifier PPC 64-bit Little Endian	pgp signature	sha-256
runZero Verifier MIPS 32-bit Big Endian	pgp signature	sha-256
runZero Verifier MIPS 32-bit Little Endian	pgp signature	sha-256
runZero Verifier MIPS 64-bit Big Endian	pgp signature	sha-256
runZero Verifier MIPS 64-bit Little Endian	pgp signature	sha-256
runZero Verifier S390X	pgp signature	sha-256

MacOS

Build	PGP sig	SHA hash
runZero Verifier x86 64-bit	pgp signature	sha-256
runZero Verifier ARM 64-bit	pgp signature	sha-256

BSD Variants

FreeBSD

Build	PGP sig	SHA hash
runZero Verifier x86 64-bit	pgp signature	sha-256
runZero Verifier x86 32-bit	pgp signature	sha-256
runZero Verifier ARM v6 32-bit	pgp signature	sha-256
runZero Verifier ARM v7 32-bit	pgp signature	sha-256

NetBSD

Build	PGP sig	SHA hash
runZero Verifier x86 64-bit	pgp signature	sha-256
runZero Verifier x86 32-bit	pgp signature	sha-256
runZero Verifier ARM v5 32-bit	pgp signature	sha-256
runZero Verifier ARM v6 32-bit	pgp signature	sha-256
runZero Verifier ARM v7 32-bit	pgp signature	sha-256

Dragonfly

Build	PGP sig	SHA hash
runZero Verifier 64-bit	pgp signature	sha-256

OpenBSD

Build	PGP sig	SHA hash
runZero Verifier 64-bit	pgp signature	sha-256

Full-scale deployment

As you get started with runZero, we recommend kicking off with our standard deployment plan and adding tasks as needed. The standard deployment plan is broken out into six stages which will help you plan out your requirements, execute the deployment, and optimize your environment based on runZero's best practices.

1. Identify key success outcomes

Total attack surface visibility

- [Active discovery on all internal assets](#)
- [Active discovery on all externally facing assets](#)
- [Passive discovery and enrichment in key network segments](#)
- [Integrate with all cloud providers and other relevant data sources](#)

Additional Resources

- [Overview](#)
- [RFC1918 coverage playbook](#)
- [Scanning OT playbook](#)

Full-spectrum exposure detection

- [Rapid Response findings and asset-level pivoting](#)
- [Network misconfiguration findings and control coverage gaps](#)
- [Vulnerability enrichment and inside-out findings](#)

Additional Resources

- [Overview](#)
- [Rapid Response blog](#)
- [Gaps in EDR playbook](#)
- [Gaps in VM playbook](#)

Risk prioritization and insights

- [Custom dashboards](#)
- [Rules and alerts](#)
- [Setting asset criticality and ownership](#)

Additional Resources

- [Overview](#)
- [Custom Dashboards documentation](#)
- [Alerting playbook](#)

Compliance, Reporting, and KPIs

- [Comply with asset inventory and discovery requirements of relevant frameworks](#)

- [Comply with secure configuration requirements of relevant frameworks](#)
- [Comply with malware protection requirements of relevant frameworks](#)
- [Comply with vulnerability management requirements of relevant frameworks](#)

Additional Resources

- [Overview](#)
- [Compliance alignment documentation](#)
- [NYDFS dashboard playbook](#)

2. Planning your deployment

This first set of tasks will help your team identify target results, get ahead of potential blockers, and help you avoid misconfigurations within runZero.

Tasks

- Identify key organizational stakeholders
 - Administrator(s) - who will be setting up runZero?
 - Integration owner(s) - who will provide credentials for each integration?
- All users take the [runZero 101 training](#)
- Administrators take the [runZero 201 training](#)
- Determine whether self-hosting is required ([docs](#) | [video](#))
- Determine if you will use SSO or local accounts for user access.
 - If SSO, you will want to start any internal processes for getting the [runZero app created](#) in your provider.
- Identify known networks and subnets for discovery and other inventory sources. ([docs](#) | [video](#))
 - If available, prepare [CSV files](#) for bulk importing subnets.
 - Firewalls should also be configured to allow the Explorer IPs to scan the entire network on all ports.
- Define organizations based on RBAC requirements, including configuring organization-level data retention and expiration thresholds ([docs](#) | [video](#)).
- Determine [Explorer](#) deployment location(s).
 - [Explorer Groups](#) can be used to logically combine Explorers for streamlined task scheduling.
- Identify required inbound integrations, prioritizing connectors for platforms such as EDR and CSPM to get full vulnerability and misconfiguration coverage.
- Familiarize the team with the runZero search to plan initial asset searches and reporting requirements.

3. Initial configuration

Once you have your plan in place, it's time to execute and run your initial scans. **Please note** that these configuration tasks are in a prioritized order to help you avoid having to reconfigure things down the road.

Tasks

- Deploy self-hosted console (if required) ([docs](#) | [video](#))
- Setup [organizations](#), ensuring expiration thresholds for stale data, integrations, and vulnerabilities are configured.
- Set up [sites](#), and [define subnets](#) for discovery ([video](#))
 - **Sites do not necessarily correspond to physical locations within runZero.** Sites are used to represent distinct networks that may have overlapping IP space
 - This also includes Public IP space and domains for external scanning.
- Install [Explorer\(s\)](#) ([video](#))
 - If planned, configure Explorer Groups to logically organize scanners for streamlined task distribution.
- Define private network addresses
 - You can set custom Private IP ranges if you use Public IP ranges internally in [Account Settings](#).
- Run initial scan ([docs](#) | [video](#))
- [Configure credentials](#) and inbound [integration](#) connections
 - [Cloud inventory sources](#)
 - [Endpoint protection](#)
 - [Vulnerability management](#)
 - [Directory services](#)
 - [Custom Integrations](#)

4. Review Assets and Exposures

Now that you have done some initial discovery and leveraged integrations, it's time to review the results. Reviewing the results and leveraging our advanced reporting features will help you expand scan scope, prioritize risk, better understand your network, and identify key exposure issues such as misconfigurations and actively exploited vulnerabilities.

Tasks

- **Review Results and Exposure Overview**
 - Review the **Risk Management Dashboard** (new default dashboard) for a centralized view of risks. This dashboard provides insights, trend data, and breakdowns of assets ([docs](#)).
 - Check the **Latest Rapid Response alerts** widget, which includes a carousel of the five most recent Rapid Response posts and displays matches in your inventory ([docs](#)).
 - Leverage the capability to create and share **multiple, custom dashboards** for different use cases or teams (e.g., Compliance or Vulnerability Management) ([docs](#)).
 - Review the Asset Inventory ([docs](#)). The inventory correlates and merges assets across all data sources (scan, passive, and integrations) to provide a single source of truth.

- Review the Asset Detail View, noting the consolidated view of attributes, vulnerabilities, and software ([docs](#)).
- **Identify and Prioritize Exposures with Findings**
 - Review the **Findings** section, which aggregates vulnerabilities and misconfigurations into prioritized risk categories (e.g., **Internet Exposure**, **Open Access**, **End-of-Life**, **Certificates**, **Vulnerability**, and **Best Practice Violations**) ([docs](#)).
- **Deep Investigation and Querying**
 - Identify risky assets using the Queries library ([docs](#)):
 - Learn query syntax ([docs](#)).
 - Apply vulnerability records to queries (e.g., for novel internal findings) ([docs](#)).
 - Filter assets and services directly by their associated **Finding Code** (e.g., `finding_code:rz-finding-internet-exposed-database`) to target remediation ([docs](#)).
 - Search for assets exposed to **Known Exploited Vulnerabilities (KEV)** lists (CISA KEV, VulnCheck) and leverage **EPSS scores** for priority assessment.
- **Track Long-Term Initiatives**
 - **Track long-term initiatives with Goals** ([docs](#)).
 - Utilize **Baseline Goals** to measure progress against specific inventory subsets (e.g., setting a goal to remediate all expired certificates or critical vulnerabilities in the `os_eol:<now` subset).
- **Review Reporting**
 - Review reports ([docs](#)).
 - Identify gaps in scanning using reports ([docs](#)).
 - Understand network segmentation ([docs](#)).

5. Optimization

After you've done your initial analysis, you will want to optimize your scans and configurations to follow best practices.

Tasks

- [Configure SNMP credentials](#) ([video](#))
- Optimize scans by adjusting scan rates and other configurations ([docs](#) | [video](#))
 - See our clickthrough of some key additional configuration options as well
- Ensure default credential and vulnerability checks are enabled on all scans.
 - Current coverage can be found [in this documentation](#).
- Configure [asset ownership](#) to streamline investigations.

6. Automation

Now that you have optimized your scans and have analyzed your runZero data, you can automate these tasks to avoid manual effort. You can leverage this automation to run scans on a recurring basis, automate queries, and generate alerting for the team.

Tasks

- Schedule recurring scan [tasks](#) and any inbound [integration tasks](#)
- [Automate queries](#) and [configure alerts](#) to align with use cases ([video](#))
- Use rules to automate tagging and setting asset criticality ([walkthrough](#))
- Setup alerts for [system events](#) like Explorer offline and scan failures.
- Configure outbound [integration](#) connections to enrich other IT and security tools
 - [CMDB](#)
 - [SIEM](#)
 - [SOAR](#)
 - **Note:** If you're utilizing a solution that runZero does not offer a standard outbound integration for at this time, be sure to review our [API documentation](#) to learn about how to export runZero data.

7. Rollout

As your runZero deployment comes to a close, you will want to ensure all users have gone through training and ensure anyone that would get value from runZero has access to the platform.

Tasks

- [Add users](#)
- Ensure all users are trained on runZero
 - Training and key documentation
 - [101 user training](#)
 - [201 administrator training](#)
 - [Use case library](#)
 - [Search](#)
 - [Reporting](#)
 - [Exporting assets](#)
 - [runZero playbooks](#)
- Identify other teams interested in the asset inventory data, such as:
 - [Enterprise security team](#)
 - runZero is typically used by security teams to achieve a complete asset inventory, find gaps in their vulnerability scanning and endpoint protection, as well as discover potential vulnerabilities.
 - [IT Operations team](#)
 - runZero is typically used by IT Operations teams to achieve a complete inventory of all assets across on-premise and cloud-based infrastructure.

This allows the team to identify misconfigurations as well as report on assets in the environment by leveraging our searching and reporting capabilities.

- [Penetration testing team](#)
 - runZero is typically used by penetration testing teams for conducting reconnaissance both internally and externally, identifying vulnerable targets, and finding ways to get to these vulnerable targets by using our reporting and searching capabilities.

Additional Resources

Now that runZero has been deployed and users have been trained on the platform, please review some of our additional resources to help answer questions you might have as well as maximize the value of runZero:

- [runZero playbooks](#)
- [Use Case Library](#)
- [Leveraging the API](#)
- [Glossary](#)
- [runZero FAQs](#)

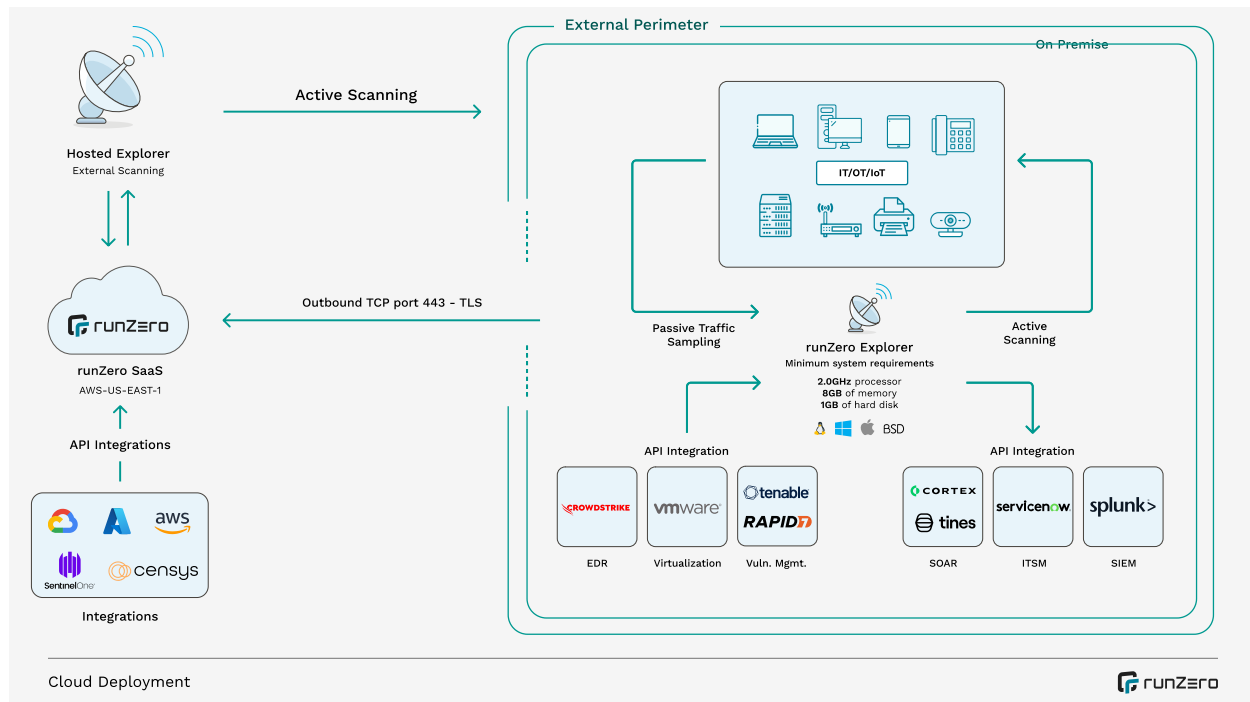
Getting help

If you need assistance at any point in this process, you can [book a session with a runZero Customer Success Engineer](#) to discuss further.

Types of networks

It is often helpful to use network examples as a starting point for planning your runZero implementation. This document breaks down a few standard network types and provides potential configurations for each. With that being said, every network has nuance, so it's likely there will be some differences for your implementation.

This is a basic overview of how discovery will be done using Explorers and scanners. By default, one Explorer will be deployed with the goal of running discovery on as much of the network as possible. If needed, more Explorers can be added for areas the primary Explorer cannot get to. You can also use a scanner for offline environments where there is no internet connectivity.



SaaS company

Network characteristics

- Flat on-premises network for offices
- Multi-cloud environment
- Hybrid workforce remote and in office

Sample runZero implementation

- Explorers
 - Corporate network Explorer that is able to get all on-premises network and cloud services private connectivity
 - One Explorer per VPC that isn't routable from on-premises networks
- Organizations
 - Single organization
- Sites
 - Primary corporate site
 - One site per VPC

Large corporation

Network characteristics

- Corporate offices with many connected branches

- Multiple M&A transactions happening at any given time, onboarding new offices every year
- Multiple data centers for internal and externally-facing applications

Sample runZero implementation

- Explorers
 - Corporate network Explorer that is able to get all on-premises networks
 - One Explorer per site with low bandwidth or legacy firewalls, proxies, etc.
 - One Explorer for each M&A transaction to understand new risks and keep data segmented
- Organizations
 - Corporate assets
 - One project per M&A transaction until the deal is complete and assets are merged
- Sites
 - Primary corporate site
 - Potentially separate sites if there is overlapping IP space in branch offices

Retail company

Network characteristics

- Flat on-premise network for corporate offices
- Data centers
- 100s of retail locations

Sample runZero implementation

- Explorers
 - Corporate network Explorer that is able to get all on-premise networks
 - One Explorer per retail location with low bandwidth or legacy firewalls, proxies, etc.
 - One Explorer per data center
- Organizations
 - Single organization
- Sites
 - Primary corporate site
 - One per retail site if they have overlapping IP space

Manufacturing company

Network characteristics

- Flat on-premise network for corporate offices
- OT environment is completely disconnected from the internet
- Field service agents VPN in from varying locations

Sample runZero implementation

- Explorers
 - Corporate network Explorer that is able to get all on-premise networks
- Scanners
 - Manufacturing plant that is not connected to the corporate networks
- Organizations
 - Single organization
- Sites
 - Primary corporate site
 - One per OT site if there is overlapping IP space

Telecommunications company

Network characteristics

- Flat on-premise network for corporate offices
- Segmented data centers
- IoT devices scattered across the country
- Field service agents VPN in from varying locations

Sample runZero implementation

- Explorers
 - Corporate network Explorer that is able to get all on-premise networks
 - One for each segmented lab
 - One for each data center
- Organizations
 - Corporate network
 - Production network
 - One per lab
- Sites
 - One primary site per organization

Managed security service provider

Network characteristics

- Many customers spinning up/down in engagements
- Mostly small, flat networks
- Some customers have OT environments that are sensitive

Sample runZero implementation

- Explorers
 - One per customer

- Scanners
 - Only needed for customers with offline networks
- Organizations
 - Project per customer initially
 - Promote project to organization for long-term use
- Sites
 - One per customer but potentially multiple if a customer has overlapping IP space

Academic organization

Network characteristics

- Many buildings/networks spread around the campus with varying connectivity
- Multiple data centers managed by different departments
- Multiple labs for research and development that are disconnected from the rest of the network and the internet

Sample runZero implementation

- Explorers
 - Main network Explorer that is able to get all on-premise networks
 - One Explorer per building with low bandwidth or legacy firewalls, proxies, etc
 - One Explorer per data center
- Scanners
 - One for each disconnected lab
- Organizations
 - Main assets
 - One for each lab
- Sites
 - One primary per organization but potentially multiple if buildings have overlapping IP space

Organizations

An organization represents a distinct entity; this can be your business, a specific department within your business, or one of your customers. All actions, tasks, Explorers, scans, and other objects managed by runZero are tied to specific organizations and isolated from each other.

Your active organization can be switched by using the dropdown selector at the top right of the runZero Console. If your default role is Viewer or higher, you can select **All Organizations** from the dropdown to view the data for all of your organizations in the Dashboard and Inventory. The Queries page also supports the All Organizations view. Pages that are not compatible with the All Organizations view will be hidden until a single organization is selected.

runZero Community Edition is limited to a single organization.

Use cases for organizations

It is common to segment your assets to multiple organizations based on a few common requirements:

- **RBAC:** Users will only be able to see the assets in the organizations or the children of organizations they're added to.
- **Environments:** It is possible you do not want your development, production, or other assets in one pool and would prefer them to be segmented.
- **Customers:** Service providers generally have one organization per customer.

Creating organizations

Organizations can be created, modified, and deactivated by going to the organizations section within the console. Click the Organizations link under Global Settings.

Organization configurations

- **Name:** Name of the organization.
- **Parent organization:** Allows for sub-organizations, limited to three levels of nesting. Permissions will be inherited based on most privilege.
- **Description:** description of the organization.
- **Stale asset expiration:** Sets the number of days before stale assets are removed from the inventory. This applies to online and offline assets not seen within the configured number of days.
- **Offline asset expiration:** Sets the number of days before offline assets are removed from the inventory. This applies to offline assets not seen within the configured number of days.
- **Stale integration attributes expiration:** Sets the number of days before stale integration attributes are removed from the inventory. Optionally keep the latest set of integration attributes per source, even if it is older than the threshold.

- **Stale vulnerability expiration:** Sets the number of days since last seen before stale vulnerabilities are permanently removed from the inventory, including archived vulnerabilities.
- **Enable vulnerability archiving:** (default: enabled) Applies to all vulnerabilities being removed from the active inventory. When disabled, vulnerabilities are deleted instead of archived. Archived metadata (such as suppression state and the original first seen date) is not retained if the vulnerability returns. Disabling this setting disables `Stale vulnerability archive`.
- **Stale vulnerability archive:** (default: 0, disabled) Sets the number of days since last seen before stale vulnerabilities are archived and removed from the active inventory. This should be set to a value less than the stale vulnerability expiration.
- **Scan data expiration:** Sets the number of days before completed scans are removed from the platform. This applies to the scan task metadata as well as the raw data files and change reports.

Projects

A project is a special type of organization designed for temporary use. They behave like organizations, and can have sites defined within them. The important difference is that projects cannot be parents of organizations or other projects, automatically become read only after 30 days, and are automatically deleted after 90 days. While organizations allow you to customize expiration times, the expiration times for projects are fixed.

Your runZero license includes the addition of up to five times as many project assets as the total number of licensed live assets. For example, if your license includes 1,000 live assets, you can have up to 5,000 project assets total. The total number of assets in organizations and projects can be seen on your [license information](#) page in the runZero console.

If you decide that you want to keep a project indefinitely, it can be converted into an organization.

Note that organizations support custom [data retention](#) settings that you can edit to your requirements.

Organization and project details

Click on the name of an organization or project to get to the details page. This page includes basic details about the organization or project, as well as its place in the organization hierarchy.

Sites

By default, your account includes a single organization, which itself contains a single site, named Primary. If the only site in an organization is deleted, a replacement will be created automatically. Similarly, if the last organization is removed, a replacement will be created. You can rename organizations and sites at any time.

Every organization has at least one site, but may have multiple sites. A site represents a distinct network segment, usually defined by addressing or accessibility. Sites in runZero do not necessarily correspond to physical sites or locations. Instead, they are used to represent distinct networks that may have overlapping address space. This allows for multiple sites to use the same RFC1918 space, something common in retail, while still being possible to differentiate their assets within the inventory.

Because sites represent separate networks, if you set up two sites and scan the same devices from both, you will end up with two copies of the resulting assets, one set for each site's network.

All analysis actions within runZero occur at the site level. For example, the [Network Map](#) builds its topology from a single site's devices, so you will likely want to avoid splitting routers and non-router assets into separate sites unless they are truly on separate disconnected networks.

Use cases for sites

For flat networks, where every IP address can reach any other address on the network, a single site is usually enough, and avoids the possibility of accidentally creating duplicate assets by scanning the same devices from multiple sites. Sites are recommended for complicated, sprawling, and highly-segmented environments.

Two circumstances that could lead to multiple sites:

- **Overlapping IP space:** sites will allow you to differentiate identical IPs that are actually different machines if you have overlapping IP space.
- **Highly complex network:** sites would not be required in this case, but they can be used as an organizational tool.

Creating sites

The ability to create sites must be enabled on your account before you can add new sites. If you do not see a **New Site** button, please reach out to support@runzero.com to have this feature enabled on your account.

To create a new site, click the **New Site** button on the top of the [sites](#) page.

Site configurations

- **Name:** The name of your site.
- **Description:** The description can help identify the purpose of the site.
- **Default scan scope:** The default scan scope will be pre-populated when creating scans for this site.
- **Default scan exclusions:** The default scan exclusions will be pre-populated when creating scans for this site.
- **Registered subnets:** Registered subnets can be used to automatically tag assets, services, screenshots, and software that fall within each subnet.

Subnet tagging

Tagging based on subnet works a little differently to directly tagging assets or tagging them via a task.

If a subnet is defined to have a specific tag, then only assets in that subnet will be given that tag. The tag will be removed from any assets not in the subnet, even if set by a task or set manually.

Importing subnets

Site subnets can be created or updated based on a CSV file import as part of the site configuration. Here is a sample CSV for import:

```
cidr,tags,description
192.168.0.0/16,purpose=endpoints,"Headquarters endpoint subnet ""quoted text""""
10.0.0.0/24,purpose=wifi,"Headquarters wireless subnet, with comma"
2001:db8::/32,"foo=bar,foobar",ipv6 subnet
1.2.3.4/16,,
```

Importing and exporting sites

Site configurations can be created or updated based on a CSV file import. [Import your CSV](#) from the sites page. The CSV format should include the following:

```
name,description,scope,exclusion,subnet_ranges,subnet_tags,subnet_descriptions
```

Your site configurations can also be exported as a CSV from the sites page.

Sites and Explorers

Sites can be tied to specific Explorers, which can help limit traffic between low-bandwidth segments. The site configuration allows a default scan scope to be defined, along with an optional list of excluded scan scopes. These fields can be used to set the scan scope for scans of the site.

If you would like to tie an Explorer to a site, navigate to the [Explorers](#) page, click the Explorer you would like to tie to the site, and then click configure. After that, you will see a dropdown with your site options.

Self-hosting runZero

Background

The self-hosted version of runZero allows you to run the entire platform on-premises or within your own cloud environment. This platform is functionally identical to the hosted service, provides a fully-offline mode, and does not send any inventory data back to runZero.

While self-hosting is less common, here are a few reasons your company might choose to:

- ISO compliance requirement
- Other compliance requirement
- Prefer data on-premise

Self-hosting must be explicitly enabled for your account. Self-hosting is available if you have a runZero Platform license, or if you are running an initial trial for a platform license. Please contact your runZero sales representative for further information.

Self-hosting is not available for the free Community Edition of runZero.

Requirements

Before you get started, make sure your system meets the following requirements.

Hardware requirements

Recommended production system requirements:

- 12 CPU Cores at 2 GHz or faster
- 1TB of local disk storage
- 128 GB of RAM

Minimum production system requirements:

- 4 CPU Cores at 2 GHz or faster
- 100 GB of local disk storage
- 32 GB of RAM (more for large sites)

Minimum testing system requirements:

- 2 CPU Cores at 2 GHz or faster
- 20 GB of local disk storage
- 16 GB of RAM (more for large sites)

Sample customer deployments

Up to 1M assets:

- Two virtual machines (one for console and one for database)
- 32 CPU Cores at 2 GHz or faster
- 1TB of local disk storage
- 128 GB of RAM

Up to 500K assets:

- 32 CPU Cores at 2 GHz or faster
- 1TB of local disk storage
- 128 GB of RAM

Up to 50K assets:

- 16 CPU Cores at 2 GHz or faster
- 1TB of local disk storage
- 64 GB of RAM

Software requirements

- PostgreSQL 14 or newer (17+ preferred, our installer can install this for you)

Supported operating systems

- Ubuntu 18.04 and newer running on x86_64
- Red Hat Enterprise Linux 7.x and newer running on x86_64
- CentOS Linux 7.x and newer running on x86_64
- Oracle Linux 7.x and newer running on x86_64
 - 8.x must be 8.4+ with UEK 5.4+ or kernel 4.18+
 - 7.x must be 7.9+ with UEK 5.4+ or kernel 3.10+
- Debian Linux 9.x and newer running on x86_64

Windows Subsystem for Linux is not supported.

The runZero console can be installed in a minimal install of Ubuntu or any RHEL derivative; you do not need to install a GUI. The default disk partitioning scheme will work, but you may want to consider using LVM if you plan to resize the disk later.

Note about Debian

By default, Debian Linux uses the `su` command instead of `sudo`. It also requires that you use the command `su -` to become root, not just `su`, in order to update the `PATH` to include system administration commands such as `useradd`. If you receive an error during installation saying executable file not found in `$PATH`, this is the most likely reason.

Connectivity

The self-hosted runZero platform requires connectivity to the runZero SaaS backend over TCP port 443 (TLS) to obtain online updates. The specific IP addresses and hostnames depend on your deployment model and region and can be found below.

United States

The console hostname is `console.runzero.com`.

IPv4

- 13.248.161.247
- 76.223.34.198

IPv6

- 2600:9000:a415:cd87:fbe5:476a:3533:69f2
- 2600:9000:a716:ee91:85f9:3c9:48c9:59b9

Germany

The console hostname is `console-eu.runzero.com`.

IPv4

- 15.197.131.232
- 3.33.248.90

IPv6

- 2600:9000:a603:e925:542d:6d40:6897:bc3a
- 2600:9000:a70e:635f:71bd:bb0a:8e43:9466

The self-hosted runZero platform will also require connectivity to the following sites over TCP port 443 during installation and for subsequent software updates.

- `apt.postgresql.org`

Platform Address & TLS Configuration

The system running the runZero platform should have a static IP address.

The Explorers need to be able to validate their HTTPS connection to the platform via a TLS certificate. By default, the runZero platform installer will set up a self-signed certificate for the console's IP address, and downloaded Explorers will be preconfigured with the appropriate URL and certificate so they can make a verified connection.



Some components of the application still reference the name "Rumble" for backwards compatibility. The documentation will be updated as these are changed.

Explorers are preconfigured with the console address and certificate during download. If the IP address or DNS name changes later, the Explorers will need to be redeployed to receive the updated address. Renewing a self-signed certificate also requires redeploying the Explorers.



It is recommended to use a real certificate from an internal or public CA for your runZero console deployment. With such a setup, explorers will not need to be reinstalled if the server's IP address is changed or the certificate is renewed. As long as the DNS name does not change and the new certificate can still be validated, explorers will continue to function.

To use a real certificate from an internal or public CA for your runZero console deployment, you will want to assign the console server a DNS name, and edit `RUNZERO_CONSOLE` in the configuration to contain the fully qualified domain name. The `TLS_CERT` and `TLS_KEY` should be set to point at the appropriate certificate and private key files. The console should then be restarted via `runzerectl restart`. Newly downloaded Explorers will then be preconfigured with the right URL and certificate.

If the platform is being deployed behind a reverse proxy, the reverse proxy may need additional configuration to pass the `Origin` header through to the runZero Console. This header is required for CSRF protection, and some functionality (such as the data tables throughout the product) will not work if the header isn't passed through, or is incorrect.

Offline mode

The self-hosted version of runZero has the ability to run in [offline mode](#). In this mode, the runZero update service is not used and [offline updates](#) must be applied manually. Enable this mode if you're in an isolated network or you don't want your self-hosted runZero console to make any connections to the internet. In addition to disabling online updates, offline mode also disables certain DNS probes that could reflect responses to the internet during a scan.

Installation steps

For offline installs please see [offline installation](#).

For installs that use your own database credentials see [Installation with your own PostgreSQL database](#).

Here's what the installation process will do:

- Set up PostgreSQL and create a dedicated user with a strong password.
- Generate TLS certificates for your IP address located in `/etc/runzero/certs`.

- Generate a configuration file at `/etc/runzero/config` and set some defaults.
- Create a `systemD` or `RC` service for the runZero platform.

Step 0: Ensure a clean starting environment

If you've previously ran the installer and encountered an issue related to disk space, memory, or repository verification, you may need to clean up some temporary files before retrying the install:

Remove any temporary files left from the previous install:

```
# rm -rf /tmp/update-tmp* /tmp/runzero-updates*
```

On RHEL and CentOS systems, clear any leftover PostgreSQL remnants:

```
# dnf remove postgresql
# rm -rf /var/lib/pgsql/
```

Use the installer binary to purge any leftovers:

```
# ./runzero-install.bin uninstall --purge
```

Step 1: Download and run the installer

- Go to <https://console.runzero.com/deploy/download/platform>.
- Copy the command directly from the download page and run it in your terminal. This will download the installer and mark it as executable. The download path for the installer is uniquely keyed.
- Run the installer.
- By default, the console will be installed to `/opt/runzero`. If you need to change this, you can use the `--install-directory` option.

Step 2: Ensure sufficient disk space

By default, the runZero console will store data in the `storage` directory underneath the install directory. By default, that will be `/opt/runzero/storage`. The `storage` directory must have enough space to store all unexpired scan data and asset data. If necessary you can override its location by setting the `RUNZERO_STORAGE_PATH` variable in the runZero configuration file `/etc/runzero/config`.

You will also want to ensure that the temporary directory has plenty of space for temporary files such as software and content updates. If the `TMPDIR` environment variable is set, that directory will be used; otherwise the default is `/tmp`.

Step 3: Initialize the admin user

After you've installed the runZero platform, you will have access to the runZero CLI `runzeroctl`.

To initialize an admin user, run:

```
runzeroctl initial [email address]
```

Step 4: Sign in to your self-hosted console

If everything is set up correctly, you can sign in to your console at <https://YourInternalIPAddress>.

Note that you may need to enable HTTPS to pass through the Linux system firewall. Example commands:

Ubuntu Linux: `sudo ufw allow https/tcp`

RHEL/CentOS/Oracle: `sudo firewall-cmd --add-service=https`

To make a `firewall-cmd` change permanent across reboots, run the command a second time with the `--permanent` flag added.

Installation with your own PostgreSQL database

runZero defaults to installing and configuring a PostgreSQL user and database for you. If you would like to provide your own details this option will allow you to override that behavior.

Requirements

- PostgreSQL 13 or newer (16+ preferred)
- [Password authentication](#) must be enabled.
- Two extensions are required: [pg_trgm](#) and [uuid-oss](#). These may be supplied as part of a contrib package rather than as part of the main PostgreSQL server install, depending on where you obtain your PostgreSQL packages.

PostgreSQL example to enable extensions and add a database/user

```
CREATE DATABASE runzero TEMPLATE='template0' LC_COLLATE = 'en_US.UTF-8' LC_CTYPE = 'en_US.UTF-8';
CREATE USER runzero WITH PASSWORD 'YOURPASSWORD';
GRANT ALL PRIVILEGES ON DATABASE runzero TO runzero;
\connect runzero;
CREATE EXTENSION IF NOT EXISTS pg_trgm;
CREATE EXTENSION IF NOT EXISTS "uuid-oss";
```

Steps to install the self-hosted runZero platform with your own database credentials

1. Run this install command as root:

```
./runzero-platform-[VERSION]-linux-amd64.bin install --manual-database
```

2. Edit your runZero configuration and add your database details in `/etc/runzero/config`. The line you need to edit is:

```
DATABASE_URL=postgres://runzero:{DB_PASSWORD}@127.0.0.1:5432/runzero?sslmode=disable
```

Change this to match your credentials. You need to set the user, password, host, port, and database name. Here is the format:

```
DATABASE_URL=postgres://{DB_USER}:{DB_PASSWORD}@{DB_HOST}:{DB_PORT}/{DB_NAME}?{DB_OPTIONS}
```

4. Verify the self-hosted runZero platform can connect to your database with this command. `sudo runzeroctl database verify`
5. Once your database is configured and verified you can restart the self-hosted runZero platform service `sudo systemctl restart runzero-console`

runZero updates

For offline updates please see [CLI update with offline mode](#).

The self-hosted runZero platform must be updated prior to first use. The `runzeroctl` command can be used to download the update and then restart the service after the update is complete.

You can update the platform and scanners at the same time or separately with the [CLI update management commands](#).

Managing users

You can manage users inside your self-hosted runZero platform console at `https://YourInternalIPAddress/team` or via the [runZero CLI](#)

Some things you can manage:

- Adding, deleting, and listing users
- Resetting passwords
- Changing default roles
- Viewing details
- Changing organization roles

CLI service management

Start the runZero service

Starts the runZero platform service.

```
runzeroctl start
```

Stop the runZero service

Stops the runZero platform service.

```
runzeroctl stop
```

Restart the runZero service

Restarts the runZero platform service.

```
runzeroctl restart
```

Install the runZero platform

Install the runZero platform service and all required dependencies such as PostgreSQL. Creates a systemd service, generates cron jobs, and generates a configuration file in `/etc/runzero/config`.

```
runzeroctl install
```

Uninstall the runZero platform

Stop and remove the runZero platform service from systemd and removes the generated cron jobs. This does not remove your PostgreSQL database, and it retains your data.

```
runzeroctl uninstall
```

Purge the runZero platform

Stop and remove the runZero platform service from systemd and removes the generated cron jobs. This will delete your runZero database and remove all the runZero directories `/etc/runzero` and `/opt/runzero`.

```
runzeroctl purge
```

You can uninstall and purge everything except the database and your PostgreSQL settings with this flag:

```
runzeroctl uninstall --purge --ignore-database
```

Run the runZero platform manually

Starts the runZero platform manually. Logs will be written to standard output.

```
runzeroctl server
```

Verify your database is reachable

Attempts to connect to your database using your self-hosted runZero platform configuration. It will either succeed or display an error.

```
runzeroctl database verify
```

CLI update management

Update the runZero platform and scanners

Updates the runZero platform service and runZero scanners. You can use the optional parameter `force` to force the update even if the current installation is the latest version.

```
runzeroctl update [--force]
```

Update the runZero platform

Updates just the runZero platform service. You can use the optional parameter `force` to force the update even if the current install is the latest version.

```
runzeroctl update-platform [--force]
```

Update the runZero scanners

Updates just the runZero scanners. You can use the optional parameter `force` to force the update even if the current installation is the latest version.

```
runzeroctl update-scanner [--force]
```

CLI user management

Create the initial administrator account

Creates the initial admin user for a new installation. You must provide an email address.

```
runzeroctl initial <email>
```

List user accounts

Lists all the users along with their email address, full name, and current roles.

```
runzeroctl user list
```

Add a user account

Creates a new user account under the initial administrator user. You must provide an email address.

```
runzeroctl user add <email>
```

Delete a user account

Deletes a user account. You must provide an email address. This cannot be undone.

```
runzeroctl user delete <email>
```

Get user details

Gets the details for a user account. You must provide an email address. Provides information such as full name, date created, last sign in IP, last sign in time, last activity, default organization role, and their current roles. You must provide an email address.

```
runzeroctl user details <email>
```

Set a user role

Sets a user's role to the role provided. Email and role must be provided. The organization is optional. If the organization isn't provided this sets their default role.

```
runzeroctl user set-role <email> [organization name or organization ID]:<role>
```

Reset a user password and MFA

This will generate and apply a new password for the specified user. The password will be printed to the terminal. You must provide an email address. This reset process also clears any associated MFA tokens.

```
runzeroctl user reset <email>
```

CLI organization management

List all organizations

Lists all the organizations by their name and ID.

```
runzeroctl organization list
```

Advanced configuration

The file at `/etc/runzero/config` can be modified to support a wide variety of configurations.

After making changes, apply them by running `runzeroctl restart`.

Email server

runZero uses an SMTP server for user account invitations and notifications. The default configuration assumes that a SMTP server is available on localhost that does not require authentication:

```
SMTP_SERVER=127.0.0.1:25
SMTP_AUTH_METHOD=none
```

runZero will automatically use STARTTLS with plaintext SMTP servers and validate the certificate. In internal environments where the SMTP server is not using a valid TLS certificate, verification can be disabled by setting:

```
SMTP_TLS_NOVERIFY=true
```

Transport-layer TLS (instead of STARTTLS) can be configured with:

```
SMTP_TLS=true
```

If authentication is required, the following three settings should be configured:

```
SMTP_AUTH_METHOD=plain
SMTP_AUTH_USER=YourUsername
SMTP_AUTH_PASS=YourPassword
```

Note: at this time the only supported authentication methods are "login" and "plain".

Emails are sent from `noreply@runzero.com` by default, but this can be changed by setting the `FROM_EMAIL` option:

```
FROM_EMAIL=runzero@yourcompany.int
```

Hostname and port

The `RUNZERO_CONSOLE` variable is used for creating inbound links, configuring deployed Explorers, and generating the default self-signed TLS certificate. This setting is how both users and deployed Explorers connect to the platform.

```
RUNZERO_CONSOLE=https://{IP ADDRESS OR HOSTNAME}:443
```

Changing this setting may require regeneration of the TLS certificate and redeployment of Explorers.

```
runzeroctl generate-certificate
```

runZero can be configured to run on a different port with the `CONSOLE_PORT` setting. This port defines where the console listens, but users and Explorers still connect to the `RUNZERO_CONSOLE` value. In most cases this should match the port specified by the `RUNZERO_CONSOLE`.

```
CONSOLE_PORT=443
```

TLS configuration

runZero will generate a self-signed TLS certificate and serve all web requests using HTTP over TLS. The standard configuration uses a self-signed certificate stored in the filesystem:

```
TLS=true
TLS_CERT=/etc/runzero/certs/cert.pem
TLS_KEY=/etc/runzero/certs/key.pem
```

The certificate and key file are PEM encoded and can be replaced with any valid certificate. Please ensure that any new certificate lists the value of RUNZERO_CONSOLE in the list of Subject Alternative Names.

Note that the key file must be unencrypted (not password protected). If you have a passphrase set, you can remove it using OpenSSL. For example:

```
# RSA key
openssl rsa -in key.pem -out newkey.pem
# DSA key
openssl dsa -in key.pem -out newkey.pem
```

If a TLS-terminating reverse proxy is used (AWS ELB, nginx, etc), TLS can be disabled at the application level:

```
TLS=false
```

Please note that while the web interface can be accessed over plain HTTP in this scenario, Explorers will refuse to connect to a plain HTTP port, and features like WebAuthn MFA will not work unless the site is accessed through TLS.

Specific TLS versions and ciphers can be configured.

TLS versions are chosen by minimum and maximum:

```
TLS_VERSION_MIN=1.2
TLS_VERSION_MAX=1.3
```

TLS ciphers may be chosen by profile name:

- **default:** A set of strong ciphers, great for most configurations
- **nist80052:** A set of strong ciphers, approved in [NIST 800-52r2](#).
- **nist80052-aes256:** A set of strong ciphers, approved in [NIST 800-52r2](#), restricted to AES-256 variants

Please note that TLS 1.3 ciphers work differently and if a specific set of ciphers is required, both TLS_VERSION_MIN and TLS_VERSION_MAX should be set to 1.2. For example, to restrict runzero to **only** NIST 800-52r2 approved ciphers using AES-256, the following configuration should be used:

```
TLS_VERSION_MIN=1.2
TLS_VERSION_MAX=1.2
TLS_CIPHERS=nist80052-aes256
```

TLS ciphers may also be chosen using comma-separated list of [cipher constants](#).

Certificate Authorities (CAs)

The server uses the system-installed certificate authorities to validate TLS connections in addition to an internal CA certificate bundle. By default, both the system certificate roots and the bundled roots are considered for all secure TLS connections.

- The `RUNZERO_TLS_IGNORE_SYSTEM_ROOTCA` setting can be set to `true` to ignore the system CA roots.
- The `RUNZERO_TLS_IGNORE_EMBEDDED_ROOTCA` setting can be set to `true` to ignore the bundled CA roots.
- The `RUNZERO_TLS_ADDITIONAL_ROOTCA` setting can be set to a file path containing additional CA roots in PEM format.

Database configuration

runZero uses a PostgreSQL database for all platform data, except for raw scan files, change reports, and images processed from scans. By default, runZero will configure a local PostgreSQL server on the same system, with a random password, and without TLS encryption:

```
DATABASE_URL=postgres://runzero:{DB_PASSWORD}@127.0.0.1:5432/runzero?sslmode=disable
```

If separate database is preferred, any PostgreSQL server running 12.x or newer should work. TLS (`sslmode=require`) should be enabled when a non-local database server is configured.

The default database pool (connection count) can be modified for high throughput environments:

```
DATABASE_POOL_COUNT=50
```

Proxy configuration

runZero makes outbound connections to receive platform updates (in online mode), to connect to third-party APIs, and to delivery webhooks for notifications. If a proxy server is required, it can be configured with this setting:

```
HTTPS_PROXY=host:port
```

A separate proxy may be configured for AWS connections. This takes precedence over

```
HTTPS_PROXY:
```

```
HTTPS_PROXY_AWS=host-for-aws-services:port
```

In addition to standard HTTP proxies, a SOCKS proxy may be configured by specifying a URL like:

```
HTTPS_PROXY=socks5://socks.example.com:1080
```


Storage configuration

runZero uses local file storage to store raw scan data, change reports, and images retrieved from assets. This storage directory must be owned by the `runzero` user and be mounted below the `/opt/runzero` path.

```
RUNZERO_STORAGE_MODE=local
RUNZERO_STORAGE_PATH=/opt/runzero/storage
```

Files within the storage directory are split up into two groups, assets and scans. The names of these can be changed by setting:

```
ASSET_BUCKET=assets
SCAN_BUCKET=scans
```

To use AWS S3 for file storage instead, the following configuration can be set:

```
RUNZERO_STORAGE_MODE=s3
ASSET_BUCKET=company-runzero-assets
SCAN_BUCKET=company-runzero-scans
```

If a non-AWS backend is used that is compatible with the S3 API, use the same AWS and bucket variables above but override `AWS_REGION` and set the `AWS_ENDPOINT_URL_S3` or `RUNZERO_STORAGE_ENDPOINT` parameter to the endpoint as appropriate. Reach out to runZero support if you run into any issues with the endpoint configuration.

If the S3 buckets are in a different region than the environment, set the `RUNZERO_STORAGE_REGION` to the correct region for the buckets.

If S3 is used, AWS must also be [configured](#), with at least the `AWS_REGION` variable set, even if a non-AWS backend is enabled.

Secret configuration

runZero uses three randomly generated secret tokens to secure the platform. These keys are hexadecimal strings generated by 16 bytes of random. Compatible values can be generated by OpenSSL:

```
$ openssl rand -hex 16
```

The authentication key used for local storage HMAC operations. This key can be rotated as long as the service is restarted afterwards:

```
RUNZERO_STORAGE_KEY_SECRET={SECRET_32_HEX}
```

The session secret key is used to sign and validate browser session cookies. This key can be rotated, but doing so will invalidate all existing web sign-ins:

```
SESSION_SECRET={SECRET_32_HEX}
```

The DB key is used for encryption of sensitive fields (user password hashes). This key cannot be rotated, as password authentication will no longer work. If this key is changed, users must reset their password from the command-line or web interface using email before they can sign back in:

```
DB_KEY={SECRET_32_HEX}
```

AWS configuration

The AWS region is required:

```
AWS_REGION=us-east-1
```

The Access Key ID and Secret must be valid and correlated to a user with read-write access to the S3 buckets and read-only access to Secrets Manager.

```
AWS_ACCESS_KEY_ID=AKIA...
AWS_SECRET_ACCESS_KEY=SECRET...
```

AWS Secrets Manager can be used to retrieve almost any configuration setting at startup. The Secrets Manager entries should match the key names of the configuration file. The secret name can be defined with:

```
AWS_SECRETS_MANAGER_KEY=runzero/production
```

In addition the variables above, most [AWS CLI environment variables](#) are also available for specific tuning.

The location of the Explorer and scanner binaries can be changed with these settings. Note that these should still live under /opt/runzero or the service will not be able to load them:

```
RUNZERO_RELEASE_DIR=/opt/runzero/agent/
RUNZERO_SCANNER_RELEASE_DIR=/opt/runzero/scanner/
```

Content security policy

In the case of a non-standard S3 configuration (or S3-like deployment, such as minio), the Content Security Policy headers need to be configured to allow external image loads.

The CSP_IMAGES setting can be used to specify one or more (comma-delimited) external image sources:

```
CSP_IMAGES=https://*.custom-storage-backend.lan
```

In addition to CSP_IMAGES, the following additional CSP settings are available:

```
CSP_SCRIPTS=https://*.myscripts.lan
CSP_FONTS=https://*.myfonts.lan
CSP_STYLES=https://*.mystyles.lan
```

Logging

The self-hosted runZero console sends its logging output to standard output. On Linux this is picked up by Linux systemd, and stored in the journal where it can be queried with the `journalctl` tool. For example, to view the most recent hour of logs, with most recent messages first, you can run the following command:

```
journalctl --unit=runzero-console --since=-1hour --reverse
```

The `journalctl` output can be piped to a text file to send to runZero support.

The systemd logging subsystem can be configured to send log messages to a local syslog daemon as well as the journal. Once logs are in syslog, they can be forwarded across the network to remote logging servers using the standard syslog protocol. Some Linux distributions, such as RHEL, are set up to forward logs from systemd to syslog by default. Other distributions, such as Ubuntu and Debian, don't include a syslog daemon in their default minimal server installs.

To configure systemd to send logs to syslog, you can use the [ForwardToSyslog option](#) in `/etc/systemd.conf`. Alternatively, some syslog daemons have an option to read the systemd journal; for example, rsyslog has [imjournal](#).

runZero's logs are output in [CEE-enhanced](#) JSON format. This is compatible with rsyslog and [syslog-ng](#), DataDog, and other common logging tools. For rsyslog, the [mmjsonparse](#) module can be used to filter the logs based on individual JSON fields, and [forward them to ElasticSearch](#) or other JSON databases.

The environment variable `LOG_MAX_LENGTH` can be set in runZero's config file to apply a limit to the length of each log line, in bytes of UTF-8 text. A value of 0 means no limit, other values below 480 are treated as 480. runZero will attempt to preserve the most valuable logging fields when truncating log output, and ensure that the result is still valid JSON. Note that the length limit is applied before any additional information systemd or syslog adds to the start of the line.

The environment variable `LOG_FORMAT` can be set to `text` to disable the CEE and JSON format, and log in plain text. For example:

```
LOG_FORMAT=text
LOG_MAX_LENGTH=512
```

HTTP timeouts

The self-hosted runZero Console allows you to change the built-in web server's HTTP timeouts. These can be changed through three configuration variables.

You are able to set the HTTP idle, read, and write timeout. The default settings are below and don't usually need to be changed.

```
HTTP_IDLE_TIMEOUT_MINUTES=3
HTTP_READ_TIMEOUT_MINUTES=60
```

```
HTTP_WRITE_TIMEOUT_MINUTES=720
```

Concurrent processing

The runZero Console can process more than one completed task at a time if the `RUNZERO_CRUNCHER_INSTANCES` option is set to a value greater than 1. Tasks are only processed concurrently when they exist within different organizations, or the tasks are within the same organization, but different sites, and the task itself does not require cross-site asset merging. Most third-party connectors and integrations require cross-site merging and are not able to take advantage of concurrent site processing within the same organization. Please note that the resource requirements for concurrent task processing scale linearly with the specified instance count.

The example below configures the console to process up to four concurrent tasks across all organizations:

```
RUNZERO_CRUNCHER_INSTANCES=4
```

Custom JavaScript

The self-hosted runZero Console allows you to include custom arbitrary JavaScript to be executed on the various runZero Console web pages. To use this feature, add your JavaScript to `/opt/runzero/etc/custom.js`. If this is your first time configuring this feature, the `custom.js` file will not exist. You will need to create the file `custom.js` inside the `/opt/runzero/etc` folder.

To enable the feature set the environment variable below to your configuration.

```
RUNZERO_CUSTOM_JS=true
```

HTTP headers

The security headers sent by the runZero Console can be disabled as needed using the following options:

```
# Disable Strict-Transport-Security
RUNZERO_DISABLE_HSTS=true
```

```
# Disable X-Frame-Options
RUNZERO_DISABLE_HXFO=true
```

```
# Disable X-Content-Type-Options
RUNZERO_DISABLE_HXCTO=true
```

```
# Disable X-XSS-Protection
RUNZERO_DISABLE_HXXP=true
```

```
# Disable Referrer-Policy
RUNZERO_DISABLE_HRP=true
```

```
# Disable Content-Security-Policy
```

```
RUNZERO_DISABLE_HCSP=true
```

```
# Disable CSRF-Protection
RUNZERO_DISABLE_CSRF=true
```

API availability

The following setting controls whether public APIs are enabled (ex: /health).

```
# Disable all unauthenticated API endpoints (/health)
RUNZERO_DISABLE_PUBLIC_APIS=true
```

Disabling entitlements

Any license entitlement can be force-disabled for the entire instance by setting `RUNZERO_DISABLE_<ENTITLEMENT>` to `true`, where `<ENTITLEMENT>` is the uppercased entitlement name. This overrides the license: it can only remove an entitlement, never grant one, and it applies to every account and organization on the instance.

For example, to disable autonomous (autoscan) discovery everywhere:

```
RUNZERO_DISABLE_AUTOSCAN=true
```

The full set of supported variables is listed below, one per entitlement. Each is commented out; set the value to `true` to disable that entitlement.

```
# Asset ownership
# RUNZERO_DISABLE_ASSET_OWNERSHIP=true

# Risky assets
# RUNZERO_DISABLE_RISKY_ASSETS=true

# Passive monitoring, traffic sampling, and PCAP import
# RUNZERO_DISABLE_PASSIVE_MONITORING=true

# Manage Explorer groups
# RUNZERO_DISABLE_EXPLORER_GROUPS=true

# Goal tracking
# RUNZERO_DISABLE_GOAL_TRACKING=true

# Unlimited organizations (not just one)
# RUNZERO_DISABLE_ORGS_UNLIMITED=true

# Unlimited projects (not just one)
# RUNZERO_DISABLE_PROJECTS_UNLIMITED=true

# Continuous scanning and variable TTL
# RUNZERO_DISABLE_SCANS_UNLIMITED=true

# Concurrent scanning on Explorers
```

```
# RUNZERO_DISABLE_SCANS_CONCURRENT=true

# Subnet ping and host ping (sampling) modes
# RUNZERO_DISABLE_SCANS_SAMPLING=true

# Advanced scan targeting
# RUNZERO_DISABLE_SCANS_TARGETING=true

# Autonomous (autoscan) discovery scanning
# RUNZERO_DISABLE_AUTOSCAN=true

# AI features
# RUNZERO_DISABLE_AI=true

# Organization API access
# RUNZERO_DISABLE_API_ORG=true

# Account (client) API access
# RUNZERO_DISABLE_API_ACCOUNT=true

# Adjust organization data retention settings
# RUNZERO_DISABLE_DATA_RETENTION=true

# Rule engine processing
# RUNZERO_DISABLE_RULES_BASIC=true

# Basic reporting
# RUNZERO_DISABLE_REPORTS_BASIC=true

# Disable external (shareable, email-verified) report links
# RUNZERO_DISABLE_DISABLE_EXTERNAL_REPORTING=true

# ServiceNow export APIs
# RUNZERO_DISABLE_INT_SERVICENOW=true

# Splunk sync APIs
# RUNZERO_DISABLE_INT_SPLUNK=true

# Cisco export APIs
# RUNZERO_DISABLE_INT_CISCO=true

# HP export APIs
# RUNZERO_DISABLE_INT_HP=true

# Censys search import
# RUNZERO_DISABLE_INT_CENSYs=true

# CrowdStrike import
# RUNZERO_DISABLE_INT_CROWDSTRIKE=true

# Microsoft 365 Defender import
# RUNZERO_DISABLE_INT_MS365DEFENDER=true

# Miradore import
```

```
# RUNZERO_DISABLE_INT_MIRADORE=true

# Nessus import
# RUNZERO_DISABLE_INT_NESSUS=true

# Rapid7 Nexpose import
# RUNZERO_DISABLE_INT_RAPID7=true

# SentinelOne import
# RUNZERO_DISABLE_INT_SENTINELONE=true

# VMware API import
# RUNZERO_DISABLE_INT_VMWARE=true

# Tenable vulnerability management
# RUNZERO_DISABLE_INT_TENABLE=true

# Prisma Cloud integration
# RUNZERO_DISABLE_INT_PRISMA=true

# Tenable Security Center integration
# RUNZERO_DISABLE_INT_TENABLE_SECURITY_CENTER=true

# Rapid7 InsightVM vulnerability management
# RUNZERO_DISABLE_INT_INSIGHTVM=true

# Qualys vulnerability management
# RUNZERO_DISABLE_INT_QUALYS=true

# Shodan integration
# RUNZERO_DISABLE_INT_SHODAN=true

# Azure AD import
# RUNZERO_DISABLE_INT_AZURE_AD=true

# LDAP import
# RUNZERO_DISABLE_INT_LDAP=true

# Intune import
# RUNZERO_DISABLE_INT_INTUNE=true

# Google Workspace import
# RUNZERO_DISABLE_INT_GOOGLE_WORKSPACE=true

# Wiz import
# RUNZERO_DISABLE_INT_WIZ=true

# Meraki integration
# RUNZERO_DISABLE_INT_MERAKI=true

# Microsoft MECM integration
# RUNZERO_DISABLE_INT_MECM=true

# Tanium integration
```

```
# RUNZERO_DISABLE_INT_TANIUM=true

# NetBox integration
# RUNZERO_DISABLE_INT_NETBOX=true

# Dragos integration
# RUNZERO_DISABLE_INT_DRAGOS=true

# Jira integration
# RUNZERO_DISABLE_INT_JIRA=true

# Palo Alto Networks integration
# RUNZERO_DISABLE_INT_PAN=true

# Manage directory users
# RUNZERO_DISABLE_DIRECTORY_USERS=true

# Manage directory groups
# RUNZERO_DISABLE_DIRECTORY_GROUPS=true

# Cloud connections (AWS, GCP, Azure)
# RUNZERO_DISABLE_INT_CLOUD=true

# Create custom integrations
# RUNZERO_DISABLE_INT_CUSTOM_INTEGRATIONS=true

# Custom integration scripts
# RUNZERO_DISABLE_INT_CUSTOM_INTEGRATION_SCRIPTS=true

# Advanced reports
# RUNZERO_DISABLE_REPORTS_ADVANCED=true

# Self-hosted platform
# RUNZERO_DISABLE_SELFHOSTED=true

# Bulk user management
# RUNZERO_DISABLE_BULK_USER_MANAGEMENT=true

# Group expiration
# RUNZERO_DISABLE_GROUP_EXPIRATION=true

# SSO group mapping
# RUNZERO_DISABLE_GROUP_MAPPING=true

# Hosted external scans (hosted zones)
# RUNZERO_DISABLE_HOSTED_ZONES=true

# Rebrand
# RUNZERO_DISABLE_REBRAND=true

# Knowledge base sidebar
# RUNZERO_DISABLE_KNOWLEDGE_BASE=true

# Customizable dashboards
```



```

# RUNZERO_DISABLE_CUSTOM_DASHBOARDS=true

# Multiple customizable dashboards
# RUNZERO_DISABLE_MULTIPLE_DASHBOARDS=true

# Query builder
# RUNZERO_DISABLE_QUERY_BUILDER=true

# Metadata minimization
# RUNZERO_DISABLE_METAMIN=true

# Offline mode
# RUNZERO_DISABLE_OFFLINE=true

# Certificate inventory
# RUNZERO_DISABLE_CERTIFICATE_INVENTORY=true

# Findings inventory
# RUNZERO_DISABLE_FINDINGS=true

# Risk management event rule types
# RUNZERO_DISABLE_RISK_EVENTS=true

# Baseline goals
# RUNZERO_DISABLE_BASELINE_GOALS=true

# Immediate analysis recalculation on rapid response publish
# RUNZERO_DISABLE_IMMEDIATE_RAPID_RESPONSE_RECALC=true

# Create new sites
# RUNZERO_DISABLE_CREATE_SITES=true

# Suppress vulnerabilities, vulnerability groups, and findings
# RUNZERO_DISABLE_SUPPRESSION=true

# Issues
# RUNZERO_DISABLE_ISSUES=true

# Archive vulnerabilities
# RUNZERO_DISABLE_ARCHIVE_VULNERABILITIES=true

```

Unofficial CPEs

When runZero can successfully fingerprint an asset's operating system, a CPE will be generated. In cases where the NIST database does not contain an official match, runZero will generate an unofficial CPE by default. This behavior can be disabled by setting `RUNZERO_GENERATE_UNOFFICIAL_CPE` to `false`:

```
RUNZERO_GENERATE_UNOFFICIAL_CPE=false
```

When unofficial CPEs are generated by runZero, they include `r0_unofficial` in the `other` field of the CPE by default. This value can be changed to any alphanumeric-constrained tag (limited

to 32 characters):

```
RUNZERO_UNOFFICIAL_CPE_TAG=custom_unofficial_tag
```

Rewriting cve.org URLs

In some circumstances, you may want to rewrite links to `cve.org` to an internal mirror or alternative site. This can be achieved in the product UI by using the environment variable `RUNZERO_REPLACE_URL_PREFIX_CVEORG`. For example:

```
RUNZERO_REPLACE_URL_PREFIX_CVEORG=https://cve.mirror.local/cve/
```

The example given would result in `cve.org` URLs in the product interface being rewritten to `https://cve.mirror.local/cve/$1` where `$1` would be replaced with the CVE identifier, such as `CVE-12345`.

Permissions

The self-hosted platform requires **root** access to install and manage from the command-line.

The platform service (`runzero-console`) runs as **root** and spawns a worker subprocess that runs as the **runzero** user account inside of a chroot environment (`/opt/runzero`). All substantive work happens within this isolated subprocess. Please note that older installations will use `rumble` instead of `runzero` in directory, file, and user names.

The following filesystem locations are used by the self-hosted platform:

`/etc/runzero`

Path	Owner	Permission	Notes
<code>/etc/runzero</code>	root	0700	Configuration files and certificates
<code>/etc/runzero/config</code>	root	0600	A plain-text configuration file
<code>/etc/runzero/certs</code>	root	0700	A directory containing the TLS certificate and key
<code>/etc/runzero/certs/cert.pm</code>	root	0600	The TLS certificate in PEM format
<code>/etc/runzero/certs/key.pm</code>	root	0600	The TLS certificate private key in PEM format

/opt/runzero

Path	Owner	Permission	Notes
/opt/runzero/tmp	runzero	0755	A temporary directory
/opt/runzero/storage	runzero	0700	Contains asset and scan artifacts
/opt/runzero/console	root	0755	Contains the platform executable
/opt/runzero/console/runzero-console.bin	root	0755	The platform executable
/opt/runzero/agent	root	0755	Contains the Explorer binaries
/opt/runzero/agent/runzero-agent-*	root	0755	The Explorer binaries
/opt/runzero/scanner	root	0755	Contains the CLI binaries
/opt/runzero/agent/runzero-scanner-*	root	0755	The CLI binaries
/opt/runzero/proc	root	0755	Contains copies of system /proc files
/opt/runzero/proc/cpuinfo	root	0644	A copy of /proc/cpuinfo
/opt/runzero/proc/meminfo	root	0644	A copy of /proc/meminfo
/opt/runzero/proc/version	root	0644	A copy of /proc/version
/opt/runzero/etc	root	0755	Contains copies of system files
/opt/runzero/etc/resolv.conf	root	0644	A copy of /etc/resolv.conf
/opt/runzero/etc/hosts	root	0644	A copy of /etc/hosts
/opt/runzero/etc/ca-certificates.crt	root	0644	A copy of the system root CA store
/opt/runzero/etc/runzero	runzero	0700	Contains instance identifiers
/opt/runzero/etc/runzero/cruncher.id	runzero	0700	A unique ID to identify the cruncher instance
/opt/runzero/etc/runzero/hub.id	runzero	0700	A unique ID to identify the hub instance
/opt/runzero/config	root	0700	Unused today

Backup and restoration

Your runZero installation and data can be backed up and restored to preserve your configuration.

runZero data backup

A backup of a self-hosted installation can be obtained by archiving the file system and database.

The file system archive includes the following paths:

- /etc/runzero
- /opt/runzero
- /lib/systemd/system/runzero-console.service
- /etc/systemd/system/multi-user.target.wants/runzero-console.service
- /usr/bin/runzeroctl

A sample file system backup command is:

```
# tar zcvf runzero-backup-fs.tar.gz /etc/runzero/ /opt/runzero/ \
/lib/systemd/system/runzero-console.service \
/etc/systemd/system/multi-user.target.wants/runzero-console.service \
/usr/bin/runzeroctl
```

The PostgreSQL database must be backed up separately. A sample command is shown below:

```
# sudo su - postgres
$ pg_dumpall -f runzero.sql && gzip runzero.sql
```

runZero data restoration

To restore the runZero install, follow these steps.

1. Stop any running runZero service:

```
# runzeroctl stop
```

2. Unpack the filesystem archive:

```
# tar -C / -zxvf /path/to/runzero-backup-fs.tar.gz
```

3. Restore the PostgreSQL database:

```
# sudo su - postgres
$ dropdb runzero; gzip -dc runzero.sql.gz | psql
```

4. Restart the runZero service:

```
sudo systemctl restart runzero-console
```

Support and debugging

The `runzeroctl` command includes a debugging tool which can collect diagnostics from your server and assemble them into a zip file which you can send to support. If you are asked to do this, the command is:

```
runzeroctl diagnostics run-script
```

Data is written to `/opt/runzero/collector`.

Alternatively you can save a copy of the script to the current directory so that you can examine it before running it:

```
runzeroctl diagnostics write-script
```

Manual migrations

Starting with version 4.0.240221.0, the runZero self-hosted upgrade process will run migrations before restarting the service. If you are running an older version of the software and would like to prevent downtime during the upgrade of a single-node self-hosted installation, the following steps can be used:

1. Obtain the self-hosted download link from the [runZero SaaS](#).
2. Download this file manually to your self-hosted systems:

```
$ curl -o platform.bin https://console.runzero.com/...../runzero-platform.bin
```

3. Mark this file as executable and run it with the `task db:migrate` parameter:

```
$ chmod u+x platform.bin; ./platform.bin task db:migrate
```

4. Once the migrations are done, install the update as usual:

```
$ runzeroctl update
```

Offline mode configuration

Supported operating systems

The runZero self-hosted console runs on the same operating systems whether you are installing online or offline. See [Self-hosting runZero — Supported operating systems](#) for the full list (Ubuntu 18.04+, RHEL 7+, CentOS 7+, Oracle Linux 7+, Debian 9+, on x86_64).

Note that for offline installs, PostgreSQL packages must be available locally. The PostgreSQL.org RPM bundles linked below cover RHEL/CentOS 8, 9, and 10. On older or other distributions, supply PostgreSQL through your own offline mirror or use `--distro-packages-only` to install whatever PostgreSQL version your OS provides (must be PostgreSQL 14 or newer).

Offline installation

The self-hosted runZero platform comes with a few options for your installation. You can utilize these options by adding flags to the install command. The current flags available are `--offline`, `--postgres-version`, `--distro-packages-only`, and `--postgres-rpm-directory`.

`--offline`

Configures the installation for operation without internet access. Network access may still be necessary during the installation to acquire dependencies. The installer uses the upstream postgresql.org packages by default and this can be disabled by specifying the options below.

`--postgres-version`

Specify a particular major version of PostgreSQL (14-18). If your operating system does not include the default version of PostgreSQL (18), you may need to use the PostgreSQL.org packages instead, which may include downloading them ahead of time for an offline installation.

`--distro-packages-only`

Install the platform without using third-party repositories. The install will try to acquire the PostgreSQL package from the configured operating system repositories (local or remote). If no repository is reachable, these packages can also be specified as a directory on the file system using the option below. Note that OS packages may be limited to older versions of PostgreSQL.

`--postgres-rpm-directory [directory]`

Install using supplied PostgreSQL RPM files (requires RHEL or CentOS). The installer will use the RPMs in the specified directory to satisfy the PostgreSQL dependencies.

By default, the console will be installed to `/opt/runzero`. If you need to change this, you can use the `--install-directory [directory]` option.

PostgreSQL RPMs required for `--postgres-rpm-directory`

There are four RPMs required for installing PostgreSQL:

- postgresql18
- postgresql18-server
- postgresql18-contrib
- postgresql18-libs

x86_64 Intel

- [RHEL/CentOS 10 RPMs](#).
- [RHEL/CentOS 9 RPMs](#).
- [RHEL/CentOS 8 RPMs](#).

64-bit ARM

- [RHEL/CentOS 10 RPMs](#).

- [RHEL/CentOS 9 RPMs.](#)
- [RHEL/CentOS 8 RPMs.](#)

Example install commands for offline mode

The first step is to download the [runZero platform](#)

RHEL/CentOS 8, 9, or 10

1. Download the required RPMs above and store them in a directory.
2. Run this install command as root:

```
./runzero-platform-[VERSION]-linux-amd64.bin install --offline
--postgres-rpm-directory [RPM_DIRECTORY]
--postgres-version=18
```

Ubuntu 18.04+, Debian 9+, RHEL 7+, CentOS 7+, or Oracle Linux 7+

1. Run this install command as root.

```
./runzero-platform-[VERSION]-linux-amd64.bin install --offline --distro-packages-only
```

Remember to use `su -` on Debian, not just `su`. On distributions whose default PostgreSQL is older than 14, supply a newer PostgreSQL via `--postgres-rpm-directory` (RHEL/CentOS 8/9/10) or your own offline mirror.



Some components of the application still reference the name "Rumble" for backwards compatibility. The documentation will be updated as these are changed.

Enabling offline mode for existing installs

- Open `/etc/runzero/config` with an editor of your choice.
- Look for `OFFLINE=` and change it to `OFFLINE=true`.
- Restart the runZero service `runzeroctl restart`.

CLI update with offline mode

The self-hosted runZero platform must be updated prior to first use.

Update the runZero platform and scanners with an offline update

- Go to <https://console.runzero.com/deploy/download/platform>.
- Copy the command directly from the download page and run it in your terminal, or you can use one of the following commands to update using the zip archive you downloaded.

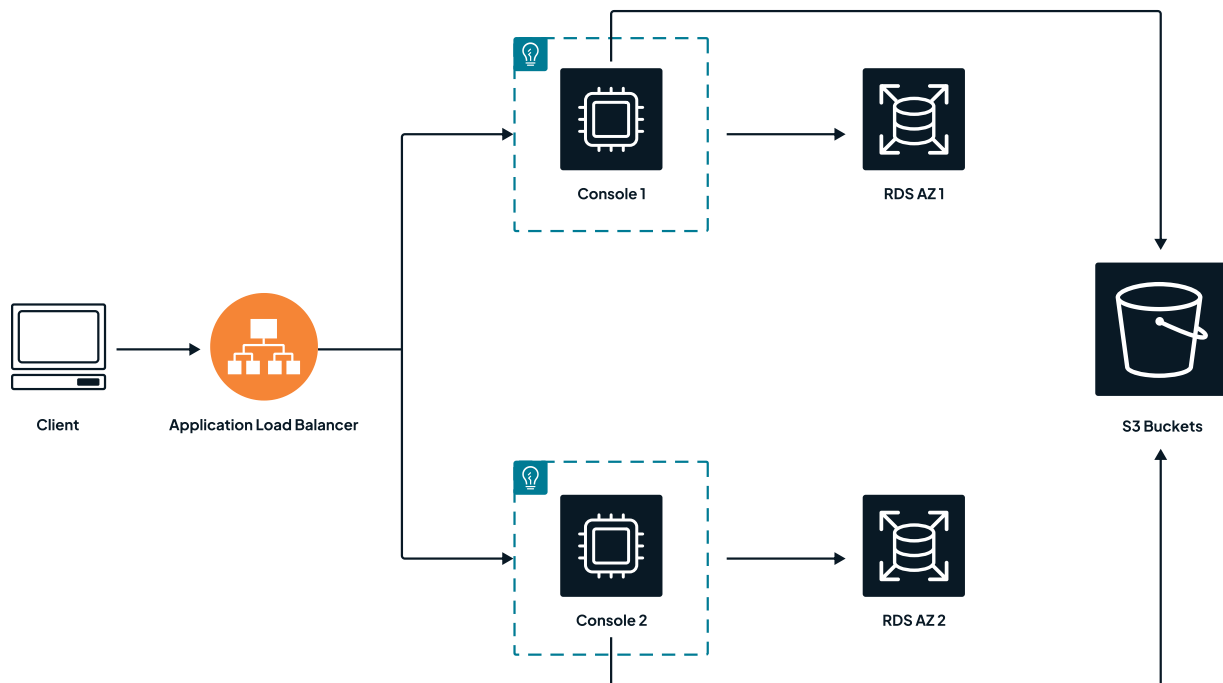
```
runzeroctl update runzero-platform-update-[VERSION].zip
```

```
runzeroctl update --offline --zip-file-path runzero-platform-update-[VERSION].zip
```

You will need to change the version to match the zip archive you downloaded.

High-availability configuration

Self-hosted installations of runZero can be configured for high-availability. For this configuration, a load balancer is used to direct traffic to multiple console servers, which use a shared PostgreSQL cluster and storage backend. The following diagram illustrates an example architecture of a high-availability installation using AWS with two availability zones.



In this diagram, an application load balancer (ALB) is terminating TLS and pointing to a target group that consists of two runZero servers, each in separate availability zones. The runZero servers use a multi-availability-zone PostgreSQL RDS instance, also configured for the same two availability zones, and both servers point to the same set of S3 storage buckets.

Installation steps

These are the general steps for installing and configuring a high-availability instance of runZero:

1. [Deploy an application load balancer.](#)
2. [Install and configure the first server node.](#)
3. [Install and configure subsequent server nodes.](#)

Deploy a load balancer with TLS termination

In order to provide a high-availability interface to the runZero cluster, an application load balancer should be used that receives TLS connections and forwards the HTTP requests to the available servers. This load balancer should be highly-available on its own. Any load balancer that can proxy large HTTP requests and websockets should be supported. The load balancer should add an X-Forwarded-For header to the HTTP requests sent onward to the runZero servers. The load balancer should also support detection of unhealthy nodes in order to handle automatic failover. The /health endpoint on the runZero servers can be used to implement health checks on the load balancer.

Prepare the first runZero server node

The general steps for installing and configuring the first runZero node are:

1. Install the runZero platform.
2. Configure the console URL and TLS/XFF settings.
3. Configure the database settings.
4. Configure the shared storage settings.
5. Verify your configuration.
6. Create the initial user account.
7. Add the runZero server to the load balancer and verify the connection.

Install the runZero platform



Some components of the application still reference the name "Rumble" for backwards compatibility. The documentation will be updated as these are changed.

1. Go to [platform download page](#) in the runZero Console.
2. Copy the URL to the platform installer from the download page. The download path for the installer is uniquely keyed to your license.
3. Download the install using Wget or cURL:

- `wget -O runzero-platform.bin https://console.runzero.com/download/platform-combined/<UNIQUE KEY>/62e41615/runzero-platform-vX.X.X-linux-amd64.bin`
- `curl -o runzero-platform.bin https://console.runzero.com/download/platform-combined/<UNIQUE KEY>/62e41615/runzero-platform-vX.X.X-linux-amd64.bin`

4. Run the following command to make the installer file executable:

```
# chmod u+x runzero-platform.bin
```

5. Run the install command with the manual database option:

```
# ./runzero-platform.bin install --manual-database
```

Configure the console URL and TLS/XFF

To configure your self-hosted installation for high-availability, the settings below should be configured in `/etc/runzero/config`, starting with the console URL and the TLS/XFF settings.

- Set `RUNZERO_CONSOLE` variable to the hostname or IP address of the load balancer:

```
RUNZERO_CONSOLE=https://{IP ADDRESS OR HOSTNAME}:443
```

- Disable TLS and configure the server to trust X-Forwarded-For headers with the following two lines:

```
TLS=false
RUNZERO_TRUST_XFF=true
```

Configure the database

The next step is to configure the database to be used by runZero.

- To do this, edit the `DATABASE_URL` value to match this format:

```
DATABASE_URL=postgres://{DB_USER}:{DB_PASSWORD}@{DB_HOST}:{DB_PORT}/{DB_NAME}?{DB_OPTIONS}
```

An example `DATABASE_URL` for AWS RDS will look like:

```
DATABASE_URL=postgres://rumble:[password]@rumbledb.rds-id.us-east-1.rds.amazonaws.com:5432/rumble?sslmode=require
```

To use a replica for read-only queries, specify `DATABASE_REPLICA_URL` with the same syntax. For RDS Aurora, specify the read-only endpoint as the replica URL. If no replicas are in use, only the `DATABASE_URL` is required, and this should point to the writer endpoint for RDS Aurora instances.

If a different type of PostgreSQL clustering is in use, make sure that the `DATABASE_URL` points to the highly-available endpoint for the active writer.

Configure shared storage

Next, configure the shared storage backend. This is used for storing asset data, raw scan data, and reports. runZero supports both mounted NFS and object storage.

- To use NFS, mount a read-write NFS share to `/opt/runzero/storage` and specify local storage mode with the following configuration options. The bucket names will be created under the NFS mount base directory.

```
RUNZERO_STORAGE_MODE=local
RUNZERO_STORAGE_PATH=/opt/runzero/storage
ASSET_BUCKET=runzero-assets
SCAN_BUCKET=runzero-scans
```

-

- To use AWS S3 as the shared storage, either configure an IAM instance role with read-write access to two S3 buckets, or specify the AWS credentials in the configuration file.
 - To specify AWS credentials specifically for runZero, use the following syntax:

```
AWS_ACCESS_KEY_ID=AKIA...
AWS_SECRET_ACCESS_KEY=[SECRET]
```

- Specify that the S3 storage mechanism should be used and set the name of the two buckets:

```
AWS_REGION=<your-region>
RUNZERO_STORAGE_MODE=s3
ASSET_BUCKET=<company>-runzero-assets
SCAN_BUCKET=<company>-runzero-scans
```

If a non-AWS backend is used that is compatible with the S3 API, use the same AWS and bucket variables above but override `AWS_REGION` and set the `AWS_ENDPOINT_URL_S3` parameter to the endpoint as appropriate. Reach out to runZero support if you run into any issues with the endpoint configuration.

In addition the variables above, most [AWS CLI environment variables](#) are also available for specific tuning.

Verify the configuration

1. After saving your configuration file, use this command to verify the self-hosted runZero platform can connect to your database:

```
sudo runzeroctl database verify
```

2. Once your settings are configured and verified you can restart the self-hosted runZero platform service:

```
sudo systemctl restart runzero-console
```

Create the initial user account

- Use the `runzeroctl initial <your-email>` command to create the initial user account.

Add the runZero server to the load balancer and verify the connection

- Point the load balancer to the new runZero server's IP on port 80 (unless modified).
- Browse to the load balancer URL over HTTPS using the configured DNS name.
- Verify that the user account configured previously can authenticate correctly.

Installing and configuring subsequent nodes

- Download and install the runZero platform binary on each node as above with the `--manual-database` option specified.
- Copy the configuration file from the first node to `/etc/runzero/config` on each subsequent node.

- After saving your configuration file, use this command to verify that the self-hosted runZero platform can connect to your database:

```
sudo runzerctl database verify
```

- Restart the platform service by running:

```
sudo systemctl restart runzero-console
```

runZero nodes should be spread across multiple availability zones and added to the load balancer's target group. Verify that requests are reaching each individual node by reviewing the syslog entries on that node.

AWS advanced configuration

AWS Secrets Manager can be used to retrieve configuration settings at startup. The Secrets Manager entries should match the key names of the configuration file. The secret name can be defined with:

```
AWS_SECRETS_MANAGER_KEY=rumble/production
```

To ensure that runZero can access the Secrets Manager, ensure that an `AWS_REGION` is set in the configuration file, and that the instance has an IAM instance role with permission to read the Secrets Manager key or the AWS credentials have been specified directly in the configuration file. With a Secrets Manager configuration, only the AWS options need to be present in the configuration file and the rest of the settings will be loaded from the Secrets Manager key. This will require copying and defining various settings from the generated configuration file to the Secrets Manager key-value pairs.

Note that many settings need to be identical across the runZero cluster for web requests and authentication to work correctly (including session secrets, DB encryption key, and the console URL).

Self-hosted troubleshooting

The runZero console includes a diagnostics collection script inspired by the need to troubleshoot a self-hosted environment. Collecting the necessary performance statistics, log files, system configuration, and profile debug capture was difficult for customers since there are many different commands and files involved. After checking permissions and dependencies, the diagnostic script gathers and compresses troubleshooting data into a convenient archive file that can be provided to runZero support. The script provides a consistent method for data collection whether it be from a system running your self-hosted console or an Explorer.

Requirements

The diagnostic script must be run from a Linux system. Additionally, the following dependencies must be installed for the script to run successfully:

Dependency	To install (Ubuntu)	To install (RHEL, CentOS, Fedora, Oracle Linux)
zip	apt install zip	dnf install zip
unzip	apt install unzip	dnf install unzip
lsscsi	apt install lsscsi	dnf install lsscsi
iostat	apt install sysstat	dnf install sysstat
curl	apt install curl	dnf install curl

The script must be run as `root` or with `sudo` in order to gather information from the system files and the `/opt/runzero` and `/etc/runzero` directories.

Running the script

Using runzeroctl

To run the script via the `runzeroctl` command, use:

```
runzeroctl diagnostics run-script
```

Output will be placed in `/opt/runzero/collector`.

Manually

To run the script manually, use the command below to write a copy of the script to the current directory:

```
runzeroctl diagnostics write-script
```

The script will be written as `runZero-collector.sh`. To run it:

```
sudo ./runZero-collector.sh
```

The script also has a `--help` option.

This script will perform the following tasks:

- Check for the required permissions.
- Check for the dependency commands (`lsscsi` / `zip` / `unzip` / `iostat` / `curl`).
- Check that the operating system is Linux.
- Check that either a runZero Explorer or runZero console is installed.
- Collect system configuration information.
- Collect system performance statistics.
- Collect database configuration information.
- Collect some basic statistics from the database.
- Collect runZero debug information from the console or log files from the Explorer.
- Create a zip file of the data collected and the log from the script.

Logging

The script will log all actions in `/tmp` and will provide the file name. Upon completion the log file is moved to the `/opt/runzero/collector` directory. This directory does not exist with the installation of a runZero Explorer or runZero console. The script will create the directory if it doesn't exist and place all files into that directory.

Data retention

runZero allows the data retention periods to be configured at the organization level. The organization settings page provides three ways to control how runZero manages your asset and scan data. Data expiration is processed as a nightly batch job based on the current settings for each organization in your account.

By default, data is retained for up to 1 year in the runZero Platform. This can be increased to up to 3 years for active subscriptions. The runZero Community Edition is limited to 30 days of retention.

Stale asset expiration

Stale asset expiration allows you to make sure that out-of-date information gets purged from runZero. Assets which have not been seen in the specified number of days are automatically deleted. The automatic deletion task runs once per day.

The stale asset expiration limit applies to both online and offline assets. For example, if an asset was online when last scanned, but hasn't been scanned (or imported via a connector) in the specified number of days, then the information is stale and the asset will be deleted. It's like running an asset search query of `last_seen:>Xdays` (where X is replaced by a number), and deleting the resulting assets.

Stale asset expiration can be set at both the Organization and Site levels. The lower value will be used. You can disable stale asset expiration by setting the number of days to zero.

Offline asset expiration

Offline asset expiration allows you to clear out information about assets that no longer exist on the network. Assets which are offline and have not been seen in the specified number of days are automatically deleted daily. Online assets are not touched. The automatic deletion task runs once per day.

Offline asset expiration applies to offline assets discovered by runZero scans. Assets that only have connector/import data are handled by stale asset expiration.

Offline status is determined by the asset state (`alive:false`). The `last_seen` value is then used to determine age against the configured expiration threshold; it is not used by itself to classify an asset as offline. The cleanup also requires scan-backed assets (`scanned:true`) with a runZero scan source.

This feature is particularly useful for guest wireless networks, or networks with short DHCP leases. It's closest to running an asset search query of `last_seen:>Xdays AND alive:false AND scanned:true` (where X is replaced by a number), and deleting the resulting assets.

Stale asset expiration is processed separately and can remove the same assets sooner when its threshold is lower.

You can disable offline asset expiration by setting the number of days to zero.

Stale integration attribute expiration

Stale integration attribute expiration allows you to clear out data from integrations that are no longer reporting an asset. Attributes for assets that have not been reported by the integration in more than a specified number of days are automatically deleted. Assets with no remaining data sources are also removed. The automatic deletion task runs once per day.

Optionally, you can keep the latest attribute set per integration, even if it is older than the specified threshold.

You can disable stale integration attribute expiration by setting the number of days to zero.

Stale vulnerability expiration

Stale vulnerability expiration allows you to remove out-of-date vulnerability data. Vulnerabilities that have not been updated in more than a specified number of days are automatically deleted. The automatic deletion task runs once per day.

You can disable stale vulnerability expiration by setting the number of days to zero.

Scan data expiration

This setting controls how long runZero will retain the raw data and metadata for Scan and Import tasks. If your organization has a maximum data retention policy for cloud services, this setting can be used to force the removal of any data older than a given number of days. The default setting of 365 preserves one year of historical scan data.

Event records

Account-level event records (found under Alerts → Events) are always retained for one year. If you need to keep copies of these for longer than one year, these records can be exported and archived through the API.

AI threads and artifacts

Data from the [AI section of the console](#) has fixed retention windows that are not configurable:

- **AI threads** (conversations, private to the user who started them) are removed 180 days after their last use. Every new turn resets the clock, so an active thread never expires; renaming one does not.
- **Unshared AI artifacts** are removed with the thread that produced them, attached files included. **Shared artifacts** are kept indefinitely while shared; unsharing one restores the thread-derived expiry.

- **AI run records** (the job and its streamed event log) are temporary: about 15 minutes while a run is active and 24 hours after it ends. The durable results are the thread and its artifacts.

Cleanup runs daily. Users can delete their own threads and artifacts at any time, and deleting a thread removes its unshared artifacts immediately.

Custom integration script versions

Custom integrations keep the [most recent 25 versions](#) of their script. Older versions are pruned as new ones are saved, and deleting a custom integration deletes its version history.

Data deletion after account termination

Upon termination of your runZero Platform subscription, your data will be deleted according to the table below.

Data Type	Default	Maximum
Stale assets (devices that have not received any new data from active, passive, or integration tasks)	180 days, unless requested earlier by emailing support[at]runzero.com	3 years (solely as set by you in runZero Platform prior to account termination)
Task data (raw data associated with each discovery task)	1 year, unless requested earlier by emailing support[at]runzero.com	3 years (solely as set by you in runZero Platform prior to account termination)
All other data (including data contained in support tickets, administrative data, etc.)	Upon written request by you to runZero at support[at]runzero.com	n/a
Any of the above data types archived as a copy on backup systems	1 year from date original data point is deleted	n/a

If you downgrade from your runZero Platform subscription to the free Community Edition, your data will be deleted according to the table below.

Data Type	Default	Maximum
Stale assets (devices that have not received any new data from active, passive, or integration tasks)	30 days, unless requested earlier by emailing support[at]runzero.com	3 years (solely as set by you in runZero Platform prior to account termination)

Task data (raw data associated with each discovery task)	10-30 days, unless requested earlier by emailing support[at]runzero.com	3 years (solely as set by you in runZero Platform prior to account termination)
All other data (including data contained in support tickets, administrative data, etc.)	Upon written request by you to runZero at support[at]runzero.com	n/a
Any of the above data types archived as a copy on backup systems	1 year	n/a

Managing access

runZero supports multiple concurrent users with a variety of roles. Roles can be set per-user on both a default and per-organization basis. The built-in roles are administrator, user, billing, annotator, viewer, and no access. There is also a superuser role available to manage global settings.

You can also define [custom roles](#) that grant exactly the permissions a team needs, and assign them anywhere a built-in role can be assigned.

The console URLs in this article use `console.runzero.com` as an example. Use the URL for your runZero deployment instead.

Where there are multiple roles defined for a user, the access granted is based on most privilege. For example, if a user has user access by being in a group, but admin access assigned directly, they will be given admin privileges.

Available roles

Superuser

The first user created within the runZero console is considered a superuser. This role allows management of global settings like subscriptions and SSO parameters, and is shown as an access level of “everything”.

If you are a superuser, you can promote someone else to be a superuser. To do this, check the row listing them, and click the *Promote to superuser* button.

If you are using [SSO authentication](#), you should configure at least one superuser with a strong password and MFA that can be used as a backup if SSO settings need to be changed in the future.

We strongly recommend having more than one superuser, particularly if you are using MFA. That way if an MFA token is lost or a superuser leaves your organization, another superuser can fix the problem.

Administrator

Administrators can modify any aspect of an organization and have the unique ability to permanently delete bulk data, create additional organizations, and reset settings for other users.

User

Users have full access to an organization and can update sites, modify assets, schedule scans, and generally use most functionality. Users are not permitted to reset other users' security credentials, bulk delete data, or delete an organization.

Billing

Billing users are unable to see any asset data, but can manage the licensing, billing, and entity settings for the account.

Annotator

Annotators have the same permissions as a viewer, except they have the ability to add tags to assets. Annotators do not have any other write-access within an organization, so they are unable to modify or remove existing tags. Modifications to existing tags must be made by a runZero user or administrator.

Viewer

Viewers have read-only access to an organization. This includes inventory data, findings, queries and reports. Viewers are not allowed to interact with tasks, modify settings, or update assets. Viewers cannot install explorers or download the CLI scanner nor do they have access to client API tokens or organization level API tokens. Certain reports cannot be generated by Viewers, including the Organization overview report, the External assets report and the Organization and site comparison report. However, previously generated versions of these reports will be visible.

No Access

The *no access* role is generally used as a default. Accounts with no access as a default are limited to those organizations where they have been granted access. If no organizations are allowed, the user is limited to managing their own account settings.

The no access global role can be used to create a single-organization user, such as a customer or third-party that needs access to the inventory for a specific organization. For consulting use cases, a single-organization user is a way to provide clients with visibility into their environment at no additional cost.

Another use for the *no access* role is to set it as the default for the account when you have no limits on who can sign in using an SSO system. You can then wait for the user to sign in and request access, before granting their newly-created account access to the appropriate organizations.

Custom roles

When none of the built-in roles is the right fit, define a [custom role](#) with the exact permissions you want. Custom roles are created once for the account on the [Roles](#) page and can be used as a user's default role, as a per-organization role, as a group role, as the SSO enrollment default, or locked to an [API client](#). See [Custom roles and permissions](#) for the permission model and how multiple roles combine.

Inviting users

To add a team member, access the [Your team](#) page, and use the *Invite user* button to send an invitation.

The **Your team** menu entry has several submenus.

- The first, *Users*, shows all users in the current client account.
- The second entry, *Restricted*, goes to a page listing users who by default have no access to any organization.
- The next entry is the name of the current organization, as selected from the organization selector at the top of the screen. The page shows only users with access to that organization.
- The *External* entry goes to a page where you can invite users from other runZero client accounts.
- Finally, *Groups*, lists the [user groups](#) available. Groups can be used to set access and permissions users have within each organization.

User details

On the Users page, you can click on a user to view their details page. The user details include a list of their effective access to each organization. These details are split into three sections:

- *User access* lists access to organizations that has been directly granted to the user.
- *Group access* lists access to organizations that they are granted because they are in a group that has access to the organization.
- *SSO group access* similarly lists access from being in a group, but in this case for groups set as a result of [SSO group roles](#).

Directly assigned user permissions can be edited using the gear icon button, either at the right side of the appropriate row of the user listing, or using the button at the top right of their user details page.

While editing user permissions using the gear icon, you are editing the explicit assigned roles. To see the resulting access levels, check the user details page.

Account settings

The *Account* page is available to superusers. It contains settings which apply to all users and organizations within the account.

Single-sign on (SSO)

runZero supports the [implementation of SSO](#) through SAML2. If you use a SAML2-compatible single sign-on (SSO) implementation, the [SSO Settings](#) page can be used to configure an Identity Provider (IdP) and allow permitted users to sign in to the runZero console.

Multi-factor authentication (MFA)

runZero supports multi-factor authentication, also known as two-factor authentication or 2FA. Physical hardware keys such as [Google TitanKey](#) and [Yubico YubiKey](#) are supported via the [WebAuthn](#) standard.



Note that the WebAuthn standard does not support use of an IP address as the relying party identifier. As a result, you cannot configure multi-factor authentication on a self-hosted console unless you are accessing the console using a domain name.

You can configure MFA policies for your account via the [Account settings page](#). If multi-factor authentication is required, users who do not have an MFA token set up will be required to set one up when they next sign in. You can choose between requiring this for all users, or only requiring it for non-SSO users. The latter option is useful if your SSO server enforces MFA use.

Once a user registers one or more MFA tokens, they will be required to use one of the tokens every time they sign in.

Note that **changing the account settings to not require MFA will not alter the MFA status of existing accounts**. Existing accounts will keep any existing MFA tokens they have registered, and will still be required to use one to sign in. To disable MFA for a user, the user must clear the MFA token registration. To do this, they can go to their [user settings page](#) and click the red "Unlink" text next to the token ID in the bottom right.

Disabling support access

If you check the box labeled *Disable support access to your account*, runZero support staff will not be allowed to switch to your account.

If you choose to disable support access, this may make it harder for runZero support to answer any questions you have. In some cases we may need you to turn support access back on so that we can help you.

Idle times and sign in duration

You can set a maximum idle session time in minutes. If set, users whose web browsers don't access runZero for the specified time period will be considered idle, and signed out.

You can also specify a maximum sign in duration. If set, users will be forced to sign in again regularly, at least once every specified period.

Account API keys

The Account API is a REST API which allows account-level operations such as adding and removing organizations and sites, adding users, and accessing the system event log. The *Generate API Key* button on the Account page can be used to generate a token which will allow access to the Account API.

Roles

The *Roles* page lists the built-in roles and any [custom roles](#) defined for the account, along with the permissions each one grants and how many users, groups, and API clients are assigned the role. Superusers and administrators of every organization can create and change custom roles there.

API clients

The *API clients* page manages [account-bound API clients](#), credentials with an explicit set of permissions and organizations that are not tied to any user account. Users manage their own [user-bound API clients](#) from [My API clients](#) in their own settings instead.

License information

The *License* page shows information about your runZero software license, including how many assets you are licensed for, how many assets you have across all organizations, and when your license renews.

Entity information

The *Entity* page allows you to update information about the legal entity runZero is licensed to. You should ensure that this information is kept up-to-date if your company changes name or location, as we use the information to calculate taxes and ensure compliance with appropriate regulations.

Audit log

The *Audit log* page shows a history of all system events relevant to the superuser, such as login events, that are not visible within the organization Events page. See [runZero events](#) for the full catalog of event actions.

Custom roles and permissions

Every role in runZero is a set of permissions. The [built-in roles](#) are fixed sets that cover the common cases. A custom role is a set you define yourself, so you can grant exactly the access a team needs instead of rounding up to Administrator or down to Viewer.

Custom roles are defined once for the whole client account and can then be assigned anywhere a built-in role can: as a user's default role, as a per-organization role for a user or a [group](#), as the [SSO enrollment default](#), or locked to an [API client](#).

The console URLs in this article use `console.runzero.com` as an example. Use the URL for your runZero deployment instead.

How permissions work

A permission is a verb applied to an area. An area is a family of records or settings, such as inventory or alerts. A verb says what may be done to it.

Verbs

Verbs are ranked, and a higher verb includes every lower one. Granting `write` on an area also grants `annotate` and `read` on that area.

Verb	What it allows
Read	Read, search, list, view detail, and export existing records. Nothing changes.
Annotate	Change user-supplied metadata on a record you can already read: tags, comments, ownership, criticality, and issue status. It never changes scan-derived facts or configuration.
Write	Create, change, and delete records, and run operations that change data.
Manage	Change the settings that govern an area for everyone else, and create or change credentials, permissions, and organization structure.

Only inventory and findings have anything that can be annotated. Every other area goes straight from read to write.

Areas

Area	Covers	Verbs available
inventory	Assets, services, software, certificates, and sites.	read, annotate, write, manage
findings	Vulnerabilities and issues.	read, annotate, write, manage
tasks	Scan tasks, scan templates, and raw task artifacts.	read, write, manage

events	The activity record. Append-only, so read is the only verb.	read
alerts	Alert rules, notification channels, and alert templates.	read, write, manage
explorers	Explorers, explorer groups, and hosted zones.	read, write, manage
sources	Integration sources and custom integrations.	read, write, manage
queries	The saved query library, including the Rapid Response feed.	read, write, manage
directory	Directory user and group records collected by integrations and scans. Harvested data, so read is the only verb.	read
users	Console users, user groups, and invitations.	read, write, manage
organizations	Organizations, their settings, and their hierarchy.	read, write, manage
credentials	The credentials runZero uses to authenticate scans and integrations.	read, write, manage
tokens	API keys, export tokens, and API clients.	read, manage
ai	AI Hub, AI threads, and query translation.	read, write, manage
reports	Reports, external reporting, and goals.	read, write, manage
account	Account settings and license information.	read, manage
content	Reference content runZero ships, such as vulnerability match rules and scan checks. Read-only for everyone.	read

Four distinctions are worth reading twice, because the names are similar and the access is not:

- **users is console identity. directory is collected data.** `users` covers the people who sign in to your console. `directory` covers the user and group records that scans and integrations bring back from LDAP, Entra ID, and similar sources. Granting one does not grant the other.
- **credentials is scan credentials. tokens is API credentials.** `credentials` covers the secrets runZero uses to log in to your systems. `tokens` covers the keys and clients that let something log in to runZero. A credential that can create another credential is a way to escalate privilege, so keep `tokens` restricted.
- **tasks includes raw scan artifacts.** A raw artifact includes the inventory and directory records the scan collected, so `tasks:read` reaches that data without a separate `inventory` or `directory` grant.
- **Permission is not the same as visibility.** A permission says what you may do; the organizations attached to the assignment say where. Both apply on every request.

What the verbs mean in practice

The same permissions drive the console, the [API](#), and [MCP](#). If a role cannot write inventory, the buttons that write inventory are not offered in the console, the matching API routes refuse the call, and the MCP tools that would change inventory are not offered to the AI client. There is no surface where a permission is enforced in one place and not another.

Some worked examples, for the areas where the line between verbs matters most:

Permission	What it allows
inventory:read	Search and view assets, services, software, certificates, and sites, and export them.
inventory:annotate	The above, plus tagging assets, setting ownership and criticality, and adding comments.
inventory:write	The above, plus editing and deleting assets, and moving them between sites.
inventory:manage	The above, plus bulk deletion and the inventory settings that apply to everyone.
tasks:read	View scan and integration task history, and download raw task artifacts, which include the inventory and directory data those scans collected.
tasks:write	Start, stop, and delete scans.
tasks:manage	The above, plus creating and changing scan templates.
findings:annotate	Change issue status and suppression state without changing the underlying findings.
sources:write	Configure and run integration sources.
sources:manage	The above, plus custom integrations.
users:read	See who is on the team and what access they have.
users:manage	Invite, change, and remove console users and groups. This is the permission that makes someone an administrator of an organization.
tokens:read	See which API credentials exist.
tokens:manage	Create, change, and revoke API keys, export tokens, and API clients, and approve MCP connections .
credentials:manage	Create, change, and delete the credentials used to authenticate scans, and list them through the API and MCP. Secret values are never returned by any permission.

An export always requires the same permission as reading the data in the console. There is no separate export permission to grant or withhold.

What a custom role cannot grant

Some privileges are structural and are never part of a role's permissions:

- **Superuser.** Global settings, subscriptions, and SSO configuration.
- **Billing.** Licensing, billing, and entity settings, which stay with the built-in Billing role.
- **Account settings.** The `account` area is reserved for built-in roles, so a custom role cannot grant it.
- **Managing roles.** A custom role can never authorize changes to roles, including its own permissions.

What the built-in roles grant

The built-in roles are the same permission model expressed as fixed sets. This table is useful as a starting point when deciding which permissions a custom role should include. A dash means no access to that area.

Area	Administrator	User	Annotator	Viewer	Billing
inventory	manage	write	annotate	read	-
findings	manage	write	annotate	read	-
tasks	manage	manage	-	-	-
events	read	read	read	read	-
alerts	manage	manage	read	read	-
explorers	manage	write	read	read	-
sources	manage	write	read	read	-
queries	manage	read	read	read	-
directory	read	read	read	read	-
users	manage	read	read	read	-
organizations	manage	read	read	read	-
credentials	manage	-	-	-	-
tokens	manage	read	-	-	-
ai	manage	write	write	write	-
reports	manage	write	read	read	-
account	read	read	read	read	manage
content	read	read	read	read	-

Superuser is not in this table because it is a flag on a user rather than a role.

Creating a custom role

Custom roles are managed from the **Roles** page under **Account** in the left navigator, at console.runzero.com/account/roles. If the page is not in your navigator, either you do not administer every organization or custom roles are not enabled for your account. The page lists the built-in roles as read-only rows alongside your custom roles, with the permissions each one grants and, for the people who manage roles, how many users, groups, and API clients are assigned the role.

The Roles page is available to superusers and to all-organization administrators, who are also the only people who can create, change, or delete a role. Everyone else can still read the role catalog, which is what lets the role dropdowns and effective-access views elsewhere in the console explain what each role allows.

To create one:

1. Click **Create role**.
2. Give it a **Name**. Names are 2 to 64 characters of letters, digits, spaces, underscores, periods, and hyphens, and must start with a letter or digit. They are unique within the account without regard to case, and cannot reuse a built-in role name or label.
3. Add a **Description**. This is the help text shown to whoever assigns the role, so describe what the role is for rather than restating the permissions.
4. Set the **Permissions**. Choose a verb for each area you want the role to grant, and leave the rest at no access. The `account` area is not offered.
5. Save. A role has to grant at least one permission.

You can also use the copy action on an existing row to start from another role's permissions, which is the quickest way to build a role that is "Viewer, plus one thing".

Editing, renaming, and deleting

Editing a role changes access immediately for everyone assigned that role. There is no separate publish step.

Renaming a role rewrites every assignment that refers to it, including SSO enrollment defaults and API client locks, so nothing is orphaned. Renaming onto a name that still has assignments is refused rather than merging the two.

Deleting a role leaves its name on the assignments that used it, but the role stops contributing any permission. Those assignments display as disabled, and effective access views show "No access" with a note that the role has been disabled. Other roles assigned to a user continue to apply. The delete confirmation lists the users, groups, API clients, and organizations affected.

Two consequences are worth planning for:

- **An API client locked to a deleted role cannot authenticate** until a role with that name exists again.
- **Recreating a role with the same name re-enables every assignment that still refers to it.** The create form warns you when a name still has assignments and asks you to

confirm. This is deliberate, so that deleting a role by mistake can be undone, but the name itself retains authority. Treat recreating a deleted role name with the same care as granting the access directly.

Assigning custom roles

A custom role appears in every role dropdown that a built-in role appears in, with its description as help text. Nothing about how you assign roles changes.

To a user

On the [Your team](#) page, use the gear icon on a user's row, or the button on their user details page, to edit their permissions. You can set a custom role as the user's default role in every organization, or per organization.

The user details page continues to show effective access per organization, split into direct user access, group access, and SSO group access, so you can see what a role actually resolves to for that person.

To a group

[User groups](#) take custom roles the same way, either as the group's default role in every organization or per organization. Group membership can come from the console or from [SSO group mappings](#).

As the SSO enrollment default

The **Default role in any organization** field on the [SSO settings](#) page accepts custom roles as well as built-in ones. Users provisioned through SSO enroll with that role.

If the role has been deleted by the time a new user enrolls, they enroll with no access rather than inheriting something unintended, and the event is recorded. Recreating the role does not retroactively change users who already enrolled, though it does re-enable their stored assignment.

To an API client

An [API client](#) can be locked to a role, so that the credential's access follows later changes to that role, or can use a role as a starting point for a one-off set of permissions. See [Role locks](#).

How multiple roles combine

A user can be assigned roles through several routes at once: their own default role, a per-organization role, a group's default role, and a group's per-organization role.

Permissions add together. In a given organization, the user gets the best built-in role that applies plus every custom role that applies, unioned. Where two roles grant different verbs on the same area, the higher one wins. A custom role can only add access, and never reduces what a built-in role already grants. If someone has Viewer by default and a custom role granting `tasks:write` in one organization, they can start scans in that organization and remain a viewer everywhere else.

A custom role can bring an organization into view. If the only thing granting a user access to an organization is a custom role, that organization appears in their organization picker and in their effective access.

Custom roles never confer structural privileges. No combination of custom roles makes someone a superuser, gives them billing access, or lets them manage roles. See [What a custom role cannot grant](#).

Each role slot contains one role. Assigning a new role to a user's default slot replaces the role that was there. To give someone several distinct sets of permissions, combine a default role with per-organization roles, or with group membership.

Auditing

Role changes are recorded in the [audit log](#) as `role-created`, `role-updated`, and `role-deleted` events. An update records which fields changed, including the previous name and permissions. A delete records how many assignments were disabled.

On the Roles page, the usage columns and the usage view list the users, groups, and API clients assigned each role, and whether it is the SSO enrollment default.

Managing user groups

User groups help streamline the management of users who need the same set of permissions. A user group explicitly sets the organizational [role](#) for users, which determines the tasks they can perform within each organization. Both built-in roles and [custom roles](#) can be assigned to a group. You can assign roles at a per-organization level or assign a single role across all organizations. Single sign-on settings can also be applied to groups through [SSO group mappings](#).



What happens if there are conflicting permissions?

runZero will always grant the role with the highest permissions level. For example, let's say an account has a viewer role for all organizations, but they've been added to a user group that has a user role for all organizations. This user will now have user-level permissions for all organizations. If the user group expires, the user's role reverts back to their account-level role.

User groups can also have an optional expiration date, which sets time-bound access to organizations within runZero for specific users. When the expiration date elapses, the user reverts back to their account-level permissions. If no expiration date is set, the user group settings will be persistent.

Creating user groups

runZero administrators and super users can create user groups.

1. Go to [Your team > Groups](#) and click Add Group.
2. Enter a name for the user group.
3. Choose the default role you'd like to assign to the group. This setting will establish the access level for every organization you have.
4. Set the per-organization roles, if you need to provide different access levels to specific organizations.
5. Set an expiration date for the user group, if you need to time-bound the permissions. When the expiration date elapses, the role for users part of the group will revert back to their user-level permissions. Otherwise, if you don't specify an expiration date, the user group will be persistent.
6. Go to the **Add users** tab and search for the users you want to add to the group. You can search by username or email.
7. When you are done adding users, save the user group. The user group will be listed on the [Groups page](#).

Adding users to user groups

To add users to multiple groups at the same time, you can use the *Edit group membership* button on the [Users page](#). The *Edit group membership* window will list every user group each user is currently in. Making changes from the *Edit group membership* window will apply to all users you have selected. Only runZero administrators and superusers can add users to user groups.

1. Go to the [Users page](#).
2. Select the users you want to add to a group.
3. Click the **Edit group membership** button.
4. Choose the user groups you want to add the users to.
5. Save your changes.

Setting an expiration date for a user group

runZero administrators and super users can set an expiration date for a user group.

1. Go to [Your team > Groups](#).
2. Find the user group you want to assign an expiration date. Click the name to open the config page.
3. Set an expiration date for the user group. After the expiration date, the user's role will revert back to their account-level permissions.
4. Save the user group. You'll be able to see the user group's expiration date from the [Groups page](#).

Viewing users in a user group

1. Go to [Your team > Groups](#).
2. Find the *Users* column in the *User groups* table, which shows the user count for the group.
3. Click the user count to query and display the users assigned to the group.

Viewing user groups assigned to a user

1. Go to [Your team > Users](#).
2. In the Groups column for each user is the number of groups that user is a member of.
3. Click on a number to show the corresponding list of groups.

Removing users from a user group

runZero administrators and super users can remove users from a user group.

1. Go to [Your team > Groups](#).
2. Find the user group you want to modify. Click the name to open the config page.
3. Go to the *Users* tab.

4. Remove the users you no longer want assigned to the group. Their permissions will revert back to their account-level ones.
5. Save your changes.

Deleting user groups

runZero administrators and super users can delete user groups.

1. Go to [Your team > User groups](#).
2. Select the user group you want to delete.
3. Click the **Delete** button.
4. Confirm you want to delete the user group.

Searching for users and user groups

When you are on [Users page](#) or [Groups page](#), you can use the following keywords to search in the table:

Keyword	Description	Example
id	User's ID.	id:123456789
name	User's name.	name:john
expires_at	Time or date the user group expires.	expires_at:>2weeks
created_at	Time or date the user group was created.	created_at:>2weeks
updated_at	Time or date the user group was last updated.	updated_at:>1year
has_expiration	Whether the group has an expiration date.	has_expiration:true
created_by_id	ID of user who created the user group.	created_by_id:123456789
created_by_email	Email of the user who created the user group.	created_by_email:user@example.com
group_id	The user group ID.	group_id:123456789
group_name	The user group's name.	group_name:group1

Group IDs can be found in the URL for the group config page

<https://console.runzero.com/groups/<groupid>/edit>.

The `group_id` keyword is only available for the users table; for the groups table, use `id`.

Bulk importing users

Instead of manually adding users one at a time, runZero administrators can add multiple users via bulk import. To bulk import users, you will need to create a CSV (comma separated values) file that contains the user information, such as their first name, last name, email, role, and organizational access.

Bulk imports will only add new users; it will not update existing users. If the file contains a user that already exists in the system, the import will not complete. You'll need to remove all duplicate users from your CSV file and import the file again.

Here are the high-level steps for bulk-importing users:

1. [Create a CSV file](#) with your user information.
2. [Import the CSV file](#) into runZero.
3. [Verify users have registered](#) their accounts.

Creating the CSV file for importing users

To create the CSV file, you can use the provided [template](#) or create a file from scratch. You'll need to follow the guidelines and include the fields outlined below.

- The first row of the CSV file is the header row and it must be included in the file. It contains the required fields needed for importing users, and those fields must follow a specific order (`first_name`, `last_name`, `email`, `all_orgs_role`, `org_roles`).
- runZero requires the `first_name`, `last_name`, and `email` fields. The organization-level fields, `all_orgs_role` and `org_roles`, are optional.
- To define the `org_roles` field, use the following format: `orgName=role`. You can use pipes to separate multiple definitions.
- Excluding the `all_orgs_role` field from the CSV will result in users being assigned a no-access role in all organizations.
- Excluding the `orgs_role` field from the CSV will result in users being assigned the role defined in the `all_orgs_role` field. If both fields are excluded from the file, all users will be assigned the no-access role. They will need to contact the runZero administrator to modify their account access.

User fields for the CSV file

The following table lists the fields you can include in your CSV file:

Field	Description	Example
<code>first_name</code>	The user's first name.	sarah
<code>last_name</code>	The user's last name.	smith
<code>email</code>	The user's email address.	ssmith@company.org

all_orgs_role	The role assigned to the user for all organizations. This is optional.	user
org_roles	The role assigned to the user for a specific site. Use pipes to separate the organizations. This field is optional.	org1=user

Your resulting CSV file should look like the following example:

```
first_name,last_name, email,all_orgs_role,orgs_role
Sarah,Smith,ssmith@company.org,user,org1=annotator|org2=annotator
```

Importing users into runZero

1. Go to your [Team Import](#) page.
2. Choose the CSV file that contains your user information and upload it to runZero.
3. Import the file.

runZero will alert you if there are issues with your file. For example, if the file contains an existing user, the import will not complete. You'll need to remove all duplicate users from your file and import it again.

Verifying users have registered

After you add new users via import, each user receives an email invitation to join your team. The email contains a link to register their email address, which prompts them to enter their name and password for their account. After they create an account, they'll be able to sign in to the runZero console.

You can visit the [Teams](#) page to find the status and last sign in for each user. Users who have completed the registration process will show an *Activated* status.

Managing external users

You can invite external users to join your runZero instance and view the organizational data available to them. The ability to add external users is useful for consultants, value-added resellers, and managed service providers who want to be able to share data from runZero with external partners and clients.

If you are a superuser, you can invite any user who has a runZero account to join your account. When you invite the user, they will receive an invitation to join your account via email. After they accept the invitation and sign in to runZero, they will see a new menu next to their organization switcher that lists all the clients they can access. Your client name will display in the list.



If the person you want to invite does not have a runZero client account, they will need to sign up for one before you can invite them as an external user. A free runZero Community Edition account can be invited as an external user.

External users are only viewable from the [External users page](#), separate from your other user lists. These users can use their same credentials to sign in and switch between clients and the organizations they've been granted access to.

Inviting an external user into your account

1. Go to **Your team > External** and click [Invite external users](#).
2. Assign the default role you want to assign the user for all organizations.
3. Enter the email address for the user. The user must have an existing runZero account.
4. Customize access for the user at the per-organization level. runZero grants the highest permission levels, when there is a conflict between the global and per-organization permissions.
5. Add a custom email subject or body, if needed. runZero uses a default email template that informs the user you have invited them to your account and provides a link to activate the invitation. The email sender will be your name and the subject line will be "Invitation to join [client] by [your name]."
6. Send the email.

After they activate their invitation, they will be able to sign in to runZero, switch clients, and view your organizational data.

Switching clients and organizations as an external user

External users who have access to another account will see a client switcher, or client dropdown menu, located next to their organization switcher at the top right of the runZero web interface. Changing the client will change the organizations you can access and view.

The account that granted you access has configured the organizations and permissions you have within the client account.

Removing an external user from your account

Removing an external user from your account will remove their access to your data. The user will no longer be able to choose your client. To remove an external user, go to the [External users page](#), select the user account you want to remove, and click **Remove user account**. Confirm you want to remove the user. They will be reverted to their primary client account and will no longer have the option to access your runZero data.

Implementing SSO

If you use a SAML2-compatible single sign-on (SSO) implementation, the [SSO Settings](#) page can be used to configure an SSO Identity Provider (IdP) and allow permitted users to sign in to the runZero console.

runZero's SSO implementation is designed to work with common SAML providers with minimal configuration, but there are a few requirements:

- Your users need to authenticate to a single domain such as `example.com`, not to multiple domains or a domain with many subdomains.
- The domain name needs to be configured in the SSO identity provider settings in runZero. This is true even for self-hosted runZero deployments.
- Your SAML IdP should provide something that looks like an email address in the standard NameID parameter. It doesn't need to be a valid email address, but it should be a unique value that has the same syntax as an email address (`user@example.com`). The field name NameID is case sensitive.
- If the NameID does not look like an email address, runZero will check the fields `email`, `user.email`, `emailaddress` and `email address` for a suitable ID. These field names are not case sensitive.
- runZero will check for the user's full name in the fields `name`, `gecos`, `user.name` and `displayname`. If no full name field is found, runZero will proceed to check for a first name in `first_name`, `firstname`, `given_name`, `user.firstname`, `givenname` OR `first name`; and for a last name in `last_name`, `lastname`, `family_name`, `user.lastname`, `surname`, `sn`, OR `last name`. These field names are not case sensitive.

Note that you must be a [superuser](#) to manage runZero SSO settings.



Once a user logs in using SSO, they can no longer login with a password, even if they were previously able to use a password. For this reason, we strongly recommend that you set up and keep a non-SSO superuser account, so that you can update the settings if SSO stops working for any reason.

Specific SSO providers

You can refer to the [Azure AD](#) or [Okta](#) pages for details on configuring these providers. The basic steps for configuration are:

1. Add runZero as an application.
2. Set up SSO in runZero.
3. Provision users to the runZero app.

For other SSO providers, the following information will help you to configure things.

Identity provider settings

To get to the identity provider settings, choose *Your team* from the left navigator, then click the SSO Settings button at the top right of the page.

The identity provider settings form is where you enter information about your SSO identity provider.

The easiest way to configure SSO is to use XML metadata, if your SSO service provides it. This will usually require that you set up runZero as an application on your SSO provider, then download an XML file. The XML can be then opened in a text editor, and copied and pasted into the box at the bottom of the runZero identity provider settings form. If the XML decodes successfully, the key fields on the form will then be set automatically.

Single Sign On mode

You can choose whether to allow regular runZero accounts, SSO-provisioned accounts, or both.

If both are allowed, you can set up backup administrator accounts in case your SSO provider is unavailable.

Domain name

The domain name is used by runZero to locate the correct SSO IdP settings when users choose to sign in via SSO. This is so that it can redirect them to the correct SSO provider to sign in. For example, if the domain is `example.com`, the runZero sign-in page will expect users to begin by entering an address of the form `username@example.com`.

The domain is also passed back by the SSO provider after sign in, and used to fetch the SSO IdP settings so that runZero can verify and process the data.

The domain is included as part of the ACS URL in the sign in request sent to the SSO provider. If you change the domain, you will need to reconfigure the SSO provider to accept the new ACS URL.

Default role in any organization

The default role given to users the first time they sign in through SSO. It accepts the [built-in roles](#), any [custom role](#) defined for the account, and Superuser.

Setting this to *No access* is a common choice when anyone in your identity provider can sign in. Users are then created with no organization access, and you grant access to the appropriate organizations after they first sign in.

If a custom role set here is deleted, users who enroll afterwards are created with no access rather than inheriting something unintended. Recreating the role restores the assignment for

those users.

Roles assigned through [SSO group mappings](#) apply on top of this default.

Issuer URL

The issuer URL, also often referred to as the entity ID, is a URL your SSO provider uses to uniquely identify itself in data passed back to runZero. You need to obtain this value from your SSO provider. Note that the value is not normalized to add or remove trailing slashes; it needs to match exactly.

Sign in URL

The sign-in URL is the URL users should be redirected to in order to begin the sign-in process for your SSO provider.

Certificate

The certificate is the PEM encoded CA certificate runZero will use to verify the signature on the data from your SSO provider.

If you paste multiple CA certificates into the box, your SSO provider will need to include a `KeyInfo` element in data passed back to runZero, to specify which certificate to check the signature against.

SSO walkthrough

To configure and debug SSO settings, it helps to understand how the SAML sign-in process works.

SAML SSO authentication is carried out using messages passed by the user's web browser. No direct connection occurs between runZero and the SSO identity provider (IdP) in either direction.

The normal sign-in process starts with the user going to the runZero console sign-in page, choosing **Sign in via SSO**, and entering their email address. The domain name of the email address is used to find the correct SSO identity provider settings in the runZero database.

runZero then redirects the user's web browser to the Sign in URL specified in those SSO IdP settings. Some additional parameters are added to the URL, according to the SAML specifications. These are verified by the SSO IdP.

The user logs in on the SSO IdP website. The IdP then redirects the user's web browser back to runZero, to the Assertion Consumer URL (ACS URL). The redirection includes a digitally signed blob of encoded data about the authenticated user. The ACS URL is shown on the service provider information tab of the SSO settings. Note that it includes the domain name, so that runZero can find the correct set of SSO settings to use to process the data.

runZero then decodes the encoded data, and checks its digital signature using the certificate that was provided as part of the identity provider settings. Assuming the certificate is valid, the decoded information about the user is then trusted by runZero. If necessary, a runZero user account is provisioned. The user is then signed in to begin a new runZero session.

Service provider information

After filling out the identity provider settings, you can switch to the service provider information tab to obtain information you need to provide to your SSO identity provider.

The Assertion Consumer URL, sometimes called the SSO URL, is the URL a person's web browser will be redirected to after they sign in successfully on the SSO identity provider. The URL contains the domain that was set up in the identity provider settings. runZero uses the domain to find the right set of SSO settings, so it can verify the data from the SSO provider. This means that if you change the domain in the identity provider settings for any reason, the ACS URL will change, and your SSO provider will need to be given the new URL.

The user sign-in URL is a URL that you can visit to begin the sign-in process. It displays the computed identity provider URL to aid in debugging.

Common problems

A user signs in on the SSO provider, but runZero refuses the sign in

The most common cause of this problem is that the IdP has been configured with an Assertion Consumer URL that does not include the domain name that was set in the SSO identity provider settings in runZero, or includes the wrong domain name.

A correct ACS URL will look like `https://console.runzero.com/auth/example.com/saml20/process` where `example.com` is the domain name configured in the runZero SSO identity provider settings.

Missing x509 Element

The full error message is "invalid SAML response: error validating response: Missing x509 Element"

This error can occur if you have entered multiple CA certificates into the certificate field of the SSO settings in runZero, but your SSO IdP didn't include a `KeyInfo` element to tell runZero which certificate to check the signature against. You will need to either configure your IdP to include information about which of the CA certificates to use as `KeyInfo`, or use a single CA certificate.

Managing SSO group mappings



Only runZero administrators can automatically map users to user groups using SSO attributes and custom rules.

SSO group mapping allows you to map your SAML attributes to user groups in runZero. In runZero, user groups explicitly set the organizational [role](#) and determines the tasks users can perform within each organization. When you set up SSO group mappings, you explicitly define the SSO attribute and value you want to use for mapping. If there is a match, runZero will apply the group settings for the user. As a result, you can ensure that SSO users are mapped to their respective groups in runZero.

For example, your IT team may need to be part of a group with administrator privileges. In this case, you can create a user group with an administrator role and then create an SSO group mapping that maps the SAML attribute that identifies your IT team to the user group. When someone from your IT team logs in to runZero, they will automatically be added with the appropriate access and permissions, all without pre-provisioning their account. After evaluating all SSO group mapping rules, runZero grants the user the highest privilege assigned for each organization. Where [custom roles](#) are involved, the permissions of every role that applies are added together.

Creating SSO group mappings

Before you create your SSO group mapping, make sure that you have [set up SSO](#) for your organization and [created user groups](#). Both must be set up in order to successfully create SSO group mappings.

Only runZero super users can create SSO group mappings.

1. Go to [Your team > SSO settings > Group mappings > Add group mapping](#).
2. In the **SSO attribute** field, enter the attribute you want to check for matching values. These values are defined in your SSO configuration.
 - For Azure AD SSO, note that the **SSO attribute** field must match the claim name from Azure AD.
3. In the **SSO value** field, provide a comma separated list of values that the attribute could match. When there is a match, runZero will apply the group permissions.
4. Click the **Group** dropdown and choose the user group that will be assigned if there is a match. The dropdown will list all user groups that have been created.
5. Save the SSO group mapping. These settings will apply the next time the user logs in to runZero.

Changes will not apply to users currently signed in. They will need to sign out and sign back in for the changes to take effect. You can forcibly sign out users to apply the SSO group mappings immediately.

Forcing a user to sign out

Changes to user permissions will not apply until the user signs out and logs back in to runZero. If you need to apply permissions immediately after setting up the SSO group mappings, you can forcibly sign out users. This will sign users out of their current session and require them to sign back in again. After they sign in, their updated permissions will be applied. Only superusers and admins who have access to all organizations can force sign-outs.

To forcibly sign out users, go to the [Teams page](#) and select the users you want to sign out. Click the sign-out button to log these users out.

Viewing SSO group mappings

To view all SSO group mappings that have been created, you can go to the [Group mappings page](#). From this page, you can create, edit, or delete group mappings as needed.

Viewing SSO group mapping assignments

To see the SSO groups that a user has been assigned to, go to the [Users page](#). From the *Groups* column, you can see the number of user groups and SSO groups the user is a part of. The number of SSO groups will be in parentheses.

Clicking the gear icon under actions will open the user settings for the user. The access summary tab will then display all of the organizations and roles they have assigned.

Deleting a group mapping

1. Go to the [Group mappings page](#).
2. Select the group you want to delete and click the **Delete** button. All users provisioned through the group mapping will revert back to their account-level permissions.

Searching for SSO group mappings

When you are on the [Group mappings page](#), you can use the following keywords to search in the table:

Keyword	Description	Example
id	User's ID.	id:123456789
sso_attribute	User's SSO attribute.	sso_attribute:department
sso_value	User SSO attribute value.	sso_value:securityteam
created_at	Time or date user group was created.	created_at:>2weeks

updated_at	Time or date user group was last updated.	updated_at:>1year
created_by_email	Email of user who created the group.	created_by_email:user@example.com
group_id	User group ID.	group_id:123456789
group_name	User group's name.	group_name:group1

Group IDs can be found by going to [the group config page](#) and enabling the ID column from the Columns menu above the data grid.

The `group_id` keyword is only available for the Users table; for the groups table, use `id`.

The `group_name` keyword is only available for the Users table.

Setting up Microsoft Entra SSO

Superusers can configure single sign-on to the runZero Console using an external identity provider (IdP), which enables authentication and user access control to the runZero Console from your single sign-on (SSO) solution. By default, runZero has SSO functionality available, but it's not a requirement to sign in to the console. You can make it a requirement or disable it completely.

Here are the high-level steps to set up SSO using Microsoft Entra (formerly Azure AD) to authenticate and manage user access to runZero:

1. [Add and configure runZero as an Entra app.](#)
2. [Download the SSO configuration metadata in XML format.](#)
3. [Set up SSO in runZero.](#)
4. [Add users to your runZero app in Entra ID.](#)

Requirements

Before you can set up SSO for the Microsoft Identity Platform:

- Verify that you have administrator privileges for Azure AD.
- Verify that you are a [superuser in runZero](#). Look for the yellow star in your account status.

Step 1: Add and configure runZero as an Azure app

The first thing you need to do is [add runZero as a non-gallery application](#) to your Azure AD setup and to [configure the settings for runZero](#) as an Azure AD application.

1. In Azure, go to **Enterprise Applications > New Application > Create your own application**.
2. Under the **What are you looking to do with your application?** section, choose the **Non-gallery application** option.
3. Name your application something like `runZero`, and then add it.

4. Go to **Azure Active Directory > Enterprise applications** and open the newly created runZero application.
5. Select the **Single sign-on** tab, and then choose **SAML** as the sign-on method.
6. For the fields on the **Configure App Settings** page, go to <https://console.runzero.com/team/sso/sp> and copy the necessary service provider details:
 - Entity ID
 - Single sign-on URL
 - SSO callback (ACS) URL
7. Enter the values into the relevant fields in the Azure AD portal.
8. **Do not** set a value for "Sign on URL (Optional)" or "Relay State (Optional)".
9. If you want to use group information from Azure to set up group membership in runZero, [configure the IdP to send group information](#).

Step 2: Download the SSO configuration metadata

While editing your application settings, you can get the download link to obtain the SSO configuration metadata in XML. You'll need this information to set up SSO in runZero.

1. On the **Configure App Settings** page, find the **SAML Signing Certificate** section.
2. Locate the XML download link under the [Federation Metadata URL](#).
3. Download the file. You'll need the contents of this file for the next step.

Step 3: Set up Entra ID SSO in runZero

Now that you have the SSO configuration metadata in XML, you can configure Entra ID SSO settings in runZero.

1. Go to <https://console.runzero.com/team/sso/idp> to access the SSO IdP provider settings page in runZero.
2. Choose one of the following modes to enable SSO:
 - **Allowed** - Enables SSO, but users still have the option to sign in without SSO.
 - **Required** - Requires users to sign in with SSO. Only superusers can sign in without SSO.
3. Enter the domain name that is associated with SSO authentication. This is likely your company domain (companyabc.com).
4. Choose a default role for SSO users. This is the [role](#) all new users will be assigned when their account is created.
5. Copy the XML you downloaded from Azure and paste it into the Metadata XML field on the runZero SSO IdP page.
6. Apply your SSO settings. The remaining IdP fields will auto-configure for you.
 - The issuer URL will look something like `https://sts.windows.net/00000000-0000-0000-0000-000000000000/` where the UUID at the end is your unique Microsoft Active Directory (tenant) ID, listed under **App registrations > Overview > Endpoints**.
 - The sign-in URL will be something like `https://login.microsoftonline.com/00000000-0000-0000-0000-000000000000/saml2` with the zero UUID replaced with your unique tenant ID.
 - The certificate will be Microsoft's PEM encoded certificate, which will be extracted automatically from the XML.

- On the Microsoft side, the redirection URL for runZero should be `https://console.runzero.com/auth/<domain>/saml20/process`, where `<domain>` is replaced with the domain specified in the runZero SSO settings.

Step 4: Add users to the runZero app in Azure

Now that you've completed the set up, you can go to the runZero app in Azure portal to [add users and assign their access](#). Any users you add to the runZero app will be viewable from the [Team members page](#) in runZero, once they have signed in to runZero.

Step 5: Update SSO group mappings to match any configured Azure groups (if applicable)

If you have created user groups within Azure, you will need to update your [SSO group mappings](#) in runZero to associate the groups created in Azure with user groups in runZero. This will ensure that the appropriate access and permissions are added to your users when they sign in to runZero.

When setting up the [SSO group mappings](#), note that the **SSO attribute** field must match the claim name from Entra ID.

Setting up Okta SSO

Superusers can configure single sign-on to the runZero Console using an external SAML identity provider (IdP), such as Okta, which enables authentication and user access control to the runZero Console without typing in credentials.

Here are the high-level steps to set up single sign-on (SSO) using Okta to authenticate and manage user access to runZero:

- [Add runZero as an application in Okta.](#)
- [Set up SSO in runZero.](#)
- [Add users to the runZero app in Okta.](#)

Requirements

Before you can set up Okta SAML:

- Verify that you have administrator privileges for Okta.
- Verify that you are a [superuser in runZero](#). Look for the yellow star in your account status.

Step 1: Add and configure runZero as an Okta app

1. Go to **Okta > Applications > Create App Integration**. When the **Create a new app integration** window appears, select **SAML 2.0** for your sign-in method.

- ☐ **OIDC - OpenID Connect**
Token-based OAuth 2.0 authentication for Single Sign-On (SSO) through API endpoints. Recommended if you intend to build a custom app integration with the Okta Sign-In Widget.
- ☒ **SAML 2.0**
XML-based open standard for SSO. Use if the Identity Provider for your application only supports SAML.
- ☐ **SWA - Secure Web Authentication**
Okta-specific SSO method. Use if your application doesn't support OIDC or SAML.
- ☐ **API Services**
Interact with Okta APIs using the scoped OAuth 2.0 access tokens for machine-to-machine authentication.

2. For the general settings, you'll need to provide a name for the app. Call the app `runZero`. You can also add a logo to make it easier for users to identify the runZero app.
3. For the SAML settings, you'll need to go to the [service provider information page](#) in runZero to find the relevant URLs.
 - **Single sign on URL** - In runZero, this is the assertion consumer service (ACS) URL.
 - **Audience URI or SIP Entity ID** - In runZero, the entity ID, or SAML audience, will be `https://console.runzero.com`.
 - **NOTE:** make sure to include the leading 'https://' when entering this field in Okta.
4. For the remaining settings, like the attribute statements, visit the Okta documentation to [learn how to configure them](#).
5. When you finish configuring the SAML settings, Okta will prompt you for some feedback on how you will be using the app. You can opt to provide feedback or skip to complete the set up.
6. After Okta creates the app, you will need to view the SAML 2.0 instructions to complete the set up. Go to the **Sign On** tab for the runZero app and view the SAML 2.0 instructions. You'll need these details for the next step.



SAML 2.0 is not configured until you complete the setup instructions.

[View Setup Instructions](#)

[Identity Provider metadata](#) is available if this application supports dynamic configuration.

Step 2: Set up SSO in runZero

1. Go to the [SSO setup page](#) in runZero. Choose one of the following modes to enable SSO:
 - **Allowed** - Enables SSO, but users still have the option to sign in without SSO.
 - **Required** - Requires users to sign in with SSO. Only superusers can sign in without SSO.
2. Enter the domain name that is associated with SSO authentication. This is likely your company domain (companyabc.com).
3. Choose a default role for SSO users. This is the [role](#) all new users will be assigned when their account is created.
4. Copy the fields from Okta into runZero.
 - **Issuer URL** - In this field, enter the **Identity Provider Issuer URL** from Okta. This will look something like `http://www.okta.com/<ID>`
 - **Sign-in URL** - In this field, enter the **Identity Provider Single Sign-On URL** from Okta. This will look something like `http://<okta-instance>/app/<app-name>/<ID>/sso/saml`.
 - **Certificate** - Copy and paste the entire contents of the X.509 certificate from Okta.
5. Apply your SSO settings.

Step 3: Add users to the runZero App in Okta

Now that you've completed the set up, you can go to the runZero app in Okta to [add and manage user access](#). After you've completed this step, your users will be able to go to your SSO sign-in URL to access runZero.

Step 4: Update SSO group mappings to match any configured Okta groups (if applicable)

If you have created user groups within Okta, you will need to update your [SSO group mappings](#) in runZero to associate the groups created in Okta with user groups in runZero. This will ensure that the appropriate access and permissions are added to your users when they sign in to runZero.

Managing licenses

As a runZero superuser or billing user, you can access and manage your organization's licensing, plan, and billing information.

Not a superuser or billing user? Please contact your organization's [superuser or billing user](#) to get help with licensing and billing information. These users are tagged with a yellow star.

How do I view my license?

If you're a superuser or billing user, go to [Account > License](#) to view your runZero licensing information.

Your licensing information will show:

- The number of organizations, sites, Explorers, and user accounts you have.
- The number of recent assets you have across all organizations.
- The number of project assets you have.
- The total number of recent assets and project assets your license supports.

What count as recent assets?

runZero calculates the number of recent assets based on how many have been seen during the last 30 days.

For assets that have been scanned by runZero, if they have been seen by a scan in the last 30 days, they are counted as recent assets.

If third party data indicates a device has been seen in the last 30 days, that will also cause the asset to count as recently seen.

License limits are uniform across all asset types. It doesn't make any difference whether the asset is online or offline, scanned or unscanned, or where the asset data was sourced from. For assets with multiple data sources, with different dates that the device was last seen at, the most recent date applies.

The number of recent assets is calculated in the background. If you delete assets it may take a short while for the number to recalculate.

When does my subscription expire?

To see when your subscription or license expires, go to [Account > License](#).

Find the line: This is a runZero [edition] subscription that expires at [date and time].

If your subscription has expired, you will see: This is a runZero [edition] subscription that expired on [date and time]. You will no longer be able to run discovery scans. To continue using

runZero, you will need to [renew your subscription](#).



Does the Community license expire? Nope, it's part of our free tier and has no expiration date. You can continue to use it as long as you don't have more than 100 recent assets.

How do I renew my subscription?

For help renewing your subscription, please contact us by [email](#) or by using this [online form](#).

How do I convert to the Community Edition?

No longer need the full platform? You can [convert it to the Community Edition](#). This plan is on our free tier and supports up to 100 recent assets.

How do I find my invoices?

Super users and billing users can access invoices from the runZero Console. Go to the [License page](#) to see all invoices.

How do I change or cancel my subscription?

Please [contact us](#) if you need to modify or cancel your subscription.

Network discovery

runZero can gather asset data through unauthenticated active scanning, passive traffic sampling, and inbound integrations.

Active scanning

The runZero Explorer and scanner perform [unauthenticated active scanning](#) of your specified networks based on the configurations you set. They leverage various network protocols to discover and fingerprint assets connected to the network.

Active scans can be configured to run once or on a schedule. Scan templates can also be used to ensure consistency across multiple scan tasks.

Traffic sampling

Explorers can be configured to perform passive traffic sampling in your environment in order to identify assets communicating across your networks. [Traffic sampling](#) is configured on a per-Explorer basis on the Explorer details page. Explorers will automatically pause traffic sampling activities in order to handle scan tasks, continuing to parse sniffed traffic once the scan completes.

The traffic sampling feature can also ingest a PCAP file to glean useful asset data.

Inbound integrations

[Inbound integrations](#) for many different platforms can be configured to enrich your runZero asset context and provide important insights into coverage and capabilities. Integrations can be configured either as connectors or scan tasks.

Active scanning

An active scan identifies all responsive devices on a given network, fingerprints these devices, and populates the asset, services, screenshot, and software inventory. Regular scans of internal and external networks is an important step in network management. Scans are configured by site, Explorer, and scan scope. The scan scope can include IP ranges, domain names, ASNs, and even entire country codes.

When creating a new scan, you have multiple parameters you can set, ranging from scheduling a date to more advanced options. To get started, login to the runZero Console, select Scan from the Data sources section of the navigation menu, and choose "Start Standard Scan". Scans can also be launched from the Inventory views.

Site

runZero organizes information into organizations and sites. Organizations are distinct entities that are useful for keeping data separate and contain a collection of sites. Sites are used to model segmented networks, particularly independent networks which use the same private IP address ranges.

For example, you might have multiple physical locations with their own local networks, all using the 10.0.0.0/8 private IP range. By defining them as sites, you can set up an Explorer for each, and the networks and assets will be treated as completely independent even if similar systems are seen at the same IP addresses in each.



runZero treats each site as a separate IP space. This enables overlapping subnets within an organization, but can result in duplication if the same network is scanned in different sites within the same organization.

Since scan analysis occurs at the site level, the boundaries you define for a site set the default scope for scans for that site.

Explorer

Select the Explorer to run the scan from, chosen from the set of registered Explorers for the site. The Explorer you choose must be able to directly communicate with the networks and addresses you define for the discovery scope. The chosen Explorer should ideally be able to reach all addresses in the scope directly, without a firewall in the way. Stateful firewalls and VPN gateways may interfere with the discovery process.

Hosted External Explorer

runZero Platform users can perform scans of public IP space using runZero-hosted Explorers. When creating a scan, choose a Hosted External Explorer in the 'Run task with' dropdown. When using this option, the discovery scope must use public IP addresses or ranges, or resolve to public IP space.

Discovery scope

The discovery scope defines the IP addresses that will be scanned. The scope uses the site settings when specified as they keyword "defaults", but may be changed on a per-scan basis as well. The scope should include at least one IP address or hostname. IPv4 address ranges can be specified in most standard formats:

- 10.0.0.1
- 10.0.0.0/24
- 10.0.0.0/255.255.255.0
- 10.0.0.1-10.0.0.255

IPv6 addresses can be specified individually, but IPv6 ranges are not supported.

Hostnames specified in the scope will be resolved at runtime by the assigned Explorer. If the hostname returns multiple IP addresses, all addresses in the response will be scanned. Hostnames can also have masks applied, indicating that the mask should expand to each resolved address of the hostname. For example, if `example.com` resolves to both `1.2.3.4` and `5.6.7.8`, the input of `example.com/24` would become `1.2.3.0/24` and `5.6.7.0/24`. IPv6 addresses returned from hostname resolution will be scanned if the Explorer has a valid IPv6 address and route to the target.



runZero load balances the scan across as many subnets as possible, in a quasi-random order.

Discovery keywords

The following keywords are supported for both scan scopes and exclusions.

- **asn4:** The `asn4:<AS number>` keyword can be used to specify IPv4 ranges associated with a given AS number.
- **country4:** The `country4:<ISO code>` keyword can be used to specify IPv4 ranges associated with a given two-character country code.
- **public** and **private:** The `public:<mode>` and `private:<mode>` keywords can be used to specify IPv4 and IPv6 addresses associated with assets in the current organization. The mode parameter can be set to `all`, `primary`, or `secondary` to indicate which IP addresses are used.

The `public` keyword selects all non-reserved IP addresses associated with organization assets. The `private` keyword selects all RFC-1918 and private use IP addresses associated with organization assets.

- **domain:** The `domain:<domain>` keyword is available to cloud-hosted users and uses the syntax `domain:<domain name>` to automatically select publicly-known hostnames for a given domain name.

Scan name

You can assign a name to your Scan task to make it easier to keep track of.

Scan speed

Specify the maximum packet rate for the overall discovery process, in network packets per second. 500 is conservative, 3000 works for most LANs including WiFi, 10000 or more may be helpful for large sites with fast connectivity.

The scan speed directly affects how long the scan will take to complete. An approximate formula is:

$$\text{time in seconds} = \text{hosts} \times \text{ports} \times \text{attempts} \div \text{scan speed}$$

The number of hosts scanned is primarily determined by the discovery scope. The number of ports is around 500 by default, and three attempts are made to connect.

The number of hosts and ports scanned can be affected by the advanced scan options, and speed can also be impacted by maximum host rate and group size; see the descriptions of the advanced scan options below.

Note also that this formula doesn't take into account time taken to take screenshots, follow web server redirects, or process the scan data.

Schedule

You can set a date and frequency for your scan task. Dates and times take into account your browser's advertised timezone.

Scans scheduled to start in the past will be launched immediately and then repeated at the specified time based at the frequency selected.

Scheduling grace period

Specify the number of hours to wait for an available Explorer before giving up on this scan. A zero or negative value will result in the scan retrying indefinitely until an Explorer becomes available.

Scan duration limit

You can specify a number of hours to limit scan duration to; if scanning is still in progress after this time has elapsed, the scan will be canceled. This does not limit processing time.

If you set this to 0, no limit is applied.

Advanced scan options

The Advanced tab can be used to display and modify additional scan settings, such as network exclusions, scan speed, the ports covered by the TCP scan, and which probes are enabled. The default settings should work for most organizations but may need to be tweaked for slow networks or unreliable links.

Maximum host rate

As well as setting an overall scan rate in packets per second, you can also control the maximum rate at which packets are sent to any single host IP address. This is useful when you have devices which are easily overloaded by network traffic. The default should be safe for most systems.

Max group size

When runZero scans your network, it spreads the scan load across many IP addresses at once. The max group size determines how many IP addresses can be actively scanned at once – allowing for the fact that hosts may take some time to respond to probes. The max group size needs to be at least as large as the overall scan speed, or else it would limit the speed of the scan to below the set value. If you provide a value that's lower than the overall scan speed, it will be increased automatically at scan time.

The max group size is mostly useful when dealing with stateful network devices that can only track a limited number of connections at once, as a way to restrict how many active TCP sessions will result from a runZero scan.

Max TTL

The IP standards define a maximum hop count for packets. In IPv4, this is called the Time To Live or TTL, while on IPv6 this is called the Hop Limit. Every device processing a packet must decrease the TTL or Hop Limit one. If this value reaches zero, the route receiving the packet must discard the packet. This setting can be used to set the maximum hop limit for scan traffic.

ToS

The IP standards define a Type of Service or ToS for packets. In IPv4, this is called the Type of Service or ToS, while on IPv6 this is called the Traffic Class or TC. The ToS or Traffic Class is used by switches and routers to prioritize network traffic. The lower bits of the IPv4 ToS are also used for congestion controller. This setting can be used to set the ToS or Traffic Class for scan traffic. Please note that the ToS/Traffic Class settings do not apply to all traffic sent by runZero, but instead are limited to the basic discovery probes. Some protocols, such as SNMP, and integrations, such as VMware, do not set the ToS/Traffic Class fields on their corresponding packets. If all scan traffic must be consistently tagged with the correct ToS or Traffic Class, this can be accomplished through settings on the managed switch port instead.

TCP ports

The **Included TCP ports** and **Excluded TCP ports** fields can be used to override the default scan ports. The string "defaults" will lookup the current default port list at scan time. The current port list is:

1	7	9	13	17	19	21	22	23	25	37
42	43	49	53	69	70	79	80	81	82	83
84	85	88	102	105	109	110	111	113	119	123
135	137	139	143	161	179	222	264	280	384	389
402	407	442	443	444	445	465	500	502	512	513
515	523	524	540	541	548	554	587	617	623	631
636	664	689	705	717	743	771	783	789	830	873
888	902	903	910	912	921	949	990	993	995	998
1000	1024	1030	1035	1080	1083	1089	1090	1091	1098	1099
1100	1101	1102	1103	1128	1129	1153	1158	1199	1211	1220
1234	1241	1260	1270	1300	1311	1337	1352	1433	1434	1440
1443	1468	1494	1502	1514	1521	1530	1533	1581	1582	1583
1604	1610	1611	1723	1755	1801	1811	1830	1883	1900	1911
2000	2002	2021	2022	2023	2024	2031	2049	2068	2074	2082
2083	2100	2103	2105	2121	2181	2199	2207	2222	2224	2323
2362	2375	2376	2379	2380	2381	2404	2443	2455	2525	2533
2598	2601	2602	2604	2638	2809	2947	2967	3000	3001	3003
3033	3037	3050	3052	3057	3071	3080	3083	3128	3142	3200
3210	3217	3220	3260	3268	3269	3273	3299	3300	3306	3311
3312	3351	3389	3460	3478	3500	3502	3628	3632	3690	3780
3790	3817	3868	3871	3872	3900	4000	4092	4322	4343	4353
4365	4366	4368	4369	4406	4433	4443	4444	4445	4567	4659
4679	4730	4786	4840	4843	4848	4900	4949	4950	4987	5000
5001	5003	5006	5007	5022	5037	5038	5040	5044	5050	5051
5060	5061	5093	5094	5168	5222	5247	5250	5275	5347	5351
5353	5355	5392	5400	5405	5432	5433	5466	5498	5520	5521
5554	5555	5560	5580	5601	5631	5632	5666	5671	5672	5678
5683	5800	5814	5900	5901	5902	5903	5904	5905	5906	5907
5908	5909	5910	5911	5920	5938	5984	5985	5986	5988	5989
6000	6001	6002	6050	6060	6070	6080	6082	6101	6106	6112

6161	6262	6274	6277	6333	6379	6405	6432	6443	6481	6502	6503
6504	6514	6542	6556	6660	6661	6667	6905	6988	7000	7001	7002
7021	7070	7071	7077	7080	7100	7144	7181	7210	7373	7443	7444
7474	7510	7547	7579	7580	7676	7700	7770	7777	7778	7780	7787
7800	7801	7860	7879	7902	7990	7999	8000	8001	8002	8003	8004
8006	8008	8009	8010	8012	8014	8020	8023	8028	8030	8080	8081
8082	8083	8084	8085	8086	8087	8088	8089	8090	8095	8098	8099
8100	8123	8127	8160	8161	8172	8180	8181	8182	8188	8193	8200
8205	8222	8265	8300	8303	8333	8400	8428	8443	8444	8445	8446
8447	8448	8449	8471	8488	8500	8503	8530	8531	8545	8649	8686
8787	8800	8812	8833	8834	8850	8871	8880	8883	8888	8889	8890
8899	8900	8901	8902	8903	8983	9000	9001	9002	9042	9060	9080
9081	9084	9090	9091	9092	9099	9100	9111	9152	9160	9200	9300
9380	9390	9391	9401	9418	9428	9440	9443	9444	9471	9495	9524
9527	9530	9593	9594	9595	9600	9809	9855	9997	9998	9999	10000
10001	10008	10050	10051	10080	10098	10162	10202	10203	10250	10255	10257
10259	10443	10616	10628	10800	11000	11099	11211	11234	11333	11434	11443
12174	12203	12221	12345	12379	12397	12401	13364	13400	13500	13778	13838
14330	15200	15671	15672	16102	16443	16992	16993	17185	17200	17472	17775
17776	17777	17778	17781	17782	17783	17784	17790	17791	17798	18245	18264
18789	18881	19000	19300	19530	19810	19888	20000	20010	20031	20034	20101
20111	20171	20222	20256	20293	21047	22222	23472	23791	23943	24442	25000
25025	25565	25672	26000	26122	27000	27017	27018	27019	27080	27888	28017
28222	28784	29418	30000	31001	31099	32400	32764	32843	32844	32845	32913
33060	34205	34443	34962	34963	34964	37718	37777	37890	37891	37892	38008
38010	38080	38102	38292	40007	40317	41025	41080	41523	41524	44334	44343
44818	45230	46823	46824	47001	47002	47290	48898	48899	49152	50000	50013
50021	50051	50070	50090	50121	50505	51443	52302	52311	53282	54321	54921
54922	54923	55553	55580	57772	61614	61616	62078	62514	65002	65535	

Prescan modes for large IP spaces

Sometimes, the scope of your IP space is unknown, subnet usage is unknown, and the total number of assets is unknown. These unknowns can make it challenging to optimize your discovery scans for efficiency and speed. And when your IP space is large, like a /16 space with a few thousand IPs in use, a full discovery scan can take more time to complete, since it looks at more than 500 TCP ports and 15 UDP ports on every address. In these types of cases, you may want to tune your scan settings to prefilter ranges and IP addresses before a full scan.

runZero has two prescan modes that you can use to run a faster scan: subnet sampling and host ping.

Subnet sampling

To speed up scans of large subnets you can use the **“Only scan subnets with active hosts”** advanced scan option. If this option is on, a prescan runs against the target space to identify the subnets with an active host. This mode leverages heuristics runZero has collected to identify addresses that are more likely to be responsive across subnets. This process allows

runZero to quickly scan larger spaces by identifying the subnets that are in use, before starting full probes. All subnets that are identified as having active hosts are then fully scanned – unless you enable host pings.

There are two tweakable parameters for subnet sampling. The sample rate determines what percentage of addresses in each subnet are prescanned to determine if the subnet should be scanned. The subnet size determines how many IP addresses are in each subnet. By default, the subnet size is 256 addresses, corresponding to a /24 subnet, and 3% of the addresses in each subnet are prescanned.

Host ping

After you have some insights on the subnets that are in use, you may want to limit the full scan to only addresses that respond to the most common ping methods, such as ICMP and some TCP and UDP ports. If you choose the **“Limit scans to pingable hosts”** advanced scan option, only hosts that respond to a ping request will be fully scanned.

The runZero Explorer uses multiple protocols for ping scans:

- Conventional ICMP ping, performed by sending an ICMP echo request and looking for an ICMP echo reply.
- TCP ping, performed by sending a TCP SYN packet to a series of common ports and seeing whether the host responds with RST or TCP SYN/ACK.
- UDP ping, performed by sending a packet to port 65535 and checking for an ICMP response of port unreachable.

The set of ports used for TCP and UDP ping can be adjusted in the LAYER2 section of the Probes and SNMP tab when setting up a scan task.

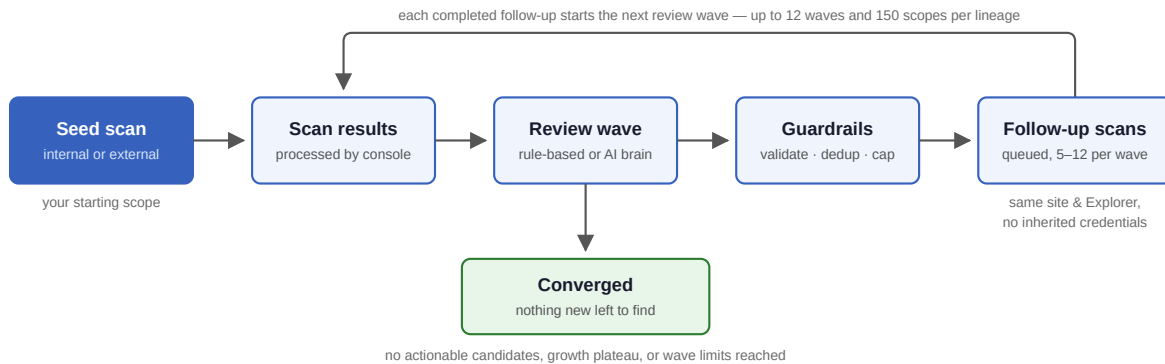
Note that it is relatively common for enterprise firewalls to be set up to block ping, or for hosts to be set up not to respond to ping requests. Limiting scans to pingable hosts can therefore result in assets being missed entirely, even if their IP addresses are probed. If your goal is to speed up scan times, subnet sampling is usually the better option.

It's possible to use both subnet sampling and limiting scans to pingable hosts at the same time, but this is not recommended except as a last resort for reducing scan times.

Autonomous discovery

Autonomous discovery is a scan mode that reviews the results of every completed scan and automatically queues bounded follow-up scans to expand coverage. Instead of asking you to enumerate every subnet or subdomain up front, runZero starts from a small seed scope, looks at the evidence each scan produces — routing tables, ARP caches, traceroute hops, TLS certificates, DNS answers — and decides what to scan next. The loop repeats until there is nothing new left to find.

Autonomous discovery works out of the box using deterministic rules and requires no AI. Accounts that have [AI enabled](#) can optionally let their own AI provider drive the analysis instead; both paths are described in [AI-assisted discovery](#).



Requirements

- The account must have the autonomous discovery entitlement, which is included with all runZero license tiers, including Community Edition.
- Internal discovery requires at least one active [Explorer](#) with access to the private networks you want to explore.
- External discovery targets public IP space and typically runs through a [Hosted External Explorer](#).
- AI is **not** required. AI-assisted analysis is optional and only used when the account has the AI entitlement and a working [AI configuration](#).

How it works

1. You launch a scan with an autonomous discovery mode selected. This seed scan runs like any other scan.
2. After the console processes the results, a **review wave** analyzes the new evidence and proposes follow-up candidates: private subnets, related hostnames and domains, or deeper scans of confirmed-live networks.
3. Every candidate is validated against the guardrails below, deduplicated against everything the lineage has already scanned, and dropped if its confidence score is below 0.30.
4. Surviving candidates are queued as child scan tasks, capped at 5 to 12 follow-ups per wave depending on how confident the wave is (8 is typical). Adjacent /24 candidates are batched into a single task, and discovered hostnames land as one multi-target scan.
5. Each follow-up keeps the autonomous mode, so the loop repeats when it completes — up to 12 waves and 150 distinct scopes per lineage.

The loop **converges** — stops queueing new work — when a wave produces no actionable candidates, novelty drops off, asset growth or vendor diversity plateaus for consecutive waves, or the iteration and scope limits are reached. Convergence is reported on the task, not treated as an error.

Guardrails

- Follow-up scans never leave the lane they started in: internal discovery only proposes private address space, and external discovery only proposes names under your seed domains.
- Site exclusions, existing scope, and everything already scanned by the lineage are always respected.
- Overly wide targets are rejected — internal candidates are never broader than a /16 (IPv4) or /48 (IPv6).
- Follow-up scans do not inherit credentials from the seed scan. Credential IDs and sensitive scan options are stripped before each child task is created.

Internal discovery

Internal mode grows coverage of private networks from topology evidence: SNMP routing tables, interface addresses and ARP caches, RIP routes, traceroute hops, CDP and LLDP management addresses, and default gateway attributes. Public routes and default routes are ignored.

Internal discovery uses a depth progression rather than deep-scanning everything immediately:

- **discovery** – a light probe of a newly inferred subnet to confirm live hosts.
- **sweep** – a coarse liveness scan of a wider private block when clustered evidence justifies it.
- **deepen** – a full default scan of a /24 (or IPv6 /48) that discovery confirmed as live. A subnet is only deepened once, and a subnet that looks like a firewall responding on every address is never treated as live.

Coverage is also tracked per scope and per dimension (liveness, ports, screenshots, and so on), so later waves can re-deepen scopes whose deep coverage has gone stale instead of only chasing new space.

External discovery

External mode starts from the public IPs and `domain:` targets in your seed scope and expands through names: TLS certificate subjects and SANs, forward DNS answers, parent domains, and observed hostname patterns such as region or environment series.

Expansion is strictly bounded by ownership. A candidate hostname or domain must share a registrable domain with your seed scope — an unrelated domain that appears in a shared certificate is dropped, and observed ASNs are never turned into IP sweeps. On the first wave, if public IPs are present, runZero may also queue a one-time passive [Shodan](#) import to enrich the seed assets before active expansion.

When scanning reveals infrastructure that cannot be enumerated over the network — cloud accounts or virtualization hosts, for example — the review records an **integration hint** suggesting the relevant [integration](#) instead of queueing a scan.

Starting an autonomous scan

From the [Scan page](#) in the Data sources section of the navigation menu, the **External** and **Internal** buttons open the scan form pre-configured for the matching mode, with a recognizable task name and a recommended starting scope. You can also configure any standard scan by hand: in the scan form, the **Autonomous discovery** control offers **Off**, **Internal**, and **External**, and the **Use recommended targets** button fills the scope with the site's default scope (internal) or your organization's domain and public egress IP (external).

- Internal discovery works best with a **Continuous** schedule and a seed scope drawn from the Explorer's local networks. Internal scopes must contain private IP addresses only — no hostnames or public ranges.
- External discovery works best with a recurring schedule — the form defaults to every 5 minutes, since hosted scans do not support the Continuous frequency — and a seed scope of public IPs, ASNs, and `domain:` targets. External scopes must not contain private addresses.
- On Community Edition licenses, autonomous scans default to an hourly schedule.

[Scan templates](#) can set a default autonomous discovery mode, so recurring scans created from a template participate automatically.

Working with follow-up scans

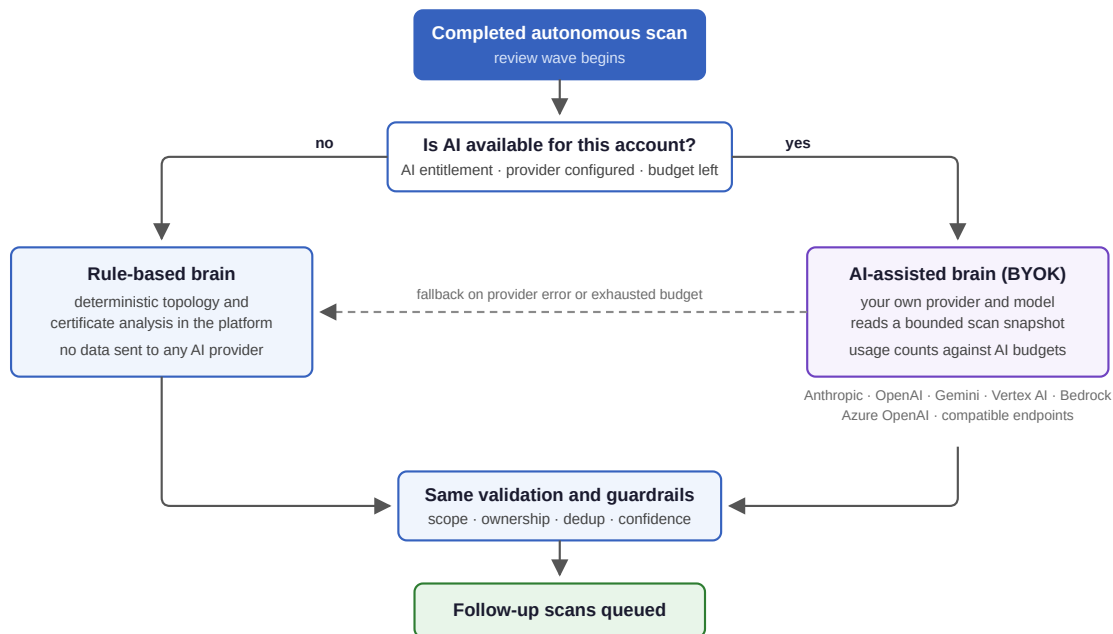
Queued follow-ups appear on the [tasks page](#) named `Autonomous follow-up: <target>`, tagged `autonomous=true`, with a description recording the wave number, confidence, and the evidence that motivated them. Each runs on the same site and Explorer (or hosted zone) as its parent.

Every autonomous task's detail page includes an **Autonomous discovery** card showing the loop status — mode, state, wave number, assets discovered, novelty, and vendor diversity — plus the latest review wave: which brain ran, every candidate it considered, its confidence, and the outcome (queued, skipped for low confidence, deferred by the wave cap, duplicate, or stop signal). When AI was used, the card also reports the token usage for the wave.

To stop the loop, stop or dismiss any task in the lineage. This cancels the entire lineage — active siblings and descendants are stopped, and no further follow-ups are queued.

AI-assisted discovery

Autonomous discovery runs one of two analysis engines, and the task details card tells you which one reviewed each wave.



Without AI: the rule-based brain

By default, review waves use the **rule-based** brain — deterministic topology and certificate analysis that runs entirely inside the runZero platform. Nothing is sent to any AI provider, there is no per-wave cost, and results are fully reproducible. This is the complete, supported experience: the rule-based brain implements all of the internal and external expansion logic described above.

With AI: bring your own key

runZero AI support uses a BYOK (bring your own key) approach — you connect a model from your own AI provider account rather than runZero supplying one. When AI is available, review waves use the **AI-assisted** brain: your configured model examines the same scan snapshot through a set of read-only tools and proposes candidates with reasoning. This can produce smarter expansion — recognizing naming conventions, prioritizing interesting segments, and stopping earlier when evidence is weak. To use it:

1. Your account must have the AI entitlement.
2. A superuser enables AI under [Account settings](#) → [AI configuration](#) by selecting or creating an AI provider credential. Supported providers include Anthropic, OpenAI, Google Gemini, Google Vertex AI, AWS Bedrock, Azure OpenAI, and OpenAI-compatible or Anthropic-compatible endpoints (OpenRouter, Ollama, internal proxies). **Verify & save** performs a live check against the provider before the configuration is activated.
3. Individual organizations can override the account provider with their own credential, or opt out of AI entirely, from the organization's AI settings.

The AI-assisted brain is subject to the same [token budgets](#) as other AI features. Daily input and output token caps can be set per account and per organization, and each wave's usage is

recorded against the `autoscan` feature in the AI usage report on the AI configuration page, as well as on the task itself.

A few properties worth knowing:

- **Same guardrails.** AI proposals are advisory. Every candidate passes exactly the same validation, scope, ownership, and deduplication checks as rule-based output before anything is queued.
- **Bounded data sharing.** The AI-assisted brain only sees a compact summary of the completed scan's results — it cannot browse your wider inventory. As with all BYOK features, data sent to your provider is governed by your agreement with that provider.
- **Automatic fallback.** If AI is unavailable for any reason — missing entitlement, a deleted credential, an exhausted budget, or a provider error mid-wave — the wave falls back to the rule-based brain and the loop continues. Autonomous discovery never stalls waiting for AI.

Limits

- A lineage runs at most 12 review waves and 150 distinct scopes.
- Each wave queues at most 5 to 12 follow-up scans; the rest are deferred to the next wave.
- Candidates below a confidence score of 0.30 are recorded but never scanned.
- Internal candidates are capped at /16 (IPv4) and /48 (IPv6) widths; external candidates must stay under a seed registrable domain.

Troubleshooting

If autonomous discovery is not behaving the way you expect, the questions and answers below may assist in your troubleshooting.

Why didn't my autonomous scan queue any follow-up scans?

1. Check the **Autonomous discovery** card on the task details page. If the state is **Converged**, the review wave found no actionable evidence — this is the loop working as intended, not a failure.
2. Review the candidates table on the same card. Candidates may have been proposed but skipped for low confidence, deduplicated against earlier waves, or deferred by the wave cap.
3. Verify the task was not stopped or dismissed while results were processing; a canceled lineage never queues follow-ups.

Why does the task show the rule-based brain when AI is enabled?

1. Confirm the account has the AI entitlement and that the [AI configuration](#) shows an active provider. A credential that has been deleted disables AI-assisted waves rather than borrowing another provider.
2. Check whether the organization has opted out of AI or has exhausted its daily token caps — budget-exhausted waves fall back to rules until the budget resets.

3. A provider error during a wave also falls back to rules for that wave; check the AI usage report for partial usage.

Why do follow-up scans run without my scan credentials?

This is by design. Inferred targets do not inherit trust from the seed scan, so credentials and sensitive scan options are stripped from every follow-up. Run a credentialed scan of the discovered scope manually if deeper authenticated data is needed.

Why did external discovery skip a hostname it found in a certificate?

External expansion only follows names whose registrable domain matches your seed scope. A hostname on a shared certificate that belongs to another organization's domain is intentionally dropped. Add the domain to a new seed scan's scope if you own it and want it explored.

Initial network scans

Background

Once you have an Explorer installed, you can start using it for network discovery. While our goal is to configure scheduled scans that we set and forget, we need to go about our first scans in a more structured manner.

The goals of our first scans are to:

- Verify the Explorer is setup properly and has everything installed
- Validate Explorer connectivity to varying parts of the network
- Determine how long scans will take at varying sizes to help with future scheduling

Your first few scans

To get started, you will want to scan a few smaller ranges to make sure everything is working as expected. Start with a few /24 network blocks from each of the RFC 1918 ranges to make sure everything looks good.

For setting up the first scan:

1. Navigate to **Sites > New Site > Create** a new temporary site within the Organization
2. Navigate to **Tasks > Scan > Standard Scan** to create a scan task
3. Chose the new site you created in step 1
4. Include a range of the RFC1918 IP addresses in the Discovery Scope, plus a small network or two that you know is in use. A suggested value for the RFC1918 range includes:

```
10.0.0.0/24,10.0.255.0/24,10.64.0.0/24,10.64.255.0/24,10.128.0.0/24,10.128.255.0/24,
10.192.0.0/24,10.192.255.0/24,10.255.0.0/24,10.255.255.0/24,192.168.0.0/24,192.168.64.0/24,
192.168.128.0/24,192.168.192.0/24,192.168.255.0/24,172.16.0.0/24,172.23.0.0/24,172.31.0.0/24,
<your networks here>
```

5. On the Advanced tab, enable the *Subnet sampling* option
6. Click on Initialize Scan

After these scans are complete, you will want to check for these things:

- Check the `ipv4.traceroute` value for assets in each RFC1918 range to verify you aren't sending traffic to an edge router or firewall.
 - Unused private IPs should have routes stubbed out to prevent traffic from being sent to the default gateway which can create a loop. You can also verify this with traceroutes from the Explorer.
- If your scan results have a large series of somewhat sequential IPs that have only ICMP or a very small number of similar ports open on them, that's probably a proxy or firewall. Check out those IPs to see if any are real. To find assets with only ICMP enabled use the inventory query `alive:t AND service_count:=1 AND service_count_icmp:=1`
 - You can add an allow rule for the Explorer IP to properly scan devices on the other side
 - Another option is to add a second Explorer on the other side of the proxy or firewall
- If you receive reports or alerts about service outages, check for session aware devices such as routers, firewalls, and proxies that are having issues handling the session load. If you run into this, there are multiple ways to approach solving the issue.
 - The simplest solution is to set up another Explorer on the other side of the device and run scans separately
 - Another option is to segment your scans on the existing Explorer, and run smaller, separate scan tasks for the network ranges on the other side of the device with lower packet per second and *max group* sizes to minimize the number of IPs that will be scanned at once
- Check how long each scan took to get an idea for how long larger scans would take
- Verify you see screenshots on ports that accept HTTP/HTTPS requests
 - If you don't see any, you likely need to install Chrome on the machine
- Check for MAC addresses
 - If you aren't seeing them you should configure SNMP
 - If SNMP is configured, you should verify community strings and check for unmanaged switches



Once you have verified that your first scan ran successfully, you can delete the temporary site and set up a real scan.

Full RFC 1918 scans

Once you have completed initial test scans, it's time to expand scanning to cover all subnets with live assets. One method of discovering all subnets with live assets is to run full RFC 1918 scans.

runZero offers a **Full RFC 1918 discovery** scan option that will discover assets across the following private address ranges as a single task.

- 10.0.0.0/8 or 10.0.0.0-10.255.255.255
- 172.16.0.0/12 or 172.16.0.0-172.31.255.255
- 192.168.0.0/16 or 192.168.0.0-192.168.255.255



The **Full RFC 1918 discovery** scan option is only recommended for small networks with limited complexity and should only be leveraged in a single site configuration.

Discovering the entire RFC 1918 private address space in a single scan can take days, if not weeks, to complete in a large complex network. runZero recommends reviewing the [Achieving RFC 1918 coverage](#) playbook for more information on scanning the entire RFC 1918 private address space.

Once you've completed scans of your private address space, review [Identifying gaps in scanning](#) to learn more about some of our built-in reports that we offer to help you get a better understanding of your network.

Identifying gaps in scanning

Background

After you have run a full network discovery scan, you can start to better understand your coverage and begin to optimize. By the end of this guide, you will understand how to use the out of the box reports in runZero to understand your gaps in network coverage.

RFC 1918 coverage

The first report to look at is the **RFC 1918 coverage** report. This report shows you which internal IPv4 subnets have been scanned, which likely contain assets, and which are still unknowns.

The scan coverage maps show all the addresses scanned within the 10.0.0.0/8, 172.16.0.0/12, and 192.168.0.0/16 ranges. See the legend to understand what percentage of each address space has been scanned. Clicking into any of the scanned subnets gives you access to the subnet grid for deeper asset analysis.

Identify scanned and un-scanned areas with the coverage map: On the flip side, red outlines indicate that there are un-scanned addresses runZero has indirect knowledge of that haven't been scanned directly. For example, this can happen when runZero finds a secondary IP address on a multi-homed device within a scanned subnet. The red boxes highlight the subnets most likely to be in use, but un-scanned.

Scan missing subnets: From the coverage report, you can launch a scan for any missing subnets in a given RFC1918 block – look for the binocular icon.

Scan missed subnets: The missing subnets will be shown in the scan scope and the subnet ping will be enabled by default. You can tune the scan configuration as needed for your environment.

Subnet utilization

The **Subnet utilization** report can provide similar visibility into your network to the RFC 1918 coverage report, but with emphasis on the subnets that contain live assets. This report will enumerate each of the subnets defined in your site definitions, and provide a count of live assets for that site and subnet, along with a utilization percentage. If there are live assets that are outside any site subnet, they will be aggregated into an inferred subnet based on the network mask size you select.

From this report, you can pivot to the asset inventory for a given subnet or initiate a new scan of a subnet. Another benefit of this report is that you can export the results as a CSV. This can be helpful for more complex data analysis and for scheduling recurring scans.

Network Map

The [Network Map](#) is the fastest way to spot scanning gaps. It overlays Layer-2 switch topology, Layer-3 routing, traceroute paths, and bridges in a single interactive view, and renders **unmapped MACs** and **unscanned traceroute hops** by default — every gap shows up as a node without an asset behind it. Search and filter directly inside the map (`multi_homed:t`, `category:OT`, etc.), then click any unmapped node to see which Explorer or scan reached it and which subnet it sits in.

Switch topology

View layer-2 link information extracted from SNMP-enabled switches. This report can be used to find unmapped assets and investigate why they aren't showing up in your scans.

Configuration for this report: The Switch Topology report uses data enumerated via SNMP to map switch ports to assets. In environments where SNMP v1 or v2 with default public or private communities are in use, this enumeration happens automatically. Non-default communities for SNMP v1/v2/v3 can also be provided in the scan configuration. Clicking on a node in this report will expand it to show its connections.

Finding unmapped MACs: This topology view is helpful when trying to understand how a given asset or switch is connected, but also provides a critical data point related to risk; the number of unmapped assets. An unmapped asset is a MAC address connected to a switch, but not found in an ARP cache or through any of the other techniques runZero uses for remote MAC address discovery.

Re-scan to properly map MAC addresses: For environments where a runZero agent is connected to each network segment, unmapped MACs may highlight VLANs or network segments that are missing from the scan scope. In environments where runZero is scanning assets multiple hops away, the unmapped asset count can provide an estimate of how well the remote segment is being identified.

Unmapped MACs

If you prefer a condensed view to see all unmapped macs, you can also use the Unmapped MACs report. This report shows all the unmapped MACs in your organization, organized by switch, and again by port. This data can be used similarly to the unmapped MACs data from the Switch Topology report.

Managing scan templates

A scan template is a predefined set of scan options and settings. If you have a scan configuration you use often, you can create a scan template to save those settings. The next time you create a scan, you can choose a template instead of manually configuring your settings. Each update you make to the scan template is automatically applied to new and recurring scans based on the template, as well as any queued scans which were set up using the template but have not started yet. With scan templates, you can save time and reduce the likelihood of misconfiguring a scan.

When you create a scan based on a scan template, you will not be able to edit the fields set by the template. However, you can configure the site, scan name, discovery scope, Explorer, and scan schedule, since these are not defined in the template.

Each time a scan runs using values from a template, the scan task is saved with a copy of the parameters. This means the task will list the values used for the scan, even if the template is modified after the scan completes.

Creating a scan template

Scan templates can be created in a few ways in runZero:

- By going to **Tasks > Task library**
- From an existing scan task
- Based on an existing scan template

Creating a scan template from scratch

1. Go to **Tasks > Task library** to view the [task library](#).
2. From the **Task library**, click **Add template**.
3. When the scan configuration form appears, enter a name for the template. Use this name to search and assign the template to a new scan.
4. [Configure the scan](#) as you normally would. The following tabs are available for you to access and configure settings:
 - **Standard** - Provide a name, description, and scan rate for the template.
 - **Advanced** - Configure excluded hosts, included ports, tags, host rates, group size, UDP probe max attempts, max TTL for all scan probe packets, ToS, screenshots, and subnet sampling.
 - **Probes and SNMP** - Enable SNMP and additional probes, and provide credentials. When you enable the SNMP probe, the scan will identify devices that are using

the SNMP protocol, using the port and credentials provided to find anything with SNMP enabled.

- **Credentials** - Enable any preconfigured [Scanning with credentials](#) you want to use for the scan. The scan uses the credentials if the defined CIDR scope matches the specific scan target.
5. Save the template.

To use this template, go to the *Templates* tab when you [configure a scan](#) or [create a Template scan](#) from the scan menu on the [inventory](#) or [tasks](#) pages.

Creating a scan template from a scan task

1. Go to **Tasks** to open the [Tasks overview page](#).
2. Find the scan task you want to use as the basis for your scan template and click the name to view the task details.
3. From the task details, click the **Copy** dropdown menu and choose **Copy for new template**.
4. When the scan configuration form appears, enter a name for the template. Use this name to search and assign the template to a new scan.
5. [Configure the scan](#) as you normally would. The following tabs are available for you to access and configure settings:
 - **Standard** - Provide a name, description, and scan rate for the template.
 - **Advanced** - Configure excluded hosts, included ports, tags, host rates, group size, UDP probe max attempts, max TTL for all scan probe packets, ToS, screenshots, and subnet sampling.
 - **Probes and SNMP** - Enable SNMP and additional probes, and provide credentials. When you enable the SNMP probe, the scan will identify devices that are using the SNMP protocol, using the port and credentials provided to find anything with SNMP enabled.
 - **Credentials** - Enable any preconfigured [Scanning with credentials](#) you want to use for the scan. The scan uses the credentials if the defined CIDR scope matches the specific scan target.
6. Save the template.

To use this template, go to the *Templates* tab when you [configure a scan](#) or [create a Template scan](#) from the scan menu on the [inventory](#) or [tasks](#) pages.

Creating a scan template based on a template

1. Go to the **Tasks > Task library** to view the [task library](#).
2. From the **Task library**, find the template you want to use as the basis for a new template and click the name to open it.
3. From the scan template configuration page, click **Copy**. runZero will duplicate and create a new scan template.
4. When the scan configuration form appears, enter a name for the template. Use this name to search and assign the template to a new scan.
5. [Configure the scan](#) as you normally would. The following tabs are available for you to access and configure settings:
 - **Standard** - Provide a name, description, and scan rate for the template.

- **Advanced** - Configure excluded hosts, included ports, tags, host rates, group size, UDP probe max attempts, max TTL for all scan probe packets, ToS, screenshots, and subnet sampling.
 - **Probes and SNMP** - Enable SNMP and additional probes, and provide credentials. When you enable the SNMP probe, the scan will identify devices that are using the SNMP protocol, using the port and credentials provided to find anything with SNMP enabled.
 - **Credentials** - Enable any preconfigured [Scanning with credentials](#) you want to use for the scan. The scan uses the credentials if the defined CIDR scope matches the specific scan target.
6. Save the template.

To use this template, go to the *Templates* tab when you [configure a scan](#) or [create a Template scan](#) from the scan menu on the [inventory](#) or [tasks](#) pages.

Applying a scan template to a scan

Instead of manually configuring a scan, you can choose to use a template instead. You can find all templates available for a scan on the **Templates** tab in your scan configuration.

You can also go to the **Task library** page and choose to **Create a scan based on this template** from the template's actions.

Editing a scan template

All updates to a scan template affect every scan that uses it. After you make a change to a template, the next scan will automatically update to use the new settings. Any scan queued to run will use the new settings.

Editing a scan that uses a scan template

Any changes to a template will affect all scans that use them. If a scan uses a template, you will need to edit the template in order to modify the scan settings. Otherwise, you will need to delete the scan and create a new scan configuration with your modifications.

When you create a scan based on a scan template, you will not be able to edit the fields set by the template. However, you can configure the standard fields, such as the site, scan name, discovery scope, Explorer, and scan schedule, since these are not defined in the template.

Deleting a scan template

Deleting a template will remove it permanently from runZero. All configurations will no longer be accessible. To delete a scan template, go to [the Task library](#), find the template you want to delete, and click **Delete**. A dialog will prompt you to confirm that you want to delete the template, and the action will be irreversible.

Scanning with credentials

The Credentials page provides a single place to store any secure credentials needed by runZero, including:

- SNMPv3 credentials
- Access secrets for cloud services like AWS and Azure
- API keys for services such as Censys and Miradore

Credentials are stored in encrypted form in the runZero database. Credentials, such as SNMP passwords, are used by runZero Explorers and are transmitted to them in encrypted form. For security reasons, the secret part of any credential cannot be viewed once entered.

When adding a credential, you can choose to make it a global credential that can be used for all organizations or to allow access only by specific organizations. The **Allow all** or **Disallow all** buttons let you quickly apply the same setting across all organizations. Individual organizations can also be toggled to allow or disallow access.

Most credential fields can be edited after the credential is saved. Some fields, like URLs, cannot be edited after saving for security reasons. Sensitive fields, such as passwords or access keys, will be hidden but can be overwritten.

Credential settings

The specific fields and options for a credential depend on the type of credential.

VMware and SNMP credentials, which are used by the runZero Explorer, allow a CIDR allow list to be specified. This can be used to limit which scanned IP addresses the credential will be used with. This feature allows you to avoid sending SNMP or VMware credentials to all scanned hosts on the network, and instead limit them to specific IP addresses or ranges.

Credential verification

Credentials can be verified when created or edited to ensure they can successfully authenticate. Choose **Verify & save** when creating or editing a credential to run the verification before saving. If verification fails, it will display an error message and then give you the option to **Save anyway**. **Save anyway** will save the most recent verification status.

Credential management

Users must have administrator-level permissions to manage credentials. Users with **Administrator** as their [default role](#) can fully manage all credentials. Users with [per-organization permissions](#) do not have access to global credentials, and are only able to manage credentials in organizations where they have administrator permissions. A shared credential cannot be deleted by an organization administrator.

Scanning with SNMP

SNMP is an open standard network protocol for collecting information about devices on a network.

There are three main versions of the protocol.

SNMP versions 1 and 2

SNMP version 1 was designed in the 1980s as an interim protocol, intended to be replaced by ISO CMIP. It was built to be used across any network common at the time, not just TCP/IP networks, so security was left up to the host network. The protocol defined a community string for arbitrary organization of groups of assets, but didn't specify how access should be granted.

SNMP version 2 attempted to introduce a security model based on "parties," but it wasn't widely adopted so a revised standard was issued as SNMP v2c. SNMP v2c removed party-based security and went back to just using community strings. The previous SNMP v2 specification is considered obsolete.

Unfortunately, both SNMP v1 and SNMP v2 send community strings in plain text across the network. There's no encryption of SNMP v1 or v2 data packets, and because the protocol is based on UDP there's no way to simply add TLS to make a secure connection.

SNMP version 3

SNMP v3 fixes the security problems of SNMP v2 by supporting both password-based authentication and encryption (referred to as privacy protection) as part of the standard. Unfortunately there are multiple algorithms supported for both the encryption and the password authentication, and you have to know which ones your network devices use. Unlike SSH for example, the protocol doesn't include any kind of negotiation of encryption methods.

SNMP v3 devices can operate in three different modes:

- noAuthNoPriv, meaning no authentication is required and there's no privacy protection (no encryption),
- authNoPriv, meaning authentication *is* required but there's still no privacy protection, and
- authPriv, meaning both authentication and privacy protection are required.

runZero's SNMP support

runZero supports SNMP v1, SNMP v2 (the common v2c variant), and SNMP v3. Scans can be performed using only v1/v2, only v3, or both.

SNMP scanning is on by default. You can turn it off or customize it using the SNMP tab when setting up a scan or a scan template.

SNMP v1 & v2 scanning

There are two ways to set up community strings for SNMP v1/v2 scanning. The first is to enter them as a comma-separated list on the SNMP tab. By default, runZero supplies the community strings `public` and `private`, as these are common defaults on network-enabled hardware such as printers and NAS servers. If you remove those defaults, runZero will not probe with them.

The second way to set up community strings is to enter them using the [credentials feature](#) as the credential type *SNMP v2 Communities*. While communities technically shouldn't be used as credentials because they're sent in plain text across the network, in practice many networks use them that way. By entering your community strings as credentials, you can use the CIDR allow list feature to control which parts of the network the community strings will be sent to, reducing the risk of their being captured by rogue devices.

If you remove all community strings from the SNMP tab and do not set up any SNMP v2 Communities as credentials, no SNMP v1/v2 scanning will be performed.

SNMP v3 scanning

Most devices which provide potentially sensitive information such as serial numbers and software versions will only do so in response to an *authenticated* SNMP v3 query. To perform authenticated SNMP v3 scanning, you will need to set up an *SNMP v3 Credential* record in runZero's credentials feature.

The authentication protocol determines the hashing algorithm used to process the authentication passphrase – that is, how the runZero Explorer logs in to the remote device.

The most common default algorithm that devices use for authentication is `sha` (HMAC-SHA-96), which is required to be supported by the SNMP v3 standards. Newer devices may support more secure variants such as SHA-256, and runZero supports up to SHA-512. HMAC-MD5-96 is also supported, as per the standards, but is best not used on your network because the MD5 algorithm is known to be insecure.

The privacy protocol determines how the data sent to and from the remote device is encrypted to prevent eavesdropping. The privacy passphrase is used as seed data to initialize the encryption.

The most common privacy algorithm, required by the original SNMP v3 standards, is CBC-DES-128. It's selected as `des` in the runZero user interface. Later RFCs added AES-128, represented as `aes`. Again, runZero supports more secure variants that may be used by newer devices, such as AES256.

As [mentioned above](#), devices may require only authentication, both authentication and privacy, or neither. For some devices there is only space for a single passphrase in the device configuration. This usually means that the same passphrase is used for both the privacy passphrase and the authentication passphrase, if the device is running in `authPriv` mode.

As with SNMP v2 community strings, you can set a CIDR allow list to determine which IP addresses SNMP v3 credentials will be sent to. This is particularly recommended if your

network is not set up to require privacy passphrases.

In addition to credentialed scanning, runZero will gather information from the pre-authentication handshake of SNMP v3.

To perform SNMP v3 scanning across Cisco switches, modify the ACLs to include the following rule:

- Newer IOS versions: `snmp-server group YourGroupName v3 auth context vlan- match prefix`
- Older IOS versions: `snmp-server group YourGroupName v3 auth context vlan-1` (repeated for every VLAN)

Choosing which versions of SNMP are scanned

To disable SNMP v1/v2 and only scan with SNMP v3, remove the community strings from the SNMP tab when setting up a scan or template and ensure that any SNMP v2 community strings stored as credentials are disabled on the Credentials tab.

To disable authenticated SNMP v3 scanning and only scan with SNMP v1/v2, ensure that any SNMP v3 credentials are disabled on the Credentials tab and make sure you have community strings specified, either on the SNMP tab or a credential.

To disable all SNMP scanning, including unauthenticated SNMP v3, switch off the toggle switch on the SNMP tab labeled *Use the SNMP protocol for discovery*.

How scans are performed

runZero will probe devices using all configured SNMP versions and all community strings. This is because devices often respond with different levels of data for SNMP v3 versus SNMP v2, and may respond with different information depending on community string.

SNMP is based on UDP, and UDP doesn't guarantee the order of data packets, so it's impossible to guarantee in what order community strings will be received by remote devices.

SNMP v3 authentication errors are not treated as scan failures. This is because it's common to have multiple sets of SNMP v3 credentials used on different parts of a network, as well as misconfigured devices with incorrect passphrases.

Checking SNMP v3 credentials and SNMP v1/v2 authentication

If you are not getting SNMP data in your assets and think that you should be, a common cause is network firewalls, which often block SNMP traffic entirely. If possible, deploy an Explorer on the other side of the firewall, on the network segment you want to scan with SNMP.

The second most common reason for not getting the expected SNMP data in assets is incorrect SNMP v3 credentials.

- If SNMP v3 authentication was attempted, you will see an *SNMP engine ID* recorded in the asset as `snmp.engineID.raw`. The engine ID is usually a long string of hex digits and is used to calculate access keys.

- If SNMP v3 authentication failed, you will see an error reported in the attribute `snmp.failedAuth`. If the error is a request timeout, that likely indicates that authentication succeeded but the encrypted data channel could not be set up, which means you may have the wrong privacy password or privacy algorithm.
- If SNMP authentication was attempted successfully (v1/v2 or v3), you will see an `snmp.credentials` attribute with the UUID of any credentials record that succeeded.
- If SNMP v1/v2 authentication via community string succeeded, you will see `snmp.secretCommunities` or `snmp.defaultCommunities` attributes, depending on the type of community string.
- If SNMP v1/v2 authentication succeeded using community strings entered directly into the task request rather than credentials records, the `snmp.secretCommunities` and/or `snmp.defaultCommunities` attributes will still be present, but `snmp.credentials` will be a zero UUID.

Debugging credential issues

It can be frustrating to debug SNMP v3 credential problems by trial and error using multiple scan attempts. Instead, you can use the `snmpwalk` utility. It's part of the [net-snmp](#) open source project, packaged for most Linux distributions and available for macOS via Homebrew. There are unofficial builds for Windows, or you can install it in the Windows Subsystem for Linux (WSL).

The `snmpwalk` utility will connect to an IP address and dump all of the information it can retrieve. Here's an example of how to use it to connect in `authPriv` mode with both authentication and privacy passphrases:

```
snmpwalk -v3 -l authPriv -a SHA -A "authentication passphrase"
-x AES -X "privacy passphrase" -u username 10.0.1.25
```

(Text above has been line-wrapped, but it's one long command.) The parameters are:

- `-v3`: switch to SNMP v3 mode.
- `-l authPriv`: run in `authPriv` mode, with both authentication and privacy required.
- `-a SHA`: authentication algorithm is SHA (SHA-128).
- `-A`: specify the authentication passphrase as the next argument.
- `-x AES`: privacy (encryption) algorithm is AES (AES-128).
- `-X`: specify the privacy passphrase as the next argument.
- `-u username`: specify the login username.
- `10.0.1.25`: the IP address to probe.

If you have the right passphrases and algorithms, you'll get data back, probably a lot of it. If not, you'll get an error message which may help identify the cause.

To check SNMP v2 community strings, you can use a command like this:

```
snmpwalk -v2c -c commstring 10.0.1.25
```

The `-v2c` argument chooses SNMP v2 (SNMP v2c) mode, and the `-c` argument specifies the community string. If you have the wrong community string, `snmpwalk` will run for a while and then print a timeout error.

Reviewing discovered SNMP services

When SNMP services are found during a scan, their protocol versions are tracked at the asset level where you will see `snmp` in addition to `snmp1`, `snmp2`, and `snmp3` depending on which version responded. This makes for performant queries via the [asset inventory](#), such as when querying for `protocol:snmp2`.

Additionally, SNMP services track how they authenticated and which protocols they used. If a commonly used value for a SNMP v2 community was used (such as `public`, `private`, some vendor defaults, and common values like `password`, `cisco`, and `community`), these will be listed in cleartext in the service details under `snmp.defaultCommunities` and asset details under `snmp.v2DefaultCommunities` with the list of communities that responded. If the SNMP v2 community is not a common value, this will be reported as `snmp.secretCommunities` with a value of `true` in the service details and in an attribute called `snmp.v2SecretCommunities` with a value of `true` in the asset details. If SNMP v3 is used, then `snmp.v3Usernames` will be populated at the asset level. Lastly, an asset attribute called `snmp.auth` is populated indicating whether `v2DefaultCommunity`, `v2SecretCommunity`, or `v3Username` was successful to authenticate to the asset.

Improving OS fingerprinting using Net-SNMP's extend feature

There are cases where the data provided by Net-SNMP on some Linux distributions is insufficient to correctly fingerprint the OS. An example of this is distributions based on Red Hat Enterprise Linux such as AlmaLinux OS, CentOS, Oracle Linux, and Rocky Linux. All of these return the exact same values for a given point release. In these cases outcomes can be improved by leveraging Net-SNMP's [extend mechanism](#) which executes a defined command and returns the value via SNMP. This requires that each target asset is using Net-SNMP as its SNMP software and modifying the SNMP configuration. The steps for doing so are:

1. Modify `/etc/snmp/snmpd.conf` and

1. Add the line:

```
extend os-release /bin/cat /etc/os-release
```

2. Grant the SNMP community string or user the runZero uses the rights to collect the data by adding a line similar to:

```
view systemonly included .1.3.6.1.4.1.8072.1.3.2
```

2. Restart the SNMP service:

```
sudo systemctl restart snmpd
```

3. Verify that the OID is accessible to the community string or account that the runZero task will be using.

```
snmpwalk -v2c -c commstring 10.0.1.25 .1.3.6.1.4.1.8072.1.3.2.3.1.2.10.111.115.45.114.101.108.101.97.115.101
```

Object Identifiers (OIDs)

While some of these OIDs are used only for specific vendors, this is a complete list of all OIDs potentially used on a given asset.

System attributes

- 1.3.6.1.2.1.1.*.0

IP addresses

- .1.3.6.1.2.1.4.20.1.3.*

MAC addresses

- .1.3.6.1.2.1.2.2.1.6.*

ARP caches

- .1.3.6.1.2.1.4.35.1.4.*
- .1.3.6.1.2.1.3.1.1.2.*
- .1.3.6.1.2.1.4.22.1.2.*

Routes

- .1.3.6.1.2.1.4.24.7.1.7.*

Port (CAM) tables

- .1.3.6.1.2.1.17.7.1.2.2.1.2.*
- .1.3.6.1.2.1.17.4.3.1.2.*

VLANs

- .1.3.6.1.2.1.17.7.1.4.3.1.1.*
- .1.3.6.1.4.1.9.9.46.1.3.1.1.4.1.*

Interface names

- .1.3.6.1.2.1.17.1.4.1.2.*
- .1.3.6.1.2.1.31.1.1.1.1.*

Serial numbers

- .1.3.6.1.2.1.47.1.1.1.1.11.*

Device models

- .1.3.6.1.2.1.47.1.1.1.1.13.*

Hostnames

- .1.3.6.1.4.1.9.2.1.3.*

Model names from Juniper switches

- .1.3.6.1.4.1.2636.3.1.2.0.*

Serial numbers from A10 devices

- .1.3.6.1.4.1.22610.2.4.1.6.2

OS release information from Net-SNMP's extend functionality (see above)

- .1.3.6.1.4.1.8072.1.3.2.3.1.2.10.111.115.45.114.101.108.101.97.115.101



Note that for Cisco Catalyst & Nexus switches, the explorer requires per-VLAN access to the ARP/FDB tables using either community indexing (v2) or contexts (v3). runZero automatically enumerates the ARP cache and port table for each discovered VLAN. Without this access, you may not see the `snmp.macs.ports` and `snmp.macs.vlans` attributes discovered for a switch and the switch may not appear on the Switch Topology report.

Please note that Cisco Catalyst & Nexus switches require per-VLAN access to the ARP/FDB tables using either community indexing (v2) or contexts (v3). runZero automatically enumerates the ARP cache and port table for each discovered VLAN.

runZero gathers information from SNMP v3 systems even when credentials are not available. This information includes the engineID as well as values from the following OIDs

- .1.3.6.1.6.3.15.1.1.1.0: `usmStatsUnsupportedSecLevels`
- .1.3.6.1.6.3.15.1.1.2.0: `usmStatsNotInTimeWindows`
- .1.3.6.1.6.3.15.1.1.3.0: `usmStatsUnknownUserNames`
- .1.3.6.1.6.3.15.1.1.4.0: `usmStatsUnknownEngineIDs`
- .1.3.6.1.6.3.15.1.1.5.0: `usmStatsWrongDigests`
- .1.3.6.1.6.3.15.1.1.6.0: `usmStatsDecryptionErrors`

Using custom fingerprints

Customers running a self-hosted instance or using the standalone scanner have the ability to use custom-written fingerprints. This can be useful in adding new fingerprint coverage for very unique or custom assets and services, such as device prototypes or proprietary applications/services. Custom fingerprints can also be used to override existing, similar runZero fingerprints by using a same-or-higher certainty value.



When using the runZero standalone scanner with custom fingerprints, you'll need to use the `RUNZERO_EXTERNAL_FINGERPRINTS` value as an environment variable when launching the scanner.

Create new fingerprints

Custom fingerprints follow the structure and format of the [open-source Recog fingerprint database](#). You can author your own fingerprint XML entries in files of similar name and format to [those found in Recog](#). For cases where an asset or service matches both a built-in runZero fingerprint and a custom fingerprint of the same kind, preference will be given to the fingerprint with higher "certainty" value(s) (e.g. `hw.certainty`, `os.certainty`, `service.certainty`). In the event of a certainty "tie" (i.e. same certainty value(s)), the custom fingerprint will be given preference.

Add new fingerprints to your self-hosted runZero instance

To ensure the self-hosted instance of runZero can properly access your custom fingerprints, they will need to exist within the runZero installation directory (`/opt/runzero` by default). The following steps will get your custom fingerprints setup for use by runZero:

- Create a new directory within the runZero install directory (e.g. `mkdir /opt/runzero/myfingerprints`)
- Update your `/etc/runzero/config` file with the new directory location (see below)
- Copy your custom fingerprint XML files into the new directory (e.g. `cp *.xml /opt/runzero/myfingerprints`)

The `/etc/runzero/config` file needs to be updated with the location of your custom fingerprints directory, which you can do by using your favorite editor to add the following line to the end of your config file (swapping `myfingerprints` with the name of the directory you added):

```
RUNZERO_EXTERNAL_FINGERPRINTS=myfingerprints
```

Note that you only need to do the directory creation and update of `/etc/runzero/config` file once. After that, you can add, remove, or modify your custom fingerprints in the directory as needed and then restart the runZero application to reload the current custom fingerprints.

Restart the runZero service

The runZero self-hosted instance will need to be [restarted](#) when custom fingerprints are added, removed, or updated. On restart, the runZero log file can be monitored to verify if the custom fingerprints were successfully applied or if an error occurred. On successful load of custom fingerprints, a log message like the following should be present:

```
@cee:{"level":"info","msg":"loaded (5) external fingerprints","time":"2022-09-12T19:51:49Z"}
```

If issues are encountered when loading or processing custom fingerprints, or if a 10 second timeout is reached, a warning message will be logged and the runZero application will continue running without any custom fingerprints.

Verify your fingerprints

Once your custom fingerprints have been added to your self-hosted runZero instance and the instance restarted, you can verify that the custom fingerprints are performing as-expected in one of the following ways:

- Running a scan task to go scan a relevant asset/service, or
- Importing an existing scan data file of the relevant asset/service

Following completion of the scan task OR the import of scan data, depending on which method you chose, you can then navigate to a [specific asset](#) or [specific service](#) and verify that your custom fingerprints are acting as-expected.

Passive sampling

runZero Explorers can identify assets, services, and vulnerabilities by passively monitoring network traffic. Passive sampling can be an alternative data source when active scanning is prohibited and helpful at identifying unknown IP ranges. Passive sampling tasks are limited to using a single CPU core and will temporarily skip packets when this limit is reached. If an active scan is run on an Explorer with a passive sampling task, the passive task will be interrupted, and then restarted once the active scan completes.

Configuring passive sampling

Passive sampling can be configured from the Explorer details screen. Choose *Deploy* from the left navigator and click on an Explorer. The network interfaces available for passive sampling will be shown on the left. Once enabled, the sampling task will process any available traffic received on these interfaces, including plain layer-2 frames from SPAN and TAP ports, encapsulated traffic using 802.1q (aka VLAN or QinQ), and layer-3 encapsulated traffic using GRE or VXLAN. While a full SPAN capture will provide the best results, any traffic, including broadcast, is useful for additional enrichment.

Passive sampling tasks will automatically enable promiscuous mode for the specified interfaces. Some virtual infrastructure, such as VMware ESXi, may require promiscuous mode to be enabled on the virtual switch before traffic is forwarded to the interface.

The *Discovery scope* specifies which IP ranges should be considered for asset, service, and vulnerability creation. Traffic that doesn't include a source or destination within an allowed range will be ignored. You can also specify *Excluded hosts* if there are IP ranges you want to skip.

The *Site* option specifies where assets should be created and updated. If the same IPs exist in other sites, this can lead to duplicate asset records.

You can set *Asset tags* for systems identified through passive traffic sampling. This can be helpful for reviewing the new assets and setting up scans to obtain more information about them.

Protocol support

aaa 4ai 5auth 4automotive 2backplane 20backup 1building-automation 5camera 1clear 56database 18deep 24directory 2discovery 57email 2encrypted 6file 9gaming 4integration 1iot 36legacy 10light 80media 2messaging 11mgmt 70mobile-core 12monitoring 8multicast 10naming 4network 1ot 56passive 14printing 9proxy 2remote-access 12routing 2security 6serial 1siem 1storage 8time 2tls 44voip 13vpn 7web 23

Filter by protocol, port, transp

TCP 227

UDP 105

TLS 1

SCTP 8

317 of 317 protocols

acop — Atlas Copco Open Protocol

TCP

Atlas Copco Open Protocol is a tightening-controller protocol used by Power Focus and Power MACS controllers on manufacturing assembly lines. Negotiates the protocol revision and queries controller identity, returning the supplier code, controller name, controller serial and software version, tool software version, and cell and channel identifiers.

Port: 4545

ot

acpp

TCP

activemq — Apache ActiveMQ OpenWire

TCP

Apache ActiveMQ OpenWire is the native binary wire protocol used by JMS clients to talk to ActiveMQ message brokers. Performs an OpenWire handshake and returns the broker version, host JVM and OS, and negotiated wire-format options.

Ports: 8161, 61616, 61617

messaging

adb — Android Debug Bridge

TCP

Android Debug Bridge is a developer protocol used to debug, install, and control Android devices over the network (typically on rooted phones, TVs, and IoT devices). Probes for an open ADB endpoint and returns the device's access state and ADB banner.

Ports: 5037, 5555

mgmtremote-access

ads — Beckhoff ADS/AMS

TCPUDP

Beckhoff ADS/AMS is the Automation Device Specification / Automation Message Specification protocol used to communicate with TwinCAT runtimes on Beckhoff PLCs and industrial PCs. Issues a ReadDeviceInfo request and returns the device name, AMS NetID, and TwinCAT runtime version.

Port: 48898

ot

advantech-adam — Advantech ADAM/EKI Config

TCPUDP

Advantech ADAM/EKI is the proprietary device-configuration service (5058/tcp and 5058/udp) served by Advantech EKI serial device servers and ADAM I/O modules. runZero sends the read-only 0x4102 network-info request (magic 4D 41 44 41, broadcast target,

empty token) and parses the reply for the device MAC, model, firmware version, and IP configuration. The read opcode is unauthenticated and has no side effects.

Port: 5058

discoverylightot

airplay — Apple AirPlay

TCP

AirPlay is Apple's wireless streaming protocol used by iOS, macOS, and Apple TV devices to mirror displays and stream audio/video. Detected from the `_airplay._tcp` and `_raop._tcp` mDNS records and HTTP banners advertised by AirPlay receivers.

discoveryiot

ajp — Apache JServ Protocol (AJP)

TCP

Apache JServ Protocol (AJP) is a binary protocol used by reverse-proxy front-ends (Apache `httpd mod_proxy_ajp`, `nginx`, `IIS`) to bridge HTTP requests to a back-end Tomcat or JBoss application server, typically on TCP/8009. runZero attributes services as AJP from external integration data (Shodan) and from banner / port hints; no active AJP probe is sent.

clearweb

amqp

TCP

AMQP is the Advanced Message Queuing Protocol used by RabbitMQ and other brokers for message-oriented middleware. Exchanges the protocol-header preamble and returns the negotiated AMQP version, broker product and version, runtime platform, cluster name, advertised SASL mechanisms, and TLS requirement.

Ports: 5671, 5672

messagingtls

ard — Apple Remote Desktop

UDP

Apple Remote Desktop is a discovery protocol used by macOS systems to advertise themselves to Apple Remote Desktop administrators on UDP/3283. runZero sends the ARD discovery request, validates the response type, and returns the advertised hostname and Apple machine model from the reply.

Port: 3283

discoverylightremote-access

arp — Address Resolution Protocol

TCP

Address Resolution Protocol is a Layer-2 protocol used to resolve IPv4 addresses to Ethernet MAC addresses on the local broadcast segment. runZero sends ARP requests for each target on the local network to discover live hosts, captures the MAC address from each reply for asset attribution and OUI-based vendor identification, and passively records ARP traffic observed during the scan.

discoverymulticastpassive

atg — Automatic Tank Gauge

TCP

Automatic Tank Gauge is a Veeder-Root serial protocol tunneled over TCP (commonly port 10001) used by ATG consoles such as the TLS-3xx and TLS-4xx series found in fuel-station forecourts. Issues the I20100 in-tank inventory and I90200 system-revision commands and returns the station name, console software and module revisions, tank count, and per-tank product names.

Port: 10001

ot

backupexec

TCP

bacnet — BACnet/IP

UDP

BACnet/IP is a building-automation protocol used for HVAC, lighting, access control, and other building systems. Issues Who-Is and ReadProperty requests and returns the device instance, vendor, model, firmware, and a summary of objects and routing devices behind the gateway.

Ports: 46808, 47808, 47809, 47810, 47811, 47812, 47813, 47814, 47815, 47816, 47817, 47818, 47819, 47820, 47821, 47822, 47823, 47824, 48808

backplanebuilding-automationdeepot

bedrock — Minecraft Bedrock

UDP

Minecraft Bedrock is the game-server query and discovery protocol used by Minecraft Bedrock Edition clients. Sends an Unconnected Ping and returns the server uptime, GUID, and the raw advertisement payload (MOTD and version fields).

Port: 19132

gaminglight

bgp — Border Gateway Protocol

TCP

Border Gateway Protocol is the inter-domain routing protocol used between autonomous systems on the Internet and inside large networks. Attempts an OPEN exchange and returns the BGP version, advertised AS number, BGP identifier, and supported capabilities.

Port: 179

routing

bitdefender-app — Bitdefender Endpoint

TCP

Bitdefender Endpoint is a proprietary check-in channel exposed by Bitdefender, NETGEAR Armor, and ThreatTrack mobile-security and VPN agents on Android and iOS endpoints.

runZero passively matches the agent's JSON app_id payload from observed banners and reports the Bitdefender vendor along with the application identifier and software version of each detected product.

Port: 7519

lightmgmtsecurity

bjnp — Canon BJNP

UDP

Canon BJNP is a vendor-specific printing and scanning protocol used by Canon's network drivers. Sends a BJNP discovery probe and returns the device type (printer or scanner), MAC address, and IPv4/IPv6 address advertised in the reply.

Ports: 8611, 8612

discoveryprinting

brother — Brother Network Discovery

UDP

Brother Network Discovery is a UDP-based broadcast protocol used by Brother printers and multi-function devices to announce themselves and respond to driver/management discovery probes. runZero parses the response to recover the printer model, firmware revision, and serial number.

discoveryiot

brother-scanner — Brother Network Discovery

TCP

Brother Network Discovery is the scanner protocol used by Brother network devices. Identifies the service from its +OK 200 / -NG 401 reply and tags the asset as a Brother scanner.

Ports: 54921, 54922, 54923

discoverylightprinting

bsap-ip — Emerson BSAP/IP

UDP

Emerson BSAP/IP is the Bristol Standard Asynchronous Protocol over IP used by Emerson ControlWave and Bristol RTUs in oil & gas SCADA. Queries node identification and, when enabled, walks the BSAP local-address hierarchy to enumerate child RTUs, returning RTU identity, firmware, and topology summaries.

Ports: 1234, 1235

backplanedeepot

c12.22 — ANSI C12.22

TCP

ANSI C12.22 is the metering-network transport used by electric-utility advanced-metering-infrastructure head-ends and relays. Issues an EPSEM Identification request and returns the device serial number, ED class, and C12.22 standard version.

Port: 1153

backplanedeepot

c37118 — IEEE C37.118 Synchrophasor

TCP

IEEE C37.118 Synchrophasor is the streaming protocol used by Phasor Measurement Units and Phasor Data Concentrators on the electric grid. Requests CFG-2 and header frames and returns PMU/PDC identification, station and channel naming, and reporting rate.

Port: 4712

backplanedeepot

cacti — Cacti Network Monitoring Web UI

TCP

Cacti Network Monitoring Web UI is the open-source RRDtool-based network graphing and monitoring console deployed by network and IT operations teams. The HTTP extractor matches the Cacti login page title, parses the embedded versionInfo string, and returns the cacti.version attribute and Cacti software identification.

lightttsweb

cassandra — Apache Cassandra CQL

TCP

Apache Cassandra CQL is the native client protocol used to query the Cassandra wide-column store. Performs a STARTUP/OPTIONS exchange and returns the cluster name, CQL version, and Cassandra release.

Ports: 9042, 9160

databasetls

cdp — Cisco Discovery Protocol

TCPUDP

Cisco Discovery Protocol is a Layer-2 announcement protocol used by Cisco devices to advertise themselves to neighbors. Passively decodes CDP frames and returns the device ID, software version, platform, capabilities, native VLAN, and management addresses advertised by the neighbor.

discoverymgmtpassive

ceph

TCP

Ceph is a cluster-internal messenger protocol used by Ceph distributed storage daemons (mon, mgr, osd, mds) to communicate. runZero identifies Ceph services from the "ceph v" banner returned on connection and tags the asset as a Ceph storage node.

storage

chargen — Character Generator

TCP

Character Generator is a legacy diagnostic service (RFC 864) that emits a stream of printable characters. Records the response banner to confirm the service.

legacylight

checkmk — Checkmk Agent

TCP

Checkmk Agent is the host-side metric exporter used by the Checkmk server to collect host metrics. Reads the agent banner and returns the agent version, build date, host operating system, architecture, and configured hostname.

Port: 6556

mgmtmonitoring

chromecast — Google Chromecast

TCP

Chromecast is the Google Cast device discovery and control protocol used by Chromecast, Google TV, Google Nest Hub, and Cast-enabled speakers. Detected via the _googlecast._tcp mDNS record and the device's HTTP /setup/eureka_info endpoint, which exposes the device name, model, build, and timezone.

discoveryiot

cip — Common Industrial Protocol

TCPUDP

Common Industrial Protocol is an industrial automation protocol used by Rockwell/Allen-Bradley and other vendors for PLC and I/O communication. Issues the List Identity and Get Attributes All requests and returns vendor, product code, revision, serial, product name, and device-type information for the controller and any backplane modules.

Port: 44818

backplanedeepot

cisco-phone — Cisco IP Phone Web Interface

TCP

Cisco IP Phone Web Interface is the embedded HTTP server exposed by Cisco SPA, 7900-series, 8800-series, and Unified IP Phones. runZero attributes services as cisco-phone from Recog banner matches and uses the response to recover the model, firmware version, MAC address, and call-manager configuration.

clearvoip

ciscosmi — Cisco Smart Install

TCP

Cisco Smart Install is a zero-touch deployment protocol commonly abused for unauthenticated configuration access. Sends the Smart Install probe, tags the asset as Cisco IOS, and records the raw response payload.

Port: 4786

mgmt

citrix — Citrix ICA Browser

UDP

Citrix ICA Browser is a UDP/1604 service used by Citrix XenApp and Virtual Apps and Desktops clients to locate published applications and server farms. runZero sends an ICA

Browser request and parses the reply to return the server-farm name and the list of advertised published applications.

Port: 1604

discoveryremote-access

cldap — Connectionless LDAP

UDP

Connectionless LDAP is a UDP-based directory protocol used to query directory servers without setting up a TCP connection. Issues a rootDSE search and returns the LDAP attributes parsed from the reply (vendor name and version, supported controls, extensions, capabilities, and SASL mechanisms).

Port: 389

directorydiscovery

click — Click Modular Router

TCP

Click Modular Router is the control socket exposed by hosts running the Click software router. Identifies the service from the Click::ControlSocket banner and tags the asset accordingly.

Port: 7734

lightmgmt

coap

UDP

CoAP is the Constrained Application Protocol (RFC 7252) used by constrained devices and IoT deployments for resource-oriented messaging. Issues a GET on /.well-known/core and returns the CoAP version, message type, response code, options, content format, and the resource list or payload from the reply.

Ports: 5683, 5684

iot

cockpit — Cockpit Linux Web Console

TCP

Cockpit Linux Web Console is the web-based Linux server-management console shipped with Red Hat, Fedora, CentOS Stream, and Debian, typically served over TLS on TCP/9090. The HTTP extractor matches the embedded environment JSON in the login page and returns the host name, OS pretty name, and OS variant identifiers.

lighttlsweb

codesys — CODESYS V3 Runtime

TCP

CODESYS V3 Runtime is the IEC 61131-3 controller runtime used by many OEMs (WAGO, Beckhoff, Schneider, Eaton, ...). Issues the runtime identification request and returns the runtime vendor, product, version, and target identification.

Ports: 1200, 1217, 2455, 11740

ot

codesys2 — CODESYS V2 Runtime

TCP

CODESYS V2 Runtime is the older 3S CODESYS V2 controller runtime used by industrial controllers from many OEMs. Performs the V2 login probe and returns the runtime version and target identification.

Ports: 1200, 2455

ot

cognex — Cognex In-Sight

TCP

Cognex In-Sight is an operator-access service exposed by Cognex In-Sight industrial machine-vision cameras for configuration and monitoring on factory networks. runZero identifies In-Sight cameras from the proprietary banner returned on connection and tags the asset as a Cognex vision system.

lightot

common-socket-connection — Raritan Common Socket Connection

TCP

Raritan Common Socket Connection is a proprietary management transport used by Raritan KVM-over-IP switches, Dominion serial consoles, and rack PDUs, typically on TCP/5000. runZero identifies CSC services from the Raritan banner returned on connection and tags the asset as a Raritan management device.

clearlightmgmt

companion-link — Apple CompanionLink

TCP

Apple CompanionLink is a discovery and pairing service used by Apple devices (iOS, macOS, tvOS, HomePod) to negotiate AirPlay, HomeKit, and Continuity sessions with paired peers, advertised via `_companion-link._tcp` on Bonjour. runZero attributes the service from the mDNS / port hint and reports the asset as Apple CompanionLink.

iotlight

control — Control Device Discovery

TCP

Control Device Discovery is the broadcast protocol used by Control/Pepperl+Fuchs RocketLinux switches and DeviceMaster serial servers to advertise their presence on the network. runZero parses the response to recover the model name, hardware/firmware revision, MAC address, IP configuration, and serial number.

discoveryot

confluence — Atlassian Confluence

TCP

Atlassian Confluence is a wiki and team-collaboration server available in Server, Data Center, and Cloud editions. The HTTP probe fetches Confluence login, dashboard, and version endpoints and returns the product name, edition, and build number.

lightttsweb

consul — HashiCorp Consul

TCP

Consul is HashiCorp's service-mesh and key/value store. Detected on the Consul HTTP API (TCP/8500), where the `/v1/status/leader` and `/v1/agent/self` endpoints disclose the datacenter, node name, build version, and Raft leader of the cluster.

Port: 8500

clearmgmt

couchdb — Apache CouchDB

TCP

CouchDB is the Apache document-oriented database fronted by an HTTP/JSON API. Detected from the welcome document at `/`, which discloses the CouchDB version, vendor name, UUID, and `--` on misconfigured deployments `--` whether the admin party is enabled.

Port: 5984

clearmgmt

crestron — Crestron Discovery

UDP

Crestron Discovery is a vendor protocol used to locate Crestron control processors and AV equipment. Sends the discovery probe and returns the device hostname, model, and firmware

version.

Port: 41794

building-automationdiscoverylight

crimsonv3 — Red Lion Crimson 3

TCP

Red Lion Crimson 3 is a configuration and runtime protocol used to read data from Red Lion Graphite, DA-series, and other industrial HMIs and gateways. Reads the manufacturer (register 0x012b) and model (register 0x012a) registers and returns those strings.

Port: 789

ot

crowd — Atlassian Crowd

TCP

Atlassian Crowd is a centralized SSO and identity-management product. runZero attributes services as Crowd from Recog matches against the application's HTTP banners and login pages, recovering the product version and build information.

clearmngmt

cspv4 — Allen-Bradley CSPv4 / PCCC

TCP

Allen-Bradley CSPv4 / PCCC is a legacy controller protocol used by SLC 5/05 and MicroLogix PLCs for register access. Issues a PCCC identify request and returns the controller family, processor type, and series/revision string.

Port: 2222

backplanedeepot

cups — Common Unix Printing System

TCP

CUPS is the Common Unix Printing System administrative web interface. runZero matches the CUPS HTTP banner and the IPP server header, returning the cupsd version and host operating system.

Port: 631

cleariot

dahua-dhip — Dahua DHIP

UDP

Dahua DHIP is a proprietary discovery and management protocol used by Dahua and OEM-rebranded IP cameras and NVRs (Amcrest, Lorex, and similar). Sends the single-byte DHIP discovery probe and returns the device serial, machine name, vendor, firmware version, MAC, and IPv4/IPv6 configuration.

Port: 37810

discoveryiotlight

daytime

TCPUDP

Daytime is a legacy diagnostic service (RFC 867) that returns the current date and time. Records the response banner, parses the timestamp, and infers an OS hint from the format.

Port: 13

legacylight

db2 — IBM Db2

TCPUDP

IBM Db2 is a relational database protocol used by Db2 LUW and Db2 for z/OS clients. Performs a DRDA EXCSAT/ACCSEC exchange and returns the server product identifier, version, and platform.

Port: 523

database

dcerpc — DCE/RPC Endpoint Mapper

TCP

DCE/RPC Endpoint Mapper is the RPC locator used to enumerate RPC services on Windows hosts. Queries the endpoint mapper and returns a summary of registered RPC interfaces, their UUIDs, versions, and bindings.

Ports: 135, 593

mgmt

dhcp

UDP

DHCP is the Dynamic Host Configuration Protocol used to lease IPv4 addresses and network configuration. Sends a DHCPDISCOVER and returns the offered server identifier, lease parameters, and any vendor-class identification revealed by the response.

Ports: 67, 68

discovery

diameter — Diameter (TCP)

TCP

Diameter (TCP) is an authentication, authorization, and accounting protocol over TCP; the successor to RADIUS, widely used in mobile-core networks. Sends a Capabilities-Exchange-Request and returns the origin host, realm, vendor, product name, and supported applications.

Port: 3868

aaamobile-coretls

diametersctp — Diameter (SCTP)

SCTPTCP

Diameter (SCTP) is the Diameter (RFC 6733) AAA protocol carried over SCTP, used between mobile-core elements (Diameter Edge Agent, DRA, HSS, PCRF, MME). Sends a Capabilities-Exchange-Request and returns the origin host, realm, vendor, product name, and supported applications.

Port: 3868

aaamobile-coretls

digi — Digi ADDP

UDP

Digi ADDP is the Advanced Device Discovery Protocol used by Digi International serial servers, cellular gateways (TransPort, IX, EX), and embedded modules to advertise themselves on the local network. runZero sends DIGI, DVKT, and DGDP discovery requests on UDP/2362 and parses the TLV reply to return the device MAC, IP, model, firmware version, and hardware revision.

Port: 2362

discoveryiotlight

digi-realport — Digi RealPort

TCP

Digi RealPort is a serial-over-TCP service exposed by Digi serial server appliances. Sends a RealPort hello and parses the product name and exposed serial-port count from the handshake reply.

Ports: 771, 1027

iotlightserial

dnp3

TCP

DNP3 is Distributed Network Protocol 3 (IEEE 1815) used in electric, water, and oil & gas SCADA between control centers (masters) and RTUs/IEDs (outstations). Performs an

unsolicited link-layer test and an Object Group 0 read and returns the outstation address, vendor, model, firmware, and device-attributes summary.

Port: 20000

backplanedeepot

dns

TCPUDP

DNS is the Domain Name System used to resolve hostnames to addresses and other resource records. Issues version.bind, hostname.bind, and recursion-test queries and returns the resolver software identification, recursion availability, and observed CHAOS-class metadata.

Ports: 53, 5353, 5355

naming

docker — Docker Engine API

TCP

Docker Engine API is the HTTP control plane exposed by dockerd. runZero queries /version and /info on unauthenticated daemons (typically TCP/2375 or 2376) to recover the engine version, API version, kernel version, operating system, architecture, and container runtime.

Ports: 2375, 2376

clearmgmt

doip — Diagnostics over IP

TCPUDP

Diagnostics over IP is an automotive diagnostics protocol used to reach in-vehicle ECUs (UDS/KWP) over Ethernet. Issues a Vehicle Identification Request and (when enabled) Entity Status, returning VIN, EID/GID, logical addresses, and reachable ECU summaries behind the gateway.

Port: 13400

automotivebackplanedeepot

dotnet-remoting — .NET Remoting

TCP

.NET Remoting is a Microsoft RPC framework used by legacy .NET Framework applications for cross-process and cross-host RPC. Identifies the service from the .NET Remoting binary-protocol prefix in the connection banner and saves the raw banner.

Port: 9090

mgmt

drbd

TCP

DRBD is the Distributed Replicated Block Device protocol used to replicate block devices between Linux nodes for high availability. Identifies the service from the connection-error banner observed on TCP/8787 and saves the raw banner.

Port: 8787

lightstorage

drobo-nasd — Drobo NASd

TCP

Drobo NASd is the management daemon used by Drobo Dashboard to administer Drobo storage appliances. Identifies the daemon from the DRINASD banner returned on TCP/5000 and saves the raw banner.

Port: 5000

lightmgmtstorage

dtls

UDP

DTLS is Datagram Transport Layer Security (RFC 6347/9147), the UDP/SCTP variant of TLS, used by WebRTC, CoAP, OpenVPN, EAP-TTLS, and other datagram services. Performs a DTLS ClientHello and returns the negotiated version, cipher suite, and any presented certificate metadata.

Ports: 443, 3391, 4433, 5246, 5349, 5684, 12346, 12366, 12386, 12406, 12426

encrypted

echo

TCPUDP

Echo is a legacy diagnostic service (RFC 862) that echoes received bytes. Records the response to confirm the service and to detect amplification-capable hosts.

Port: 7

legacylight

eero-ebid — eero EBID

UDP

eero EBID is a proprietary discovery protocol used by Amazon eero mesh Wi-Fi access points to advertise the extender beacon identifier between mesh nodes on the local segment.

runZero attributes the service from the EBID hint and applies eero-specific fingerprinting to identify the asset as an eero mesh extender.

discoveryiotpassive

eerogw — eero Gateway

UDP

eero Gateway is a proprietary discovery protocol used by Amazon eero mesh gateways to advertise themselves to companion mobile applications on the local network. runZero attributes the service from the eero gateway hint and applies eero-specific fingerprinting to identify the asset as an eero gateway node.

discoveryiotpassive

elasticsearch

TCP

Elasticsearch is the Elastic search/analytics engine. runZero queries the root HTTP endpoint to recover the cluster name, node name, Elasticsearch version, Lucene version, and build hash.

Port: 9200

clearmgmt

epm — DCE/RPC Endpoint Mapper Service

TCP

EPM is the surface name used by runZero for services attributed to the Microsoft DCE/RPC Endpoint Mapper following Recog/banner-based fingerprinting. The lower-level wire protocol is decoded as dcerpc; this label captures hosts where only fingerprint evidence (banners, version strings) was available.

Port: 135

clearmgmt

epmd — Erlang Port Mapper Daemon

TCP

EPMD is the Erlang Port Mapper Daemon used by Erlang and Elixir distributed nodes (including RabbitMQ and CouchDB) to advertise registered node names and the dynamic ports they listen on. runZero issues the NAMES_REQ to enumerate registered nodes and their listening ports.

Port: 4369

clearmgmt

epo — Trellix/McAfee ePolicy Orchestrator (ePO)

TCP

Trellix/McAfee ePolicy Orchestrator (ePO) is the central management console for Trellix (formerly McAfee) endpoint-security agents. The HTTP probe fetches the ePO login page and returns the product name, build number, and version metadata exposed in the page markup.

lightsecuritytlsweb

erlangdp — Erlang Distribution

TCP

Erlang Distribution is the inter-node messaging protocol used between Erlang and Elixir nodes. Queries EPMD for registered node names and (when nodes are discovered) performs a distribution handshake on the first node, returning the EPMD names list, node name and hostname, distribution version, supported flags, and handshake status.

Port: 4369

messaging

erldp — Erlang Distribution Protocol

TCP

Erlang Distribution Protocol is the distribution protocol used between Erlang and Elixir VM nodes for inter-node messaging (also commonly referred to as ErlDP). runZero matches the Erlang distribution handshake on the wire and reports the protocol alongside the active erlangdp scanner so port-scan results are tagged consistently with EPMD-discovered nodes.

clearmessaging

etcd — etcd v3 API

TCP

etcd is the distributed key-value store used by Kubernetes and other CoreOS-derived projects. Detected via the v3 HTTP/gRPC API (typically TCP/2379), where /version reports the etcd-server and etcd-cluster versions.

Port: 2379

clearmgmt

etcd2 — etcd v2 API

TCP

etcd2 is the legacy v2 HTTP API exposed by older etcd deployments. Detected via /v2/stats/self and /version, which expose the cluster name, member ID, and etcd version on unauthenticated installations.

Ports: 2379, 4001

clearmgmt

ethercat

UDP

EtherCAT (Ethernet for Control Automation Technology, IEC 61158) fieldbus used for high-speed motion control and distributed I/O on machine-control and CNC segments. Queries master and slave registers and returns the master vendor identification and a summary of slaves discovered on the segment.

Port: 34980

backplanedeepot

fgfm — FortiGate to FortiManager

TCP

FGFM is the proprietary TLS-wrapped management protocol used by Fortinet FortiGate firewalls to register with and receive configuration from a FortiManager. Detected by the FGFM TLS server certificate and banner; runZero records the device serial number and model where exposed.

Port: 541

clearmgmt

finger

TCP

Finger is a legacy user-information service (RFC 1288) historically exposed on TCP/79, today seen mostly on Cisco IOS devices and embedded printers. runZero reads the Finger banner returned on connection and extracts the printer model (HP JetDirect-style) or Cisco IOS identification, and reports the service for legacy-protocol exposure tracking.

legacy

fins — Omron FINS

UDP

Omron FINS is the Factory Interface Network Service used by Omron CJ, CS, NJ, and NX PLCs and related automation devices on factory floors. runZero records FINS-derived asset attributes (controller model, firmware) emitted by the active omronfins scanner and uses the FINS protocol identifier when categorizing assets and assigning OT asset functions.

Port: 9600

ot

firebird — Firebird SQL

TCP

Firebird SQL is the relational database wire protocol used by the Firebird open-source database engine. Performs the Firebird connection handshake and returns the server architecture, protocol version, and Firebird release.

Port: 3050

database

focas — Fanuc FOCAS

TCP

Fanuc FOCAS is an Ethernet protocol used to monitor and control Fanuc CNC machine tools and robots (Open CNC API Specification, FOCAS2/Ethernet). Issues the system-info call and returns the CNC series, version, machine number, and (when enabled) per-path machining data.

Port: 8193

backplanedeepot

fortigate-to-fortimanager

TCP

FortiGate-to-FortiManager is the Fortinet FGFM management protocol used by FortiManager to manage FortiGate firewalls. Inspects the FGFM TLS handshake and returns the FortiGate model, firmware, and serial number embedded in the certificate.

Port: 541

mgmtsecurity

fox — Tridium Niagara Fox

TCP

Tridium Niagara Fox is the building-automation control protocol used by Niagara Framework JACE controllers and supervisors. Sends the Fox hello and returns the Fox version, station name, host ID, host name, OS name and version, JVM name and version, brand identifier, and authentication agent.

Ports: 1911, 4911

building-automationot

ftp

TCP

FTP is the standardized file-transfer protocol (RFC 959). Reads the FTP greeting and issues SYST/HELP/AUTH probes, returning the server software, system type, supported features, and TLS-availability indicators.

Ports: 21, 2121, 9090

clearfiletls

ganglia

TCP

Ganglia is a distributed monitoring system commonly deployed on HPC clusters and Linux server farms. runZero identifies Ganglia services from the GANGLIA_XML document returned by gmond/gmetad and captures the banner so cluster identification and host metrics are available for inventory.

monitoring

gesrtp — GE SRTP

TCP

GE SRTP is the Service Request Transport Protocol used to communicate with GE/Emerson PACSystems, Series 90, and RX3i/RX7i PLCs. Issues a controller-identification request and returns the model, firmware, sweep state, and slot configuration.

Port: 18245

ot

giop — GIOP / CORBA IIOP

TCP

GIOP / CORBA IIOP is the OMG General Inter-ORB Protocol (the wire format under CORBA IIOP). Identifies the service from the GIOP magic in the connection banner.

Port: 535

mgmt

git — Git Smart Protocol

TCP

Git Smart Protocol is the native transport used by git:// servers for clones, fetches, and pushes. Sends an upload-pack advertisement request and returns the advertised refs summary, server capabilities, and HEAD reference.

Port: 9418

file

git-http — Git Smart HTTP Service

TCP

Git Smart HTTP Service is the Git smart-HTTP transport (git-upload-pack and git-receive-pack endpoints) used by GitLab, Gitea, Bitbucket, cgit, and bare git-http servers for clones, fetches, and pushes. The HTTP extractor matches the _gitlab_session cookie and parses the manifest body, returning the git-http protocol attribution and the GitLab manifest hash.

lighttlswsweb

googlewifi — Google Wifi / Nest Wifi

TCP

Google Wifi is the local management API exposed by Google Wifi and Nest Wifi mesh access points. Detected via mDNS (_googlecast._tcp) and the local HTTP setup endpoints, which reveal the device's hardware model, build version, and mesh role.

discoveryiot

gpsd

TCP

GPSD is the GPS daemon JSON-over-TCP protocol used to share location and timing data from connected GNSS receivers. Sends a ?WATCH request and identifies the service from the GPSD banner returned in the response.

Port: 2947

iot

gtpc — GTP-C

UDP

GTP-C is the GPRS Tunneling Protocol control plane that carries signaling between mobile-core nodes (SGSN/GGSN, MME/SGW/PGW). Sends an Echo Request and returns the GTP version, restart counter, and supported features.

Ports: 2123, 2152

mobile-core

gtpprime — GTP'

UDP

GTP' is the GTP charging variant used to ship CDRs from mobile network elements to a Charging Gateway. Sends an Echo Request and returns the GTP' version and node identification.

Port: 3386

mobile-core

gtpu — GTP-U

UDP

GTP-U is the GPRS Tunneling Protocol user plane that encapsulates subscriber traffic between mobile-core nodes and base stations. Sends an Echo Request and returns the GTP-U version and observed extension-header support.

Port: 2152

mobile-core

gvcp — GigE Vision Control

UDP

GigE Vision Control is the AIA GVCP protocol used to discover, configure, and trigger industrial machine-vision cameras over Ethernet. Sends a Discovery_Cmd and returns the camera manufacturer, model, serial, firmware, MAC, and supported GVCP version.

Port: 3956

discoveryot

gvsp — GigE Vision Streaming

UDP

GigE Vision Streaming is the AIA GVSP protocol used to transport image and chunk data from machine-vision cameras to host applications. Passively classifies stream packets and returns the streaming state, packet format, and block identifier.

Port: 20202

otpassive

hartip — HART-IP

TCP

HART-IP is an industrial protocol used to tunnel HART process-instrument traffic over TCP/UDP through gateways and multiplexers. Issues HART command 0 and returns the gateway identification, and (when enabled) walks sub-device indices via Cmd 84 to enumerate connected field instruments.

Port: 5094

backplanedeepot

hicp — HMS HICP/SHICP

UDP

HMS HICP/SHICP is the HMS Industrial Networks discovery protocol used to discover and configure Anybus and Netbiter industrial gateways. Sends the HICP discovery probe and returns the device hostname, MAC, IP configuration, and firmware revision.

Port: 3250

discoveryot

hiddiscoveryd — HID DiscoveryD

UDP

HID DiscoveryD is the discovery service used to locate HID VertX/Edge access-control panels and readers. Sends the discovery probe and returns the device model, firmware, and primary network configuration.

Port: 4070

building-automationdiscoveryiotlight

hikvision — Hikvision IP Camera/NVR Web

TCP

Hikvision IP Camera/NVR Web is the HTTP management interface for Hikvision (and OEM) IP cameras and recorders. Identifies the product family from the WWW-Authenticate realm and pins the firmware version using either the embedded ?version= query strings on the login page assets, or the Last-Modified header on /doc/page/login.asp combined with a known build-date table.

cameraiotlightweb

hostmeta — Web Host Metadata

TCP

Web Host Metadata is the document defined by RFC 6415 and exposed at /.well-known/host-meta or host-meta.json, commonly used by federated-identity, WebFinger, and ActivityPub deployments. runZero sets the hostmeta protocol on the asset when the active HTTP probe successfully retrieves the document and surfaces the discovered links for inventory.

lightweb

hsms — SEMI HSMS / SECS-GEM

TCP

SEMI HSMS / SECS-GEM is the High-Speed SECS Message Services transport that carries SECS-II/GEM messages for semiconductor fab equipment (SEMI E37). Performs the HSMS Select handshake and an S1F1 Are-You-There, returning the equipment model identifier, software revision, and (when enabled) Equipment Constant subsystem summary.

Port: 5000

backplanedeepot

http

TCP

HTTP is the Hypertext Transfer Protocol used by the World Wide Web and most modern APIs. Issues HEAD/GET probes and runs HTTP-specific extractors, returning server software, response codes and headers, page titles and generators, favicons, and any application fingerprints recognized by extractor rules.

Ports: 80, 3000, 4567, 5000, 5985, 8000, 8001, 8080, 8081, 8082, 8200, 8443, 8888, 9001, 9080, 9090, 9100

tlsweb

http2 — HTTP/2

TCP

HTTP/2 is a binary, multiplexed framing protocol for HTTP (RFC 7540), negotiated via TLS ALPN or the HTTP/2 cleartext upgrade. runZero negotiates HTTP/2 when offered and feeds the response into the standard HTTP analyzer so headers, server identification, and TLS attributes are captured.

tlsweb

https

TCP

HTTPS is HTTP carried over TLS, the dominant transport for web applications, REST APIs, and management UIs. Implemented in runZero as the standard HTTP scanner over a TLS connection; the protocol is reported as https (rather than http) whenever the connection

negotiates TLS or the port hint is flagged as TLS-only, and the full TLS handshake metadata (certificate, ciphers, fingerprints) is recorded alongside the HTTP response.

encryptedtlsweb

iax2

UDP

IAX2 is the Inter-Asterisk eXchange version 2 VoIP signaling and media protocol used between Asterisk PBXes. Sends a POKE and returns the responder's IAX2 version and PBX identification.

Port: 4569

voip

icmp — ICMP Echo (Ping)

TCP

ICMP Echo (Ping) is the IPv4/IPv6 ICMP Echo Request/Reply exchange (RFC 792, RFC 4443) issued by the host-discovery probe to confirm host liveness, capture round-trip times, and observe TTL/Hop-Limit and IP TOS fields.

discoverylight

ics-trace — ICS Passive Trace

TCP

ICS Passive Trace is a synthetic protocol used internally to record evidence from passive analysis of ICS/OT traffic when active OT probing is disabled. runZero attributes the asset under ics-trace and emits the protocol identifiers and attributes observed in the passive trace.

otpassive

ident

TCP

Ident is the Identification Protocol (RFC 1413), a legacy user-identity lookup service. Sends an ident query against the connecting socket and returns the operating-system field and any user-identity string disclosed by the response.

Port: 113

legacylight

idrac — Dell iDRAC

TCP

iDRAC is the Dell Integrated Dell Remote Access Controller out-of-band management interface. runZero attributes services as iDRAC from the SSH/HTTPS banners, redfish endpoints, and TLS certificates issued for the controller, recovering the firmware version and service tag.

clearmgmt

iec104 — IEC 60870-5-104

TCP

IEC 60870-5-104 is a SCADA telecontrol protocol used between control centers and RTUs/substation gateways, primarily in electric power and rail. Sends TESTFR (and, when enabled, STARTDT/General Interrogation) and returns the common ASDU address, originator address, and any device-identity ASDUs received.

Port: 2404

deepot

iec61850-goose — IEC 61850 GOOSE

TCP

IEC 61850 GOOSE is the Generic Object Oriented Substation Event Layer-2 multicast protocol (EtherType 0x88B8) used by substation IEDs for peer-to-peer trip and status signaling. runZero passively decodes GOOSE frames, attributes the publisher MAC as an IEC 61850 IED, and records the GoCB reference and dataset.

multicastotpassive

iec61850-mms — IEC 61850 MMS

TCP

IEC 61850 MMS is the Manufacturing Message Specification mapping used by substation Intelligent Electronic Devices for monitoring, control, and reporting. Opens an MMS session and (when enabled) issues Identify, returning the IED vendor, model, firmware, and logical-device summary.

Port: 102

backplanedeepot

iec61850-sv — IEC 61850 Sampled Values

TCP

IEC 61850 Sampled Values is the Layer-2 multicast protocol (EtherType 0x88BA) used by substation merging units to publish synchronized current and voltage measurements to protection and control IEDs. runZero passively decodes SV frames, attributes the publisher MAC as a merging unit, and records the SvCB reference and dataset.

multicastotpassive

igel — IGEL Discovery

UDP

IGEL Discovery is the discovery protocol used by IGEL OS thin clients to advertise themselves to the IGEL Universal Management Suite (UMS), deployed in VDI and remote-desktop environments. Sends the IGEL discovery probe and returns the endpoint hostname, hardware model, IGEL OS version, MAC, and the UMS-server registration state.

Port: 30005

discoverylightmgmt

igel-discovery — IGEL UMS Discovery

UDP

IGEL UMS Discovery is the UDP broadcast used by IGEL OS thin clients to locate their Universal Management Suite server. Distinct from the igel management protocol decoded over TCP, this entry captures discovery-only sightings on UDP/30005.

Port: 30005

discoverymgmt

iis — Microsoft Internet Information Services (IIS)

TCP

Microsoft Internet Information Services (IIS) is the HTTP server bundled with Windows Server, commonly hosting ASP.NET, Outlook Web Access, and SharePoint. The HTTP fingerprinter inspects Server and X-Powered-By headers and default landing pages and returns the IIS version and ASP.NET version hints.

lighttlswb

ike

UDP

IKE is the Internet Key Exchange protocol used to negotiate IPsec security associations. Sends IKEv1/IKEv2 SA proposals and vendor-ID probes and returns the negotiated proposal summary, supported transforms, and any vendor-ID strings that disclose the gateway implementation.

Ports: 500, 4500

vpn

ikev2

UDP

IKEv2 is the Internet Key Exchange version 2 protocol (RFC 7296) used by IPsec VPN gateways on UDP/500 and UDP/4500. runZero sends an IKE_SA_INIT request and parses the

responder SPI, accepted transform set, and any vendor-ID payloads that disclose the gateway implementation.

Ports: 500, 4500

vpn

infinispan — Infinispan Hot Rod

TCP

Infinispan Hot Rod is a binary client/server protocol used by Red Hat Data Grid and JBoss-family caches. Performs the Hot Rod ping and returns the server version, topology identifier, and supported protocol version.

Port: 11222

database

influxdb

TCP

InfluxDB is the time-series database from InfluxData. runZero queries /ping and /health on the HTTP API to recover the server build, X-Influxdb-Version header, and on permissive deployments the list of available databases.

Port: 8086

clearmgmt

intermapper

TCP

InterMapper is the probe interface exposed by Fortra (formerly HelpSystems) InterMapper network-monitoring agents installed on monitored servers and appliances to report status to a central InterMapper server. Reads the InterMapper service banner and returns the product name and agent version.

Port: 8181

lightmonitoring

ipmi

UDP

IPMI is the Intelligent Platform Management Interface used for out-of-band server management on BMCs (iLO, iDRAC, IMM). Performs an IPMI 2.0 RMCP+ Get Channel Authentication Capabilities exchange and, when credentials are configured, returns supported cipher suites, authentication algorithms, and the BMC vendor/firmware identification.

Port: 623

authmgmt

ipp-browse — IPP Browse

UDP

IPP Browse is the legacy CUPS browse protocol used by macOS and Linux print servers to advertise IPP print queues to the local segment via UDP/631 broadcasts. runZero passively decodes browse packets and active responses and returns the advertised queue URI, printer name, and CUPS server identification, attributing the asset as a print server.

Port: 631

discoverylightmulticastprinting

ippbrowse — IPP Browse

UDP

IPP Browse is the legacy CUPS browse protocol used by Unix print servers to advertise IPP print queues via UDP/631 broadcasts. Passively decodes browse packets and returns the advertised queue URI, printer name, and CUPS server identification.

Port: 631

discoverylightmulticastprinting

irc

TCP

IRC is the Internet Relay Chat protocol, a text-based group messaging protocol. Reads the IRC server greeting and runs a NICK/USER probe, returning the server software, version, and 005-numeric capability summary.

Ports: 6667, 6668, 6669, 6697, 7000, 7001

clearmessagingtls

iscsi

TCP

iSCSI is the Internet Small Computer Systems Interface protocol used to expose block storage over IP. Sends a SendTargets request and returns the iSCSI Target Name list, target portals, and authentication-method summary.

Port: 3260

storage

iua — IUA (SCTP)

SCTP

IUA (SCTP) is the ISDN User Adaptation Layer (RFC 4233) carried over SCTP, used to backhaul ISDN signaling in SIGTRAN networks. Verifies the SCTP association and IUA payload protocol identifier and returns endpoint identification.

Port: 9900

voip

jabber

TCP

Jabber is the legacy product name for XMPP, still used by Cisco Jabber, ejabberd, Openfire, and similar chat/presence servers. runZero identifies Jabber services from the <stream:stream> or jabber.org-namespaced response returned on connection and tags the asset alongside the active xmpp scanner results.

messagingtls

java-object — Java Object Serialization

TCP

Java Object Serialization is the binary stream format produced by java.io.ObjectOutputStream, often indicating exposed RMI, JMX, or JBoss endpoints when seen on the wire. Inspects the magic header and returns the serialization-protocol version and any class-name hints disclosed in the stream.

lightmgmt

java-rmi — Java RMI

TCP

Java RMI is the Remote Method Invocation protocol used by Java applications. Performs the JRMP handshake and a registry list, returning the RMI-registry version and the names and stub classes of bound objects.

Port: 1099

mgmt

jdbc-hsqldb — HyperSQL JDBC

TCP

HyperSQL JDBC is the JDBC database server protocol used by the HyperSQL (HSQLDB) engine. Identifies the service from the 'HSQLDB JDBC Network Listener' banner returned on connection.

Port: 9001

database

jdwp — Java Debug Wire Protocol

TCP

Java Debug Wire Protocol is the unauthenticated JVM debugging transport used by IDEs and debuggers to control a JVM. Performs the JDWP handshake and Version command, returning the JDK version, JVM vendor, and process description.

Ports: 3999, 5000, 5005, 8000, 8453, 8787, 8788, 9001, 18000

mgmt

jetdirect — HP JetDirect

TCP

HP JetDirect is the raw printing protocol on TCP/9100 (PJM banner port). Sends a PJM INFO ID/STATUS probe and returns the printer make/model, firmware, page count, and PJM device-attribute summary.

Ports: 9100, 9101, 9102

lightprinting

jira — Atlassian Jira

TCP

Atlassian Jira is an issue-tracking and project-management server available in Server, Data Center, and Cloud editions. The HTTP probe fetches Jira login, dashboard, and REST endpoints and returns the product name, edition, and build number.

Ports: 80, 443, 8080

lighttlswsweb

jms — JMS / JMX-RMI Port Mapper

TCP

JMS / JMX-RMI Port Mapper is the OpenMQ/GlassFish imqbroker port-mapper service that publishes the names, transports, and ports of bound JMS and JMX-RMI endpoints (default TCP/7676). Queries the port mapper and returns the broker version and the bound endpoint names, transports, and ports.

Port: 7676

messagingmgmt

kafka — Apache Kafka

TCP

Apache Kafka is a distributed event-streaming wire protocol. Sends an ApiVersions request and returns the broker identifier, supported API versions, and (when an unauthenticated MetadataRequest is permitted) cluster and topic-name summaries.

Ports: 9092, 9093, 9094

messagingtls

kasa — TP-Link Kasa

TCP

TP-Link Kasa is the smart-home control protocol used by Kasa plugs, bulbs, and switches. Sends the obfuscated SYS_INFO query and returns the device alias, model, firmware, hardware version, and MAC.

Port: 9999

iotlight

kerberos

TCPUDP

Kerberos is the network authentication protocol used by Active Directory and many enterprise services. Sends an AS-REQ for a benign principal and returns the realm, KDC error code, and supported encryption types.

Ports: 88, 464, 749, 750

aaa

knxnet — KNXnet/IP

UDP

KNXnet/IP is the IP-tunneling encapsulation used to tunnel and route KNX building-automation telegrams (lighting, HVAC, shading). Sends a SEARCH_REQUEST and returns the device serial, MAC, KNX individual address, supported services, and friendly name.

Port: 3671

building-automationot

l2t — L2TP (UDP 1701)

UDP

L2TP (UDP 1701) is the Layer 2 Tunneling Protocol on UDP/1701, used by VPN concentrators and remote-access gateways (often paired with IPsec) to tunnel PPP sessions. Sends an L2TP SCCRQ and returns the host name, vendor name, and firmware revision AVPs disclosed by the responder.

Ports: 1701, 2228

vpn

l2tp

UDP

L2TP is the Layer 2 Tunneling Protocol used to carry PPP sessions, commonly with IPsec for VPN. Sends an SCCRQ and returns the host name, vendor name, and firmware revision AVPs.

Port: 1701

vpn

landesk — Ivanti / LANDesk Agent

TCP

Ivanti / LANDesk Agent is the endpoint-management agent for Ivanti (formerly LANDesk).

Reads the agent banner and returns the agent version and bound management server.

Port: 9595

lightmgmt

langflow

TCP

Langflow is an open-source visual builder for LLM-powered applications. runZero attributes services as Langflow from Recog matches against the application's HTTP responses and OpenAPI document, recovering the application version.

aiclearweb

lantronix — Lantronix Discovery

UDP

Lantronix Discovery is the discovery protocol used to locate Lantronix serial-to-Ethernet device servers. Sends the discovery probe and returns the device model, firmware, MAC, and configured serial-port settings.

Port: 30718

discoveryiotlight

ldap

TCP

LDAP is the Lightweight Directory Access Protocol (RFC 4511) used by Active Directory, OpenLDAP, 389 Directory Server, and other directory services for authentication and identity lookups. Queries the rootDSE and returns the supported LDAP versions, naming contexts, supported controls, and any forest or domain identifiers exposed.

Ports: 389, 636, 3268, 3269

authdirectorytls

lexmark — Lexmark Discovery

TCP

Lexmark Discovery is the printer/MFP network discovery protocol used by Lexmark devices. Sends the discovery probe and returns the device model, serial, firmware, and printer/MFP

capabilities.

Port: 10000

discoverylightprinting

librechat

TCP

LibreChat is an open-source self-hosted LLM chat front-end. runZero attributes services as LibreChat from Recog matches against the web UI banners and configuration endpoints.

Ports: 443, 3080

aiclearweb

liebert — Vertiv Liebert IntelliSlot

TCP

Vertiv Liebert IntelliSlot is the system running OT communications cards by Vertiv, often with HTTP servers and a predictable path for card information.

lightweb

lldp — Link Layer Discovery Protocol

TCP

Link Layer Discovery Protocol is the IEEE 802.1AB Layer-2 discovery protocol used by switches, routers, IP phones, and hypervisors to advertise themselves to neighbors. runZero passively decodes LLDP frames seen during the scan and returns the chassis ID, port ID, system name, system description, capabilities, and management addresses of each neighbor.

discoverymgmtmulticastpassive

llmnr

LLMNR is Link-Local Multicast Name Resolution (RFC 4795) used by Windows hosts to resolve single-label names on the local link when DNS is unavailable. Sends an LLMNR query and returns the responding hostname and the IP version of the answering host.

Port: 5355

lightmulticastnaming

lockdownnd — Apple lockdownnd

TCP

Apple lockdownnd pairing service exposed on TCP/62078 by iPhone, iPad, and iPod touch devices and used by iTunes, Finder, Apple Configurator, and MDM tooling. Reads the lockdownnd query response and returns the device class, product type, iOS or iPadOS version, serial number, and unique device identifier (UDID).

Port: 62078

lightmgmt

lpd

TCP

LPD is the standardized BSD line-printer protocol (RFC 1179). Sends a Receive-Job command and returns the raw printer banner from the daemon for downstream make and model fingerprinting.

Port: 515

lightprinting

lsv2 — Heidenhain LSV/2

TCP

Heidenhain LSV/2 is a control protocol used by Heidenhain TNC CNC controls (TNC 640, iTNC 530, TNC 320). Queries control identification and returns the NC software type, version, and (when enabled) NC software-options bitmask.

Ports: 8000, 8001, 8002, 8003, 8004, 19000

deepot

lwm2m — OMA LwM2M

UDP

OMA LwM2M is the OMA Lightweight M2M device-management protocol layered on CoAP and used to manage constrained IoT endpoints, sensors, and cellular modules. Sends a CoAP GET for /.well-known/core and returns whether the LwM2M registration directory and bootstrap-server resources are advertised along with a server-implementation hint.

Ports: 5683, 5783

iotmgmt

m2pa — M2PA (SCTP)

SCTP

M2PA (SCTP) is the MTP2 Peer Adaptation Layer (RFC 4165) over SCTP used in SIGTRAN to carry SS7 MTP2 between signaling gateways. Establishes an SCTP association, verifies the M2PA payload protocol identifier, and returns the M2PA message class, message type, link state, and any error code or info string in the reply.

Port: 3565

mobile-corevoip

m2ua — M2UA (SCTP)

SCTP

M2UA (SCTP) is the MTP2 User Adaptation Layer (RFC 3331) over SCTP used in SIGTRAN deployments. Establishes an SCTP association, verifies the M2UA payload protocol identifier, and returns the M2UA message class, message type, and any error code or info string in the reply.

Port: 2904

mobile-corevoip

m3ua — M3UA (SCTP)

SCTP

M3UA (SCTP) is the MTP3 User Adaptation Layer (RFC 4666) over SCTP, the most common SS7-over-IP transport. Establishes an SCTP association, verifies the M3UA payload protocol identifier, and returns the M3UA message class, message type, and any error code or info string in the reply.

Port: 2905

mobile-corevoip

managesieve

TCP

ManageSieve is a protocol used by mail clients to manage Sieve mail-filter scripts on the server. Reads the capabilities response and returns the implementation name, version, supported SASL mechanisms, and STARTTLS availability.

Ports: 2000, 4190

emailtls

matter

UDP

Matter is the Connectivity Standards Alliance smart-home application protocol used by Matter-certified devices over Wi-Fi and Thread. runZero identifies Matter devices from _matter._tcp mDNS records, parses the VP and DT TXT fields, and resolves them against the bundled Matter vendor table to return the vendor name, product name, and device type.

discoveryiotpassive

mbus-tcp — M-Bus over TCP

TCP

M-Bus over TCP is the EN 13757 Meter-Bus protocol tunneled over TCP, used by gateways aggregating utility meters (heat, water, gas, electric). Sends REQ_UD2 to the gateway and

(when enabled) walks primary addresses 1-250 to enumerate connected meters, returning meter manufacturer, identification, version, and medium.

Ports: 8888, 8889

backplanedeepot

mcp — Model Context Protocol

TCP

Model Context Protocol (MCP) is Anthropic's JSON-RPC protocol used by AI agents to connect to external tools and data sources. runZero attributes services as MCP from Recog matches against the server's HTTP/SSE handshake and the initialize response, which exposes the server name, version, and advertised capabilities.

aiclear

mdns

UDP

mDNS is Multicast DNS used by Bonjour, Avahi, and other zero-configuration networking stacks. Sends a service-enumeration query and returns the advertised service types, instance names, ports, hostnames, and TXT-record metadata.

Port: 5353

discoverylightmulticastnaming

megaco — Megaco / H.248

UDP

Megaco / H.248 is a media gateway control protocol (RFC 3525) used between softswitches and media gateways in carrier and enterprise VoIP networks. Sends a ServiceChange probe and returns the media-gateway identifier (MID) and the negotiated H.248 protocol version.

Ports: 2944, 2945

voip

melsecq — Mitsubishi MELSEC-Q

TCP

Mitsubishi MELSEC-Q is a protocol used to communicate with Mitsubishi MELSEC PLCs.

Issues a CPU model-name read using SLMP 3E (and, when enabled, 4E) and returns the CPU model name and CPU type code along with the matching MELSEC product CPE.

Ports: 5006, 5007

backplanedeepot

memcache — Memcache (text)

TCPUDP

Memcache (text) is the ASCII text wire protocol exposed by the Memcached distributed in-memory key-value cache and compatible servers. Issues a text VERSION/STATS request and returns the daemon version, current connections, and items and bytes held in cache.

Port: 11211

cleardatabasetls

meshcop — Thread Mesh Commissioning

UDP

Thread Mesh Commissioning is the Mesh Commissioning Protocol used by Thread border routers (Apple HomePod, Google Nest Hub, eero, Nordic OTBR) to advertise commissioning endpoints to companion mobile apps. runZero attributes the asset as a Thread border router from the meshcop hint and surfaces the advertised network name, extended PAN ID, and vendor identification.

discoveryiotpassive

mgcp

UDP

MGCP is the Media Gateway Control Protocol (RFC 3435) used between call agents and media gateways in VoIP networks. Sends an AUEP probe and returns the response code along with any endpoint identifiers and packages disclosed in the reply.

Ports: 2427, 2727

voip

mikrotik-bandwidth — MikroTik Bandwidth Test Server

TCP

MikroTik Bandwidth Test Server is the proprietary throughput-testing service exposed by MikroTik RouterOS devices. runZero detects the listener via its banner; presence indicates the device is reachable for line-rate tests, which can be abused for amplification.

Port: 2000

clearmgmt

mikrotikwinbox — MikroTik Winbox

TCP

MikroTik Winbox is the protocol used by the Winbox utility to administer RouterOS devices. Sends the index-request and returns the RouterOS architecture, version, and bootloader identification.

Ports: 2000, 8291, 8728

mgmt

milvus — Milvus Vector Database

TCP

Milvus is an open-source vector database used by retrieval-augmented LLM applications. runZero attributes services as Milvus from Recog matches against the gRPC and HTTP endpoints and the management UI banners.

Port: 19530

aiclearmgmt

minecraft — Minecraft Java

TCP

Minecraft Java is the server query/list-ping protocol used by Minecraft Java Edition. Sends a status-request and returns the server MOTD, version, protocol number, current and maximum player counts, and any sample player names disclosed.

Port: 25565

gaming

mms — ISO 9506 MMS (IEC 61850)

TCP

ISO 9506 MMS (IEC 61850) is the Manufacturing Message Specification over RFC 1006/COTP, the application protocol used by IEC 61850 substation Intelligent Electronic Devices (IEDs). The probe issues an A-ASSOCIATE plus MMS Initiate-Request and parses the Initiate-Response for vendor identity, negotiated MMS version, and supported services.

Port: 102

backplanedeepot

modbus — Modbus/TCP

TCP

Modbus/TCP is a protocol used to read and write registers on PLCs, RTUs, drives, and meters. Issues a Read Device Identification (function 43/MEI 14) and returns the vendor, product code, revision, vendor URL, product name, and (when configured) extended identification objects.

Port: 502

backplanedeepot

mongodb — MongoDB Wire Protocol

TCP

MongoDB Wire Protocol is the document-database wire protocol used by MongoDB drivers. Sends an unauthenticated isMaster/hello and returns the server version, build environment, replica-set role, and observed authentication requirements.

Ports: 27017, 27018, 27019, 28017

databasetls

mountd — NFS mountd

TCPUDP

NFS mountd is the companion daemon to NFS that authorizes mount requests and enumerates exports, registered through rpcbind on Unix file servers and NAS appliances. runZero locates mountd through rpcbind, sends an EXPORT (procedure 5) call for each advertised version, and returns the exported directory list and per-export host or netgroup access lists.

filestorage

moxa — Moxa Discovery

UDP

Moxa Discovery is the proprietary UDP service (4800/udp) used by the Moxa configuration utilities to locate NPort, OnCell, and MGate devices. Sends an identify request and returns the device MAC address and vendor, confirming a Moxa device.

Port: 4800

discoveryot

mqtt

TCP

MQTT is a lightweight publish/subscribe messaging protocol used by IoT devices and brokers. Sends a CONNECT and returns the broker's CONNACK response code, supported MQTT version, and any properties advertised by the broker.

Ports: 1883, 8883

iotmessagingtls

mssql — Microsoft SQL Server (TDS)

TCPUDP

Microsoft SQL Server (TDS) is the Tabular Data Stream protocol used by Microsoft SQL Server and Sybase ASE database engines for client-server query traffic. Sends a TDS PRELOGIN and returns the server version, encryption requirement, and named-instance identification.

Ports: 1433, 1434

databasetls

mssql-replica — Microsoft SQL Server Replica

TCP

Microsoft SQL Server Replica is the Always-On availability-group and database-mirroring endpoint that exchanges replica traffic on TCP/5022 separately from the user TDS endpoint on TCP/1433. runZero hints TCP/5022 as mssql-replica during the MSSQL probe and tags the asset as a SQL Server replica endpoint when the standard TDS handshake is not offered.

cleardatabaselight

mtconnect

TCP

MTConnect is a protocol used by CNC machine tools, robots, and additive-manufacturing systems to publish device state over an HTTP/XML REST API. Issues a GET /probe and returns the agent version, instance ID, sender host, and per-device manufacturer, model, serial number, and UUID.

Ports: 5000, 7878

discoveryot

munin

TCP

Munin is the protocol used by the Munin master to poll plugins on hosts. Reads the node banner and returns the node hostname, Munin version, and configured plugin list summary.

Port: 4949

monitoring

mysql — MySQL / MariaDB

TCP

MySQL / MariaDB is the binary client-server wire protocol used by MySQL, MariaDB, and Percona Server database engines. Reads the server-greeting packet and (when credentials are configured) authenticates, returning the server version, capability flags, supported authentication plugins, and TLS availability.

Ports: 3306, 33060

databasetls

mysqlx — MySQL X Protocol

TCP

MySQL X Protocol is a Protocol-Buffer-based wire protocol exposed by MySQL Server (default TCP/33060) for MySQL Shell and X DevAPI document-store and CRUD clients. Sends a CapabilitiesGet message and returns the supported X-Protocol capabilities and TLS requirements.

Port: 33060

database

natpmp — NAT-PMP

UDP

NAT-PMP is a protocol used by clients to request port forwards from a NAT gateway (RFC 6886). Sends a public-address request and returns the gateway's external IPv4 address, response code, and seconds-since-epoch counter.

Port: 5351

lightmgmt

nats

TCP

NATS is a lightweight publish/subscribe and request/reply messaging system. Reads the NATS INFO message and returns the server identifier, version, host, port, and authorization/TLS requirements.

Ports: 4222, 6222, 8222

clearmessagingtls

ndmp

TCP

NDMP is the Network Data Management Protocol used by enterprise backup software to coordinate backups of NAS devices. Opens a CONNECT_OPEN session and returns the NDMP protocol version reported by the server along with the connection status and any reason text.

Port: 10000

backupstorage

neo4j — Neo4j Bolt

TCP

Neo4j Bolt is the graph-database wire protocol used by Neo4j drivers and clients. Performs the Bolt handshake and returns the negotiated Bolt version and server release.

Ports: 7473, 7474, 7687

databasetls

netbios — NetBIOS Session Service

TCP

NetBIOS Session Service is the TCP/139 transport used by the legacy NetBIOS-over-TCP framing of SMB. runZero records the session-service banner alongside the NetBIOS name and workstation/server resource records reported by the host. The application-layer SMB protocol is decoded separately.

Port: 137

clearmgmt

netbios-dgm — NetBIOS Datagram Service

UDP

NetBIOS Datagram Service is the UDP/138 broadcast/datagram side of NetBIOS-over-TCP. runZero observes datagrams to recover the NetBIOS computer and workgroup names announced by Windows hosts and SMB servers.

Port: 138

discovery

netbios-ns — NetBIOS Name Service

UDP

NetBIOS Name Service is a protocol used for legacy Windows name registration and resolution. Sends a NetBIOS node-status query and returns the registered names, node type, and any associated MAC address.

Ports: 137, 138

naming

netis

UDP

Netis is an identifier for Netis and Netcore SOHO routers, including the well-known UDP/53413 administrative backdoor exposed by historical firmware. runZero attributes the service from the netis-specific hint and tags the asset as a Netis router so the historical backdoor exposure is surfaced in inventory and reports.

iotmgmt

netop-remote-control — Netop Remote Control

TCP

Netop Remote Control is a commercial remote-administration product from Netop (formerly Danware), commonly deployed in classroom, kiosk, and retail environments. runZero attributes the asset from the Netop host banner observed on the standard service port and tags the service as remote-access for inventory and exposure reporting.

clearlightremote-access

nfs

TCPUDP

NFS is the Network File System used to share files across Unix-like systems (Sun RPC program 100003). Issues a NFS NULL ping and a MOUNT EXPORT call (via portmap) and returns the supported NFS versions and the list of exported filesystems and allowed clients.

Port: 2049

filestorage

nrpe — Nagios NRPE

TCP

Nagios NRPE is the protocol used by Nagios/Icinga to run checks on remote hosts. Sends a _NRPE_CHECK probe and returns the NRPE protocol version and any version banner disclosed.

Port: 5666

monitoringtls

ntp

UDP

NTP is the Network Time Protocol used to synchronize clocks across networks (RFC 5905). Sends mode-3 client and mode-6 control queries and returns the stratum, reference identifier, refid clock source, and (when enabled) implementation/version strings disclosed by mode-6 readvar.

Port: 123

time

omronfins — Omron FINS

UDP

Omron FINS is the Factory Interface Network Service protocol used to communicate with Omron CJ/CS/NJ/NX PLCs and related automation devices. Issues a Controller Data Read (0501) and returns the controller model and firmware version along with the FINS handshake banner.

Port: 9600

ot

opcda — OPC Classic (OPC DA)

TCP

OPC Classic (OPC DA) is the OLE for Process Control data-access standard layered over Microsoft DCOM, widely deployed in legacy SCADA, HMI, and historian gateways. runZero detects OPC DA servers when the DCERPC scanner observes the OPCEnum interface UUID advertised by the endpoint mapper.

deepot

opcua — OPC UA

TCP

OPC UA is a vendor-neutral industrial information-model and data-access standard. Performs a GetEndpoints request and returns the application URI, product URI, server-certificate metadata, and per-endpoint security policies and identity tokens.

Ports: 4840, 4843, 48050

backplanedeepot

openvpn

UDP

OpenVPN is the tunnel control-channel protocol used by OpenVPN Community Edition and OpenVPN Access Server VPN gateways. Sends a P_CONTROL_HARD_RESET_CLIENT_V2 packet and returns the local and remote OpenVPN session identifiers from the server's hard-reset reply.

Port: 1194

encryptedvpn

oracle — Oracle TNS

TCP

Oracle TNS is the Transparent Network Substrate listener protocol used by Oracle Database servers. Sends a TNS Connect carrying a VERSION command and returns the listener version, instance name, and supported services.

Ports: 1521, 1522, 1525, 2483, 2484

database

oracledb — Oracle Database (TNS)

TCP

Oracle Database (TNS) is a lightweight Oracle Net listener probe used to identify Oracle Database servers without negotiating a session. Sends a minimal TNS Connect, parses the Refuse/Accept/Resend reply, and returns the packet type, VSNNUM, error code, and disclosed listener version.

Ports: 1521, 1522, 1525

database

orion — SolarWinds Orion Platform

TCP

SolarWinds Orion Platform is a Windows-based network-management suite that bundles NPM, NCM, NTA, IPAM, SAM, UDT, and related modules. The HTTP extractor matches the Orion footer in the web console, parses the platform release and component list, and returns orion.version, orion.components, and SolarWinds Orion software identification.

lightttsweb

panasonicictv — Panasonic TV

TCP

Panasonic TV is the network control service exposed by Panasonic Viera and similar consumer smart TVs for remote-control companion apps. runZero identifies these televisions from the proprietary control-protocol banner returned on connection, applies Panasonic TV fingerprinting, and attributes the asset as a Panasonic television.

iotlight

panxmlapi — Palo Alto Networks PAN-OS XML API

TCP

Palo Alto Networks PAN-OS XML API is the authenticated management API (system info, ARP/MAC/neighbor caches, interfaces) issued against Palo Alto Networks firewalls and Panorama. Used by the runZero scanner with a user-supplied API key to enumerate adjacent assets and device facts.

authdeeptls

pca — Symantec pcAnywhere

UDP

Symantec pcAnywhere is a remote-access protocol. Sends the pcAnywhere status probe and returns the host name, status, and capability flags disclosed in the response.

Port: 5632

lightremote-access

pcworx — Phoenix Contact PCWorx

TCP

Phoenix Contact PCWorx is a runtime protocol used to program and interact with ILC-series and other Phoenix Contact controllers. Queries controller identification and returns the PLC type, model number, and firmware version, date, and time.

Port: 1962

ot

pega — Pega Platform

TCP

Pega Platform is a low-code business process management and CRM application server from Pegasystems used for enterprise workflow automation. The HTTP extractor matches Pega in the page title or body, parses the version span, and returns the pega.version attribute and Pegasystems Pega software identification.

lightttsweb

pfcp

UDP

PFCP is the Packet Forwarding Control Protocol used in 5G/LTE mobile cores between control-plane and user-plane functions. Sends a PFCP Heartbeat Request and returns the recovery time stamp and supported feature flags.

Port: 8805

mobile-core

pop3

TCP

POP3 is the Post Office Protocol version 3 used by mail clients to retrieve messages from a server. Reads the POP3 greeting and runs CAPA, returning the server software banner, supported capabilities, and STARTTLS availability.

Ports: 110, 995

emailtls

postgresql

TCP

PostgreSQL is the frontend/backend wire protocol used by PostgreSQL database servers and compatible engines such as Amazon RDS/Aurora and CockroachDB. Performs an SSLRequest followed by a StartupMessage and returns the server version, supported authentication mechanisms, advertised server parameters, and TLS availability.

Ports: 5432, 5433, 6432

databasetls

powerlink — Ethernet POWERLINK

UDP

Ethernet POWERLINK is a real-time industrial Ethernet protocol from the EPSG, used for deterministic motion control and I/O between managing nodes and controlled nodes on machine tools, packaging lines, and robotics cells. Passively decodes POWERLINK frames and returns the node identifier, vendor identifier, product code, and revision.

otpassive

pptp

TCP

PPTP is the Microsoft Point-to-Point Tunneling Protocol legacy VPN. Sends a Start-Control-Connection-Request and returns the protocol version, vendor, firmware revision, and host name.

Port: 1723

vpn

printerid — Printer Identification

TCP

Printer Identification is a vendor-specific service exposed on TCP/9200 by HP, Lexmark, and other network printers and MFPs. runZero parses the model string returned by the device, records it as printerid.model, and uses the value as a synthetic fingerprint to identify the printer make and model during asset categorization.

lightprinting

proconos — Phoenix Contact ProConOS

TCP

Phoenix Contact ProConOS is the PLC runtime protocol from KW-Software/Phoenix Contact, used by ILC, RFC, and OEM-rebadged controllers running the ProConOS or ProConOS eCLR runtime. Issues a runtime identification query and returns the ladder-logic runtime version, PLC type, project name, boot project, and project source-code identifier.

Port: 20547

ot

profinet

UDP

PROFINET is an industrial Ethernet protocol used for cyclic and acyclic real-time communication with PROFINET I/O devices and PLCs. Performs a Read Identification request and returns the device vendor, order number, serial, software/hardware revision, and (when enabled) discovered slot/subslot module list.

Ports: 34962, 34963, 34964

backplanedeepot

profinet-dcp — PROFINET DCP

UDP

PROFINET DCP is the Discovery and basic Configuration Protocol used at Layer 2 to identify and assign names/IPs to PROFINET stations. Passively decodes DCP Identify announcements and returns the station name, vendor and device identifiers, and device role.

discoveryotpassive

prosoft — ProSoft Discovery Service

TCP

ProSoft Discovery Service is the UDP discovery protocol used by ProSoft Technology industrial gateways and radios. runZero parses the response to recover the device model, firmware revision, MAC address, and IP configuration.

Port: 1718

discoveryot

psdisco — PlayStation Discovery

UDP

PlayStation Discovery is the Sony PlayStation 4/5 console-discovery service (HTTP/1.1-style SRCH messages on UDP/987 and UDP/9302) used by Remote Play and second-screen companion apps. Sends a SRCH probe and decodes the response, returning the host id, host name, host type, system version, discovery-protocol version, and running-app title metadata.

Ports: 987, 9302

discoverygamingiotlight

qdrant — Qdrant Vector Database

TCP

Qdrant is an open-source vector search database used by retrieval-augmented LLM applications. runZero attributes services as Qdrant from Recog matches against the HTTP API root and the /metrics endpoint, which expose the server version.

Ports: 6333, 6334

aiclearmgmt

qotd — Quote of the Day

TCPUDP

Quote of the Day is the RFC 865 diagnostic service historically exposed by Unix inetd hosts and frequently abused for UDP amplification. Passively decodes QOTD replies in TCP and UDP captures and returns the protocol tag along with the truncated quote text or banner observed in the response.

Port: 17

legacylight

qualys — Qualys Cloud Agent / Scanner Appliance

TCP

Qualys Cloud Agent / Scanner Appliance is the web management interface exposed by Qualys vulnerability-management components, including the on-premises Scanner Appliance and Cloud Agent UI. The HTTP fingerprinter inspects these pages and returns the product name, appliance role, and any version identifiers disclosed.

lightsecuritytlsweb

quic

UDP

QUIC is the IETF transport (RFC 9000), an encrypted UDP-based transport that carries HTTP/3 and is increasingly used by CDNs, web servers, and SaaS endpoints. runZero

attributes the service as QUIC when the long-header Initial packet is observed on a probed UDP port and tags the asset for web / TLS exposure tracking.

encryptedtlsweb

radius

UDP

RADIUS is an AAA protocol used in Wi-Fi, VPN, and network-access control. Sends an Access-Request with an invalid principal and returns the response code, any Reply-Message and NAS-Identifier, and the list of attributes present in the reply.

Ports: 1645, 1646, 1812, 1813

aaa

raritan-csc — Raritan CommandCenter

TCP

Raritan CommandCenter is the Common Socket Connection (CSC) management protocol used by Raritan CommandCenter Secure Gateway appliances and adjacent Raritan KVM/serial console managers. Passively detects the <CSC/> banner emitted on connection and tags the asset as a Raritan CommandCenter device.

lightmgmt

rdp

TCP

RDP is the Microsoft Remote Desktop Protocol used by Windows Remote Desktop Services. Performs an X.224 Connection-Request and returns the supported security protocols, NLA requirement, and (when TLS is offered) certificate-derived hostname/version metadata.

Ports: 3389, 3390

remote-accesstls

redfish

TCP

redis

TCP

Redis is an in-memory data-structure store and message broker. Issues PING/INFO/AUTH probes and returns the Redis version, role, mode, and authentication or protected-mode requirements.

Ports: 6379, 16379, 26379

databasetls

rexec

TCP

rexec is the BSD Remote Execution protocol (legacy, transmits credentials in cleartext).

Detects an rexec listener and returns the responsiveness and any host-identification banner observed.

Port: 512

clearlegacylightremote-access

riak — Riak Protocol Buffers

TCP

Riak is a distributed NoSQL key/value store from Basho. This entry covers the Protocol Buffers transport on TCP/8087 used by the native Riak clients; the HTTP API is reported separately as riak-http.

Port: 8098

clearmgmt

riak-http — Riak HTTP API

TCP

Riak HTTP API is the REST interface exposed by Basho Riak nodes (typically TCP/8098). runZero queries /stats and /riak/ to recover the node name, ring size, and Riak version.

Port: 8098

clearmgmt

rip

UDP

RIP is a distance-vector IGP. Sends a RIP Request and returns the RIP version and any advertised routes disclosed by the responder.

Port: 520

routing

rlogin

TCP

rlogin is the legacy BSD remote-login protocol (RFC 1282) superseded by SSH. runZero records the banner and any login prompt returned by the server, and flags the service as a clear-text credential exposure.

Port: 513

clearmgmt

roomalert — AVTECH Room Alert

TCP

AVTECH Room Alert is an environmental monitoring appliance. Reads the device banner and returns the model, OS version, MAC address, and IP address.

Port: 9999

iotlightmonitoring

rpcbind — ONC RPC / rpcbind

TCPUDP

ONC RPC / rpcbind is the portmap service used to discover Sun RPC programs (NFS, NIS, ...). Queries the portmap dump and returns the registered program list with versions, protocols, and ports.

Port: 111

discoverymgmt

rsync — rsync (SSH-tunneled)

TCP

rsync is the rsync daemon wire protocol exposed on TCP/873 by file mirrors, backup servers, and software-distribution archives. Reads the server greeting, performs the version handshake, and returns the protocol version, raw banner, and module list when the server permits enumeration.

Port: 873

file

rsyncd — rsync daemon

TCP

rsync daemon is the standalone rsync service (rsync://) listening on TCP/873. Reads the daemon greeting and lists modules, returning the rsync version, available module names, and module comments.

Port: 873

clearfilelight

rtmp

TCP

RTMP is the Adobe Real-Time Messaging Protocol used to stream audio, video, and data between Flash players and media servers. Performs the RTMP handshake and returns the protocol version byte echoed by the server.

Port: 1935

media

rtps — OMG RTPS / DDS

UDP

OMG RTPS / DDS is the Real-Time Publish-Subscribe wire protocol from the Object Management Group; underlies DDS in robotics, ROS 2, and industrial IoT. Sends an SPDP participant announcement and returns the participant GUID, vendor identifier and name, and protocol version.

Ports: 7400, 7401, 7410, 7411

discoverymessagingot

rtsp

TCP

RTSP is a streaming-control protocol used by IP cameras, NVRs, and media servers to control streams. Issues an OPTIONS request and returns the server software, supported methods, and any session-description metadata disclosed by DESCRIBE.

Ports: 554, 8554

iotmediatls

s7comm — Siemens S7Comm

TCP

Siemens S7Comm is the protocol used to program and exchange data with SIMATIC S7-300, S7-400, S7-1200, and S7-1500 PLCs. Issues SZL identification reads and returns the module name, plant identification, copyright, serial, module type, hardware/firmware version, and (when enabled) backplane rack/slot module list.

Port: 102

backplanedeepot

sadp — Hikvision SADP

UDP

Hikvision SADP is the Search Active Devices Protocol used to discover Hikvision and OEM IP cameras and NVRs. Sends a SADP Inquiry and returns the device serial, model, firmware, MAC, IP configuration, and activation state.

Port: 37020

discoveryiotlightmulticast

sccp — Cisco SCCP / Skinny

TCP

Cisco SCCP / Skinny is a call-control protocol used by Cisco IP phones registering with CallManager/Unified Communications Manager. Sends a Register message and returns the call-manager response and station identification.

Ports: 2000, 2443

voip

sctpun — SCTP Tunnel

UDP

SCTP Tunnel is runZero's identifier for SCTP-over-UDP encapsulation (RFC 6951) used to traverse middleboxes that block native SCTP. It is reported when an SCTP INIT is observed inside a UDP/9899 datagram.

Port: 9899

clearmgmt

securemote — Check Point SecuRemote

TCP

Check Point SecuRemote is a Check Point VPN topology discovery service. Sends the topology query and returns the gateway hostname and server identifier disclosed in the

response.

Port: 264

securityvpn

sentinel — Redis Sentinel

TCP

Redis Sentinel is the high-availability supervisor for Redis. runZero issues SENTINEL ping and SENTINEL master to recover the sentinel version, monitored master name, and quorum configuration on unauthenticated instances.

Port: 26379

clearmgmt

sercos-iii — SERCOS III

UDP

SERCOS III is a real-time industrial Ethernet protocol used for drives, servos, and I/O in machine tools and packaging machinery. Passively decodes SERCOS III frames and returns the slave count, cycle time, and vendor and device codes.

otpassive

servicetag — Sun Service Tag

TCPUDP

Sun Service Tag is a discovery service used to inventory Sun/Oracle hardware and software. Sends the discovery probe and returns the registered product instance, instance URN, and version.

Port: 6481

discoverymgmt

sgsap — SGsAP (SCTP)

SCTP

SGsAP (SCTP) is the SGs interface (3GPP TS 29.118) over SCTP between MME and MSC for SMS over SGs and CSFB. Verifies the SCTP association and SGsAP payload protocol identifier and returns endpoint identification.

Port: 29118

mobile-core

sip

TCPUDP

SIP is a signaling protocol used to establish voice, video, and messaging sessions. Sends an OPTIONS request and returns the response code, server/user-agent strings, allowed methods, and any contact and via metadata disclosed.

Ports: 5060, 5061

tlsvoip

sixnet-udr — Sixnet UDR

TCPUDP

Sixnet UDR (Universal Data Register) is the station control protocol served by sxether_client on 1594/tcp and 1594/udp on Red Lion / Sixnet SixTRAK and VersaTRAK RTUs. runZero identifies it passively - a reply that leads with the 0x29 sync byte and carries a valid CCITT CRC - without ever sending a UDR frame, because the channel is unauthenticated by default and can carry destructive file and I/O-register writes. A reachable listener is flagged as an exposure.

Port: 1594

discoverylightot

slp

TCPUDP

SLP is the Service Location Protocol used to discover services on a LAN (RFC 2608); commonly exposed by VMware ESXi and printers. Sends an attribute and service-type request and returns the SLP version, advertised services, and per-service attribute summary.

Port: 427

discovery

smb — SMB / CIFS

TCP

SMB / CIFS is the file-sharing and IPC protocol used by Windows and Samba. Negotiates SMB1/2/3 and reads tree/share metadata, returning the dialect, signing/encryption requirements, server OS, NetBIOS/computer name, domain, and (when permitted) the list of shares.

Ports: 139, 445

filestorage

smb1 — SMBv1

TCP

SMBv1 is the legacy Server Message Block dialect (CIFS / NT LM 0.12) deprecated by Microsoft and disabled by default since Windows 10 1709 / Server 2019. runZero flags this dialect specifically because it is required by the EternalBlue and related exploits and should be disabled wherever possible.

Ports: 139, 445

clearmgmt

smb2 — SMBv2

TCP

SMBv2 is the Server Message Block dialect family introduced in Windows Vista / Server 2008 (dialects 2.0.2 and 2.1). runZero records the negotiated dialect, signing requirements, server GUID, and operating-system version reported during NEGOTIATE_PROTOCOL and SESSION_SETUP.

Port: 445

clearmgmt

smb3 — SMBv3

TCP

SMBv3 is the Server Message Block protocol family introduced in Windows 8 / Server 2012 (dialects 3.0, 3.0.2, 3.1.1). runZero records the negotiated dialect, signing and encryption capabilities, and any pre-auth integrity hash advertised by the server.

Port: 445

clearmgmt

smtp

TCP

snmp

UDP

SNMP is the Simple Network Management Protocol used to monitor and configure network devices and servers. Walks system.* and selected enterprise OIDs over SNMPv1/v2c (and SNMPv3 when configured) and returns sysDescr, sysObjectID, sysName, location, contact, and a vendor/device-type fingerprint derived from the response.

Ports: 161, 162, 10161, 10162

authmgmt

socks — SOCKS Proxy

TCP

SOCKS is a generic proxy protocol with two incompatible versions (SOCKS4 and SOCKS5). This entry captures sightings where only the SOCKS family was identified; the version-

specific decoders socks4 and socks5 record the negotiated authentication methods and supported commands.

Port: 1080

clearmgmt

socks4

TCP

SOCKS4 is a proxy protocol (SOCKS version 4) used by client applications and proxy servers (Squid, Dante, SSH dynamic forwarding) to relay TCP connections through an intermediary. Sends a CONNECT request to a benign target and returns the proxy responsiveness and the SOCKS reply status code observed.

Ports: 1080, 1081

clearlightproxy

socks5

TCP

SOCKS5 is a proxy protocol (SOCKS version 5, RFC 1928) used by client applications, Tor (9050/9150), and proxy servers such as Dante, 3proxy, and Squid to relay TCP and UDP through an intermediary. Sends a method-selection request and returns the SOCKS version, supported authentication methods, and proxy reachability.

Ports: 1080, 1081, 9050, 9150

lightproxy

solr — Apache Solr

TCP

Apache Solr is the Lucene-based enterprise search platform. runZero attributes services as Solr from the X-Solr-Version response header, the admin-UI banners, and the /solr/admin/info/system JSON, which exposes the Solr and Lucene versions, JVM details, and host operating system.

Port: 8983

clearmgmt

some-ip — AUTOSAR SOME/IP

UDP

AUTOSAR SOME/IP is the Scalable service-Oriented MiddlewarE over IP used between automotive ECUs on in-vehicle Ethernet for service discovery and RPC. Sends a SOME/IP-SD FindService and returns the advertised service IDs and service count.

Ports: 30490, 30491

automotiveot

sonarqube

TCP

SonarQube is the static analysis and code-quality platform from SonarSource. runZero attributes services as SonarQube from the application's HTTP banners and the /api/system/status endpoint, recovering the server version and edition.

Port: 9000

clearmgmt

sonicwall-sgms — SonicWall GMS Agent

TCP

SonicWall GMS Agent is the Global Management System agent used to manage SonicWall firewalls. Identifies the agent from its TCP banner.

Port: 3023

lightmgmtsecurity

splunk — Splunk Enterprise / Universal Forwarder Web UI

TCP

Splunk Enterprise / Universal Forwarder Web UI is the HTTP-served management interface for Splunk's SIEM and log-collection platform. The HTTP extractor inspects splunkd and Splunk Web responses and returns the product edition and version.

lightsiemtlswb

spotify-connect — Spotify Connect

TCP

Spotify Connect is the device-discovery and remote-control protocol used by Spotify clients to find playback endpoints (smart speakers, AV receivers, set-top boxes). Detected via the _spotify-connect._tcp mDNS record and the /zc HTTP endpoint exposed by the device.

discoveryiot

ssdp

UDP

SSDP is the Simple Service Discovery Protocol used by UPnP devices to advertise services. Sends an M-SEARCH and returns the advertised service types, USN, server string, and Location URLs of the responding devices.

Port: 1900

discoveryiotlightmulticast

ssh

TCP

SSH is the Secure Shell remote-access and tunneling protocol. Reads the SSH banner, runs a KEX-init exchange, and (when credentials are configured) authenticates, returning the server software string, supported KEX/host-key/cipher/MAC algorithms, host keys and fingerprints, and accepted authentication methods.

Ports: 22, 2222, 22222

encryptedremote-access

steam — Steam Server Discovery

UDP

Steam Server Discovery is the Valve Steam Remote Play / In-Home Streaming LAN broadcast protocol. Sends a CMsgRemoteClientBroadcastDiscovery and returns the hostname, client/instance/device IDs, client version, OS type, public IP, and Steam Deck / VR / Remote Play status.

Port: 27036

discoverygaminglight

stun

UDP

STUN is the Session Traversal Utilities for NAT protocol (RFC 5389) used for NAT discovery in WebRTC and VoIP. Sends a binding request and returns the SOFTWARE attribute and the observed XOR-MAPPED-ADDRESS reported by the server.

Ports: 3478, 3479, 5349, 5350

voip

sua — SUA (SCTP)

SCTP

SUA (SCTP) is the SCCP User Adaptation Layer (RFC 3868) over SCTP, used to carry SS7 SCCP signaling over IP. Verifies the SCTP association and SUA payload protocol identifier and returns endpoint identification.

Ports: 2904, 14001

mobile-corevoip

subversion — Apache Subversion

TCP

Subversion is the Apache version-control system. This entry covers detections produced by integration data and Recog banners; runZero's active svn decoder negotiates the svnserve protocol on TCP/3690 to recover the repository UUID, root URL, and supported capabilities.

Port: 3690

clearmgmt

sunrpc — Sun RPC Portmapper

UDP

Sun RPC Portmapper (also known as rpcbind) is the legacy ONC RPC service registry. This entry covers Recog-only matches against banners that did not return a structured rpcbind dump; the active rpcbind decoder enumerates registered programs and their dynamic ports.

Port: 111

clearmgmt

svn — Subversion

TCP

Subversion is the Apache Subversion svn:// version-control protocol. Reads the SVN greeting and returns the minimum and maximum supported protocol versions, supported capabilities, and offered authentication mechanisms.

Port: 3690

file

sybase — Sybase / SAP ASE (TDS 5.0)

TCP

Sybase / SAP ASE (TDS 5.0) is the Tabular Data Stream 5.0 wire protocol used by Sybase ASE and SAP Adaptive Server Enterprise. Sends a TDS prelogin probe and returns the server version reported in the prelogin response.

Port: 5000

database

syslog

TCPUDP

Syslog is a standard event-logging protocol used to forward log messages between hosts and collectors. Identifies syslog listeners over UDP/TCP and decodes RFC 3164/5424 messages to report the priority, facility, severity, version, hostname, and application name.

Ports: 514, 6514

monitoringtls

tcpmux

TCP

TCPMUX is the TCP Port Service Multiplexer (RFC 1078), a legacy diagnostic service. Queries the registered service list and returns the disclosed names.

legacylight

teamviewer

TCP

TeamViewer is the proprietary binary protocol used by the TeamViewer remote-access client to reach the TeamViewer cloud over TCP/5938. Sends a TeamViewer ping/hello probe and identifies the service from the response magic and command byte.

Port: 5938

lightrmote-access

telnet

TCP

tenable-agent-id — Tenable Agent Identifier

TCP

Tenable Agent Identifier is the synthetic protocol runZero uses to track the unique agent UUID reported by Tenable Nessus and Tenable.io agents observed via integrations. It is not a network protocol and has no associated wire-level scan.

integration

tftp

UDP

TFTP is the Trivial File Transfer Protocol (RFC 1350) used for boot images, firmware, and config transfer. Sends a benign read request and returns the responsiveness and any error-code metadata that discloses the server implementation.

Port: 69

file

thinprint

TCP

ThinPrint is the Cortado virtual-printing protocol used by Citrix, VMware Horizon, and Microsoft RDS deployments to redirect print jobs from session hosts to client-side printers. Detected from the TPAutoConnect listener banner on TCP/4000.

Port: 4000

clearmgmt

time

TCPUDP

Time is the legacy 32-bit time-of-day service (RFC 868). Reads the response to confirm the service and detect amplification-capable hosts.

Port: 37

legacylighttime

tls — TLS / SSL

TCP

TLS / SSL is the Transport Layer Security encrypted-transport substrate used by HTTPS and most modern Internet protocols. Performs a TLS handshake and returns the negotiated version and cipher suite, supported versions and extensions, and the full server-certificate chain with subject, issuer, SANs, validity, and key metadata.

Ports: 443, 5986, 6443, 8443, 9443

encrypted

turn

UDP

TURN is the Traversal Using Relays around NAT protocol (RFC 5766) used as a media relay for WebRTC and VoIP. Sends an Allocate request and returns the SOFTWARE attribute, mapped and relayed addresses, allocation lifetime, requested transport, and any error code.

Ports: 3478, 3479, 5349, 5350

voip

ubnt — Ubiquiti Discovery

UDP

Ubiquiti Discovery is a device-discovery protocol used by UISP/UNMS and the Ubiquiti Discovery Tool. Sends the discovery probe and returns the device hostname, model, firmware, MAC, and IP configuration.

Port: 10001

discoverylightmgmt

unitronics — Unitronics PCOM

TCP

Unitronics PCOM is the proprietary protocol used to communicate with Unitronics Vision and Samba/UniStream PLC+HMI controllers. Queries the controller identification and returns the

model and OS version.

Port: 20256

ot

upnp — Universal Plug and Play

TCP

UPnP is the device-control protocol layered on top of SSDP. runZero fetches the device-description XML referenced in the SSDP LOCATION header to recover the friendly name, manufacturer, model, serial number, UPnP UUID, and the list of advertised services.

cleardiscoveryiot

uscan — Apple Image Capture (uscan)

TCP

uscan is the AirScan/Mopria HTTP scanner protocol used by macOS Image Capture, iOS Notes, and Mopria-compatible clients to discover and drive network scanners. Detected via the `_uscan._tcp` mDNS record and the `/eSCL/ScannerCapabilities` XML returned by the device.

cleardiscoveryiot

uscans — Apple Image Capture over HTTPS (uscans)

TCP

uscans is the TLS-protected variant of the AirScan/Mopria scanner discovery protocol, advertised via the `_uscans._tcp` mDNS record. runZero fetches the HTTPS `/eSCL/ScannerCapabilities` document to recover the device make, model, firmware, and supported scan profiles.

discoveryiot

vault — HashiCorp Vault

TCP

Vault is HashiCorp's secrets-management platform. runZero queries `/v1/sys/health` and `/v1/sys/seal-status` on the HTTP API to recover the Vault version, cluster name, cluster ID, sealed/initialized state, and replication mode.

Port: 8200

clearmgmt

vmauthd — VMware vmauthd

TCP

VMware vmauthd is the authentication daemon listening on VMware ESXi and Workstation hosts. Identifies the service from its 220 greeting banner.

Port: 902

lightmgmt

vmware

TCPTLS

vnc — VNC / RFB

TCP

VNC / RFB is the Virtual Network Computing remote-desktop protocol, also known as the Remote Frame Buffer protocol. Reads the protocol-version handshake and returns the RFB version, supported security types, and any disclosed vendor banner.

Ports: 5800, 5900, 5901, 5902, 5903

remote-accesstls

vsdp — Vivotek Search Discovery Protocol

UDP

VSDP is the UDP broadcast discovery protocol used by Vivotek IP cameras and video servers. runZero parses the response to recover the camera model, firmware version, MAC address, and IP configuration.

Port: 3702

discoveryiot**vxlan** — Virtual eXtensible LAN**UDP**

VXLAN is the L2-over-UDP tunneling protocol (RFC 7348) used by data-center overlay networks. runZero observes VXLAN encapsulation on UDP/4789 and records the VXLAN Network Identifier (VNI) of the inner Ethernet frame.

Port: 4789

clear**waveu** — Wave/UE Discovery**UDP**

WaveU is the UDP discovery service used by Crestron AirMedia and selected Wave-branded conferencing endpoints to advertise their presence to companion mobile and desktop apps. runZero parses the response to recover the model and firmware revision.

discoveryiot**wbsm** — IBM Web-Based System Manager**TCP**

IBM Web-Based System Manager (WSM/WebSM) is the remote-administration protocol used by the Java WSM client to manage AIX systems. Identifies the service from its banner.

Port: 9090

lightmgmt**wdbrpc** — VxWorks WDB Agent**UDP**

WDB RPC is the Wind River Workbench Debug Bridge agent built into many VxWorks images and exposed (often unintentionally) on UDP/17185. runZero issues a target-info query to recover the VxWorks version, BSP name, CPU type, and target name; presence of WDB indicates the host can be remotely controlled without authentication.

Port: 17185

clearmgmtot**webmin****UDP**

Webmin is a web-based Unix administration suite that exposes a UDP discovery service on UDP/10000 alongside Usermin and Virtualmin. Sends the "webmin" query and returns the advertised Webmin server IP, port, and HTTP/HTTPS scheme used to reach the management UI.

Port: 10000

lightmgmtweb**wiznet** — WIZnet Discovery**UDP**

WIZnet Discovery is a device-discovery protocol used by WIZnet serial-to-Ethernet modules and embedded TCP/IP chips. Sends the discovery probe and returns the raw advertised configuration fields.

Ports: 5000, 50001

discoveryiotlight**wsd** — Web Services Dynamic Discovery**UDP**

Web Services Dynamic Discovery (WS-Discovery) is the SOAP-over-UDP multicast discovery protocol used by Windows printers, scanners, and ONVIF IP cameras. runZero issues a Probe to ff02::c / 239.255.255.250 and parses the ProbeMatch response to recover the device's endpoint reference, types, scopes, and metadata URLs.

Port: 3702

cleardiscoveryiot

wsman — WS-Management

TCP

WS-Management (WS-Man) is the SOAP-over-HTTP/HTTPS management protocol used by Windows Remote Management (WinRM), iDRAC, iLO, and IPMI baseboard controllers. runZero issues an Identify request to recover the product vendor, product version, and supported protocol versions.

Ports: 5985, 5986

clearmgmt

x11 — X11 / X Window System

TCP

X11 / X Window System is the X Window System display protocol used by Unix graphical desktops; if exposed, it allows remote display capture and input injection. Performs an X11 connection-setup probe and returns the X.Org/XFree86 vendor string, protocol-major version, and the access state of the server.

Ports: 6000, 6001, 6002, 6003, 6004, 6005, 6006, 6007, 6008, 6009, 6010, 6011, 6012, 6013, 6014, 6015

clearlightremote-access

x2ap — X2AP (SCTP)

SCTP

X2AP (SCTP) is the X2 Application Protocol (3GPP TS 36.423) over SCTP used between LTE eNodeBs. Verifies the SCTP association and X2AP payload protocol identifier and returns endpoint identification.

Port: 36422

mobile-core

xdmcp

UDP

XDMCP is the X Display Manager Control Protocol used by X Window System login managers (xdm, gdm, kdm). Sends an XDMCP query and returns the responding manager's hostname and supported authentication types.

Port: 177

legacylightremote-access

zabbix — Zabbix Agent

TCP

Zabbix Agent is the monitoring agent used by the Zabbix server to collect host metrics. Sends an agent.version request and returns the agent version, derived CPE, and whether remote commands are enabled.

Ports: 10050, 10051

monitoringtls

zabbix-agent — Zabbix Agent

TCP

Zabbix Agent is the host-side collector polled by a Zabbix server over TCP/10050. runZero requests the agent.version and agent.hostname items to recover the agent build and configured hostname from unauthenticated agents.

Port: 10050

clearmgmt

zebra — Zebra Discovery

UDP

Zebra Discovery is a network discovery protocol used by Zebra Technologies label and barcode printers. Sends the discovery probe and returns the printer hostname disclosed in

the response.

Port: 6101

discoverylightprinting

zookeeper — Apache ZooKeeper

TCP

Apache ZooKeeper is a distributed coordination service whose wire protocol exposes four-letter administrative commands. Sends a four-letter (ruok/srvr/conf) command and returns the access state, mode (leader/follower/standalone), node count, and ZooKeeper version when the command is permitted.

Ports: 2181, 2888, 3888

databasemgmt

zyxel — Zyxel Device Web Management

TCP

Zyxel Device Web Management is the HTTP management interface exposed by Zyxel switches and routers (e.g. GS1200 series), serving a system_data.js endpoint that advertises model, firmware version, MAC, hostname, and IP configuration.

lightnetworktlsweb

Additional protocols

- anydesk
- banner
- citrixica
- h323
- iec60870-5-104
- imap
- ipp
- ipsec
- pulsar
- smpp
- snpp
- spice
- sstp
- wireguard
- xmpp

Default ports

runZero scans include extensive defaults and can be configured to cover any TCP or UDP port range.

What TCP ports does runZero scan?

runZero scans the following TCP ports by default:

7	13	17	21	22	37	53	80	88	102	110
111	113	135	137	139	179	264	389	427	443	445
464	502	512	513	514	515	523	535	541	554	593
631	636	749	750	771	789	873	902	995	1027	1080
1081	1099	1153	1200	1217	1433	1434	1521	1522	1525	1594
1718	1723	1883	1911	1935	1962	2000	2049	2121	2181	2222
2375	2376	2379	2404	2443	2455	2483	2484	2888	2947	3000
3023	3050	3080	3260	3268	3269	3306	3389	3390	3690	3868
3888	3999	4000	4001	4190	4222	4369	4545	4567	4712	4786
4840	4843	4911	4949	5000	5005	5006	5007	5037	5058	5060
5061	5094	5353	5355	5432	5433	5555	5666	5671	5672	5800
5900	5901	5902	5903	5938	5984	5985	5986	6000	6001	6002
6003	6004	6005	6006	6007	6008	6009	6010	6011	6012	6013
6014	6015	6222	6333	6334	6379	6432	6443	6481	6514	6556
6667	6668	6669	6697	7000	7001	7473	7474	7519	7676	7687
7734	7878	8000	8001	8002	8003	8004	8080	8081	8082	8086
8098	8161	8181	8193	8200	8222	8291	8443	8453	8500	8554
8728	8787	8788	8883	8888	8889	8983	9000	9001	9042	9050
9080	9090	9092	9093	9094	9100	9101	9102	9150	9160	9200
9418	9443	9595	9999	10000	10001	10050	10051	11211	11222	11740
13400	16379	18000	18245	19000	19530	20000	20256	20547	22222	25565
26379	27017	27018	27019	28017	33060	44818	48050	48898	54921	54922
54923	61616	61617	62078							

What UDP ports does runZero scan?

runZero scans the following UDP ports by default:

7	13	17	37	53	67	68	69	88	111	123
137	138	161	162	177	389	427	443	464	500	514
520	523	623	631	749	750	987	1194	1234	1235	1433
1434	1594	1604	1645	1646	1701	1812	1813	1900	2049	2123
2152	2228	2362	2427	2727	2944	2945	3250	3283	3386	3391
3478	3479	3671	3702	3956	4070	4433	4500	4569	4789	4800
5000	5058	5060	5061	5246	5349	5350	5351	5353	5355	5632
5683	5684	5783	6101	6481	6514	7400	7401	7410	7411	8611
8612	8805	9302	9600	9899	10000	10001	10161	10162	11211	12346
12366	12386	12406	12426	13400	17185	19132	20202	27036	30005	30490

30491 30718 34962 34963 34964 34980 37020 37810 41794 44818 46808 47808
47809 47810 47811 47812 47813 47814 47815 47816 47817 47818 47819 47820
47821 47822 47823 47824 48808 48898 50001

Scanning IoT and OT

Can I safely scan my IoT and OT environments?

Yes. IoT and OT equipment is often sensitive to high packet rates or malformed traffic, and past experiences with aggressive scanners have led many teams to put a “don’t scan” rule in place for these networks. runZero was purpose-built to operate safely in these environments, so most organizations can confidently include their IoT and OT assets in their active inventory.

runZero discovers assets using a lightweight active scan engine called the [Explorer](#). The Explorer can be deployed almost anywhere on your network — no SPAN or TAP ports to configure, and no agents to install on individual devices. Because discovery is performed actively from a single point on the network, you don’t have to modify the environment you’re trying to inventory.

What makes the Explorer safe for sensitive environments?

The Explorer was engineered from the ground up with fragile OT and IoT systems in mind. It is **not** based on `nmap`, `masscan`, or any other open source or commercial scanner. The Explorer only sends well-formed, standards-compliant traffic — it never sends malformed packets and never attempts to exploit vulnerabilities.

Safety controls built into every scan include:

- **Conservative, configurable scan rates.** Defaults are 1000 packets per second across the entire scan and 40 packets per second per host. Both values are adjustable, and traffic is balanced evenly across all targets in scope.
- **IP and port exclusions.** Critical assets, subnets, and individual TCP/UDP ports can be excluded globally or per scan.
- **Per-protocol probe controls.** Every UDP service probe — including each OT/IoT-specific probe — can be enabled or disabled individually.
- **Connection-aware pacing.** The Explorer balances SYN and ACK traffic and watches for port-exhaustion conditions on both the scanner and the target.
- **Configurable max group size.** Limits how many targets are probed concurrently, which directly controls the connection load placed on stateful middle boxes such as firewalls, proxies, and small routers.
- **Targeted port coverage.** Only TCP and UDP ports that yield actionable fingerprinting data are probed — not all 65535. The list is fully adjustable for specialized equipment using non-standard ports (see [Ports scanned by runZero](#)).
- **Per-protocol behavioral tuning.** Probe behavior is tailored to known-fragile services. For example, the Explorer will collect a banner from `9100/tcp` on a printer but will never send active probe data that could result in printed “garbage.”

For a step-by-step rollout plan, see the [Scanning OT networks](#) playbook.

Where should I deploy the Explorer?

There are two common patterns:

- **Co-locate with existing vulnerability scanners.** These hosts typically already have allow-lists, network reachability, and session-table sizing that account for scan traffic, which simplifies deployment.
- **Deploy dedicated Explorers at remote and OT sites.** Placing an Explorer on the same network segment as the assets it scans gives the best layer-2 visibility, eliminates middle-box concerns, and lets you tune scans independently per location.

For isolated OT networks with no internet access, deploy a [self-hosted runZero console](#) or use the [offline scanner](#) so Explorers never need to leave the OT zone.

OT/ICS vendors runZero fingerprints

runZero recognizes a wide range of OT, ICS, and IoT vendors out of the box, including:

- ABB
- Allen-Bradley / Rockwell Automation
- AutomationDirect
- Beckhoff
- BARIK
- Cisco
- CODESYS
- Control Solutions
- Control Techniques
- Danfoss
- Delta Electronics
- Digi
- Emerson
- Festo
- GE
- GENEREX
- GLC Controls
- Hilscher
- Hirschmann
- HMS Networks
- Honeywell
- ifm
- Johnson Controls
- Lantronix
- Linor Koda
- Mitsubishi Electric
- Moxa
- Murrelektronik
- Omron
- Phoenix Contact

- Pilz
- Pressac
- ProSoft Technology
- Red Lion
- Rittal
- Saia-Burgess
- Schneider Electric
- SICK
- Siemens
- Turck
- WAGO
- Weidmüller
- Westermo
- Yokogawa

IoT and OT protocols

runZero ships dedicated probes for the most common industrial, building-automation, and field-bus protocols. Each probe is standards-compliant and designed to elicit only the information needed to fingerprint a device. All probes can be toggled individually in your [scan templates](#).

Probe	Protocol	Typical port(s)	Notes
BACNET	BACnet/IP	47808/udp	HVAC, lighting, building management. Enumerates routed devices and BBMD/FDT entries as sub-assets.
ETHERNETIP	EtherNet/IP (CIP)	44818/tcp+udp, 2222/udp	Rockwell / Allen-Bradley and other CIP devices. Enumerates backplane modules as sub-assets.
MODBUS	Modbus/TCP	502/tcp	Tune <code>modbus-identification-level</code> to match device capabilities. Enumerates connected unit IDs as sub-assets.
S7COMM	Siemens S7Comm	102/tcp	Set <code>s7comm-request-extended-information</code> to true for richer data. Enumerates rack/slot modules as sub-assets.
KNXNET	KNXnet/IP	3671/udp	Building automation; enumerates KNX devices behind the IP interface.
DNP3	DNP3	20000/tcp+udp	Some outstations only talk to one master at a time — test before broad rollout.
PROFINET	PROFINET	34962-34964/udp	Cyclic and acyclic real-time control; enumerates slot/subslot modules.
ETHERCAT	EtherCAT	34980/udp	Enumerates slaves on the EtherCAT segment.
FINS	Omron FINS	9600/tcp+udp	Omron PLCs and compatible controllers; enumerates connected units.

MELSECQ	Mitsubishi MELSEC-Q	5006-5007/tcp	Returns CPU model and type; enumerates backplane CPUs.
HARTIP	HART-IP	5094/tcp	Multiplexers in front of HART field instruments; walks Cmd 84 sub-device indices.
IEC104	IEC 60870-5-104	2404/tcp	SCADA telecontrol; enumerates outstation addresses.
MMS	IEC 61850 MMS	102/tcp	Substation IEDs; enumerates logical devices.
ADS	Beckhoff ADS	48898/tcp	Beckhoff TwinCAT runtime; enumerates ADS sub-devices.
C37118	IEEE C37.118 Synchrophasor	4712/tcp	PMUs reported by Phasor Data Concentrators.
ATG	Veeder-Root Automatic Tank Gauge	10001/tcp	Fuel-tank monitoring at retail / fleet sites.
BSAP-IP	Emerson BSAP/IP	1234-1235/udp	Emerson ControlWave / Bristol RTUs in oil & gas SCADA.
C12.22	ANSI C12.22	1153/tcp	Electric-utility AMI head-ends and relays.
CSPV4	Allen-Bradley CSPv4 / PCCC	2222/tcp	Legacy SLC 5/05 and MicroLogix PLCs.
DOIP	Diagnostics over IP	13400/tcp+udp	In-vehicle ECUs over Ethernet; enumerates reachable ECUs.
FOCAS	Fanuc FOCAS	8193/tcp	Fanuc CNC machine tools and robots.
HSMS	SEMI HSMS / SECS-GEM	5000/tcp	Semiconductor fab equipment (SEMI E37).
MBUS-TCP	M-Bus over TCP	8888-8889/tcp	Tunneled EN 13757 utility-meter aggregators.
OPCUA	OPC UA	4840/tcp, 4843/tcp	Vendor-neutral industrial information model and data access.

Fifteen of these probes (BACnet, CIP, Modbus, KNXnet, S7Comm, DNP3, PROFINET, EtherCAT, Omron FINS, MELSEC-Q, HART-IP, IEC 60870-5-104, IEC 61850 MMS, Beckhoff ADS, IEEE C37.118) also enumerate downstream devices behind the gateway and report each as its own sub-asset linked back to the gateway. See [Protocol gateways](#) for how that works and how to query the resulting sub-assets.

You can find the complete list in our [protocol catalog](#).

runZero recommends starting with the OT-specific probes disabled, validating a limited scan against a representative subset of devices, and then enabling additional protocols as your

team gains confidence. See the [Scanning OT networks](#) playbook for a recommended phased rollout.

Platform integrations

Inbound integrations

Enriching runZero results with data from other tools

The runZero platform offers integrations with several sources of asset data, allowing users to enrich their asset inventory and identify assets and subnets that are not effectively managed or protected. By leveraging product APIs and export/import functionality, runZero can pull data from many IT and security tools to extend visibility across your organization's network.

Supported integrations

Cloud and virtualization

- [Amazon Web Services](#)
- [Google Cloud Platform](#)
- [Microsoft Azure](#)
- [Prisma](#)
- [VMware](#)
- [Wiz](#)

Endpoint protection

- [CrowdStrike Falcon](#)
- [Microsoft 365 Defender](#)
- [Microsoft Intune](#)
- [Miradore MDM](#)
- [SentinelOne](#)
- [Tanium API Gateway](#)

Endpoint management

- [Microsoft Endpoint Configuration Manager \(MECM\)](#)

Asset and identity management

- [Google Workspace](#)
- [Microsoft Active Directory](#)
- [Microsoft Entra ID \(formerly Azure AD\)](#)

Vulnerabilities and risk

- [Censys Search & Data](#)
- [Censys Platform](#)
- [Dragos](#)
- [Qualys VMDR](#)
- [Rapid7](#)
 - [InsightVM](#)
 - [Nexpose](#)
- [Shodan](#)
- [Tenable](#)
 - [Tenable Vulnerability Management](#)
 - [Tenable Nessus Professional](#)
 - [Tenable Security Center](#)
 - [Tenable Nessus \(file import\)](#)

Network management

- [Cisco Meraki Dashboard](#)
- [Tailscale](#)

Custom integrations

If the solution you want to draw data from isn't available as a current runZero integration, Platform and Community users can leverage the [custom integrations](#) feature to add asset data from custom sources. The primary path is [custom integration scripts](#), which run on your Explorers or on the console and can pull from REST APIs or directly over SSH, SMB, WinRM, WMI, and SQL; a library of over a hundred ready-to-use scripts is available in the [runzero-custom-integrations](#) repository and from the example picker in the console's script editor. Where AI is enabled, the console can also [generate a script](#) from a plain-language description. Custom asset data can also be imported through the API or the [runZero Python SDK](#).

Each custom integration is either available to every organization in the account or only to the organizations it has been granted to, and administrators can only edit the integrations belonging to organizations they administer. See [organization access](#) for details.

Scan probes or connector tasks

Most integrations can be run either as a scan probe or a connector task.

Scan probes run as part of a scan task. The scan task can be used to scan your environment and sync integrations at the same time. To run an integration as a scan probe:

1. Configure a scan task from the *Scan* menu in your inventory or tasks page.
2. Activate the integration under the *Probes* tab.

3. Activate the correct credential under the *Credentials* tab.
4. Configure, activate, or deactivate other scan task configuration options as preferred.

Connector tasks run independent of scan tasks in order to allow more finely tuned scheduling of integration syncs and asset scans. Connector tasks are run from the runZero cloud by default, but can be configured to run from an Explorer in your organization if preferred. To run an integration as a connector task:

1. Configure a connector task from the [Integrations page](#) or the *Integrate* menu in your inventory or tasks page.
2. Select an Explorer from the *Explorer* menu (optional).
3. Configure, activate, or deactivate other connector task configuration options as preferred.

Importing integration data

Some integrations can be used by importing data from that platform into runZero. For example, .nessus files from [Tenable Nessus](#) and .xml files from [Rapid7 Nexpose](#) can both be ingested without requiring a connection to their APIs.

Automatic asset merge

How runZero maps integration assets to assets:

- For hosts that can be matched to an existing runZero asset, asset-level attributes will be updated, and integration-specific attributes will be added.
- For hosts that cannot be matched with an existing runZero asset, a new asset will be created in the site specified when the integration task is set up.

runZero is able to merge integration data into existing assets by the following, in priority order:

1. MAC address
2. IP address (3-day window)
3. Hostname

Assets from integrations can also be manually merged into runZero assets using the *Merge* button on the [Asset Inventory](#) page.

Removing an integration data source

When an integration is removed as a data source, the associated attributes are removed from your runZero assets. Since some asset attribute fields are merged, it is possible that attributes populated by both runZero scans and the integration could be deleted. Rescanning the affected assets will resolve this issue.

Excluding integration attributes

Your account and organizations can be configured to prevent the collection of certain integration attributes; for example, personally identifiable information (PII) such as usernames or email addresses. Please note that some attributes are crucial to runZero's merging and fingerprinting methods, and excluding these attributes may lead to inaccuracies in your inventory.

Excluded attributes can be configured in your [Account settings](#) under *Personally Identifiable Information (PII) collection* to apply across all organizations, or configured within individual [Organization](#) settings to override the account-level settings.

You can choose to use the default list of attributes or define your own. The attributes must be in the format `@source.resourceType.attributeName`. For example, `@crowdstrike.dev.firstLoginUser` excludes the attribute "firstLoginUser" from CrowdStrike devices.

Only integrations that import PII support attribute exclusions, at this time those integrations are:

- CrowdStrike
- SentinelOne
- Tanium
- Google Workspace
- Intune
- Miradore

Attribute exclusions are not supported for the Directory Users and Directory Groups inventories. The AzureAD, LDAP, and Google Workspace integrations import directory users and groups by default, but this can be disabled in the task configuration.

Source names and IDs

The table below maps the source name to the source ID for querying assets and vulnerabilities.

ID	Name	Description
-1	custom	Custom
1	runzero	runZero
2	miradore	Miradore
3	aws	AWS
4	crowdstrike	CrowdStrike
5	azure	Azure
6	censys	Censys

7	vmware	VMware
8	gcp	GCP
9	sentinelone	SentinelOne
10	tenable	Tenable
11	nessus	Nessus
12	rapid7	Rapid7
13	insightvm	InsightVM
14	qualys	Qualys
15	shodan	Shodan
16	azuread	AzureAD
17	ldap	LDAP
18	ms365defender	MS365Defender
19	intune	Intune
20	googleworkspace	GoogleWorkspace
21	sample	Sample
22	tenablesecuritycenter	TenableSecurityCenter
23	packet	Packet
24	wiz	Wiz
25	meraki	Meraki
26	mecm	MECM
27	tanium	Tanium
28	simulator	Simulator
29	netbox	NetBox
30	cip	CIP
31	pan	Palo Alto Networks
32	prisma	Prisma
34	dragos	Dragos
35	bacnet	BACnet
36	modbus	MODBUS
37	knxnet	KNXnet
38	insightvmcloud	InsightVMCloud

39	s7comm	S7comm
40	dnp3	DNP3
41	profinet	PROFINET
42	ethercat	EtherCAT
43	omronfins	OmronFINS
44	melsecq	MELSEC-Q
45	hartip	HART-IP
46	c37118	C37.118
47	ads	ADS
48	opcua	OPC-UA
49	s7	S7
50	iec104	IEC60870-5-104
51	mms	IEC61850-MMS
52	censysplatform	CensysPlatform
10001	tailscale	Tailscale
10002	ninjaone	NinjaOne

Amazon Web Services

runZero integrates with Amazon Web Services (AWS) to provide better visibility across your cloud environment. This integration imports data from each applicable API to add detailed information to your asset inventory:

- [AWS EC2 API](#)
- [AWS RDS API](#)
- [AWS ELBv1 API](#)
- [AWS ELBv2 API](#)
- [AWS Lambda API](#)
- [AWS FSx API](#)
- [AWS Inspector2 API](#)

Syncing with AWS allows you to quickly identify the number of EC2 instances, elastic load balancers, relational database services, FSx file systems, and VPCs you have running, as well as their region, account, and more.

This integration supports the import of all running EC2 instances, RDS instances, Lambda functions, FSx file systems, VPCs, and active application, network, gateway, or classic load balancers. It can be configured to connect to a single AWS account or to all accounts in your organization and imports data across multiple regions.

Getting started

The following AWS resource types are supported:

- EC2 instances
- Elastic load balancers
- RDS instances
- Lambda instances
- FSx file systems
- VPCs

To set up the AWS integration, you'll need to:

1. Configure AWS to allow API access through runZero.
2. Add the AWS credential to runZero, which includes the access key and secret key.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the AWS integration to sync your data with runZero.

Requirements

Before you can set up the AWS integration:

- Make sure you have access to the AWS console.
- Make sure you are using [AWS Organizations](#) if you want to connect to multiple accounts.
 - When using [AWS Organizations](#) with compatible credentials, such as the [IAM Role Credential](#), we enumerate member accounts to sync all data.

Step 1: Create an AWS credential

There are multiple options for creating an AWS credential:

- [IAM Role Credential](#) (recommended) - runZero authenticates to AWS using an IAM role that you create in your AWS account. No long-term access keys are needed.
 - Note that IAM Role credentials are not supported for self-hosted instances, GovCloud-based accounts, or Explorer tasks, and IAM Key credentials should be used in this case instead.
- [IAM Role Credential - Manual Account Selection](#) (alternative) - runZero authenticates to AWS in the same manner as the IAM Role Credential, but using manually-specified account IDs rather than AWS Organization subaccounts. Recommended only for advanced use cases.
- [IAM Key Credential](#) (deprecated) - runZero authenticates to AWS using an IAM user and long-term access keys.

Option A: IAM Role Credential

When using an IAM role to provide runZero with authenticated access to AWS, you'll need to establish an **External ID**. An external ID is a unique ID provided by runZero for your IAM role. You'll use this external ID when configuring your IAM role to ensure that only your runZero account can access the role.

To learn more about the use of external IDs in cross-account access, refer to [AWS' documentation on the confused deputy problem](#).

Step 1a.1: Create a runZero credential and external ID

To generate an external ID, you'll need to create an AWS IAM Role credential in the runZero console:

1. In the runZero console, navigate to the [Credentials page](#) and click **Add Credential**.
2. Choose **AWS IAM Role** from the list of credential types.
3. Provide a name for the credential, like `AWS IAM Role`.
4. Enter the full ARN of the AWS IAM role you plan to create in the next step into the AWS IAM role ARN field. An IAM role ARN has the format `arn:aws:iam::123456789012:role/runzero-sync`, where `123456789012` is your AWS account ID and `runzero-sync` is the name of the role you will create.

If you don't know your AWS account ID, that's ok. You can use `arn:aws:iam::XXXXXXXXXXXX:role/runzero-sync` as a placeholder and update it later.

5. Click **Save** to save the credential and generate an external ID.

Note: Clicking Verify & Save will not work because you haven't created the IAM Role in AWS yet. You'll come back and test the credential in Step 1a.3.

6. Copy the external ID that was generated. You'll need it in the next step.

Step 1a.2: Create AWS IAM role(s)

You must create an IAM role using the directions below in every AWS account you want to sync with runZero. If you utilize AWS Organizations, you must reuse the same role name and external ID from your credential in each account.

The preferred way to create an IAM Role is using CloudFormation StackSets:

1. Create a `runzero-cloudformation-stackset.json` file with the following content. Replace `<externalID>` with the external ID generated using the runZero console.

```
"Resources": {
  "IamrunZeroAccessRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
```

```

"AssumeRolePolicyDocument": {
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Principal": {
      "AWS": "arn:aws:iam::336004405641:root"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "StringEquals": {
        "sts:ExternalId": "<externalID>"
      }
    }
  }]
},
"ManagedPolicyArns": [
  "arn:aws:iam::aws:policy/AmazonEC2ReadOnlyAccess",
  "arn:aws:iam::aws:policy/AmazonRDSReadOnlyAccess",
  "arn:aws:iam::aws:policy/AWSLambda_ReadOnlyAccess",
  "arn:aws:iam::aws:policy/AmazonFSxReadOnlyAccess",
  "arn:aws:iam::aws:policy/AWSOrganizationsReadOnlyAccess",
  "arn:aws:iam::aws:policy/AmazonInspector2ReadOnlyAccess"
]
}
}
}

```

2. Go to **CloudFormation > StackSets** and click **Create StackSet**.
3. Select **Template is ready** and **Upload a template file**. Upload the template you downloaded earlier.
4. Enter a name for the StackSet, for example runZero-IAM-Role. Click next.
5. (Optional) Set tags. Click next.
6. Set deployment options. Click next.
7. Review and create the StackSet.
8. The StackSet will create an IAM Role of the format: `arn:aws:iam::<accountID>:role/runzero-sync`. Copy the full ARN of the role you created. You'll need this in Step 1a.3.

Alternatively, to create the IAM Role(s) manually:

1. Sign in to the AWS console.
2. Go to **Identity and Access Management (IAM) > Roles** to create a role that will provide access to runZero.
3. Click the **Create role** button
4. On the "Select trusted entity" step:
 1. Select **AWS account** in the **Trusted entity type** panel
 2. Select **Another AWS account** in the **An AWS account** panel
 3. Enter 336004405641 for the account ID. This is the account runZero uses to access your AWS account.
 4. Check the **Require external ID...** checkbox and enter the external ID that was generated in Step 1a.1.
 5. Click the **Next** button to proceed to the next step.

5. On the “Add permissions” step, select the following AWS managed policies and click the **Next** button to proceed to the next step. These grant runZero read-only access to read inventory from your AWS account.
 - AmazonEC2ReadOnlyAccess to sync EC2, ELB, and VPC resources
 - AmazonRDSReadOnlyAccess to sync RDS instances
 - AWSLambda_ReadOnlyAccess to sync Lambda functions
 - AmazonFSxReadOnlyAccess to sync FSx file systems
 - AWSOrganizationsReadOnlyAccess to sync AWS Organization account information
 - AmazonInspector2ReadOnlyAccess to import [AWS Inspector](#) vulnerability findings for EC2 instances and Lambda functions
6. On the “Name, review, and create” step, enter a relevant role name, such as runzero-sync, and description for the role. Once confirmed click the **Create role** button.
7. Copy the full ARN of the role you just created. It will have the format `arn:aws:iam::<accountID>:role/<roleName>`. You'll need this in Step 1a.3.

Additional Notes:

- Often, payer accounts are not able to be StackSet-configured by default. In these cases, the StackSet approach can configure all member accounts, and the payer account may need to be manually configured.
- If you intend to sync with AWS regions that are not enabled by default, [change the region compatibility of the global STS endpoint](#). To do this, in the AWS console navigate to IAM → Account Settings → Security Token Service (STS). Then change the Global Endpoint to be compatible with “All AWS Regions.”

Step 1a.3: Update the runZero credential

1. Return to the runZero console, navigate to the [Credentials](#) page and edit to the Credential you created in Step 1a.1.
2. Update the **AWS IAM role ARN** field with the full ARN of the role you created in Step 1a.2.
3. Use the **Verify & Save** button to save and test the credential.

You're now ready to set up and activate the connection to bring in data from AWS. Continue to [Step 2](#).

Option B: IAM Role Credential - Manual Account Selection

This option should only be chosen when the default IAM Role Credential does not provide sufficient flexibility in targeting a subset of AWS accounts. Please consider this carefully, as there is additional operational overhead in using it.

Step 1b.1: Create a runZero credential and external ID

This step should look very similar to step 1a.1.

To generate an external ID, you'll need to create an AWS IAM Role credential in the runZero console:

1. In the runZero console, navigate to the [Credentials](#) page and click **Add Credential**.

2. Choose **AWS IAM Role - Manual Account Selection** from the list of credential types.
3. Provide a name for the credential, like `AWS IAM Role - Manual`.
4. Enter the name of the role you are using for all accounts into the AWS IAM Role field.
5. Enter a newline-separated list of the account IDs you would like to import asset information from. An AWS account ID will be a 12-digit number, so an entry would look like `123456789012`.
6. Click **Save** to save the credential and generate an external ID.

Note: Clicking Verify & Save will not work because you haven't created the IAM Role in AWS yet. You'll come back and test the credential in Step 1b.3.

7. Copy the external ID that was generated. You'll need it in the next step.

Step 1b.2: Create AWS IAM role(s)

Refer to step 1a.2, as this uses the same setup as that credential type. Note that the setup will need to be repeated for each account that you intend to import asset data from, as this credential type does not use AWS Organizations like the IAM Role Credential.

Step 1b.3: Update the runZero credential

1. Return to the runZero console, navigate to the [Credentials](#) page and edit to the Credential you created in Step 1b.1.
2. Use the **Verify & Save** button to save and test the credential.

You're now ready to set up and activate the connection to bring in data from AWS. Continue to [Step 2](#).

Option C: IAM Key Credential

If you created an IAM Role using Option 1 above, you may skip this step and proceed to [Step 2](#) below. This section describes how to authenticate runZero to AWS using an IAM user and long-term access keys. Using long-term access keys is not recommended. We recommend using an IAM Role instead.

Step 1c.1: Create an AWS IAM user

1. Sign in to the AWS console. If you are using AWS Organizations and intend to sync with multiple accounts, sign into the AWS Organizations management account, sometimes called the "payer" or "root" account.
2. Go to **Identity and Access Management (IAM) > Users** and select or create a user that will provide API access to runZero.
3. Click **Add permissions > Attach existing policies directly**. Search for and attach the following policies based on the services you would like to sync:
 - `AmazonEC2ReadOnlyAccess` to sync EC2, ELB, and VPC resources
 - `AmazonRDSReadOnlyAccess` to sync RDS instances

- AWSLambda_ReadOnlyAccess to sync Lambda functions
 - AmazonFSxReadOnlyAccess to sync FSx file systems
 - AWSOrganizationsReadOnlyAccess to sync AWS Organization account information
 - AmazonInspector2ReadOnlyAccess to import AWS Inspector vulnerability findings for EC2 instances and Lambda functions
4. Add an inline policy to allow assuming runZero roles in sub-accounts (only required if using AWS Organizations):

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Effect": "Allow",
    "Action": "sts:AssumeRole",
    "Resource": "arn:aws:iam::*:role/<rolename>"
  }]
}
```

5. From the user summary screen, open the **Security credentials** tab and click on **Create access key**.
6. Save the **Access key ID** and **Secret access key**. You'll need them in Step 1c.3.
7. If you intend to sync with AWS regions that are not enabled by default, [change the region compatibility of the global STS endpoint](#) to be compatible with All AWS Regions. To do this, in the AWS console navigate to IAM → Account Settings → Security Token Service (STS). Then change the Global Endpoint to be compatible with "All AWS Regions."

Step 1c.2: Add an IAM role to sub-accounts

If you are using AWS Organizations and want runZero to sync with all accounts in your organization, you must create an IAM role in each sub-account that allows the IAM user you created in Step 1c.1 to assume the role. If you aren't using AWS Organizations, you may skip this step and proceed to Step 1c.3.

In every AWS sub-account:

1. Go to **Identity and Access Management (IAM) > Roles** and click **Create role**.
2. Choose **Another AWS Account** for the type of trusted entity.
3. For Account ID, enter the ID for your organization's management account.
4. Click **Next: Permissions**.
5. Attach the AmazonEC2ReadOnlyAccess policy if you want to sync EC2, ELB, and VPC resources.
6. Attach the AmazonRDSReadOnlyAccess policy if you want to sync RDS instances.
7. Attach the AWSLambda_ReadOnlyAccess policy if you want to sync Lambda functions.
8. Attach the AmazonFSxReadOnlyAccess policy if you want to sync FSx file systems.
9. Attach the AmazonInspector2ReadOnlyAccess policy if you want to import AWS Inspector vulnerability findings for EC2 instances and Lambda functions.
10. Click **Next: Tags** and add tags optionally.
11. Click **Next: Review** and provide a name for the role. (The role must be named the same for all accounts)
12. Click **Create role**.

Step 1c.3: Add the AWS credential to runZero

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Provide a name for the credential, like `AWS IAM User`.
3. Choose **AWS Access & Secret** from the list of credential types.
4. Provide the following information:
 - **AWS access key** - Access key ID obtained from Step 1c.1.
 - **AWS secret access key** - Secret access key obtained Step 1c.1
 - **AWS role** - Assumed role used to connect to other accounts in your organization. This is the role name you used in Step 1c.2.
 - Select the region(s) that you want to sync.
5. Save the credential.

Step 2: Choose how to configure the AWS integration

The AWS integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 3: Set up and activate the AWS integration to sync data

After you add your AWS credential, you'll need to set up a connector task or scan probe to sync your data.

Step 3a: Configure the AWS integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new AWS-only assets are created.

1. Activate a [connection to AWS](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like `AWS sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**:
 - Choose the site you want to add your assets to. All newly discovered assets will be stored in this site. You can also choose to **Automatically create a new site per VPC** or **Automatically create a new site per account**, and runZero will take care of creating the sites for newly discovered assets or accounts.
 - Choose whether to **automatically delete stale AWS assets**. If you check this option, runZero will automatically delete AWS assets previously seen in AWS that were not found in the most recent sync.
 - Choose whether to include AWS assets that are not currently running. If you check this option, runZero will import AWS asset data for assets that are not in a running state.

6. Under **Service options**, select the services you would like to sync data from. You must choose at least one.
7. If you want to import [AWS Inspector](#) vulnerability findings for your EC2 instances and Lambda functions, switch the **Import AWS vulnerabilities** toggle to *Yes*. This requires the credential's IAM role or user to have the `AmazonInspector2ReadOnlyAccess` managed policy (or equivalent `inspector2:ListFindings` permission) attached in each account you sync — see [Step 1: Create an AWS credential](#) above. Findings are only imported for resources that AWS Inspector has already scanned and matched to an existing EC2 instance or Lambda function.
8. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
9. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 3b: Configure the AWS integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the AWS toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the AWS toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 4: View AWS assets

After a successful sync, you can [go to your inventory](#) to view your AWS assets. These assets will have an AWS icon listed in the **Source** column.

To filter by AWS assets, consider running the following queries:

- [View all AWS assets:](#)

```
source:aws
```

- [View all AWS EC2 instances:](#)

```
source:aws AND has:"@aws.ec2.instanceID"
```

- [View all AWS Elastic Load Balancers:](#)

```
source:aws AND (has:"@aws.elb.loadBalancerArn" OR has:"@aws.elb.loadBalancerName")
```

- [View all AWS VPCs:](#)

```
source:aws AND has:"@aws.vpc.vpcID"
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by the AWS APIs.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Amazon Web Services integration unable to connect?

1. Are you getting any data from the AWS integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the AWS integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Why aren't AWS Inspector vulnerabilities showing up on my EC2 or Lambda assets?

1. Confirm the **Import AWS vulnerabilities** option is enabled on the connector task or scan probe. This is separate from selecting the EC2 or Lambda service options.
2. Confirm the IAM role or user used by the credential has the `AmazonInspector2ReadOnlyAccess` managed policy (or a custom policy granting `inspector2:ListFindings`) attached in the specific AWS account and region being synced. This permission is required in *every* member account, not just the payer/root account. If this permission is missing, the task log will show an `AccessDeniedException` similar to:

```
AccessDeniedException: User: arn:aws:sts::<account-id>:assumed-role/<role-name>/runZeroAWSIntegrationSession is
authorized to perform: inspector2:ListFindings on resource: arn:aws:inspector2:<region>:<account-id>:/findings/1
because no identity-based policy allows the inspector2:ListFindings action
```

3. Confirm that AWS Inspector scanning (EC2 deep inspection and/or Lambda standard scanning) is activated for the account and region, and that the resource has actually been scanned. runZero only imports findings that AWS Inspector has already generated — it does not trigger new scans.
4. For Lambda functions, confirm the function has been invoked or updated within the last 90 days, is using the `$LATEST` version, and is not excluded from scanning by tag or a customer-managed KMS key.

Enriching scans with EC2

As part of a discovery scan, runZero will automatically enrich scanned assets with data from the AWS EC2 API when available. runZero assets will be updated with internal IP addresses, external IP addresses, hostnames, MAC addresses, and tags, along with other EC2-specific attributes, such as the account ID and instance type.

No additional configuration is needed in runZero to get this data enrichment. However, you may need to [modify the permissions](#) associated with the instance's IAM role.

Find Explorers with EC2 enrichment capabilities

To use the EC2 enrichment capabilities, the runZero Explorer must be running on an instance with permissions to describe your EC2 instance list. This can be configured through an IAM role associated with the instance as well as by configuring the AWS credentials for the root user account.

To identify the Explorers with this capability, view your [registered Explorers](#). Any Explorer with the cloud icon indicates that it can enumerate EC2 instances.

Scans run from these Explorers will merge any EC2 instance fields into the asset automatically for any in-scope targets matched against the instance list.

Add permissions to describe instances

To allow for EC2 instance data enrichment, you will need to add the EC2 `ec2:DescribeInstances` permission for the instance role configured for your instance. Visit the [Amazon docs](#) to learn how to create and update policies.

Basically, your policy will look like:

```
{
  "Version": "2012-10-17",
  "Statement": [{
    "Sid": "VisualEditor0",
    "Effect": "Allow",
    "Action": [
      "ec2:DescribeInstances",
    ],
    "Resource": "*"
  }]
}
```

From the IAM UI, go to **Roles > Permissions > Attach policies** and search for the EC2 service. From the actions, select `DescribeInstances`, which is located under List.

You can also configure credentials on the instance by running `aws configure` as root, instead of using the IAM instance role.

After you save your policy, you'll need to restart your Explorer. The easiest way to do this is to force an update from the Explorer menu.

Keep in mind if your configuration uses one region, but the instance is located in another, it will use the instance's region instead for all API requests.

Attributes runZero gets from the EC2 API

When runZero determines that an IP address is also an EC2 instance, it will enrich the existing runZero data with EC2 metadata. runZero will be able to pull in more hostnames based on AWS asset tags, MAC addresses, internal and external IPs.

Some attributes that runZero is able to get from the EC2 API metadata:

- aws.accountID
- aws.architecture
- aws.availabilityZone
- aws.hypervisor
- aws.imageID
- aws.instanceID
- aws.instanceType
- aws.ipv4
- aws.ipv6
- aws.keyName
- aws.launchTimeTS
- aws.macs
- aws.privateDNS
- aws.privateIP
- aws.publicDNS
- aws.publicIP
- aws.region
- aws.rootDeviceName
- aws.rootDeviceType
- aws.scanner.instanceID
- aws.scanner.instanceType
- aws.state
- aws.subnetID
- aws.tags
- aws.tenancy
- aws.virtualizationType
- aws.vpcID

Additionally, runZero will also report on other things that respond to the scan, but don't match an EC2 entry. You may see things like Amazon RDS, temporary ELBs, and Lambdas in your inventory, but you will only see EC2 metadata on EC2 assets.

Censys Search & Data

runZero supports importing assets from the [Censys Search API](#).

Censys Search API

To get started with the Censys Search API, you will need to register for a [Censys Search account](#). Once you have done so, you can find your API credentials in the [My Account](#) section.

In runZero, go to the Credentials page, and click [Add Credential](#). Select *Censys Search API Key* as the credential type, and enter your API ID and API secret.

You can now go to your [asset inventory](#), click the **Connect** button, and choose **Censys Search API**. Select the credential you just created from the Censys Search credential dropdown.

Configuration

There are two modes for connecting runZero to the Censys Search API.

- **Custom Query** mode - runZero runs a Censys search query you specify, and then imports all of the results into runZero. The search query should be in [Censys Search Language](#). It is a good idea to test your query using the main [Censys Search 2.0](#) interface before running an import task.
- **All Assets** mode - runZero assembles a list of public IP addresses from all of the assets in the selected site, and then uses the API to find Censys Search information for those addresses. The information found is imported into runZero and merged into the appropriate assets.

As with a runZero scan, you'll need to select a site to contain the scan data. The usual task scheduling options are available.

When you have finished editing the Censys Search configuration, click **Activate Connection**.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Censys Search integration unable to connect?

1. Are you getting any data from the Censys Search integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this integration documentation and follow the steps exactly.

3. If the Censys Search integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Censys Platform

runZero supports importing assets from the [Censys Platform API](#).

- [Importing assets from the Censys Platform API](#)

Censys Platform API

To get started with the Censys Platform API, you will need to register for a [Censys Platform account](#). Once you have done so, you can find your API credentials in the [My Account](#) section.

In runZero, go to the Credentials page, and click [Add Credential](#). Select *Censys Platform API Key* as the credential type, and enter your API ID and API secret.

You can now go to your [asset inventory](#), click the **Connect** button, and choose **Censys Platform API**. Select the credential you just created from the Censys Platform credential dropdown.

Configuration

There are two modes for connecting runZero to the Censys Platform API.

- **Custom Query** mode - runZero runs a Censys query you specify and then imports all the results into runZero. The search query should be in [Censys Query Language](#). It is a good idea to test your query using the main [Censys Platform](#) interface before running an import task.
- **All Assets** mode - runZero assembles a list of public IP addresses from all the assets in the selected site, and then uses the API to find Censys information for those addresses. The information found is imported into runZero and merged into the appropriate assets.

As with a runZero scan, you'll need to select a site to contain the scan data. The usual task scheduling options are available.

When you have finished editing the Censys Platform configuration, click **Activate Connection**.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Censys Platform integration unable to connect?

1. Are you getting any data from the Censys Platform integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this integration documentation and follow the steps exactly.
3. If the Censys Platform integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Cisco Meraki Dashboard

runZero integrates with Cisco Meraki Dashboard by importing data from the [Cisco Meraki Dashboard API](#). This integration allows you to sync data about your devices and network clients from Meraki to provide better visibility of your network.

Getting started with Meraki

To set up an integration with Meraki, you'll need to:

1. Generate an API key for your Cisco Meraki Dashboard administrator account.
2. Configure the Meraki credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Requirements

Before you can set up the Meraki integration:

- Make sure you have administrator access to the Meraki Dashboard.

Step 1: Generate an API key in Cisco Meraki Dashboard

1. Sign in to Cisco Meraki Dashboard with an administrator account.
2. Check that access to the Cisco Meraki Dashboard API is enabled. This must be enabled for each organization you would like to sync.
 1. Navigate to **Organization > Settings** and enable **API access** under **Dashboard API access**.
3. Navigate to **My profile** to generate an API key. The API key will inherit the same permissions as the account that created it.

Step 2: Add the Meraki API key to runZero

1. Go to the [Credentials page](#) in runZero.
2. Choose **Meraki API Key** from the list of credential types.
3. Provide a name for the credential, like Meraki.
4. Provide the following information:
 - **Meraki API URL** - The API Endpoint URL used to access the Cisco Meraki Dashboard API.
 - **Meraki API key** - The API key for the Cisco Meraki Dashboard administrator account.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Save the credential.

You're now ready to set up and activate the connection to bring in data from Meraki.

Step 3: Choose how to configure the Meraki integration

The Meraki integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your Meraki credential, you'll need to sync your data from Meraki.

Step 4a: Configure the Meraki integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Meraki-only assets are created.

1. Activate a connection to [Meraki](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Optionally provide a list of organization IDs or names to include in the import. The list must be comma-separated. We will only import data for the organizations specified.
4. Optionally provide a list of network IDs or names to include in the import. The list must be comma-separated. We will only import data for the networks specified.
5. Optionally provide a comma-separated list of VLANs to exclude from the import. We will not import devices associated with the specified VLANs.
6. If you want to exclude assets that are not associated with a VLAN, set the **Exclude clients with no VLAN** option to **Yes**.
7. Optionally provide a comma-separated list of SSIDs to exclude from the import. We will not import devices connected on the specified SSIDs.

8. Enter a name for the task, like `Meraki Sync` (optional).
9. Choose the Explorer to perform this connector task from (optional).
10. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
11. Enter a description for the task (optional).
12. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
13. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
14. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Meraki integration as a scan probe

You can run the Meraki integration as a scan probe so that the runZero Explorer will pull your Meraki assets into the runZero Console.

In a new or existing scan configuration:

- Ensure that the `MERAKI` option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct `MERAKI` credential to *Yes* in the *Credentials* tab.

Step 5: View Meraki assets

After a successful sync, you can [go to your inventory](#) to view your Meraki assets. These assets will have a Cisco icon listed in the **Source** column.

To filter by Meraki assets, consider running the following queries:

- [View all Meraki assets:](#)

```
source:Meraki
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from Meraki.

CrowdStrike Falcon

runZero integrates with CrowdStrike by importing data through the [CrowdStrike Falcon API](#). This integration allows you to sync and enrich your asset inventory, as well as ingesting vulnerability data from Falcon Spotlight and software data from Falcon Discover. Adding your CrowdStrike data to runZero makes it easier to find things like endpoints that are missing an EDR agent.

Getting started

To set up the CrowdStrike integration, you'll need to:

1. Configure CrowdStrike to allow API access through runZero.
2. Add the CrowdStrike credentials, which will include the client ID and client secret, and CrowdStrike base API URL in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the CrowdStrike integration to sync your data with runZero.

Requirements

Before you can set up the CrowdStrike integration:

- Make sure you have access to the CrowdStrike admin portal.

Step 1: Configure CrowdStrike to allow API access to runZero

1. Sign in to CrowdStrike.
2. Go to **Support > API Clients and Keys**. When the **API Key** page appears, choose to add a new API client.
3. Provide the following details for the API client:
 - **Client name:** API client name, such as runZero.
 - **API scope:**
 - To ingest host details, include read permissions for **Hosts** and **Host Groups**.
 - To ingest vulnerability data, include read permissions for **Vulnerabilities**.
 - To ingest software data, include read permissions for **Assets**.
4. When you are done, add the client. An API client created window appears and shows you the client ID and client secret. You'll need them to configure the integration in runZero.
5. Copy the client ID and client secret now. You may not be able to get them later.

Step 2: Add the CrowdStrike credentials to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like CrowdStrike Falcon.
2. Choose **CrowdStrike Falcon API key** from the list of credential types.
3. Provide the following information:
 - **CrowdStrike client ID and CrowdStrike client secret** - To generate your client ID and client secret, go to **Support > API Clients and Keys > OAuth2 API clients > Add new API Client** in your CrowdStrike portal.
 - **CrowdStrike API URL** - Your organization-specific base URL, which will depend on your account type. You can find this in the CrowdStrike API Swagger documentation.
 - For a **US-1** account use `api.crowdstrike.com`
 - For a **US-2** account use `api.us-2.crowdstrike.com`
 - For a **US-GOV-1** account use `api.laggar.gcw.crowdstrike.com`
 - For a **EU-1** account use `api.eu-1.crowdstrike.com`
4. If you want other organizations to be able to use these credentials, select the *Make this a global credential* option. Otherwise, you can configure access on a per organization basis.

5. Save the credentials. You're now ready to set up and activate the connection to bring in data from CrowdStrike.

Step 3: Choose how to configure the CrowdStrike integration

The CrowdStrike integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the CrowdStrike integration to sync data

After you add your CrowdStrike credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the CrowdStrike integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new CrowdStrike-only assets are created.

1. Activate a [connection to CrowdStrike](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure that it has access to the organization you are currently in.
3. Set the severity and risk levels you want to import (optional).
4. Set the *Fingerprint only* toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
5. Add an [filter](#) for imported assets (optional).



If the CrowdStrike API key is configured with access to Falcon Discover or Falcon Spotlight, software and vulnerability data will only be imported for the assets included in the filtered results.

6. Enter a name for the task, like `CrowdStrike sync` (optional). 7. Choose the Explorer to perform this connector task from (optional). 8. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site. 9. Enter a description for the task (optional). 10. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the *_Exclude unknown assets_* toggle to *_Yes_*. By default, the integration will include assets that have not been scanned by runZero. 11. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set. 12. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks] (<https://console.runzero.com/tasks>) to see when the next sync will occur.

Step 4b: Configure the CrowdStrike integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the CrowdStrike toggle to **Yes**, and change any of the default options if needed.
4. On the Credentials tab, set the CrowdStrike toggle for the credential you wish to use to **Yes**.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 5: View CrowdStrike assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your CrowdStrike assets and vulnerabilities. These will have a CrowdStrike icon listed in the **Source** column.

To filter by CrowdStrike attributes, consider running the following queries:

- [View all CrowdStrike assets](#)

```
source:crowdstrike
```

- [Find assets that have a CrowdStrike EDR agent installed](#)

```
edr.name:crowdstrike
```

- [Find Windows assets, excluding servers, that are missing a CrowdStrike EDR agent](#)

```
os:windows and not type:server and not edr.name:CrowdStrike
```

- [View all CrowdStrike vulnerabilities](#)

```
source:crowdstrike
```

- [View all CrowdStrike software results](#)

```
source:crowdstrike
```

Click into each asset or vulnerability to see its individual attributes. runZero will show you the attributes returned by the CrowdStrike API, with the exception of policies.

Filtering Crowdstrike assets

An optional filter can be applied to Crowdstrike integration tasks. runZero uses Crowdstrike's Falcon Query Language (FQL) for filtering. FQL follows the syntax `<property>:[operator]<value>`. Multiple expressions can be combined for more complex filtering by adding a + between expressions. An OR expression can also be leveraged with comma separated expressions.

Properties

The following are some useful CrowdStrike properties that can be used in an FQL expression to filter assets. Details on additional attributes that are available for filtering can be found by reviewing [CrowdStrike's API documentation](#).

CrowdStrike Property	runZero Attribute	Description	Example
external_ip	externalIP	The external IP address of the device	18.191.169.203
first_seen	firstSeen	A timestamp of when the device was first seen by CrowdStrike	2022-01-08T19:42:34Z
hostname	hostname	The hostname of the device	EXPLORER-01
last_seen	lastSeen	The timestamp of when the device was last seen by CrowdStrike	2022-09-13T19:14:30Z
local_ip	localIP	The local IP address of the device	192.168.1.100
mac_address	macAddress	The mac address of the interface communication with CrowdStrike	0a-6e-20-4a-e6-56
os_version	osVersion	The operation system version of the device	Ubuntu 20.04
platform_name	platformName	The platform running on the device	Linux
product_type_desc	productTypeDesc	The type of device	Server

Operators

The following operators can be used in an FQL expression to filter assets.

Operator	Description
!	Not equal to
>	Greater than
>=	Greater than or equal to
<	Less than
<=	Less than or equal to
~	Text match. Tokenizes the string, ignoring spaces, case and punctuation
!~	Does not text match. Tokenized the string, ignoring spaces, cases and punctuation

*	Wildcard matching. Matches one or more characters

Example Filters

The following are examples of filters that can be applied to the CrowdStrike sync.

Search Filter	Description
hostname:'WIN10*'	Import all devices where the hostname starts with WIN10
platform_name:'Linux'	Import all Linux devices
product_type_desc:'Server'	Import all devices that CrowdStrike identifies as a Server
hostname:'PROD*'+platform_name:'Linux'	Import all Linux devices with a hostname that starts with PROD
local_ip:'192.168.1.100'	Only import the device with a local IP address of 192.168.1.100
local_ip:!'192.168.1.100'	Import all devices, excluding 192.168.1.100
local_ip.raw:*'192.168.1.*'	Import all devices with a local IP address in the 192.168.1.0/24 range
(local_ip.raw:*'192.168.1.*'), (local_ip.raw:*'192.168.2.*')	Import all devices with a local IP address in the 192.168.1.0/24 or 192.168.2.0/24 range
local_ip.raw:!'192.168.1.*'	Import all devices, excluding devices with a local IP address in the 192.168.1.0/24 range
local_ip.raw:!'192.168.1.*'+local_ip.raw:!'192.168.2.*'	Import all devices, excluding devices with a local IP in the 192.168.1.0/24 and 192.168.2.0/24 ranges

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the CrowdStrike integration unable to connect?

1. Are you getting any data from the Crowdstrike integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.

- In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
- 2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
- 3. If the CrowdStrike integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
- 4. If the integration endpoint is on-premises, verify they are running the integration task from an Explorer with access to the CrowdStrike host.

How can I solve the following CrowdStrike error?

Unable to collect software data for CrowdStrike devices: invalid response 403 Forbidden

This error occurs if your API client is missing the **Assets** API scope. The integration requires read-only permissions for **Assets** in order to collect software information. Host and vulnerability data should be collected just fine, though. This can be remedied by returning to [step 1](#) of the CrowdStrike documentation above and enabling read permissions for **Assets**.

Custom Integration Scripts

Custom integration scripts let you import asset data from any source that runZero does not already integrate with, or export runZero data to another system. Scripts are written in [Starlark](#), a Python-like language, and run in a sandbox on one of your [Explorers](#) or on the runZero console itself.

runZero 5.1 significantly expanded what custom integration scripts can do:

- An embedded [CONFIG block](#) describes the integration and its parameters. The console uses it to generate a typed credential form, validate input, apply defaults, and route secrets through encrypted storage.
- Shared [TLS and HTTP option suites](#) (`OPTIONS_TLS` and `OPTIONS_HTTP`) add standard connection controls to any script without copying boilerplate parameters.
- [report_assets](#) streams assets to runZero page-by-page, so large imports no longer need to be held in memory.
- A CONFIG-level [matchBehavior policy and per-asset trust_* flags](#) give scripts control over how imported assets merge with existing inventory.
- A much larger [library of built-in modules](#), including typed `kwargs` accessors, HTTP helpers with retry support, parsers for CSV/XML/streaming JSON, JWT and AWS SigV4 signing, and direct-protocol modules for SSH, SMB, WinRM, WMI, and SQL.
- The `runzero` script CLI command gained a [--validate mode](#) that smoke-tests a script's CONFIG and HTTP/TLS wiring against a local dummy server, and an [--output mode](#) that writes real scan output for inspection.
- The custom integration page in the console became a [script workbench](#) with an editor, [test runs against real endpoints](#), a shipped example library, version history, and — where AI is enabled — a [generative AI assistant](#) that drafts and revises scripts.

A library of ready-to-use integrations built on these capabilities is available in the [runzero-custom-integrations](#) GitHub repository, including a [boilerplate template](#) to start from.

Overview

To set up a custom integration script:

1. Write the script, starting with a `CONFIG` block and a `main` function.
2. Test it locally with the [runZero CLI](#) (`runzero script`).
3. Create the custom integration in the console and paste in the script.
4. Create a credential; the form is generated from the script's `CONFIG` parameters.
5. Create an integration task to run the script on a schedule.

Writing a script

The Starlark language

Starlark is a dialect of Python with some notable differences:

1. There is no exception handling (`try/except`). Use return values to signal errors.
2. There is no f-string `f'{var}'` formatting. Use `"{}".format(var)` for string interpolation.
3. The standard library is limited to the [built-in modules](#) provided by runZero.

The custom integration sandbox also disallows recursion, top-level `if/for` statements, and reassigning global variables. `while` loops and the `set()` type are supported.

The entrypoint

Every script must define a `main` function. runZero calls it with the task's arguments and keyword arguments:

```
load('runzero.types', 'ImportAsset')

def main(*args, **kwargs):
    asset = ImportAsset(
        id='device-1',
        hostnames=['web1.example.com'],
        os='Linux',
        osVersion='6.1',
        manufacturer='Example Corp',
        model='VM',
    )
    return [asset]
```

- `args` receives any positional arguments configured on the task, as strings.
- `kwargs` receives the credential fields and task keyword arguments, as strings. Declare every key your script reads in `CONFIG["params"]`, and read them with the typed [kwargs accessors](#).

- For inbound integrations, `main` returns a list of `ImportAsset` objects (or a single `ImportAsset`). Returning `None` is valid when assets are streamed with `report_assets` instead.
- For outbound integrations, `main` performs its export work and returns `None`.

Streaming large imports with `report_assets`

Returning one large list from `main` keeps every asset, plus the raw API responses used to build them, in memory at once. For sources with large inventories, stream each page to runZero as it is processed instead. `report_assets` is a predeclared builtin, so no `load()` is needed:

```
load('runzero.types', 'ImportAsset')

def fetch_page(cursor):
    # Replace with a real paginated API call.
    if cursor == None:
        return [{"id": "a-1"}, {"id": "a-2"}], "page-2"
    return [{"id": "b-1"}], None

def main(*args, **kwargs):
    total = 0
    cursor = None
    while True:
        page, cursor = fetch_page(cursor)
        if not page:
            break
        assets = [ImportAsset(id=item["id"]) for item in page]
        total += report_assets(assets) # stream this page to runZero
        if not cursor:
            break
    print("reported {} assets".format(total))
    return None # nothing buffered in main
```

`report_assets` accepts a single asset, several positional assets, or a list/tuple of assets, and returns the number reported. Assets streamed with `report_assets` and assets returned from `main` are both imported, so partial adoption is safe.

The singular `report_asset` (also predeclared) takes exactly one `ImportAsset` (or `None`, a no-op) and returns 1 or 0. Streaming already batches internally, so reporting each asset as it is built is the simplest correct pattern — there is nothing to gain by accumulating a page into a list first:

```
for device in devices:
    report_asset(build_asset(device))
```

For the pagination loop itself, the predeclared `pager` builtin guards against a cursor that never advances. `p = pager(label="devices")` returns an object whose `p.next()` is used as the `while` condition; if the loop reaches the page ceiling (`maxPages`, default 1,000,000), the script stops with an error naming the label instead of silently truncating or spinning forever. `max_pages()` returns the effective ceiling.

The CONFIG block

The `CONFIG` block is an embedded, declarative description of the integration. When you save a script in the console, runZero extracts `CONFIG` (without executing the script) and uses it to:

- Render a typed credential form with labels, groups, and conditional fields.
- Validate submitted values (required, min, max, pattern, options) and apply default values before main runs.
- Route `secret` parameters through encrypted storage and redact their values from script logs and error messages.
- Refuse to run the script on runZero versions older than `minVersion`.

```
CONFIG = {
  "id": "runzero-example",
  "name": "Example integration",
  "type": "inbound",
  "description": "Imports devices from the Example API.",
  "version": "1",
  "minVersion": "5.1.0",
  "params": [
    {"key": "url", "label": "Example API URL", "type": "url", "required": True},
    {"key": "api_token", "label": "API token", "type": "secret", "required": True},
    {"key": "page_size", "label": "Page size", "type": "int", "required": False,
     "default": 100, "min": 1, "max": 1000},
  ],
  "includes": {
    "tls_": OPTIONS_TLS,
    "http_": OPTIONS_HTTP,
  },
}
```

```
load("runzero.types", "ImportAsset")
# ...rest of script
```

Rules for the block itself:

- `CONFIG` must be the **first top-level statement** in the script. Comments and blank lines may appear before it; `load(...)` calls, constants, and other statements may not.
- Every value must be a literal: strings, numbers, `True/False/None`, lists, tuples, and dicts with string keys. Function calls, variable references, and arithmetic are rejected. The only exception is `CONFIG["includes"]`, which may reference the predeclared option-set identifiers `OPTIONS_TLS` and `OPTIONS_HTTP`.
- Scripts without a `CONFIG` block continue to run with the legacy `access_key` and `access_secret` credential fields, so existing integrations keep working unchanged.
- `sourceId` and `sourceName` are reserved for runZero's own shipped integrations. A script that declares either is refused when you save it.

Top-level fields

Field	Description
id	Stable lower-case identifier, for example runzero-tailscale.
name	Display name for the integration.
type	inbound, outbound, or internal (descriptive).
description	Short summary of what the integration does.
version	The script's own revision, bumped when the script changes, for example "1".
minVersion	Minimum runZero version required to run the script, for example 5.1.0. Older Explorers refuse to run the script and report a clear upgrade message.
params	List of parameter definitions .
includes	Shared option suites , keyed by prefix.
atLeastOneOf	List of parameter-key groups; at least one key in each group must be set.
exactlyOneOf	List of parameter-key groups; exactly one key in each group must be set.
rejectUnknown	Reject keys that are not declared in <code>params</code> . Unknown keyword arguments are always rejected at run time for CONFIG-based scripts.
validationMode	How <code>runzero script --validate</code> exercises the script: the default expects an HTTP request; "compile" only checks that CONFIG parses and <code>main</code> exists. Use "compile" for direct-protocol (SSH/SMB/WMI/WinRM/SQL) integrations.
maxPages	Page ceiling enforced by the predeclared pager builtin ; defaults to 1,000,000. A script may lower the effective ceiling by passing <code>limit=</code> to <code>pager</code> , but not raise it.
matchBehavior	Integration-wide merge policy , a space-separated string of flags.
assetType	Asset type applied to every asset the integration reports, overridable per asset with the <code>assetType</code> field on <code>ImportAsset</code> .
assetTypeBehavior	Map of asset type to a <code>matchBehavior</code> -style flag string, layered over the integration-wide <code>matchBehavior</code> for assets of that type. Also accepted as <code>sourceTypeBehavior</code> ; declaring both with different values is an error.
ownershipAttributes	Attribute keys the integration supplies that should be treated as ownership data.

trustOS, trustOSVersion, trustType	Trust this integration's OS, OS version, or device type over other sources, for every asset it reports. The per-asset <code>trust_*</code> fields on ImportAsset do the same thing for one asset.

Parameter definitions

Each entry in `params` describes one credential form field and one keyword argument delivered to `main`.

Supported type values: `string`, `secret`, `int`, `float`, `bool`, `enum` (requires `options`), `url`, `textarea`, and `json`.

Field	Description
<code>key</code>	Keyword argument name. Must match <code>^[a-zA-Z_][a-zA-Z0-9_]*\$</code> and match the name the script reads.
<code>label</code>	Form label.
<code>description</code>	Help text shown under the field.
<code>type</code>	One of the types above. <code>secret</code> values get masked input, encrypted storage, and log redaction.
<code>secret</code>	Treat the value as a secret regardless of its <code>type</code> . Equivalent to <code>type: "secret"</code> for masking, storage and redaction.
<code>required</code>	Whether the field must be set.
<code>default</code>	Default value applied when the field is left blank. Not allowed on <code>secret</code> parameters.
<code>placeholder</code>	Placeholder text for the form field.
<code>options</code>	Allowed values for <code>enum</code> parameters.
<code>multi</code>	For <code>enum</code> : allow multiple comma-separated selections.
<code>min / max</code>	Numeric bounds for <code>int/float</code> ; length bounds for <code>string/secret/textarea</code> .
<code>pattern</code>	Regular expression the value must fully match (<code>string</code> , <code>secret</code> , <code>textarea</code> , <code>url</code>).
<code>aliases / caseInsensitive</code>	For <code>enum</code> : alternate spellings that normalize to a canonical option before <code>main</code> runs.
<code>dependsOn, visibleIf, visibleIfValue</code>	Show this field only when another declared field is set (optionally to a specific value).
<code>requiredIf, requiredIfValue</code>	Make this field required when another declared field is set (optionally to a specific value).
<code>group</code>	Section heading used to group fields in the form.

How values reach your script

- Each declared parameter arrives in `kwargs` under its key. Values are delivered as strings; use the [kwargs module](#) (`get_string`, `get_int`, `get_bool`, `get_list`, ...) to read them with type coercion and defaults.
- Declared default values are applied, enum aliases are normalized, and validation runs before `main` is called.
- CONFIG-based scripts reject unknown keyword arguments.
- Credential keys starting with `_` (for example `_integration_id`) are stored on the credential but never forwarded to the script.
- For backward compatibility, the credential's `access_key` and `access_secret` fields are still injected when the task does not provide them.

Shared TLS and HTTP option suites

Most integrations talk to an HTTPS API, and most of them need the same connection controls: trusting a private CA, pinning a certificate, presenting a client certificate, disabling validation in a lab, or sending a specific `User-Agent`. Rather than declaring these parameters in every script, add the predeclared option suites to `CONFIG["includes"]`:

```
"includes": {
  "tls_": OPTIONS_TLS,
  "http_": OPTIONS_HTTP,
},
```

Each include expands into a set of parameters. The dict key (`"tls_"`, `"http_"`) is a prefix prepended to every generated parameter key. In the credential form, suite parameters render after the script's own fields as collapsed sections named for the suite (**TLS**, **HTTP**); a section holding an empty required field is forced open with a **Needs attention** badge. When the same suite is included under two prefixes, each section is named for its prefix, for example **TLS (src_tls)**.

OPTIONS_TLS

With the conventional `tls_` prefix, `OPTIONS_TLS` generates these keyword arguments:

Generated kwarg	Type	Default	Description
<code>tls_disable_validation</code>	<code>bool</code>	<code>False</code>	Allow connections to endpoints with untrusted TLS certificates.
<code>tls_ca_cert</code>	<code>textarea</code>	—	PEM-encoded certificate authorities to trust when validating the endpoint certificate.
<code>tls_peer_hash</code>	<code>string</code>	—	SHA-256 fingerprint of the endpoint certificate to trust (certificate pinning). Multiple pins can be separated by commas, semicolons, spaces, or newlines.

tls_client_cert	textarea	—	PEM-encoded client certificate to present for mutual TLS.
tls_client_key	secret	—	PEM-encoded private key for the client certificate. Required when tls_client_cert is set.

OPTIONS_HTTP

With the conventional http_ prefix, OPTIONS_HTTP generates:

Generated kwarg	Type	Default	Description
http_user_agent	string	""	Optional User-Agent header sent with HTTP requests.

Using the option suites

Declaring the includes only creates the form fields; the script must pass the collected values to the HTTP client. The `kwargs` module does this in one call:

```
load("http", "get_json", "bearer")
load("kwargs", "get_string", "get_url_base", "get_http_options")

def main(*args, **kwargs):
    http_options = get_http_options(kwargs, headers={
        "Authorization": bearer(get_string(kwargs, "api_token")),
        "Accept": "application/json",
    })
    data, err = get_json("{}v1/devices".format(get_url_base(kwargs)), **http_options)
```

`get_http_options(kwargs, prefix="http_", tls_prefix="tls_", headers=None)` gathers the suite values into a dict of keyword arguments (`headers=` and `tls=`) that can be splatted into any `http` module function (`get`, `post`, `get_json`, `post_json`, `oauth2_token`, and so on). The mapping is:

Suite kwarg	Effect
tls_disable_validation	tls={"insecure": True}
tls_ca_cert	tls={"ca_pem": ...}
tls_client_cert / tls_client_key	tls={"client_cert_pem": ..., "client_key_pem": ...}
tls_peer_hash	tls={"thumbprints": [...]}
http_user_agent	headers={"User-Agent": ...} (only when the header is not already set)

Use `get_http_tls(kwargs, "tls_")` when a script only needs the `tls=` dict and manages headers itself. The `tls=` dict can also be built by hand; it accepts the keys `insecure`, `server_name`, `ca_pem`, `client_cert_pem`, `client_key_pem`, and `thumbprints`, and rejects anything else.

Two things to be aware of:

- `runzero script --validate` verifies the wiring: a script that declares `OPTIONS_HTTP` or `OPTIONS_TLS` but never passes the collected options to an HTTP call fails validation.
- The stateful `requests.Session` object does not accept the `tls=` dict; only its `insecure_skip_verify` constructor flag is available. Prefer the `http` module functions when the TLS suite matters.

Multiple endpoints

Scripts that talk to more than one endpoint can include a suite more than once under different prefixes, and collect each set separately:

```
"includes": {
    "src_tls_": OPTIONS_TLS,
    "src_http_": OPTIONS_HTTP,
    "dst_tls_": OPTIONS_TLS,
    "dst_http_": OPTIONS_HTTP,
},
```

```
src_options = get_http_options(kwargs, "src_http_", "src_tls_", src_headers)
dst_options = get_http_options(kwargs, "dst_http_", "dst_tls_", dst_headers)
```

A complete example

The script below puts the pieces together: a CONFIG block with typed parameters and both option suites, option-suite plumbing via `get_http_options`, paginated fetching with `get_json`, and streaming import via `report_assets`. It passes `runzero script --validate as-is`, and can be used as a starting point for a real integration (see also the [boilerplate template](#) on GitHub).

```
CONFIG = {
    "id": "runzero-example",
    "name": "Example integration",
    "type": "inbound",
    "description": "Imports devices from the Example API.",
    "version": "1",
    "minVersion": "5.1.0",
    "params": [
        {"key": "url", "label": "Example API URL", "type": "url", "required": True},
        {"key": "api_token", "label": "API token", "type": "secret", "required": True},
    ],
    "includes": {
        "tls_": OPTIONS_TLS,
        "http_": OPTIONS_HTTP,
    },
},

load("runzero.types", "ImportAsset", "to_custom_attributes")
load("net", "network_interface")
load("http", "get_json", "bearer")
```



```

def build_asset(device):
    return ImportAsset(
        id=str(device["id"]),
        hostnames=[device.get("hostname", "")],
        os=device.get("os", ""),
        networkInterfaces=[network_interface(mac=device.get("mac"),
                                             ips=device.get("ips", []))],
        customAttributes=to_custom_attributes(device),
    )

def main(*args, **kwargs):
    require(kwargs, "url", "api_token")
    base_url = get_url_base(kwargs)
    http_options = get_http_options(kwargs, headers={
        "Authorization": bearer(get_string(kwargs, "api_token")),
        "Accept": "application/json",
    })

    total = 0
    page = 1
    while True:
        data, err = get_json("{} /v1/devices".format(base_url),
                        params={"page": str(page)}, retries=2, **http_options)
        if err:
            print("request failed: {}".format(err))
            break
        devices = data or []
        if not devices:
            break
        total += report_assets([build_asset(d) for d in devices])
        page += 1

    print("imported {} devices".format(total))
    return None

```

```
runzero script --filename example-integration.star --validate
```

```
INFO example-integration.star validated with 1 HTTP request(s)
```

```
INFO validated 1 script(s) with dummy HTTP/TLS server https://127.0.0.1:52084
```

Controlling how assets merge

By default, imported assets merge with existing inventory using the asset `id`, MAC address, IP address, and hostname. Merge policy is declared once for the whole integration, as the `matchBehavior` key of the [CONFIG block](#):

```

CONFIG = {
    # ...
    "matchBehavior": "no-mac-break no-ip-break no-name-break",
}

```

`matchBehavior` accepts a space-separated string of flags built from `no- + (id|mac|ip|name) + (-match|-break)`, plus the break-only `no-type-break`, which allows merges across an integration's

own asset types. Two presets cover most cases:

- "no-mac-break no-ip-break no-name-break" — use when your source supplies a **stable, unique id** (vendor UUID, serial number). The id still drives merges, but differing MACs, IPs, or names will not disqualify a merge with an existing asset.
- "no-id-match no-id-break" — use when your source only emits **ephemeral or per-run ids**. The id is ignored and merging falls back to MAC, IP, and hostname.

`matchBehavior` was formerly a per-asset field on `ImportAsset`. It no longer is: a script that reads or sets `ImportAsset.matchBehavior` fails with an error pointing at the CONFIG key.

Two related controls remain per-asset:

- `assetType` on `ImportAsset` overrides the CONFIG-level `assetType` for one asset. When the CONFIG also declares `assetTypeBehavior`, the asset's type selects which merge policy applies to it.
- `trust_device_type`, `trust_os`, and `trust_os_version` (booleans) apply the script's `deviceType`, `os`, and `osVersion` values to the asset fingerprint even when runZero cannot normalize them through its fingerprint engine. The CONFIG-level `trustType`, `trustOS`, and `trustOSVersion` do the same for every asset the integration reports.

Leave `matchBehavior` unset to keep the default matcher behavior, which is correct for most integrations. See the [asset identity guidance](#) in the custom integrations repository for a deeper treatment.

Testing scripts with the runZero CLI

The [runZero CLI](#) includes a `script` sub-command for developing and debugging integration scripts locally, using the same Starlark engine and modules as the Explorer.

```
runzero script [--filename file] [--args a] [--kwargs key=value]
runzero script repl [--filename file]
```

Flag	Description
<code>-f, --filename</code>	Script file to load and run. <code>--validate</code> also accepts a directory.
<code>--args</code>	Positional argument passed to <code>main</code> (repeatable).
<code>--kwargs key=value</code>	Keyword argument passed to <code>main</code> (repeatable).
<code>--validate</code>	Validate the script CONFIG and smoke-test HTTP/TLS wiring against a local dummy server.
<code>-o, --output</code>	Directory to write scan output to (<code>scan.runzero.gz</code>).
<code>--overwrite</code>	Replace the output directory if it already exists.
<code>--custom-integration-id</code>	Integration UUID stamped on exported records; required with <code>--output</code> .

<code>--starlark-allow-cidrs</code>	Restrict every network connection the script makes to the listed CIDRs or IPs (default deny). Enforced after DNS resolution, across the <code>http</code> , <code>requests</code> , <code>socket</code> , <code>direct-protocol</code> , and <code>SQL</code> modules.
<code>--starlark-block-cidrs</code>	Block script connections to the listed CIDRs or IPs (default allow). Same coverage as <code>--starlark-allow-cidrs</code> .

Running scripts

Save a minimal script as `hello.star`:

```
def main(*args, **kwargs):
    print("Hello {}".format(kwargs.get("name", "world")))
```

```
runzero script --filename hello.star --kwargs name=Dave
```

```
INFO script: Hello Dave!
INFO script completed successfully with 0 returned asset(s)
```

`--args` and `--kwargs` can be repeated to pass as many values as needed:

```
runzero script --filename hello.star --args one --args two --kwargs api_token=foo --kwargs url=https://api.example.com
```

Note that a local run calls `main` with exactly the arguments you provide; `CONFIG` defaults and validation are applied by the console and Explorer at task time, and by `--validate` locally.

Validating scripts

`--validate` checks a script end-to-end without touching the real vendor API:

```
runzero script --filename example.star --validate
```

```
INFO example.star validated with 1 HTTP request(s)
INFO validated 1 script(s) with dummy HTTP/TLS server https://127.0.0.1:52084
```

For HTTP integrations, validation parses the `CONFIG` block, generates type-appropriate placeholder values for every parameter, initializes the script, calls `main`, and transparently routes all HTTP requests to a local TLS server that returns canned responses. It fails if the script never makes an HTTP request, and if declared `OPTIONS_HTTP/OPTIONS_TLS` options never reach the HTTP client.

Scripts with `"validationMode": "compile"` (templates and direct-protocol integrations) stop after verifying that `CONFIG` parses and `main` exists.

Passing a directory validates every `.star` file under it:

```
runzero script --filename ./my-integrations/ --validate
```

A successful validation is a `CONFIG` and wiring check, not proof that the vendor API accepts your credentials or returns the expected payload — test against the real API with `--kwargs` for that.

Exporting scan output

To inspect exactly what a script would import, write real scan output to a directory:

```
runzero script --filename example.star --kwargs api_token=MY_TOKEN -o ./out --overwrite --custom-integration-id 111111
```

This writes `./out/scan.runzero.gz`, a gzip-compressed stream of JSON records — one per asset — in the same format an integration task produces. Without `--output`, assets streamed via `report_assets` are discarded with a warning and only counted.

REPL

`runzero script repl` starts an interactive session with all modules preloaded. With `--filename`, the script's functions and globals are available to call directly:

```
$ runzero script repl --filename hello.star
>>> main(**{"name": "Dave"})
INFO script: Hello Dave!
>>> load('json', json_decode='decode')
>>> data = json_decode('{"greeting": "hello"}')
>>> print(data["greeting"])
INFO script: hello
```

Exit the REPL with `^D`.

Building and testing a script in the console

You do not have to write a script in a local editor and paste it in. The custom integration page in the console is a **script workbench**: an editor with a test-credentials panel, a live output console, and — where it is enabled — an AI assistant that can draft or revise the script for you. Everything is on the page itself; there is no separate workspace to open or close.

The workbench is laid out as:

- a **pinned action bar** that stays on screen while the page scrolls. Its left half holds the **verdict row** — one line stating the result of the last thing you did (validated, ran, saved, loaded, or generated; empty until there is something to report) — and the example picker. Its right half holds **Revert**, **Validate**, **Run test** (which becomes **Stop** while a run is going), and **Full screen**.
- the **editor**, with a rail beside it carrying two tabs: **Test credentials** (the default) and **Generative AI**. Switching tabs never interrupts a test run or a generation. The editor has its own resize grabber for height; there are no drag dividers.
- the **output console** directly under the editor, a capped scroller carrying the detail: test-run logs and reported assets, per-line compile errors, and the progress and transcript of an AI generation.

Saving is the page's **Save** (or **Update**) button, or **Cmd/Ctrl+S** from anywhere in the workbench. A workbench save reports **Script changes saved** in the verdict row rather than a toast. On a brand-new integration, **Cmd/Ctrl+S** creates the integration in place: the URL moves to the edit page without disturbing the editor, a generation, or a running test.

Full screen fills the window with the editor alone. **Esc** (or the **Exit full screen** button) leaves it, Cmd/Ctrl+S still saves, and the label notes **unsaved changes** whenever the editor differs from the last saved script. Full screen is unavailable while a generation is writing into the editor.

While a test run or a generation is in flight, the **Enable custom integration script** toggle refuses to turn off and says why. Turning the toggle off otherwise stashes the editor content locally; turning it back on restores it.

An empty editor shows a starter panel with three ways in: **Generate with AI**, **Load a blank template**, and **Browse examples**.

Starting from an example

The example picker in the action bar (**Select an example...**) loads a shipped integration into the editor as a starting point. These are the same scripts published in the [runzero-custom-integrations repository](#), grouped as **Start from scratch** (a blank template with commented examples), **Import assets from a vendor**, and **Other automations**. The library ships through content updates, so it grows and refreshes between releases.

Loading an example replaces what is in the editor, so it asks first when you have something there. On an integration that has already been saved, your previous script is written to [version history](#) before it is replaced, so you can always load it back. Loading also fills in an empty integration name and description from the example, and picking the example already in the editor reports **No change**.

The integration remembers which example a script came from and at which version. When the shipped copy no longer matches yours, a banner says **A newer version of the *name* example is available**, notes when you have local modifications, and offers **Update to v_N_**. Updating also asks first, and also snapshots what you have.

Generating a script with AI

Where generative AI is enabled for your account, the workbench rail has a **Generative AI** tab (an empty editor also offers **Generate with AI** directly). The tab holds a short form:

- **What should this integration do?** — describe the vendor, the API, and the data you want imported, in your own words. (When revising an existing script, the field asks **What should change?**) The **Authentication**, **Asset fields** and **Scope** buttons under the box insert the three questions the assistant most needs answered and cannot guess: how the API authenticates, which response fields are the address, MAC and hostname, and what to leave out.
- **Documentation links (optional)** — one vendor API documentation URL per line, up to 10. The field says how many of what you entered will actually be read.

Press **Generate** to write a new script, or **Update script** to revise the one already in the editor. The assistant works for several minutes: it consults the shipped integration library for the same vendor, reads the documentation you linked, and writes the script. Before it answers, the finished script must pass runZero's real validators — a script that fails is sent back to the assistant to fix, shown as **Fixing validation errors** — and any `load()` lines the script forgot are added mechanically afterward.

Progress appears in the output console below the editor — the pages it reads, the tools it calls, and its own commentary — and the script itself streams into the editor as it is written. You can switch back to the **Test credentials** tab while it works; the run keeps going and keeps reporting in the verdict row.

Stop — in the AI form, or beside the transcript in the output console — cancels the run. Nothing else does: an AI run is deliberately independent of the browser, so leaving the page or closing the tab leaves it running. Returning to the integration page picks the run back up, including anything that streamed while you were away. Every attempt also runs as a live turn on its own [AI thread](#), with the running indicator, the work log, and its own Stop, so a run that failed or that you left behind is still there to look at — the output console links it as **View this request in AI Threads**.

If a run fails or is stopped before it finishes, whatever script was in the editor beforehand is put back. A run can also finish without a script — the assistant declining, asking a question, or answering in prose — and that answer is shown under **What the model said**.

When it succeeds, the assistant's own account of what it built stays in the output console under **What the model says it built**, beside the code. **Clear** removes the transcript when you are done with it.

What the assistant is given, and what it is not

Only three things are sent to the model: your description, the documentation URLs you supplied, and — when you are revising — the script currently in the editor. No inventory, asset, organization or credential data is sent, ever.

The assistant has four tools: it can fetch a documentation page, list and read runZero's shipped example integrations, and validate a candidate script. Validation parses and compiles the script only — nothing is executed, no network call is made, and nothing is saved.

Documentation pages are fetched under the same protections as the rest of the platform: private and internal addresses are blocked, and the content is treated as untrusted throughout, because a page the assistant reads is written by someone else.

Read the script before you save or run it. It is drafted from third-party documentation and it will run with your integration credentials.

Limits

- One authoring run is bounded at 10 minutes.
- Documentation fetching within a run is bounded at 10 pages and 90 seconds in total.
- A script sent for revision must be under 48 KiB. Above that the run still works, but it is generated from your description alone, and the form says so.
- Generative AI requires the AI feature to be enabled for the account and the organization, in addition to the custom integration script entitlement, and administrator access — the same requirement as authoring a script by hand.
- Generations count against the same [AI run limits](#) as threads: 4 running AI jobs per user and 12 per account.

Where the generated script goes

When the run finishes, the script is loaded into the editor and, if the integration has already been saved, committed straight away as a new version — the output console notes **Saved as version N** — so a generation is never one navigation away from being lost. A script that does not compile is not saved as a version; the output console says so and leaves it in the editor for you to fix. On an integration that has not been saved yet, it reminds you to save the integration to keep the script.

Each run also produces an [AI artifact](#) holding the script. Artifacts start private to you and can be shared; an unshared artifact is [retained](#) as long as the thread it came from. Artifacts are listed under **Artifacts** in the AI section of the console, where **Use in custom integration** loads one back into the integration it was written for (or into a new one). The handoff lands directly on the workbench: the editor flashes, and the verdict row confirms what was loaded — for a new integration, with a suggested name derived from the artifact's title that you can change before saving.

Validating and previewing

Validate runs the same checks the save path runs: the size cap, the Starlark parse and compile, the [CONFIG block](#) extraction and schema, and the reserved-name rules. Nothing is executed. Failures are reported per line in the output console, and the editor marks the line.

The **Credential fields** disclosure below the workbench lists every parameter the CONFIG block declares — label, key, type, whether it is required and secret, and its default — refreshed automatically when the integration loads and after each save. It is the quickest way to check that a CONFIG block says what you meant. To see the rendered form itself, switch the test rail to **Enter values for this run only**, which shows the same fields the person creating the credential will see.

Running a test

Run test executes the script in the editor immediately — including changes you have not saved — and streams back its output, a preview of the assets it reported, and a plain-language verdict.

A test run is not a task, and differs from one in ways that matter:

- **It runs on the console, not on an Explorer.** Internal and link-local addresses are blocked — on self-hosted consoles too — so an integration that targets an endpoint inside your network cannot be tested this way. Test that one from an Explorer with the [runZero CLI](#) instead.
- **Nothing is imported.** No task is created, no assets reach your inventory, and nothing is written to your organization. The assets shown are a preview and are discarded when the run ends.
- **Credentials are not stored.** You can run against a saved credential, or switch the rail to **Enter values for this run only**. Values typed in for a run are held for the length of that run and never saved; to schedule an integration whose CONFIG requires values, you still need a saved credential.

Picking a saved credential shows its stored values in the rail, so the choice is not blind: non-secret values appear verbatim, and secret values show only that a value is stored — they are never sent to the browser. Your non-secret selections (credential mode, credential choice, non-secret parameter values) are remembered per integration in the browser between visits; secret values start empty every time.

Before anything executes, the run checks the credential values against the script's CONFIG schema; values that can never satisfy it are refused up front, naming the parameter keys (never the values). The rail also warns beside **Run test** when no credential is selected and the CONFIG declares required fields.

Every credential value supplied to a run — saved or typed in — plus every CONFIG `secret` value is redacted from everything the run prints, including error text, crash output, and the asset preview.

Test run limits

Limit	Value
Wall clock	120 seconds
Memory	256 MiB
Computation	2,000,000,000 interpreter steps; time spent waiting on an API does not count
Assets	100, after which the run stops and reports success
Assets shown in full	the first 10, with up to 24 attributes each, values truncated at 256 bytes
Log output	500 lines, 4 KiB per line

You can run one test at a time, up to six runs a minute; the account as a whole is capped at twenty starts a minute.

Reaching the 100-asset ceiling is reported as a **success**: it means the script worked and was stopped early because it is a test. A real task is not capped this way.

Every run ends with a named outcome rather than a bare failure — it completed, it stopped at the asset limit, you stopped it, the script raised an error, it ran out of time, memory, computation or stack, the sandbox stopped it, it stopped unexpectedly, or it never started — each with a short explanation of what to look at next. A run that completes without reporting any assets says so: it usually means the response was parsed differently than expected, or a filter in the script excluded everything.

A run that could not be started at all is explained the same way, including the compiler's own line and column when the script did not compile, so you do not have to press **Validate** to find out what was wrong.

Version history

Every change to a script is snapshotted, including versions committed by an [AI generation](#). The **Script version history** disclosure on the custom integration page — its title carries the count and the current version — opens a table listing each version with its author, timestamp, size, and the example it came from, when it came from one. The most recent 25 versions are kept.

Load, offered on every version other than the current one, puts a stored version into the editor so you can look at it or work from it. It asks first when the editor holds something else. Nothing is written until you save, and saving it creates a new version rather than rewriting history.

Revert appears in the action bar whenever the editor differs from the last saved script, and restores that saved script locally. Like Load, it only changes the editor; it asks first, because the editor may be the only copy of what you have typed.

Adding the integration to your console

Step 1: Create the custom integration

1. Go to the [Custom sources page](#) and click **Add custom integration**.
2. Provide a name (no spaces) and optionally an icon (PNG, up to 256×256).
3. Choose whether the integration is available to every organization in the account or only to specific organizations. See [organization access](#).
4. Toggle **Enable custom integration script**, then paste in your script, [start from an example, or generate one with AI](#).
5. Click **Validate** to check the script syntax and CONFIG block, and optionally [run the script against a real endpoint](#) before you schedule it.
6. Click **Save** — or **Save & create task**, which saves and lands on the task form with the integration pre-selected.

Step 2: Create the credential

1. Go to the [Credentials page](#) and click **Add Credential**.
2. Choose **Custom Integration Script Secrets** as the credential type.
3. Select the custom integration. The form shows the fields declared in the script's CONFIG block, including any option-suite fields such as `tls_disable_validation` and `http_user_agent`.
4. For legacy scripts without a CONFIG block, provide the **Access key (legacy)** and **Access secret (legacy)** values, which are passed to the script as the `access_key` and `access_secret` kwargs.
5. To let other organizations use this credential, select the *Make this a global credential* option.
6. Save the credential.

Step 3: Create the task

1. Go to the [custom integration task page](#), click **Integrate** on the Tasks page and choose **Custom Scripts**, or use one of the shortcuts that pre-select the integration: **Save & create task** on the integration page, or the play action on its row of the Custom sources grid.
2. Select the custom integration and the credential created above. Saving the task validates the pair: the script must compile, and the credential must satisfy every parameter the CONFIG block requires — a save with no credential, or a credential missing required parameters, fails with an error naming the parameter keys. (Legacy scripts without a CONFIG block skip this check.)
3. Select the Explorer to run the script from, or leave the selection empty to run the task on the console itself. Console-hosted tasks cannot reach private or link-local addresses — and on runZero-hosted consoles they run under a tighter budget — so use an Explorer for anything that targets your internal network.
4. Set the site, description, and schedule as appropriate.
5. Activate the connection to start the task.

Once the task completes, assets appear in your inventory and can be found with the search `custom_integration:<name>`. Tasks list the integration with its own icon, and the task search `custom_integration:` accepts a name or a UUID — including the UUID of a deleted integration.

Organization access

A custom integration is either available to every organization in the account, or available only to the organizations it has been granted to. Custom integrations that existed before this option was added are available to every organization, so their tasks keep working until an administrator narrows the scope of an integration.

Users must have administrator-level permissions to manage custom integrations. Users with **Administrator** as their [default role](#) can manage every custom integration. Users with [per-organization permissions](#) only see the integrations available to their organizations, and can only edit an integration if they have administrator permissions in every organization it is available to. An integration that is available to every organization can only be edited by a superuser or a user whose default role is Administrator.

Reading an integration's script and its configuration parameters requires administrator permissions in one of the organizations the integration has been granted to; the script of an integration available to every organization is visible only to superusers and users whose default role is Administrator. Users, viewers and annotators can see that an integration exists, including its name and icon, so that the asset data it collected stays readable, but they cannot see its script or its configuration parameters. Selecting an integration when configuring a task requires at least user-level (write) access in an organization the integration is available to; viewers and annotators cannot.

An integration task can only use a custom integration that is available to the organization the task belongs to. Access is checked again each time the task runs, so removing an organization's access also stops tasks that were already scheduled, with an error naming the integration and organization. Deleting an organization removes it from every integration's

access list; an integration that was scoped only to that organization is kept, but is visible only to superusers and account-wide administrators until it is re-scoped.

The same organization scope is exposed on the [account API](#): custom integration objects carry `global` and `organizationIds` fields (mutually exclusive), and organization-scoped API tokens only see integrations shared with their organizations.

Deleting a custom integration

Deleting a custom integration requires typing the word `delete` to confirm, and the confirmation states the impact: the saved script versions are deleted, the data the integration added to assets is removed, and scheduled tasks that use it are **paused** rather than deleted — each shows the reason, and can be re-pointed at another integration. Credentials are kept and can be used by another integration.

Script limits and sandbox behavior

- Scripts are limited to 1 MiB of source. A script sent to the [AI assistant](#) for revision has a much smaller ceiling of 48 KiB.
- Scripts can only `load()` the [registered modules](#); there is no filesystem access or relative import.
- A script run is bounded by an execution-step ceiling and a wall clock, and the deadline also bounds HTTP requests and `time.sleep` calls. A task running on an Explorer or a self-hosted console gets 24 hours and 720 billion steps; one running on a runZero-hosted console gets 2 hours and 240 billion steps. A [console test run](#) is much tighter again, at 120 seconds. Steps are consumed by computation only, so time spent waiting on an API does not count against the ceiling. Self-hosted operators can override both budgets with the `RUNZERO_CUSTOM_INTEGRATION_MAX_STEPS` and `RUNZERO_CUSTOM_INTEGRATION_MAX_RUN_SECONDS` environment variables.
- HTTP response bodies are capped at 1 GiB, and a script's cumulative network reads — across `http`, `requests`, `socket`, `SSH`, and `SMB` — are capped at 4 GiB per run. Other per-call caps include 64 MiB per `SMB` file read, 16 MiB per `SSH` command output stream, and `gzip` decompression bounded at 1 GiB and a 500:1 expansion ratio (see the [library reference](#)).
- A script can hold at most 256 open connections and sessions (`sockets`, `SSH`, `SMB`, `WinRM`, `WMI`, `SQL`) at once; closing one frees its slot.
- Values of `secret` parameters are automatically redacted from `print` output, progress messages, and error text.
- On an Explorer task, every network module — `http`, `requests`, `socket`, and the direct-protocol modules (`runzero.ssh`, `runzero.smb`, `runzero.winrm`, `runzero.wmi`, `runzero.sql`) — can reach the internal addresses visible to that Explorer. On console-hosted runs (both [test runs](#) and tasks run without an Explorer), private and link-local addresses are blocked for every module. The CLI's `--starlark-allow-cidrs` and `--starlark-block-cidrs` flags pin script egress for local runs.

Available libraries

Load only what you use; each module is available via `load(...)`. Full signatures and runnable examples for every module are on the [Starlark libraries](#) page.

Module	Provides
runzero.types	ImportAsset, NetworkInterface, Service, ServiceProtocolData, Software, Vulnerability, to_custom_attributes
kwargs	Typed accessors: require, get_string, get_bool, get_int, get_float, get_list, get_url_base, get_http_tls, get_http_options
coerce	Total (never-raising) conversions for messy API data: as_text, as_dict, as_list, dicts, as_int, as_float, as_bool, dedupe
http	HTTP verbs, get_json/post_json with retries, bearer/basic/oauth2_token, url_encode/url_parse/url_join, multipart
requests	Stateful HTTP Session with sticky headers and cookies
net	ip_address, network_interface, normalize_mac, ip_network, ip_in_network, resolve, plus identity screens routable_ip, routable_ips, clean_hostname, clean_hostnames, mac_key
json	encode, decode, encode_indent, indent
jsonstream	iter_array, iter_lines for large JSON/NDJSON responses
csv	read_all, read_rows, write_all, write_dicts
xml	parse into an element tree
re	RE2 regular expressions: match, find_all, sub, split, compile
time	now, parse_time, parse_ts, parse_duration, from_timestamp, sleep
uuid	new_uuid
base64 / hex / base32	Standard encodings, including raw and URL-safe variants
crypto	Hashes, HMAC, AWS SigV4 signing, CSPRNG output
jwt	encode, decode, decode_unverified
gzip	compress, decompress
flatten_json	flatten nested structures
runzero.progress	report, info, warn task progress in the console
socket	Raw TCP/UDP/TLS connections
runzero.ssh / runzero.smb / runzero.winrm / runzero.wmi / runzero.sql	Direct-protocol collection from systems without a REST API

The predeclared names `report_assets`, `report_asset`, `pager`, `max_pages`, `OPTIONS_TLS`, and `OPTIONS_HTTP` are always available without a `load()`.

Existing Custom Integrations

Name	Setup Instructions	Integration Code
Absolute Secure Endpoint	Link	Link
AdGuard Home	Link	Link
Akamai Guardicore Centra	Link	Link
Armis Centrix	Link	Link
Asimily	Link	Link
Audit Log to Webhook	Link	Link
Automox	Link	Link
BMC Helix Discovery	Link	Link
Bitdefender GravityZone	Link	Link
Bitsight	Link	Link
Carbon Black	Link	Link
Checkmk Raw Edition	Link	Link
Cisco Cyber Vision	Link	Link
Cisco ISE	Link	Link
Cisco Secure Endpoint	Link	Link
Claroty CTD	Link	Link
Claroty xDome	Link	Link
Cortex XDR	Link	Link
CyberArk EPM	Link	Link
Cybereason	Link	Link
Cyberint	Link	Link
Cyberwatch	Link	Link
Device42	Link	Link
Digital Ocean	Link	Link
Drata	Link	Link
EfficientIP SOLIDserver	Link	Link

ExtraHop Reveal(x)	Link	Link
Extreme Networks CloudIQ	Link	Link
Fleet (osquery)	Link	Link
Foreman	Link	Link
Forescout CounterACT	Link	Link
Forescout Risk and Exposure Management	Link	Link
Forescout eyeInspect	Link	Link
Frontline VM	Link	Link
GLPI	Link	Link
Ghost Security	Link	Link
Greenbone (GMP) Import	Link	Link
Greenbone (GMP) Scan Launch	Link	Link
HCL BigFix	Link	Link
HPE Aruba ClearPass	Link	Link
Halcyon	Link	Link
Home Assistant	Link	Link
Icinga 2	Link	Link
Illumio Core	Link	Link
Infoblox NIOS	Link	Link
Ivanti Neurons	Link	Link
JAMF	Link	Link
JumpCloud	Link	Link
Kandji	Link	Link
Kenna Security	Link	Link
Kubernetes	Link	Link
Lansweeper	Link	Link
LibreNMS	Link	Link
LimaCharlie	Link	Link
Linux via SSH	Link	Link
ManageEngine Endpoint Central	Link	Link
Maze	Link	Link

Microsoft Defender for IoT	Link	Link
Microsoft SQL Server databases	Link	Link
Microsoft WSUS	Link	Link
MikroTik RouterOS	Link	Link
Miradore	Link	Link
Mosyle	Link	Link
Nautobot	Link	Link
Netdata	Link	Link
Netdisco	Link	Link
Netskope	Link	Link
Nexthink	Link	Link
NinjaOne	Link	Link
Nozomi Networks	Link	Link
Nutanix Prism	Link	Link
OCS Inventory NG	Link	Link
OPNsense	Link	Link
Open-Audit Community	Link	Link
OpenNMS Horizon	Link	Link
OpenWrt	Link	Link
Palo Alto Networks Device Security	Link	Link
Pi-hole	Link	Link
Portainer / Docker Engine	Link	Link
Proxmox	Link	Link
PuppetDB	Link	Link
Quest KACE SMA	Link	Link
Red Hat Insights	Link	Link
SAP LeanIX	Link	Link
Scale Computing	Link	Link
Scan Passive Assets	Link	Link
Slurp'it	Link	Link
Snipe-IT	Link	Link

Snow License Manager	Link	Link
SolarWinds Information Service	Link	Link
Sophos Central	Link	Link
Stairwell	Link	Link
Sumo Logic	Link	Link
Synology DSM	Link	Link
TP-Link Omada	Link	Link
Tactical RMM	Link	Link
Tailscale	Link	Link
Tenable OT Security	Link	Link
Trellix ePolicy Orchestrator	Link	Link
Trend Micro Vision One	Link	Link
TrueNAS	Link	Link
Ubiquiti UniFi Network	Link	Link
Ubiquiti UniFi Protect	Link	Link
Ubiquiti UniFi Site Manager	Link	Link
Unraid	Link	Link
Uptycs	Link	Link
Vulnerability Workflow	Link	Link
Wazuh	Link	Link
Windows SMB shares	Link	Link
Windows WMI	Link	Link
Workspace ONE UEM	Link	Link
Zabbix	Link	Link
exe.dev	Link	Link
iTop	Link	Link
ntopng	Link	Link
pfSense	Link	Link
phpIPAM	Link	Link
runZero Task Sync	Link	Link

Starlark libraries

This page is the reference for the Starlark modules available to [custom integration scripts](#), with a runnable example for each. Save any example as a `.star` file and run it with the [runZero CLI](#):

```
runzero script --filename example.star
```

Load only the names you use; every module is available via `load(...)`. The names `report_assets`, `report_asset`, `pager`, `max_pages`, `OPTIONS_TLS`, and `OPTIONS_HTTP` are predeclared and never need a `load()`.

```
load("runzero.types", "ImportAsset", "NetworkInterface", "Service",
      "ServiceProtocolData", "Software", "Vulnerability",
      "to_custom_attributes")
load("kwargs", "require", "has", "get", "get_string", "get_bool", "get_int",
      "get_float", "get_list", "get_url_base", "get_http_tls",
      "get_http_options")
load("http", http_get="get", http_post="post", "head", "put", "patch", "delete",
      "get_json", "post_json", "url_encode", "url_parse", "url_join",
      "multipart", "bearer", "basic", "oauth2_token")
load("requests", "Session", "Cookie")
load("net", "ip_address", "network_interface", "normalize_mac",
      "ip_network", "ip_in_network", "resolve",
      "routable_ip", "routable_ips", "clean_hostname", "clean_hostnames",
      "mac_key")
load("coerce", "as_text", "as_dict", "as_list", "dicts",
      "as_int", "as_float", "as_bool", "dedupe")
load("json", json_encode="encode", json_decode="decode")
load("jsonstream", "iter_array", "iter_lines")
load("csv", csv_read="read_all", csv_write="write_dicts")
load("xml", xml_parse="parse")
load("re", re_match="match", re_find_all="find_all", re_sub="sub")
load("time", "now", "parse_time", "parse_ts", "parse_duration", "sleep")
load("uuid", "new_uuid")
load("base64", base64_encode="encode", base64_decode="decode")
load("hex", hex_encode="encode", hex_decode="decode")
load("base32", b32_encode="encode", b32_decode="decode")
load("crypto", "sha256", "hmac_sha256", "sign_v4", "random_hex")
load("jwt", jwt_encode="encode", jwt_decode="decode")
load("gzip", gzip_compress="compress", gzip_decompress="decompress")
load("flatten_json", "flatten")
load("runzero.progress", progress_report="report", progress_info="info")
load("socket", "tcp", "udp", "tls")
load("runzero.ssh", ssh_dial="dial")
load("runzero.smb", smb_dial="dial")
load("runzero.winrm", winrm_dial="dial")
load("runzero.wmi", wmi_dial="dial")
load("runzero.sql", sql_connect="connect")
```

runzero.types

Constructors for the objects a script imports. Every keyword argument accepts both camelCase (osVersion) and snake_case (os_version) spellings.

- `ImportAsset(id, ...)` — `id` is required and must be a stable, unique string (up to 1024 characters). Optional fields: `hostnames`, `domain`, `os`, `osVersion`, `manufacturer`, `model`, `deviceType`, `assetType`, `tags`, `firstSeenTS`, `lastSeenTS`, `networkInterfaces`, `services`, `software`, `vulnerabilities`, `customAttributes`, `runZeroID`, `trust_device_type`, `trust_os`, `trust_os_version`. Merge policy is no longer set per asset: declare `matchBehavior` in the **CONFIG block** instead — passing it to `ImportAsset` is an error.
- `NetworkInterface(macAddress=None, ipv4Addresses=[], ipv6Addresses=[])` — or build one with `net.network_interface`, which handles mixed and messy input.
- `Service(address, port, transport, ...)` — the first three are required; `vendor`, `product`, `version`, `protocolData`, and `customAttributes` are optional.
- `ServiceProtocolData(name, attributes={})` — protocol-specific details attached to a `Service`.
- `Software(id, ...)` — optional `vendor`, `product`, `version`, `update`, `cpe23`, `serviceAddress`, `serviceTransport`, `servicePort`, `installedAt`, `installedSize`, `installedFrom`, and more.
- `Vulnerability(id, ...)` — optional `name`, `description`, `solution`, `cve` (must match CVE-YYYY-NNNN), `cpe23`, `cvss2BaseScore`, `cvss3BaseScore`, `severityRank/riskRank` (0=Info through 4=Critical), `severityScore/riskScore`, `exploitable`, `service linkage`, and `timestamps`.
- `to_custom_attributes(value, ...)` — coerces arbitrary nested data into the string `-> string` map required by `customAttributes`. Nested dicts are flattened with `separator` (default `"."`), lists joined with `list_join` (default `" "`), empty values dropped unless `drop_empty=False`, and `prefix`, `exclude`, `max_key`, `max_value`, and `max_entries` are available for fine control. The platform caps custom attributes at 1024 entries, with keys up to 256 and values up to 1024 characters.

Limits worth knowing: up to 256 network interfaces and 100 hostnames (260 characters each) per asset; up to 100 tags. `hostnames` and `tags` silently drop empty strings, but a `None` entry in `hostnames` is an error that aborts the run — `hostnames=[device.get("hostname")]` fails whenever the field is absent. Pass `hostname` fields through `net.clean_hostnames`, which also screens out placeholder names.

```
load("runzero.types", "ImportAsset", "Service", "ServiceProtocolData",
     "Software", "Vulnerability", "to_custom_attributes")
load("net", "network_interface")
```

```
def main(*args, **kwargs):
    nic = network_interface(
        mac="9C:B6:D0:F1:D2:04",
        ips=["192.0.2.5", "fe80::1%eth0", "[2001:db8::1]:443"],
    )
    svc = Service(
        address="192.0.2.5",
        port=443,
        transport="tcp",
        vendor="nginx",
        product="nginx",
        version="1.25.3",
```

```

ServiceProtocolData(name="http", attributes={"server": "nginx"}),
    ServiceProtocolData(name="tls", attributes={"subject": "CN=acme"})),
],
)
asset = ImportAsset(
    id="device-123",
    hostnames=["web1.acme.local"],
    os="Linux",
    osVersion="6.1",
    deviceType="Server",
    manufacturer="Acme",
    model="Rack-42",
    tags=["prod", "web"],
    networkInterfaces=[nic],
    services=[svc],
    software=[Software(id="pkg-1", vendor="F5", product="NGINX", version="1.25.3",
        serviceAddress="192.0.2.5")],
    vulnerabilities=[Vulnerability(id="vuln-1", name="Example finding",
        cve="CVE-2023-0001", severityRank=3,
        severityScore=8.1, riskRank=3, riskScore=8.1)],
    customAttributes=to_custom_attributes({
        "name": "web1",
        "labels": ["prod", "edge"],          # joined with ","
        "sys": {"os": "linux", "ver": 5},    # flattened with "."
        "empty": "",                        # dropped
    }),
)
return [asset]

```

kwargs

Typed, validating accessors over the `**kwargs` dict passed to `main`. Values arrive as strings; these helpers coerce them and apply defaults.

- `require(kwargs, *keys)` — error if any key is missing or blank.
- `has(kwargs, key)` — True when the key is present and non-empty.
- `get(kwargs, key, default="")` / `get_string(...)` — aliases.
- `get_bool(kwargs, key, default=False)` — accepts true/false, 1/0, yes/no, y/n, on/off.
- `get_int(kwargs, key, default=0)` / `get_float(kwargs, key, default=0.0)`
- `get_list(kwargs, key, default=[], sep=",")` — splits a delimited string, or passes a list through.
- `get_url_base(kwargs, key="url", default="")` — extracts scheme + host from a URL kwarg, dropping any path or query.
- `get_http_tls(kwargs, prefix="tls_")` — collects the [OPTIONS_TLS include](#) into a `tls=` dict.
- `get_http_options(kwargs, prefix="http_", tls_prefix="tls_", headers=None)` — collects the `OPTIONS_HTTP` and `OPTIONS_TLS` includes into a dict of keyword arguments for any http function.

```
load("kwargs", "require", "has", "get_string", "get_bool", "get_int", "get_list")
```

```
def main(*args, **kwargs):
```

```

client_id = get_string(kwargs, "client_id")
page_size = get_int(kwargs, "page_size", default=100)
include_offline = get_bool(kwargs, "include_offline", default=False)
regions = get_list(kwargs, "regions", default=[])
print(client_id, page_size, include_offline, regions)
if has(kwargs, "region"):
    print("region override:", get_string(kwargs, "region"))

```

```
runzero script --filename example.star --kwargs client_id=abc --kwargs client_secret=xyz --kwargs include_offline=yes
```

coerce

Total conversions for messy API data: nothing in this module raises, so a malformed or missing field degrades to a default instead of aborting the import.

- `as_text(value, default="", join="")` — any value to a string; lists are joined with `join` when set.
- `as_dict(value)` — a dict, or `{}` for anything else.
- `as_list(value, wrap=True)` — a list; scalars are wrapped in a one-element list unless `wrap=False`.
- `dicts(value)` — the dict elements of a list, dropping everything else.
- `as_int(value, default=0)` / `as_float(value, default=0.0)` — numbers from ints, floats, or numeric strings.
- `as_bool(value, default=False)` — the usual true/false spellings.
- `dedupe(values, fold_case=False)` — order-preserving deduplication.

```
load("coerce", "as_text", "as_int", "as_list", "dicts")
```

```

def main(*args, **kwargs):
    device = {"name": None, "port": "8443", "ips": "192.0.2.5",
              "disks": [{"id": 1}, {"oops", {"id": 2}}]}
    print("name:", as_text(device["name"], default="unknown"))
    print("port:", as_int(device["port"]))
    print("ips:", as_list(device["ips"]))    # a scalar becomes a one-item list
    print("disks:", dicts(device["disks"])) # non-dict entries dropped

```

http

Stateless HTTP requests, JSON convenience wrappers, auth helpers, and URL utilities.

Verbs — `get(url, headers=None, params=None, timeout=60, insecure_skip_verify=False, tls=None)` and `head(...)` take the same arguments; `post`, `put`, `patch`, and `delete` additionally accept `body=` (bytes/string) or `json=` (auto-encoded dict). All verbs default to a 60-second timeout, and `params=` replaces any query string already present in the URL. Each returns a response struct with `status_code` (int), `status` (string), `headers` (values are lists of strings), and `body`.

get_json / post_json — the common fetch-check-decode pattern in one call, returning a (data, err) tuple. `err` is `None` on success, or a short string: "status 401: ..." for an HTTP failure, or the transport error text for a connection failure. A 2xx response with an empty body decodes to `None`. Both also accept `insecure_skip_verify=` and `tls=`. Transient failures are retried by default: `retries` defaults to 3, with exponential backoff (`retry_backoff`, default 1 second, doubling up to `retry_max_backoff`, default 30 seconds) on transient statuses (408, 425, 429, 500, 502, 503, 504 by

default; override with `retry_on=[...]`, honoring `Retry-After` headers. Pass `retries=0` to opt out — do so for non-idempotent writes.

Auth helpers — `bearer(token)` and `basic(username, password)` format `Authorization` header values. `oauth2_token(token_url, client_id, client_secret, scope=None, audience=None, grant_type="client_credentials", extra=None, headers=None, timeout=60, insecure_skip_verify=False, tls=None)` performs an OAuth token exchange and returns the access token string.

URL utilities — `url_encode(params)` builds a query string; `url_parse(url)` returns a struct (`scheme`, `host`, `hostname`, `port`, `path`, `query`, `fragment`, ...) or `None`; `url_join(base, ref)` resolves a relative reference (useful for pagination next links); `multipart(fields)` builds a multipart/form-data body and returns (`body_bytes`, `content_type`).

```
load("http", "get_json", "url_parse", "url_join", "bearer")
```

```
def main(*args, **kwargs):
    url = "https://hacker-news.firebaseio.com/v0/topstories.json"
    data, err = get_json(url, headers={"Accept": "application/json"}, retries=2)
    if err:
        print("fetch failed:", err)
        return None
    print("top story IDs:", data[:5])

    u = url_parse(url)
    print("host:", u.host, "path:", u.path)
    print("next page:", url_join(url, "/v0/beststories.json"))
    print("auth header:", bearer("example-token"))
```

In an integration, collect the [option-suite settings](#) once and splat them into every call:

```
http_options = get_http_options(kwargs, headers={"Authorization": bearer(token)})
data, err = get_json(url, params={"limit": 100}, **http_options)
```

The `tls=` argument accepts a dict with the keys `insecure`, `server_name`, `ca_pem`, `client_cert_pem`, `client_key_pem`, and `thumbprints` (SHA-256 pins); unknown keys are an error. Use the raw verbs instead of `get_json` when you need response headers, cookies, or the status code directly.

Where an egress filter applies (console-hosted runs, or the CLI's `--starlark-allow-cidrs/--starlark-block-cidrs` flags), a request to a refused address aborts the script rather than returning an `err` tuple.

requests

A stateful HTTP session with sticky headers and a cookie jar. Verbs (`get`, `post`, `put`, `patch`, `delete`, `head`) accept headers, cookies, params, body, json, and timeout (default 60 seconds), and return the same response struct as the `http` module.

Note that `Session` does not accept the `tls=` dict; only `Session(insecure_skip_verify=True)` is available. Prefer the `http` module when the [TLS option suite](#) matters.

```
load("requests", "Session", "Cookie")
load("json", json_decode="decode")
```

```
def main(*args, **kwargs):
    session = Session()
    session.headers.set("Accept", "application/json")
    session.headers.set("User-Agent", "runZero-Example/1.0")

    url = "https://hacker-news.firebaseio.com/v0/topstories.json"
    response = session.get(url)
    if response.status_code == 200:
        data = json_decode(response.body)
        print("top story IDs:", data[:5])
    else:
        print("request failed with status:", response.status_code)
```

`session.headers` exposes `get(key)` and `set(key, value)` (`value=None` deletes a header). `session.cookies` exposes `get(url)`, `set(url, cookies)`, and `clear()`; `Cookie(name, value, path="", domain="", secure=False, ...)` builds a full cookie object for `cookies.set`.

net

IP, MAC, and DNS helpers, plus identity screens that keep placeholder values out of asset matching.

- `ip_address(s)` — validates one address; the result exposes `.version` (4 or 6) and stringifies to canonical form.
- `network_interface(mac=None, ips=None, ipv4=None, ipv6=None)` — builds a `NetworkInterface` from messy input: mixed v4/v6 lists, `addr:port` and `[addr]:port` suffixes, and `%zone` IDs are handled, duplicates dropped, and up to 99 addresses kept per family. Returns `None` when nothing usable remains, so if `nic:` guards work.
- `normalize_mac(s, preserve_bits=True)` — canonical lowercase colon form from any common format (colons, dashes, Cisco dotted, bare hex); returns `None` for unparseable input. Pass `preserve_bits=False` to clear the locally-administered bit, the historical matching form; `network_interface` stores the asset MAC with the bit cleared regardless.
- `mac_key(s)` — `normalize_mac` plus rejection of values that cannot identify a device: all-zero and broadcast addresses and synthetic `ip-<addr>` placeholders return `None`.
- `routable_ip(value, exclude=None)` / `routable_ips(values, exclude=None)` — the canonical identity-usable address (or `None`), rejecting loopback, unspecified, link-local, multicast, and broadcast addresses while keeping RFC1918, CGNAT, and ULA space. Accepts bracketed, `host:port`, `%zone`, and CIDR-suffixed input. `exclude=[...]` overrides the exclusion list; `exclude=[]` disables it.
- `clean_hostname(value, extra=None, max_length=253)` / `clean_hostnames(values, ...)` — a usable hostname or `None`, rejecting placeholders (`localhost`, `unknown`, `none`, `-`, `n/a`), IP-shaped values, all-numeric names, over-long names, and illegal DNS characters. `extra=[...]` adds source-specific placeholders; the plural form also deduplicates case-insensitively.
- `ip_network(cidr)` — struct with `cidr`, `version`, `prefix`, `network`, `broadcast`, `netmask`, and a `contains(ip)` method.
- `ip_in_network(ip, cidr)` — one-shot membership check; `False` for malformed or mixed-family input.
- `resolve(host, timeout=10)` — A/AAAA lookup returning a list of IP addresses; returns an empty list (never an error) for unresolvable input.

```

load("net", "ip_address", "network_interface", "normalize_mac",
     "ip_network", "ip_in_network", "resolve",
     "routable_ip", "clean_hostnames", "mac_key")

def main(*args, **kwargs):
    addr = ip_address("192.0.2.5")
    print("ip:", addr, "version:", addr.version)
    print("mac:", normalize_mac("9CB6.D0F1.D204"))

    net10 = ip_network("10.0.0.0/8")
    print("contains:", net10.contains("10.1.2.3"))
    print("in network:", ip_in_network("10.1.2.3", "10.0.0.0/8"))

    for ip in resolve("localhost"):
        print("resolved:", ip, ip.version)

    nic = network_interface(mac="9c-b6-d0-f1-d2-04",
                           ips=["192.0.2.5", "fe80::1%eth0", "[2001:db8::1]:443"])
    print("nic:", nic)

    print("routable:", routable_ip("127.0.0.1"), routable_ip("10.1.2.3:8443"))
    print("hostnames:", clean_hostnames(["web1.acme.local", "localhost", None]))
    print("mac key:", mac_key("00:00:00:00:00:00")) # None: not an identity

```

json

```

load("json", json_encode="encode", json_decode="decode")

def main(*args, **kwargs):
    data = {"name": "runZero", "features": ["scan", "API", "integrations"]}
    encoded = json_encode(data)
    print("encoded:", encoded)
    decoded = json_decode(encoded)
    print("decoded name:", decoded["name"])

```

The module also provides `encode_indent` and `indent` for pretty-printed output.

jsonstream

Stream large JSON documents without materializing them. `iter_array(body, path=None)` iterates the elements of an array (optionally at a dot-separated path inside the document); `iter_lines(body)` iterates NDJSON / JSON-lines input. Both accept strings or bytes — pass a response body directly.

```

load("jsonstream", "iter_array", "iter_lines")

def main(*args, **kwargs):
    big_json = '{"data": {"items": [{"id": 1}, {"id": 2}]}}'
    for item in iter_array(big_json, path="data.items"):
        print("streamed id:", item["id"])

```

```
for line in iter_lines('{ "a": 1}\n{ "a": 2}'):
    print("ndjson:", line["a"])
```

csv

`read_all(text, delimiter=",", comment="", header=True)` returns a list of dicts keyed by the header row (or a list of lists with `header=False`); `read_rows` is the header-less form. `write_all(rows)` and `write_dicts(rows, fields=None)` serialize back to CSV strings.

```
load("csv", csv_read="read_all", csv_write="write_dicts")
```

```
def main(*args, **kwargs):
    rows = csv_read("id,name\n1,web1\n2,web2\n")
    print("first row:", rows[0]["name"])
    print("as csv:", csv_write(rows))
```

xml

`parse(text)` returns an element tree. Elements expose `tag`, `text`, `tail`, `attrib`, and `children`, plus the methods `find(path)`, `find_all(path)`, `get(name, default="")`, and `text_all()`.

```
load("xml", xml_parse="parse")
```

```
def main(*args, **kwargs):
    doc = xml_parse("<devices><device><name>web1</name></device></devices>")
    print("name:", doc.find("device/name").text)
    print("count:", len(doc.find_all("device")))
```

re

Regular expressions using Go RE2 syntax. `match(pattern, string)` and `search(...)` return a struct (`.match`, `.start`, `.end`, `.groups`, and `.named` for named groups) or `None`. Also available: `find_all`, `find_all_groups`, `sub(pattern, repl, string, count=-1)`, `split`, `escape`, and `compile(pattern)` for a reusable pattern object.

```
load("re", re_match="match", re_find_all="find_all", re_sub="sub")
```

```
def main(*args, **kwargs):
    matches = re_find_all(r"id=(\d+)", "id=10 id=20")
    print("matches:", matches)
    print("cleaned:", re_sub(r"\s+", " ", "a b\tc"))
    m = re_match(r"(?P<major>\d+)\.(?P<minor>\d+)", "5.1")
    print("named groups:", m.named)
```

time

The standard Starlark time module plus a sandbox-aware `sleep`.

- `parse_time(s)` — parses RFC 3339 strings into a time value with `year`, `month`, `day`, `hour`, `minute`, `second`, `unix`, and `unix_nano` fields, plus `format(layout)` and `in_location(name)` methods. It raises

on input it does not recognize, and a raise from a builtin aborts the whole script — use it for literals, and `parse_ts` for API-supplied values.

- `parse_ts(value, default=None, assume_utc=True, clamp_to_now=True, unit="s")` — a never-raising timestamp parser for values a vendor API returns: epoch ints, floats, and numeric strings (unit selects s/ms/us/ns) plus a wide set of datetime layouts, with zone-less values read as UTC. Non-positive epochs yield default, and future values are clamped to now, because the platform drops an `ImportAsset` whose first- or last-seen timestamp is ahead of the clock.
- `parse_duration(s)` — parses "90m", "1h30m", "250ms", and similar into a duration with hours, minutes, seconds, and related fields. Time and duration values support arithmetic.
- `now()` — the current time. `from_timestamp(sec)` converts a Unix timestamp.
- `sleep(duration)` — accepts a duration string or value, and respects the task deadline.

```
load("time", "now", "parse_time", "parse_ts", "parse_duration", "sleep")
```

```
def main(*args, **kwargs):
    t = parse_time("2026-08-14T15:00:00Z")
    print("unix:", t.unix, "year:", t.year)
    print("formatted:", t.format("2006-01-02"))

    seen = parse_ts("2026-08-14 15:00:00") # zone-less, read as UTC
    print("api time:", seen)
    print("bad input:", parse_ts("not-a-date")) # None, never a crash

    d = parse_duration("90m")
    print("minutes:", d.minutes)
    print("90 minutes from now:", now() + d)
    sleep("250ms")
```

uuid

```
load("uuid", "new_uuid")
```

```
def main(*args, **kwargs):
    print("generated UUID:", new_uuid())
```

base64, hex, and base32

base64 provides `encode/decode` plus `raw_encode/raw_decode` (unpadded), `url_encode/url_decode` (URL-safe alphabet), and `raw_url_encode/raw_url_decode` (both); its decode functions return strings. hex provides `encode/decode`, and base32 provides `encode/decode` plus unpadded `raw_encode/raw_decode`; their decode functions return bytes.

```
load("base64", base64_encode="encode", base64_decode="decode")
load("hex", hex_encode="encode", hex_decode="decode")
load("base32", b32_encode="encode")
```

```
def main(*args, **kwargs):
    enc = base64_encode("user:pass")
    print("base64:", enc, "->", base64_decode(enc))
    print("hex:", hex_encode("hi"))
    print("base32:", b32_encode("hi"))
```

crypto

Hashes, HMAC, AWS request signing, and random values.

- **Hashes:** sha1, sha256, sha512, md5 — accept strings, return hex.
- **HMAC:** hmac_sha1/hmac_sha256/hmac_sha512(key, data, output="hex") and the generic hmac(algorithm, key, data, output="hex"); output may be "hex", "base64", "base64_raw", or "bytes".
- sign_v4(method, url, headers, body, access_key, secret_key, region, service, session_token=None, timestamp=None) — returns the AWS Signature V4 headers (Authorization, X-Amz-Date, X-Amz-Content-Sha256, and X-Amz-Security-Token when a session token is given).
- random_bytes(n) / random_hex(n) — cryptographically secure random output.

```
load("crypto", "sha256", "hmac_sha256", "random_hex", "sign_v4")
```

```
def main(*args, **kwargs):
    print("sha256:", sha256("test"))
    print("hmac:", hmac_sha256("key", "message"))
    print("random:", random_hex(8))

    headers = sign_v4("GET", "https://ec2.us-east-1.amazonaws.com/?Action=DescribeInstances",
                      {}, "", "AKIDEXAMPLE", "EXAMPLESECRET", "us-east-1", "ec2")
    print("signed headers:", sorted(headers.keys()))
```

jwt

encode(claims, key, algorithm="HS256", headers=None), decode(token, key, algorithms=None) (verifies the signature), and decode_unverified(token) (returns {"header": ..., "claims": ...} without verification). HS, RS, PS, ES, and EdDSA algorithm families are supported; the none algorithm is rejected.

```
load("jwt", jwt_encode="encode", jwt_decode="decode", "decode_unverified")
```

```
def main(*args, **kwargs):
    token = jwt_encode({"sub": "runzero", "exp": 1893456000}, "example-signing-key")
    print("claims:", jwt_decode(token, "example-signing-key"))
    print("header:", decode_unverified(token)["header"])
```

gzip

compress(data) and decompress(data) operate on bytes and return bytes. Decompression is bounded at 1 GiB of output and a 500:1 expansion ratio to guard against decompression bombs.

```
load("gzip", gzip_compress="compress", gzip_decompress="decompress")
```

```
def main(*args, **kwargs):
    original = bytes("Hello, runZero!")
    compressed = gzip_compress(original)
    print("compressed length:", len(compressed))
    print("round trip:", gzip_decompress(compressed))
```

flatten_json

`flatten(input, separator="_", root_keys_to_ignore=None)` flattens nested structures into a single-level dict. For building `CustomAttributes`, prefer [to_custom_attributes](#), which also stringifies values and applies the platform limits.

```
load("flatten_json", "flatten")
```

```
def main(*args, **kwargs):
    flat = flatten({"a": {"b": 1, "c": 2}, "d": 3})
    print("flattened:", flat)    # {"a_b": 1, "a_c": 2, "d": 3}
```

runzero.progress

Surface progress and log lines from a running task in the runZero console.

- `report(pct, msg="")` — percentage is clamped to 0–100; calls within 250 ms are coalesced; messages are truncated to 256 bytes.
- `info(msg) / warn(msg)` — emit log lines through the task logger.

```
load("runzero.progress", progress_report="report", progress_info="info",
      progress_warn="warn")
```

```
def main(*args, **kwargs):
    progress_report(0, "starting sync")
    progress_info("fetched page 1")
    progress_warn("retrying after 429")
    progress_report(100, "done")
```

report_assets, report_asset, and pager (predeclared)

Stream `ImportAsset` values to runZero while the script runs, instead of returning them all from `main`. No `load()` is required. See [streaming large imports](#) for the pagination pattern.

- `report_asset(asset)` — report exactly one asset (or `None`, a no-op), returning 1 or 0. Streaming batches internally, so per-record reporting is the preferred pattern; there is nothing to gain by accumulating a page into a list first.
- `report_assets(...)` — accepts a single asset, several positional assets, or a list/tuple, returning the count.
- `pager(label="pages", limit=0)` — a loop guard for pagination. `p.next()` is the `while` condition; reaching the page ceiling ([CONFIG](#) `maxPages`, default 1,000,000) is an error naming the label, never a silent stop. `limit=` may lower the effective ceiling but not raise it, and `p.page` is the 1-based current page.
- `max_pages()` — the effective page ceiling.

```
load("runzero.types", "ImportAsset")
```

```
def main(*args, **kwargs):
    a = ImportAsset(id="asset-1")
    b = ImportAsset(id="asset-2")
```

```

report_asset(a)          # one asset; the preferred streaming pattern
    report_assets(a, b)   # several positional assets
    n = report_assets([a, b]) # a list, returning the count
    print("reported", n)

p = pager(label="devices")
while p.next():
    print("page", p.page)
    break # a real script would fetch until the API runs dry
return None

```

socket

Raw TCP, UDP, and TLS connections for simple line- or byte-oriented protocols.

- `tcp(host, port, timeout=30, tls=None, insecure_skip_verify=False, server_name=None)` — `tls=` accepts a bool or the same dict of TLS overrides as the [http module](#) (`insecure`, `server_name`, `ca_pem`, `client_cert_pem`, `client_key_pem`, `thumbprints`), so `kwargs.get_http_tls()` output can be splatted straight into a socket.
- `udp(host, port, timeout=30)`
- `tls(host, port, timeout=30, insecure_skip_verify=False, server_name=None, tls=None)`

Sockets expose `send(data, timeout=None)`, `recv(max=..., timeout=None)` (max defaults to 16 MiB, per-call ceiling 200 MiB), `recv_exact(n, ...)`, `recv_line(...)` and `recv_until(delim, ...)` (max defaults to 1 MiB), `starttls(...)` (TCP only, also accepts `tls=`), `set_timeout(seconds)`, and `close()`, plus the attributes `local_addr`, `remote_addr`, `is_tls`, `closed`, and `network`. Always `close()` sockets when done.

On an Explorer task, sockets can reach the internal addresses visible to that Explorer, the same as the direct-protocol modules — scope the Explorer and the credential to the intended system. On console-hosted runs, private and internal addresses are blocked for every module, sockets included.

```

load("socket", "tls")

def main(*args, **kwargs):
    host = kwargs.get("host", "example.com")
    sock = tls(host, 443, timeout=10)
    print("connected to:", sock.remote_addr, "tls:", sock.is_tls)
    sock.close()

```

Direct-protocol modules

For sources without a REST API, these modules open authenticated connections from the Explorer the task runs on, and can reach internal addresses visible to that Explorer. They return session objects — always `close()` them when done: a script can hold at most 256 open connections and sessions at once, and network reads across every module share a 4 GiB per-run budget. Scripts using these modules should declare `"validationMode": "compile"` since `--validate` cannot dial a real endpoint.

The examples below read their targets from `--kwargs` so they can run against a test system:

```
runzero script --filename example.star --kwargs host=192.0.2.10 --kwargs username=svc --kwargs password=...
```

runzero.ssh

`dial(host, username, password=None, private_key=None, private_key_passphrase=None, host_key=None, host_keys=None, insecure_ignore_host_key=False, port=22, timeout=30)`. There is no trust-on-first-use: pin the expected host key(s) — in `authorized_keys` or PEM form, for example from `ssh-keyscan` — or explicitly pass `insecure_ignore_host_key=True`; combining pins with the `insecure` flag is an error. Password auth also answers keyboard-interactive prompts, and timeouts clamp at 600 seconds.

Sessions provide:

- `run(command, stdin=None, timeout=0)` — returns `(stdout, stderr, exit_code)`; a timeout of 0 uses the session timeout. `stdin` is capped at 1 MiB and each output stream at 16 MiB; `run_command` is an alias.
- `stream(command, stdin=None, timeout=0)` — starts a long-lived remote command and returns a stream with `send(data)`, `recv(max=..., timeout=None)` (max defaults to 1 MiB; `b""` on EOF), `close_stdin()`, `close()`, and the attributes `exit_code` (None until `exit`), `stderr`, and `closed`. Output is consumed incrementally with no per-stream cap, so a stream can process more than `run` can return.
- `open_unix(path) / open_tcp(host, port)` — port forwarding to a remote Unix socket or a TCP address reachable from the remote host, returning a full [socket](#) object. This is how a script reaches a service that only listens locally on the remote system, such as a management socket or a loopback-only listener.

Sessions also expose `host`, `port`, `username`, and `close()`.

```
load("runzero.ssh", ssh_dial="dial")
```

```
def main(*args, **kwargs):
    host = kwargs.get("host")
    if not host:
        print("pass --kwargs host=... --kwargs username=... --kwargs password=... to run")
        return None
    session = ssh_dial(host=host, username=kwargs["username"],
                      password=kwargs.get("password"),
                      insecure_ignore_host_key=True)
    stdout, stderr, code = session.run("uname -a")
    session.close()
    print("exit:", code, "output:", stdout)
```

runzero.smb

`dial(host, username, password="", domain="", nt_hash=None, port=445, timeout=30)` — `DOMAIN\user` shorthand is accepted in `username`, and timeouts clamp at 600 seconds. Sessions provide `list_shares()` and `mount(share)`; a mounted share offers `read(path, limit=0)` (single reads capped at 64 MiB), `list(path="/")`, `stat(path)`, `exists(path)`, `unmount()`, `close()`, and a `name` attribute.

Access refusals are values, not errors: `mount()` and `list()` return `None` on access denied — distinct from an empty list — so a script can skip a share it cannot read and keep going. `read()` and `stat()` still raise; `exists()` is `stat`'s soft counterpart.

```
load("runzero.smb", smb_dial="dial")
```

```
def main(*args, **kwargs):
    host = kwargs.get("host")
    if not host:
        print("pass --kwargs host=... --kwargs username=... --kwargs password=... to run")
        return None
    session = smb_dial(host=host, username=kwargs["username"], password=kwargs["password"])
    print("shares:", session.list_shares())
    session.close()
```

runzero.winrm

dial(host, username, password, port=0, https=False, insecure_skip_verify=False, ca_cert=None, auth="ntlm", timeout=60) (port 0 selects the default, 5985 or 5986 with https=True). Sessions provide run(command) (alias run_command), run_powershell(script) (alias run_ps), and wql(query, namespace="root/cimv2"), plus the attributes host, port, and https.

```
load("runzero.winrm", winrm_dial="dial")
```

```
def main(*args, **kwargs):
    host = kwargs.get("host")
    if not host:
        print("pass --kwargs host=... --kwargs username=... --kwargs password=... to run")
        return None
    session = winrm_dial(host, kwargs["username"], kwargs["password"])
    stdout, stderr, code = session.run_powershell("Get-ComputerInfo | Select-Object OsName")
    session.close()
    print(stdout)
```

runzero.wmi

dial(host, username, password, transport="tcp", namespace="//./root/cimv2", target_name="", timeout=60, port=0, smb_port=445) (port 0 selects the default, 135). Sessions provide query(query, limit=0, page=100) returning a list of dicts.

```
load("runzero.wmi", wmi_dial="dial")
```

```
def main(*args, **kwargs):
    host = kwargs.get("host")
    if not host:
        print("pass --kwargs host=... --kwargs username=... --kwargs password=... to run")
        return None
    session = wmi_dial(host, kwargs["username"], kwargs["password"])
    rows = session.query("SELECT Caption, Version FROM Win32_OperatingSystem")
    session.close()
    print(rows)
```

runzero.sql

connect(driver, dsn, timeout=30, max_open_conns=4, max_idle_conns=4, conn_max_lifetime=0) with drivers postgres, mysql, and mssql (and common aliases). Sessions provide query(query, params=None,

`limit=1000000, timeout=0)` returning a list of dicts (row limit up to 1,000,000), and `exec(query, params=None, timeout=0)` returning a dict with `rows_affected` and `last_insert_id`. A timeout of 0 uses the session timeout; timeouts clamp at 600 seconds. DSNs are restricted to network connections: file-reading options (such as `sslkey`, `passfile`, or `allowAllFiles`) and Unix-socket or named-pipe transports are rejected.

```
load("runzero.sql", sql_connect="connect")

def main(*args, **kwargs):
    dsn = kwargs.get("dsn")
    if not dsn:
        print("pass --kwargs dsn=postgres://user:pass@host:5432/db to run")
        return None
    db = sql_connect("postgres", dsn)
    rows = db.query("SELECT version()")
    db.close()
    print(rows)
```

Dragos

runZero integrates with Dragos to help expand your OT visibility by importing assets and vulnerabilities from the Dragos API.

Getting started with Dragos

To set up an integration with Dragos, you'll need to:

1. Generate a Dragos API ID and API Secret with the following privileges:
 - `asset:read`
 - `detection:read`
 - `vulnerability:read`
2. Configure the Dragos credential in runZero.
3. Activate the integration to pull your data into runZero.

Step 1: Generate a Dragos API ID and API Secret

1. Log into the Dragos console and navigate to Admin, then navigate to Users. Under the user, click the Add New API Key button.
2. When the Generate New API Key box appears, name your API ID and API Secret, then click Generate Key.
3. Copy the ID and Secret for later.

Step 2: Add the Dragos API ID and API Secret to runZero

1. Go to the [Credentials](#) page in runZero.
2. Choose **Dragos API ID & Secret** from the list of credential types.
3. Provide a name for the credential, like Dragos.
- 4.

4. Provide the following information:
 - **Dragos API ID** - Your Dragos API ID from Step 1.
 - **Dragos API Secret** - Your Dragos API Secret from Step 1.
 - **Dragos API URL** - The URL to your Dragos instance, without any trailing slashes. For example, `https://my.instance.dragos.cloud`.
 - **Insecure** - Enable this option to approve authenticating with untrusted endpoints. When enabled, certificate validation is disabled. Use with caution.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Verify and save the credential. Note that if the URL provided is an internal IP address, verification is unsupported, but the credential can be saved and the integration will still be usable when run on an Explorer.

You're now ready to set up and activate the connection to bring in data from Dragos.

Step 3: Choose how to configure the Dragos integration

The Dragos integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your Dragos credential, you'll need to sync your data from Dragos.

Step 4a: Configure the Dragos integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Dragos-only assets are created.

1. Activate a connection to [Dragos](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Enter a name for the task, like *Dragos Sync* (optional).
4. Enter a description for the task (optional).
5. Choose the Explorer to perform this connector task from (optional).
6. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
7. If you want to exclude assets that have not been scanned by runZero from your integration import, select the **Exclude assets that cannot be merged into an existing asset** option. By default, the integration will include assets that have not been scanned by runZero.
8. If you want to skip importing vulnerabilities detected on Dragos assets, select the **Disable importing vulnerabilities from the Dragos inventory** option. By default, the integration will import both assets and vulnerabilities.

9. Choose the combination of **Priorities**, **Severities**, **Tags**, and **Subnets** criteria to filter assets by. These criteria correspond to the available filter criteria within Dragos.
10. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
11. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Dragos integration as a scan probe

You can run the Dragos integration as a scan probe so that the runZero Explorer will pull your Dragos assets into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *DRAGOS* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct *DRAGOS* credential to *Yes* in the *Credentials* tab.

Step 5: View Dragos assets

After a successful sync, you can [go to your inventory](#) to view your Dragos assets. These assets will have a Dragos icon listed in the **Source** column.

To filter by Dragos assets, consider running the following queries:

- [View all Dragos assets](#):

```
source:Dragos
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from Dragos.

Google Cloud Platform

The Google Cloud Platform (GCP) integration provides visibility into your cloud assets by synchronizing your GCP cloud inventories with runZero. runZero also integrates with other cloud providers, such as [Microsoft Azure](#) and [Amazon AWS](#). Similarly to other integrations, you will need to add the [Scanning with credentials](#) needed to authenticate to GCP and set up a connector in runZero. runZero will pull in GCP compute instance VMs, pulling in GCP attributes that will be viewable from each asset.

The following GCP asset types are supported:

- Compute Engine instances
- Load balancers
- Cloud SQL

Requirements

- Verify you have a [Google Cloud service account](#) with the Compute Network Viewer and Cloud SQL Viewer [roles](#).
 - This service account will need to be granted access to each project that you want the integration to gather data from.
- [Download a key](#) for the GCP service account.
- Verify you have these [GCP APIs enabled](#) on each project:
 - Compute Engine
 - Cloud SQL Admin

How to set up the Google Cloud Platform integration

Here are the high-level steps to set up the Google Cloud Platform integration:

- [Create a Google Cloud Platform credential](#) in runZero.
- Choose whether to configure the integration as [a scan probe or connector task](#).
- [Activate the integration](#) for Google Cloud Platform.
- [View your GCP assets](#).

Step 1: Create Google Cloud Platform credentials

1. Go to the [Credentials page](#) and click **Add Credential**.
2. From the **Credentials type** dropdown, choose **GCP Service Account Key**.
3. Provide a name for the credential, like `GCP`.
4. Set the **Include all projects** toggle to *Yes* if you want runZero to gather asset data from all GCP projects that the service account has access to. If set to *No*, the integration will only gather asset data from the project specified in the key file.
5. Click **Choose file** to upload the service account key file you downloaded from GCP.
6. If you want other organizations to be able to use this credential, select the **Make this a global credential** option. Otherwise, you can configure access on a per-organization basis.
7. Save the credential. You're now ready to set up and activate the connection to bring in data from Google Cloud Platform.

Step 2: Choose how to configure the GCP integration

The GCP integration can be configured as either a [scan probe or a connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 3: Activate the Google Cloud Platform integration

After you add your GCP credential, you'll need to set up a connector task or scan probe to sync your data.

Step 3a: Configure the GCP integration as a connector task

1. [Activate a connection to GCP](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like Google Cloud Platform sync.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. To organize your assets logically, choose the site you'd like to use to add your assets to. You can choose an existing site or add them to a new site when the sync occurs. Assigning your assets to a site helps organize and group your assets. You can automatically generate a new site per GCP project by selecting this option from the task configuration.
6. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
7. Activate the connection when you are done. The sync will run on the defined schedule. You can check the [Scheduled tasks](#) to see when the next sync will occur.

Step 3b: Configure the GCP integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the GCP toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the GCP toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 4: View your Google Cloud Platform assets

After a successful sync, you can [go to your inventory](#) to view your GCP assets. These assets will have a Google icon listed in the **Source** column.

To filter by GCP assets, consider running the following queries:

- [View all GCP assets](#):

```
source:gcp
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by GCP.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Google Cloud Platform integration unable to connect?

1. Are you getting any data from the GCP integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the GCP integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Google Workspace

runZero integrates with Google Workspace to allow you to sync and enrich your asset inventory, as well as gain visibility into users and groups. Adding your Google Workspace data to runZero makes it easier to find unmanaged assets on your network. The Google Workspace integration supports ChromeOS, Mobile, and Endpoint [registered asset types](#).

Requirements

- Verify or create a new [Google service account](#) in whichever project is most suitable.
- [Create and download a key](#) for the Google service account. Save this JSON file.
- Verify that you have the Admin SDK and Cloud Identity APIs enabled for the project. Use the search box in the [API Library](#) to find each API and then enable it.
- Enable domain-wide delegation in the [Google Workspace console](#)
 - Add a new API client using the unique numeric ID of service account as the Client ID
 - Enable the following OAuth scopes for this API client:

```
https://www.googleapis.com/auth/admin.directory.user.readonly,
https://www.googleapis.com/auth/admin.directory.group.readonly,
https://www.googleapis.com/auth/admin.directory.device.mobile.readonly,
https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly,
https://www.googleapis.com/auth/cloud-identity.devices.readonly
```

- Optionally, enter each OAuth scope individually:
 - https://www.googleapis.com/auth/admin.directory.user.readonly
 - https://www.googleapis.com/auth/admin.directory.group.readonly
 - https://www.googleapis.com/auth/admin.directory.device.mobile.readonly
 - https://www.googleapis.com/auth/admin.directory.device.chromeos.readonly
 - https://www.googleapis.com/auth/cloud-identity.devices.readonly

How to set up the Google Workspace integration

These are the high-level steps to set up the Google Cloud Platform integration:

- [Create a Google Workspace credential](#) in runZero.
- Choose whether to configure the integration as a [scan probe or connector task](#).
- [Activate the connection](#) for Google Workspace.
- [View your results](#).

Step 1: Create a Google Workspace credential

1. Go to the [Credentials page](#) and click **Add Credential**.
2. From the **Credentials type** dropdown, choose **Google Workspace Client Secret**.
3. Provide a name for the credential, like *Google Workspace*.
4. In the *Admin account email* field, provide the email address of an administrator account with access to the assets, users, or groups you wish to import.
5. If you want to import from an organization other than the one your administrator account belongs to, provide a *Customer ID*. By default, runZero will use the Customer ID associated with the service account. (Optional)
6. Click **Choose file** to upload the service account key file you downloaded from Google Workspace.
7. If you want other organizations to be able to use this credential, select the **Make this a global credential** option. Otherwise, you can configure access on a per-organization basis.
8. Save the credential. You're now ready to set up and activate the connection to bring in data from Google Workspace.

Step 2: Choose how to configure the Google Workspace integration

The Google Workspace integration can be configured as either a [scan probe or a connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. Configuring the integration as a scan probe is useful if you are running self-hosted runZero Platform and your console cannot access Google Workspace. For most situations it will be easier to set up a scheduled connection to sync your data from Google Workspace.

Step 3: Activate the Google Workspace integration

After you add your GCP credential, you'll need to set up a connector task or scan probe to sync your data.

Step 3a: Configure the Google Workspace integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where the data is organized.

1. [Activate a connection to Google Workspace](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like `Google Workspace sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. To organize your assets logically, choose the site you'd like to use to add your assets to. You can choose an existing site or add them to a new site when the sync occurs. Assigning your assets to a site helps organize and group your assets.
6. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
7. Activate the connection when you are done. The sync will run on the defined schedule. You can check the [Scheduled tasks](#) to see when the next sync will occur.

Step 3b: Configure the Google Workspace integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the `GoogleWorkspace` toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the `GoogleWorkspace` toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 4: View Google Workspace assets

After a successful sync, you can [go to your inventory](#) to view your Google Workspace assets. These assets will have a Google Workspace icon listed in the **Source** column.

To filter by Google Workspace assets, consider running the following queries:

- [View all Google Workspace assets:](#)

```
source:googleworkspace
```

- [View runZero assets not connected to Google Workspace:](#)

```
source:runzero AND NOT source:googleworkspace
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by Google Workspace.



The Google Workspace integration provides details about users and groups in addition to enriching asset inventory data. Go to [Inventory > Users](#) or [Inventory > Groups](#) to view the data provided by Google Workspace. Use the query `source:googleworkspace` to filter your results.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Google Workspace integration unable to connect?

1. Are you getting any data from the Google Workspace integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Google Workspace integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Microsoft 365 Defender

runZero integrates with Microsoft 365 Defender to allow you to sync and enrich your asset, software, and vulnerability inventory. Adding your Microsoft 365 Defender data to runZero makes it easier to find assets missing EDR protection.

Getting started

To set up the Microsoft 365 Defender integration, you'll need to:

1. Configure Microsoft 365 Defender to allow API access through runZero.
2. Add the Microsoft 365 Defender credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the Microsoft 365 Defender integration to sync your data with runZero.

Requirements

Before you can set up the Microsoft 365 Defender integration:

- Make sure you have access to the Microsoft Azure portal.

Step 1: Register an Azure application for Microsoft 365 Defender API access

runZero can authenticate to the Microsoft 365 Defender API using a client secret. Register an application to configure Microsoft 365 Defender API access.

1. Sign in to the Microsoft Azure portal.
2. Go to *App registrations* and click on *+ New registration*.
 - Provide a name.
 - Select the supported account types.
 - Optionally add a redirect URI.
3. Click *Register* to register the application.
4. Once the application is created, you should see the Overview dashboard. Note the following information:
 - Application (client) ID
 - Directory (tenant) ID
5. From the application's details page, go to *Manage > API permissions* and choose *+ Add a permission*.
6. Select the second tab called *APIs my organization uses* to view available APIs.
7. Select `WindowsDefenderATP` from the list of Microsoft APIs.
8. Select the permissions type `Application permissions` to configure a *client secret*.
9. Search for and select the following required permissions:
 - Windows Defender ATP API permissions:
 - `Machine.Read.All`
 - `Software.Read.All`
 - `Vulnerability.Read.All`
10. Click *Add permissions* to save the permissions to the application.
11. Click *Grant admin consent* to grant consent for the permissions to the application.
- 12.

To generate the client secret, perform the following steps from the app management pages:

- Navigate to *App registrations* and select the application you created.
- Go to *Certificates & secrets* and click on *+ New client secret*.
 - Enter a description.
 - Select the expiration.
- Click *Add* to create the client secret and save the client secret value.

Step 2: Add an Azure Client Secret credential to runZero

This type of credential can be used to sync all resources in a single directory (across multiple subscriptions).

1. Go to the [Credentials page](#) in runZero and click *Add Credential*.
2. Provide a name for the credential, like *Azure Client Secret*.
3. Choose *Azure Client Secret* from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure client secret** - To generate a client secret, go to *App registrations*, select your application, go to *Manage > Certificates & secrets* and click on *New client secret*.
 - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to *App registrations* and select the application.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 3: Choose how to configure the Microsoft 365 Defender integration

The Microsoft 365 Defender integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the Microsoft 365 Defender integration to sync data

After you add your Microsoft 365 Defender credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the Microsoft 365 Defender integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Microsoft 365 Defender-only assets are created.

1. Activate a [connection to Microsoft 365 Defender](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure it has access to the organization you are currently in.
3. Enter a name for the task, like `Microsoft 365 Defender sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to.
6. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
7. If you want to include assets that have stopped reporting to the Microsoft 365 Defender service in your integration import, switch the **Include inactive assets** toggle to *Yes*. By default, the integration will not include assets that are marked as inactive.
8. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Microsoft 365 Defender integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the Defender365 toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the MS365Defender toggle for the credential you wish to use to *Yes*.

- Click *Initialize scan* to save the scan task and have it run immediately or at the scheduled time.

Step 5: View Microsoft 365 Defender assets

After a successful sync, you can [go to your inventory](#) to view your Microsoft 365 Defender assets. These assets will have an Active Directory icon listed in the **Source** column.

To filter by Microsoft 365 Defender assets, consider running the following queries:

- View all Microsoft 365 Defender assets:

```
source:ms365defender
```

- View runZero assets not connected to Microsoft 365 Defender:

```
source:runzero AND NOT source:ms365defender
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by Microsoft 365 Defender.

Filtering Microsoft 365 Defender assets

An optional filter can be applied to Microsoft 365 Defender integration tasks. runZero uses Microsoft Graph \$filter query parameter to filter assets. GraphQL follows the syntax `<property> [operator] <value>`. Multiple expressions can be combined for more complex filtering by adding `and` or `or` between expressions.

Properties

Any property that runZero imports from Microsoft 365 Defender can be used to apply a filter. The following are some examples.

Defender Property	runZero Attribute	Description	Example
computerDNSName	@ms365defender.dev.computerDNSName	The hostname of the device	EXPLORER-01
osPlatform	@ms365defender.dev.osPlatform	The operation system of the device	Windows11, Android
healthStatus	@ms365defender.dev.healthStatus	The health status of the Defender agent that is installed	Active, Inactive

isAADJoined	@ms365defender.dev.isAADJoined	Boolean value specifying whether the device is registered in Azure AD	true, false
managedBy	@ms365defender.dev.managedBy	Description of how the device is managed	Intune

Operators

The following are common operators that can be used in a Microsoft 365 Defender filter.

- Equal to (eq)
- Not equal to (ne)
- Has (has)
- Less than (lt)
- Greather than (gt)
- Less than or equal to (le)
- Greater than or equal to (ge)

The following are common functions that can be used in a Microsoft 365 Defender filter. Functions follow the syntax `function(<property>, <value>)`.

- Starts with (startswith)
- Ends with (endswith)
- Contains (contains)

Example Filters

The following are examples of filters that can be applied to a Microsoft 365 Defender integration.

Search Filter	Description
<code>not(osPlatform eq 'Android')</code>	Import all assets except those with an Android operating system
<code>not(osPlatform eq 'Android') and not(osPlatform eq 'iOS')</code>	Import all assets except those with an Android or iOS operating system
<code>startswith(computerDNSname, 'PROD')</code>	Import all devices with a hostname that starts with PROD
<code>not(startswith(computerDNSname, 'DEV'))</code>	Import all devices except those with a hostname that starts with DEV

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Microsoft 365 Defender integration unable to connect?

1. Are you getting any data from the Microsoft 365 Defender integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Microsoft 365 Defender integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Microsoft Active Directory

runZero integrates with Microsoft Active Directory (AD) via LDAP to allow you to sync and enrich your asset inventory, as well as gain visibility into domain users and groups. Adding your AD data to runZero makes it easier to find assets that are not part of your domain.

Getting started

To set up the Active Directory integration, you'll need to:

1. Add the AD credential in runZero.
2. Choose whether to configure the integration as [a scan probe or connector task](#).
3. Activate the Active Directory integration to sync your data with runZero.

Requirements

Before you can set up the LDAP integration, make sure you have credentials for an LDAP account.

Step 1: Add the LDAP credential to runZero

1. Go to the [Add credential page](#) in runZero. Provide a name for the credentials, like LDAP.
2. Choose **LDAP Username & Password** from the list of credential types.
3. Provide the following information:
 - **LDAP username** - The username you want to use with the LDAP integration. The account used for this integration does not require any special permissions. The

following username formats are accepted:

- Distinguished Name (DN): `CN=[username],CN=Users,DC=[domain],DC=[tld]`
- User Principle Name (UPN): `[username]@[domain].[tld]`
- Domain\Username: `[domain]\[username]`
- **LDAP password** - The password for the username to be used with the LDAP integration.
- **LDAP base DN** - The base distinguished name for LDAP searches. This field requires distinguished name format: `DC=[domain],DC=[tld]`. Note that only entities underneath the specified base DN will be imported into runZero.
- **LDAP URL** - The URL for your LDAP server. This field supports `IP[:port]` notation as well as `hostname.domain.tld[:port]`. This field requires that the URL entered begins with `ldap://` (for insecure LDAP connections) or `ldaps://` (for secure LDAP connections). For example: `ldaps://ad.example.com:636`
- **LDAP insecure** - Set this to *Yes* if you want to attempt authentication without a verified thumbprint. By default, runZero will attempt to connect with LDAPS but will fall back to LDAP+StartTLS then LDAP. LDAP without StartTLS will only work if this toggle is set to *Yes*.
- **LDAP thumbprints** (optional) - A set of `IP[:port]=SHA256:B64HASH` or `hostname.domain.tld=SHA256:B64HASH` pairs to trust for authentication.
 - You will need to scan your LDAP server with runZero in order to obtain the TLS thumbprint. The [TLS fingerprints service attribute report](#) lists all previously seen fingerprints. The TLS thumbprints used for self-signed certificates will only work with LDAPS. If you want to use LDAP+StartTLS with a self-signed certificate, you will need to set the **Insecure** option to *Yes*.
 - If *LDAP insecure* is set to *No* and no thumbprints are provided:
 - With a self-signed certificate, the connection will fail because the certificate chain cannot be verified.
 - With a valid certificate from a public CA, the connection will work without thumbprints.
- 4. If you want all other organizations to be able to use this credential, select the **Make this a global credential** option. Otherwise, you can configure access on a per-organization basis.
- 5. Save the credential. You're now ready to set up and activate the connection to bring in data from LDAP.

Step 2: Choose how to configure the Active Directory integration

The Active Directory integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 3: Set up and activate the Active Directory integration to sync data

After you add your Active Directory credential, you'll need to set up a connector task or scan probe to sync your data.

Step 3a: Configure the Active Directory integration as a connector task

After you add your LDAP credential, you'll need to set up a connection to sync your data from LDAP. A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new LDAP-only assets are created.

1. Activate a [connection to Active Directory](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Enter a name for the task, like `LDAP sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to.
6. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
7. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 3b: Configure the Active Directory integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the LDAP toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the LDAP toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 4: View Active Directory assets

After a successful sync, you can [go to your inventory](#) to view your LDAP assets. These assets will have an Active Directory icon listed in the **Source** column.

To filter by LDAP assets, consider running the following queries:

- [View all LDAP assets:](#)

```
source:ldap
```

- [View runZero assets not connected to LDAP:](#)

```
source:runzero AND NOT source:ldap
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by LDAP.



The LDAP integration provides details about users and groups in addition to enriching asset inventory data. Go to [Inventory > Users](#) or [Inventory > Groups](#) to view the data provided by LDAP.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Microsoft Active Directory integration unable to connect?

1. Are you getting any data from the Microsoft Active Directory integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Microsoft Active Directory integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Microsoft Active Directory host.

Microsoft Azure

runZero integrates with Microsoft Azure to deliver greater visibility into your cloud assets. This integration imports data through each applicable API to enrich your asset inventory:

- [Virtual Machines API](#)
- [Virtual Machine Scale Sets API](#)
- [Load Balancers API](#)
- [AzureSQL API](#)
- [Web Apps API](#)

Syncing with Azure allows you to view information about your asset's OS profile, storage profile, and more. This integration imports assets that are in a running state.

Getting started

The following Azure resource types are supported:

- Virtual Machines
- Virtual Machine Scale Sets
- Azure SQL
- Azure Cosmos DB
- Load Balancers
- Function Apps

To set up the Azure integration, you'll need to:

1. Configure Azure to allow API access through runZero.
2. Add an Azure credential to runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the Azure integration to sync your data with runZero.

Requirements

Before you can set up the Azure integration, make sure you have access to the Microsoft Azure portal.

Step 1: Configure Azure to allow API access through runZero

1. Sign in to the Microsoft Azure portal.
2. Go to **Azure Active Directory > App registrations** and click on **New registration**.
 - Provide a name.
 - Select the supported account types.
 - Optionally add a redirect URI.
3. Click **Register** to register the application.
4. Once the application is created, you should see the **Overview** dashboard. Save the following information:
 - Application (client) ID
 - Directory (tenant) ID
5. Give the client access to the subscriptions you want to sync. From the subscription details page, go to **Access Control (IAM)** and select **Add > Add role assignment**. Enter the following:
 - Role: Reader
 - Assign access to: User, group, or service principal
 - Under **Select**, search for the name of the application you created. Click on your application to add it to the **Selected members** list below.
6. Click **Save** to save the role assignment.
7. Navigate to **Azure Active Directory > App registrations** and select the application you created.
8. Go to **Certificates & secrets** and click on **New client secret**.
 - Enter a description.
 - Select the expiration.
9. Click **Add** to create the client secret. Save the following information:
 - Client secret value

Step 2: Add the Azure credential to runZero

The credential used for the Azure integration can be either a client secret or a username & password.

Step 2a: Add an Azure Client Secret credential to runZero

This type of credential can be used to sync all resources in a single directory (across multiple subscriptions).

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Provide a name for the credential, like `Azure Client Secret`.
3. Choose **Azure Client Secret** from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to **Azure Active Directory > App registrations** and select the application.
 - **Azure client secret** - To generate a client secret, go to **Azure Active Directory > App registrations**, select your application, go to **Certificates & secrets** and click on **New client secret**.
 - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to **Azure Active Directory > App registrations** and select the application.
 - Enable the **All subscriptions** option to sync all resources in your directory. If enabled, you can add subscription IDs to the **Exclude subscription IDs** text area to exclude specific IDs. If disabled, you can add specific subscription IDs by entering them in the **Include subscription IDs** text area. The subscription ID can be found in the Azure portal if you go to **Subscriptions** and select the subscription. For both of these text areas, please enter one subscription ID per line.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per organization basis.
6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 2b: Add an Azure Username & Password credential to runZero

This type of credential can be used to sync all resources across directories. Alternatively, you can add one **Azure Client Secret** credential for each Azure directory you want to sync.

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Provide a name for the credential, like `Azure User/Pass`.
3. Choose **Azure Username & Password** from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to **Azure Active Directory > App registrations** and select the application.

- **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to **Azure Active Directory > App registrations** and select the application.
 - **Azure username** - The username for your Azure cloud account. This cannot be a federated user account.
 - **Azure password** - The password for your Azure cloud account.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per organization basis.
 6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 3: Choose how to configure the Azure integration

The Azure integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the Azure integration to sync data

After you add your Azure credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the Azure integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Azure-only assets are created.

1. Activate a [connection to Azure](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like *Azure sync*.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. Under **Service options**, select the services you would like to sync data from. You must choose at least one.
7. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
8. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Azure integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).

2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the Azure toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the Azure toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 5: View Azure assets

After a successful sync, you can [go to your inventory](#) to view your Azure assets. These assets will have an Azure icon listed in the **Source** column.

To filter Azure assets, consider running the following queries:

- [View all Azure assets:](#)

```
source:azure
```

- [View all Azure VMs](#)

```
has:"@azure.vm.vmID"
```

- [View all Azure virtual machine scale set VMs](#)

```
has:"@azure.vms.vmsID"
```

- [View all Azure load balancers](#)

```
has:"@azure.lb.id"
```

- [View all AzureSQL instances](#)

```
has:"@azure.azsql.id"
```

- [View all Azure Cosmos DB instances](#)

```
has:"@azure.cosmos.id"
```

- [View all Azure Function Apps](#)

```
@azure.functionapp.kind:"functionapp"
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by the Azure APIs.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Microsoft Azure integration unable to connect?

1. Are you getting any data from the Microsoft Azure integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Microsoft Azure integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Microsoft Endpoint Configuration Manager (MECM)

runZero integrates with Microsoft Endpoint Configuration Manager (MECM), formerly System Center Configuration Manager (SCCM), by importing data from the MECM MSSQL database. This integration allows you to sync data about your devices from MECM, making it easier to find unmanaged devices in your network.

Getting started with MECM

To set up an integration with MECM, you'll need to:

1. Identify or create a database user with read access to the MECM database.
2. Configure the MECM credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Step 1: Identify or create a database user for access to MECM

1. Identify an existing database user with read access to the database.
2. Alternatively, create a dedicated read-only database user for this integration. More details on creating a new database user can be found in Microsoft's documentation - [Create a database user](#).

Step 2: Add the MECM database connection string to runZero

1. Go to the [Credentials](#) page in runZero.
2. Choose **MECM Database Connection String** from the list of credential types.
3. Provide a name for the credential, like MECM.
- 4.

4. Provide the database connection string, using one of the following formats:
 - `Server=host,port;Database=database-name;User Id=user-id;Password=password;`
When using this format, the values should not contain a semicolon (;). You can use single or double quotes to escape any special characters.
 - `sqlserver://username:password@host/instance?database=value&param=value`
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Save the credential.

You're now ready to set up and activate the connection to bring in data from MECM.

Step 3: Choose how to configure the MECM integration

The MECM integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your MECM credential, you'll need to sync your data from MECM.

Step 4a: Configure the MECM integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new assets are created.

1. Activate a connection to [MECM](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Enter a name for the task, like MECM Sync (optional).
4. Choose the Explorer to perform this connector task from (optional).
5. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. Enter a description for the task (optional).
7. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
8. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
9. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the MECM integration as a scan probe

You can run the MECM integration as a scan probe so that the runZero Explorer will pull your MECM devices into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *MECM* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct *MECM* credential to *Yes* in the *Credentials* tab.

Step 5: View MECM assets

After a successful sync, you can [go to your inventory](#) to view your MECM assets. These assets will have a Microsoft icon listed in the **Source** column.

To filter by MECM assets, consider running the following queries:

- [View all MECM assets](#):

```
source:mecm
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from MECM.

Microsoft Entra ID

runZero integrates with Microsoft Entra ID (formerly Azure AD) to allow you to sync and enrich your asset inventory, as well as gain visibility into Entra ID users and groups. Adding your Entra ID data to runZero makes it easier to find assets that are not part of your domain.

Note that Entra ID is still referred to as Azure AD within the runZero product.

Getting started

To set up the Entra ID integration, you'll need to:

1. Configure Entra ID to allow API access through runZero.
2. Add the Entra ID credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the Entra ID integration to sync your data with runZero.

Requirements

Before you can set up the Entra ID integration, make sure you have access to the Microsoft Azure portal.

Step 1: Register an Azure application for Entra ID API access

runZero can authenticate to the Entra ID API using either a username and password or a client secret. Register an application to configure Entra ID API access.

1. Sign in to the Microsoft Azure portal.
2. Go to *App registrations* and click *+ New registration*.
 - Provide a name.
 - Select the supported account types.
 - Optionally add a redirect URI.
3. Click *Register* to register the application.
4. Once the application is created, go back to the main Azure portal page and select *App registrations*. You should be able to find the application you just registered. (It may help to select the *Owned applications* tab.) Click on the app's name
5. You should see the an overview of the app registration. Note the following information:
 - Application (client) ID
 - Directory (tenant) ID
6. Click on the display name of the application to go to its management pages.
7. Select the *Manage > Authentication* section on the left. Set **Allow public client flows** to *Yes* and then save the configuration.
8. Go to *API permissions* and click *+ Add a permission*.
9. Select *Microsoft Graph* from the list of Microsoft APIs.
10. Select the correct permissions type for your needs:
 - **Username & password:** select *Delegated permissions*
 - **Client secret:** select *Application permissions*
11. Search for and select the following required permissions:
 - *Device.Read.All*
 - *Group.Read.All*
 - *User.Read.All*
12. Click *Add permissions* to save the permissions to the application.
13. Click *Grant admin consent* to grant consent for the permissions to the application.
14. If using a client secret, also perform the following steps:
 - Navigate to *Azure Active Directory > App registrations* and select the application you created.

- Go to *Certificates & secrets* and click on *New client secret*.
 - Enter a description.
 - Select the expiration.
- Click *Add* to create the client secret and save the client secret value.

Add the Entra ID credential to runZero

Step 2a: Add an Azure Username & Password credential to runZero

1. Go to the [Credentials page](#) in runZero and click *Add Credential*.
2. Provide a name for the credential, like *Azure Username & Password*.
3. Choose *Azure Username & Password* from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure username** - The username for your Azure cloud account. This cannot be a federated user account.
 - **Azure password** - The password for your Azure cloud account.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per organization basis.
6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 2b: Add an Azure Client Secret credential to runZero

This type of credential can be used to sync all resources in a single directory (across multiple subscriptions).

1. Go to the [Credentials page](#) in runZero and click *Add Credential*.
2. Provide a name for the credential, like *Azure Client Secret*.
3. Choose *Azure Client Secret* from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure client secret** - To generate a client secret, go to *App registrations*, select your application, go to *Certificates & secrets* and click on *+ New client secret*.

- **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - Select the **Access all subscriptions in this directory (tenant)** option to sync all resources in your directory. Otherwise, specify the Azure subscription ID - The unique ID for the subscription that you want to sync. This can be found in the Azure portal if you go to *Subscriptions* and select the subscription.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per organization basis.
 6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 3: Choose how to configure the Entra ID integration

The Entra ID integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the Entra ID integration to sync data

After you add your Entra ID credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the Entra ID integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Entra ID-only assets are created.

1. Activate a [connection to Entra ID](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure it has access to the organization you are currently in.
3. Optionally provide a filter following the [Microsoft Graph API filter syntax](#). We will only import devices that match the filter.
4. Enter a name for the task, like *Entra ID sync*.
5. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
6. Under **Task configuration**, choose the site you want to add your assets to.
7. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.

8. If you want to include assets in your integration import that the Entra ID account has marked as inactive, switch the **Include inactive assets** toggle to *Yes*. By default, the integration will not include assets that are marked as inactive.
9. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Entra ID integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the **Azure AD** toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the **Azure AD** toggle for the credential you wish to use to *Yes*.
5. Click *Initialize scan* to save the scan task and have it run immediately or at the scheduled time.

Step 5: View Entra ID assets

After a successful sync, you can [go to your inventory](#) to view your Entra ID assets. These assets will have an Active Directory icon listed in the *Source* column.

To filter by Entra ID assets, consider running the following queries:

- [View all Entra ID assets:](#)

```
source:azuread
```

- [View runZero assets not connected to Entra ID:](#)

```
source:runzero AND NOT source:azuread
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by Entra ID.



The Entra ID integration provides details about users and groups in addition to enriching asset inventory data. Go to [Inventory > Users](#) or [Inventory > Groups](#) to view the data provided by Entra ID.

Filtering Entra ID assets

An optional filter can be applied to Entra ID integration tasks. runZero uses Microsoft Graph \$filter query parameter to filter assets. GraphQL follows the syntax <property> [operator] <value>. Multiple expressions can be combined for more complex filtering by adding an and or between expressions.

Properties

Any property that runZero imports from Entra ID can be used to apply a filter. The following are some examples.

Entra ID Property	runZero Attribute	Description	Example
displayName	@azuread.dev.displayName	The hostname of the device	EXPLORER-01
operatingSystem	@azuread.dev.operatingSystem	The operation system of the device	Windows
operatingSystemVersion	@azuread.dev.operatingSystemVersion	The version of the specified operating system	10.0.x
manufacturer	@azuread.dev.manufacturer	The manufacturer of the device	Dell Inc.
model	@azuread.dev.model	The model of the device	Precision 3560
isManaged	@azuread.dev.isManaged	Boolean value specifying whether device is managed	true, false
managementType	@azuread.dev.managementType	Description of how the device is managed	MDM, MicrosoftSense
deviceOwnership	@azuread.dev.deviceOwnership	Description of who owns the device	Company, Personal

Operators

The following are common operators that can be used in an Entra ID filter.

- Equal to (eq)
- Not equal to (ne)
- Has (has)
- Less than (lt)
- Greater than (gt)
- Less than or equal to (le)
- Greater than or equal to (ge)

The following are common functions that can be used in an Entra ID filter. Functions follow the syntax `function(<property>, <value>)`.

- Starts with (startswith)
- Ends with (endswith)

Example Filters

The following are examples of filters that can be applied to an Entra ID integration.

Search Filter	Description
<code>not(startsWith(operatingSystem, 'Android'))</code>	Import all assets except those with an Android operating system
<code>not(operatingSystem eq 'iOS') and not(operatingSystem eq 'iPad')</code>	Import all assets except those with an iOS or iPad operating system
<code>startswith(displayName, 'PROD')</code>	Import all devices with a hostname that starts with PROD
<code>not(startswith(displayName, 'DEV'))</code>	Import all devices except those with a hostname that starts with DEV
<code>deviceOwnership eq 'Company' or isManaged eq true</code>	Import all devices that are owned by company or that are configured as managed devices

For more information about filter syntax and additional examples, check [Microsoft's Graph API documentation](#).

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Azure Active Directory integration unable to connect?

- 1.

Are you getting any data from the Entra ID integration?

- Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
 3. If the Entra ID integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

How do I solve the following Entra ID errors?

- (invalid_client) AADSTS7000218: The request body must contain the following parameter: 'client_assertion' or 'client_secret'
 1. This error can be corrected by enabling **Allow Public Client Flows** in Entra ID. This can be accomplished by entering the Application details page under *Authentication > Advanced Settings*. From here you can toggle the **Allow Public Client Flows** setting to *Yes*.
 2. It can also be helpful to ensure that application permissions were granted correctly when registering the Azure application for Entra ID API access. To do this, navigate to the API permissions settings page and ensure that each of the API/Permissions have the *type* set to *application*. Also make sure that the permission granted is *Grant Admin Consent for Default Directory*.
- failed to get Entra ID groups: invalid response: 403 (403 Forbidden)

This error is likely due to an issue with credentials. Please review the documentation and check your credentials to ensure everything was entered correctly and no steps were accidentally skipped.

Microsoft Intune

runZero integrates with Microsoft Intune to allow you to sync and enrich your asset inventory. Adding your Microsoft Intune data to runZero makes it easier to find unmanaged assets on your network. Data added includes the [discovered apps](#) from Intune. Managed apps (those pushed to devices by Intune) are not currently reported.

Getting started

To set up the Microsoft Intune integration, you'll need to:

1. Configure Microsoft Intune to allow API access from runZero.
2. Add the Microsoft Intune credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the Microsoft Intune integration to sync your data with runZero.

Requirements

Before you can set up the Microsoft Intune integration:

- Make sure you have access to the Microsoft Azure portal.

Step 1: Register an Azure application for Microsoft Intune API access

runZero can authenticate to the Microsoft Intune API using either a username and password or a client secret. Register an application to configure Microsoft Intune API access.

1. Sign in to the Microsoft Azure portal.
2. Go to *App registrations* and click on *+ New registration*.
 - Provide a name.
 - Select the supported account types.
 - Optionally add a redirect URI.
3. Click *Register* to register the application.
4. Once the application is created, go back to the main Azure portal page and select *App registrations*. You should be able to find the application you just registered. (It may help to select the *Owned applications* tab.) Click on the app's name
5. You should see an overview of the app registration. Note the following information:
 - Application (client) ID
 - Directory (tenant) ID
6. From the application's details page, go to *Manage > API permissions* and choose *+ Add a permission*.
7. Select *Microsoft Graph* from the list of Microsoft APIs.
- 8.

Select the correct permissions type for your needs:

- **Username & password:** select *Delegated permissions*
- **Client secret:** select *Application permissions*

9. Search for and select the following required permission:

- DeviceManagementManagedDevices.Read.All
- User.Read.All
- DeviceLocalCredential.ReadBasic.All (only required if you choose to enable the Include LAPS Information option within the integration)

10. Click *Add permissions* to save the permissions to the application.

11. Click *Grant admin consent* to grant consent for the permissions to the application.

12. If using a client secret, also perform the following steps from the app management pages:

- Navigate to *App registrations* and select the application you created.
- Go to *Certificates & secrets* and click on *+ New client secret*.
 - Enter a description.
 - Select the expiration.
- Click *Add* to create the client secret and save the client secret value.

Step 2: Add the Microsoft Intune credential to runZero

Adding the Microsoft Intune credential requires adding an Azure username and password or an Azure Client Secret to runZero. The following sub-steps break down each task.

Step 2a: Add an Azure Username & Password credential to runZero

1. Go to the [Credentials](#) page in runZero and click *Add Credential*.
2. Provide a name for the credential, like *Azure User/Pass*.
3. Choose *Azure Username & Password* from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure username** - The username for your Azure cloud account. This cannot be a federated user account.
 - **Azure password** - The password for your Azure cloud account.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization

basis.

6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 2b: Add an Azure Client Secret credential to runZero

This type of credential can be used to sync all resources in a single directory (across multiple subscriptions).

1. Go to the [Credentials page](#) in runZero and click *Add Credential*.
2. Provide a name for the credential, like *Azure Client Secret*.
3. Choose *Azure Client Secret* from the list of credential types.
4. Provide the following information:
 - **Azure application (client) ID** - The unique ID for the registered application. This can be found in the Azure portal if you go to *App registrations* and select the application.
 - **Azure client secret** - To generate a client secret, go to *App registrations*, select your application, go to *Manage > Certificates & secrets* and click on *New client secret*.
 - **Azure directory (tenant) ID** - The unique ID for the tenant. This can be found in the Azure portal if you go to *App registrations* and select the application.
5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Save the credential. You're now ready to set up and activate the connection to bring in data from Azure.

Step 3: Choose how to configure the Microsoft Intune integration

The Microsoft Intune integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independent of a scan, from either the cloud or one of your Explorers, and only performing the integration sync.

Step 4: Set up and activate the Microsoft Intune integration to sync data

After you add your Microsoft Intune credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the Microsoft Intune integration as a connector task

A connection requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Microsoft Intune-only assets are created.

1. Activate a [connection to Microsoft Intune](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure it has access to the organization you are currently in.
3. Optionally provide a filter to import only devices that match it. You can find more details in the [Filtering Intune assets](#) section below.
4. If you wish to retrieve LAPS information from Intune, enable the Include LAPS Information option. This functionality also has additional permission requirements; please see the [Intune API Access](#) section above. Please note that **runZero does not collect passwords from LAPS when using this option**.
5. Enter a name for the task, like `Microsoft Intune sync`.
6. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
7. Under **Task configuration**, choose the site you want to add your assets to.
8. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
9. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Microsoft Intune integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the Intune toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the Intune toggle for the credential you wish to use to *Yes*.
5. Click *Initialize scan* to save the scan task and have it run immediately or at the scheduled time.

Step 5: View Microsoft Intune assets

After a successful sync, you can [go to your inventory](#) to view your Microsoft Intune assets. These assets will have an Active Directory icon listed in the *Source* column.

To filter by Microsoft Intune assets, consider running the following queries:

- [View all Microsoft Intune assets:](#)

```
source:intune
```

- [View runZero assets not connected to Microsoft Intune:](#)

```
source:runzero AND NOT source:intune
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by Microsoft Intune.

Filtering Intune assets

An optional filter can be applied to Intune integration tasks. runZero uses the Intune reports API to retrieve assets from the `DevicesWithInventory` report. This API accepts a filter field to narrow down the results.

Properties

The `DevicesWithInventory` report supports filtering based on the following properties:

- CreatedDate
- LastContact
- CategoryName
- CompliantState
- ManagementAgents
- OwnerType
- ManagementState
- DeviceType
- JailBroken
- EnrollmentType
- PartnerFeaturesBitmask

For the most up-to-date list of properties, please refer to Microsoft's documentation on the [DevicesWithInventory](#) report. Note that not all the properties listed on that table are filterable. Scroll just below the table to find a separate list of properties that can be used in filters.

Operators

There is no official documentation on the filter syntax or supported operators for the Intune reports API, but based on our testing, the following operators may be supported. In practice,

we've found that some of these, such as `not` and `in`, can produce errors, so your results may vary.

- Equal to (`eq`)
- Not equal to (`ne`)
- Logical negation (`not`)
- In (`in`)
- Has (`has`)
- Less than (`lt`)
- Greater than (`gt`)
- Less than or equal to (`le`)
- Greater than or equal to (`ge`)

The syntax appears to be similar to the `$filter` query parameter used elsewhere in Microsoft Graph. Examples for this syntax can be found in [Use the \\$filter query parameter](#). We recommend testing filters carefully and starting with basic operators like `eq` and `ne` for best results.

Example Filters

The following are examples of filters that can be applied to an Intune integration.

Search Filter	Description
<code>DeviceType eq 'android'</code>	Import all android devices
<code>LastContact ge '2023-02-23 23:50:01.0000000'</code>	Import devices that checked in after a specific date and time

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Microsoft Intune integration unable to connect?

1. Are you getting any data from the Microsoft Intune integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Microsoft Intune integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

How do I solve the following Microsoft Intune error:

- (invalid_client) AADSTS7000218: The request body must contain the following parameter: 'client_assertion' or 'client_secret'

This error means that you need to enable Public Client Flows in Azure. To do so, follow these steps:

1. Navigate to the App Registration page in the Azure portal
2. Choose *Authentication* from the left navigation
3. Select *Advanced Settings*
4. Toggle the *Allow Public Client Flows* switch at the bottom of the page to *Yes*

Miradore MDM

runZero integrates with Miradore mobile device management (MDM) to deliver greater visibility into your mobile assets. This integration imports data from the [Miradore API](#) to enrich your asset inventory. Syncing with Miradore allows you to view information about device hardware, OS version, associated user, and more. This integration imports all enrolled devices.

Getting started

To set up the Miradore integration, you'll need to:

1. Sign in to your Miradore web portal and create a new API key.
2. Add the Miradore credential to runZero, which includes the endpoint hostname and API key.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the Miradore integration to sync your data with runZero.

Requirements

Before you can set up the Miradore integration:

- Make sure you have access to the Miradore MDM portal as an administrator.

Step 1: Create a Miradore API key

1. Sign in to your Miradore portal as an administrator.
2. Click on the **System > Infrastructure diagram** link from the left-side navigation.
3. Scroll down to find the **API** node in the diagram. Click this and choose **Create API key**.
4. Give this new key a name and copy the secret value.

Step 2: Add the Miradore credential to runZero

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Provide a name for the credential, like `Miradore MDM`.
3. Choose **Miradore MDM API Key** from the list of credential types.
4. Provide the following information:
 - **Name** - Give this credential a unique name (ex: `Miradore`)
 - **Miradore endpoint hostname** - The URL for your Miradore portal.
 - **Miradore API key** - The API key created in step 1.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Miradore.

Step 3: Choose how to configure the Miradore integration

The Miradore integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the Miradore MDM integration to sync data

After you add your Miradore credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the Miradore integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Miradore-only assets are created.

1. Activate a [connection to Miradore](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like `Miradore sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
6. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
7. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Miradore integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the Miradore toggle to **Yes**, and change any of the default options if needed.
4. On the Credentials tab, set the Miradore toggle for the credential you wish to use to **Yes**.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 5: View Miradore assets

After a successful sync, you can [go to your inventory](#) to view your Miradore assets. These assets will have a Miradore icon listed in the **Source** column.

To filter by Miradore assets, consider running the following queries:

- [View all Miradore assets](#):

```
source:miradore
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by the Miradore API.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Miradore integration unable to connect?

1. Are you getting any data from the Miradore integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Miradore integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Miradore host if it is on-premises.

NetBox CMDB

runZero integrates with NetBox configuration management database (CMDB) using the [NetBox REST API](#) to enrich your asset inventory.

NetBox limitations

runZero explicitly supports NetBox version 4. Older versions (3.x) may be compatible, but are not specifically tested or supported.

Since NetBox data entry is free-form, the runZero integration may not be a good fit for your organization, and we strongly recommend testing this integration in a Project first before configuring it for a production Organization.

The NetBox integration will import Devices and Virtual Machines, as well as their associated resources.

The NetBox integration will NOT create entities for anything other than Devices and Virtual Machines.

The Device and Virtual Machine records are imported as runZero assets with the following fields:

- Device or Virtual Machine
 - ID in the form of `device.<ID>` or `vm.<ID>`
 - Name
 - FirstSeenTS and LastSeenTS
 - Serial
 - AssetTag (Device only)
 - Interfaces
 - IP addresses for PrimaryIP / PrimaryIP4 / Primary IP6 / OOB IP
 - Additional IP addresses using the `ip<index>.field` syntax
 - NAT Inside and NAT Outside addresses
 - MAC Addresses (if entered in NetBox)
 - CreatedTS and UpdatedTS
 - ID, DNSName, Name, Comments, and Description
 - Module Name, MTU, Speed (Device only)
 - Tagged and Untagged VLANs
 - Custom Fields
 - Prefixes
 - VLAN and CIDR
 - ID, Comments, and Description
 - Role Name
 - Custom Fields
 - Location
 - ID and Name
 - Latitude and Longitude
 - Chained Location Path

- Custom Fields
- Rack
 - ID and Name
 - Position and Rack Face
- Status
- Manufacturer
 - ID, Name, and Model
 - Comments, Description, and Part Number
 - Custom Fields
- Device Role
 - ID, Name, Description
 - Custom Fields
- Platform
 - ID & Name
 - Custom Fields
- Site
 - ID, Name, Description, Comments
 - Facility, Physical Address, Shipping Address
 - Region, Group
 - Custom Fields
- Virtual Chassis
 - ID, Name, Comments, and Description
 - Master ID and Master Name
 - VC Position and Priority
- Interface Count
- Inventory Item Count
- Custom Fields

The following top-level attributes are rollups across all sub-fields:

- Addrs
- Names
- MACs
- VLANs

NetBox asset matching

runZero will use the Name, IP Address, and MAC Addresses to match NetBox Devices and Virtual Machines to runZero assets. If there isn't a direct match, runZero will create a new asset record, unless the "Exclude assets that cannot be merged into an existing asset" option is enabled. Since NetBox data is typically static and network assets tend to change over time, there is a good chance that many assets will not match. MAC Addresses are the most reliable way to correlate these, but many organizations do not fill out the MAC Address fields in NetBox.

NetBox asset fingerprinting

runZero will use the `mfg.name` as the hardware vendor and `mfg.model` as the hardware model for Devices. runZero will report Virtual Machine vendors by matching the Cluster Type field to a

known list of common hypervisors (Xen, VMware, Proxmox, etc.). The `platform.name` field, with and without the `mfg.name` field as a prefix will be used to find an operating system match.

The NetBox operating system data in `platform.name` will be treated as low-confidence relative to agent-based sources like EDRs and CMDB.

The NetBox hardware values will be treated similarly, with live agent-based sources taking priority.

NetBox-to-runZero custom field mapping

The NetBox integration allows for users to map custom fields within NetBox to attributes within runZero. These mappings will override the `mfg.name`, `mfg.model`, and `platform.name` attributes as noted above.

Please note that in the field mapping widget within the NetBox task configuration, the “NetBox field” inputs must be the **name** properties of the custom fields within NetBox.

The following runZero attributes are supported for custom field mapping:

- `hostnames`: The hostnames to apply to the asset within runZero. Multiple hostnames within the NetBox custom field must be comma-separated and no hostname may contain tab characters. This custom field must be a `Text` type.
- `manufacturer`: The manufacturer to apply to the asset within runZero. This custom field must be a `Text` type.
- `model`: The model to apply to the asset within runZero. This custom field must be a `Text` type.
- `device_type`: The type of device to apply to the asset within runZero. This custom field must be a `Text` type.
- `first_seen_ts`: The “first seen time” to apply to the asset within runZero. This custom field must be either a `Date & time` or `Date` type.
- `os`: The type of operating system to apply to the asset within runZero. This custom field must be a `Text` type.
- `os_version`: The version of the operating system to apply to the asset within runZero. This custom field must be a `Text` type.

Usage within the runZero CLI scanner

To use the custom field mapping functionality within the runZero CLI scanner, provide a JSON array with the following example structure as an argument, where in each array item the “key” represents the NetBox custom field name and the “value” represents the runZero attribute:

```
--netbox-field-mappings '[{"key":"netboxHostnames","value":"hostnames"}, {"key":"netboxManufacturer","value":"manufacturer"}]
```

In this example, `netboxHostnames` is the NetBox custom field being mapped to the runZero `hostnames` attribute and `netboxManufacturer` is the NetBox custom field being mapped to the runZero `manufacturer` attribute.

Searchable time attributes

The NetBox integration will process all Date & time or Date NetBox attributes and create searchable attributes from them within runZero. For example, a NetBox attribute called "lastChecked" with a value of 2024-07-26T15:30:00Z will have a new attribute created from it called lastCheckedTS with a Unix epoch timestamp value of 1722017400.

Getting started

To set up the NetBox integration, you will need:

1. A NetBox instance in the cloud or that is reachable by one of your runZero Explorers.
2. An administrative account in the NetBox instance.
3. A v1 NetBox API key (we do not currently support v2 API keys, which can be identified by their prefix of nbt_).

Step 1: Create a NetBox API key

1. Sign in to your NetBox portal.
2. Click on your name in the upper right and select **API Tokens** from the drop-down.
3. Click **Add a Token** and define a new token with no expiration and without Write permissions.
4. Give this new key a name and copy the secret value.

Step 2: Add the NetBox credential to runZero

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Provide a name for the credential, like NetBox CMDB.
3. Choose **NetBox API Key** from the list of credential types.
4. Provide the following information:
 - **Name** - Give this credential a unique name (ex:NetBox)
 - **NetBox URL** - The URL for your NetBox portal.
 - **NetBox API key** - The API key created in step 1.
 - **Insecure** - Set this to Yes if you want to attempt authentication without a verified thumbprint.
 - **NetBox thumbprints** (optional) - A set of IP=SHA256:B64HASH OR domain.tld=SHA256:B64HASH pairs to trust for authentication.
 - You may scan your NetBox instance with runZero in order to obtain the TLS thumbprint. The [TLS fingerprints service attribute report](#) lists all previously seen fingerprints.
 - If *insecure* is set to *No* and no thumbprints are provided:
 - With a self-signed certificate, the connection will fail because the certificate chain cannot be verified.
 - With a valid certificate from a public CA, the connection can work without thumbprints.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from NetBox.

Step 3: Configure the NetBox integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new NetBox-only assets are created.

1. We strongly recommend testing NetBox in a new Project first before applying this to your production Organization. Create a new Project.
2. Activate a [connection to NetBox](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
3. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization or project you are currently in.
4. Enter a name for the task, like `NetBox sync`.
5. Leave the default settings to create a one-off sync task.
6. Under **Task configuration**, choose the site you want to add your assets to. NetBox assets that do not match existing assets will use this site by default.
7. If you want to exclude assets that have not been scanned by runZero from your integration import, tick the **Exclude assets that cannot be merged into an existing asset** checkbox.
8. Activate the connection when you are done. The task will run. You can always check the [Tasks](#) page to track the status.

Step 4: View NetBox assets

After a successful sync, you can [go to your inventory](#) to view your NetBox assets. These assets will have a NetBox icon listed in the **Source** column.

To filter by NetBox assets, consider running the following queries:

- [View all NetBox assets:](#)

```
source:netbox
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by the NetBox API.

Review the new NetBox assets and confirm that the NetBox-sourced asset count is roughly in-line with your overall inventory and the total in NetBox itself.

If there are far fewer NetBox-sourced assets in runZero, you may want to avoid the final step until NetBox is updated with better correlation data.

Step 5: Enable Unknown Assets in Recurring Sync

Configure a new recurring NetBox task, with **Exclude assets that cannot be merged into an existing asset** disabled. This will create new assets for anything in NetBox that runZero can't correlate to in your asset inventory. If this process creates significant duplicates, delete all assets matching the filter `source_count:1 AND source:netbox`, and re-enable the *Exclude unknown assets* option in your recurring task until the NetBox records have better correlation data (MAC addresses, hostnames, etc.).

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the NetBox integration unable to connect?

1. Are you getting any data from the NetBox integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the NetBox integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the URL
 - 404 - hitting an unknown URL on the server
 - 401 - not authorized, likely a credential issue
 - 403 - access denied, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the NetBox host if it is on-premises.

Why aren't my custom field mappings aren't working properly?

1. Ensure you are using the `name` of the custom field and **not** the `label`.
2. Ensure the custom fields are applied to the correct types. The NetBox integration only supports the "Device" and "Virtual Machine" types within NetBox.
3. If your hostnames are becoming malformed within runZero, ensure they are comma-separated if there are multiple hostnames within the custom field. Additionally, ensure there are no tab characters present within the custom field.
4. All of the custom fields except for the `first_seen_ts` attribute must be the `Text` type within NetBox. `first_seen_ts` must be either a `Date & time` OR `Date` type.

Qualys VMDR

runZero integrates with Qualys VMDR by importing data from the Qualys KnowledgeBase [API](#).

Asset inventory

There is a column on the asset inventory page showing the count of vulnerabilities detected by Qualys for each asset. When a single asset is selected, the vulnerabilities table lists all the results related to that asset. The vulnerability count can be impacted by the type of vulnerability scan as well as the import settings selected.

Vulnerabilities table

The **Vulnerabilities** tab of the inventory lists all vulnerability results that have been imported from Qualys. The table lists every result, and selecting a result will take you to the page for the impacted asset.

Severity and risk scores

Qualys assigns all vulnerabilities a severity rating (Minimal, Medium, Serious, Critical, Urgent). runZero normalizes the severities shown in the vulnerability inventory to be consistent across the runZero Console.

runZero Severity	Qualys Severity
Info	1 / Minimal
Low	2 / Medium
Medium	3 / Serious
High	4 / Critical
Critical	5 / Urgent

runZero will also normalize risk scores assigned by Qualys. A risk score of 0.0 will be shown as `none` in the runZero Console, and all other risk scores will match the assigned severity level.

Getting started with Qualys

To set up the Qualys VMDR integration, you'll need to:

1. Create or obtain user credentials with access to the Qualys API.
2. Configure [CVSS scoring in Qualys](#).
3. Add the Qualys API username, password, and account API URL in runZero.
4. Choose whether to configure the integration as [a scan probe or connector task](#).
5. Activate the Qualys integration to pull your data into runZero.

Requirements

Before you can set up the Qualys VMDR integration:

- Make sure you have access to the Qualys Cloud Platform portal.

Step 1: Add the Qualys credentials to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like `Qualys`.
2. Choose **Qualys Username & Password** from the list of credential types.
3. Provide the following information:
 - **Qualys username** - the username you want to use to connect to the Qualys API.
 - **Qualys password** - the password for your Qualys API username.
 - **Qualys account API URL** - the URL of the Qualys API for the relevant account. The expected format is `https://ip:port` OR `https://domain.tld:port`. This URL is unique for [each Qualys user](#).
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential. You're now ready to set up and activate the connection to bring in data from Qualys VMDR.



The role of the Qualys API user will determine which VM scanned hosts are visible to the integration. Managers may view all VM scanned hosts in the subscription. Auditors have no permission to view VM scanned hosts. Unit Managers may view VM scanned hosts in the user's assigned business unit. Scanners and Readers may view VM scanned hosts in the user's account, but must be assigned to the assets through asset groups in VM/VMDR.

Step 2: Choose how to configure the Qualys integration

The Qualys integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 3: Set up and activate the Qualys integration to sync data

After you add your Qualys credential, you'll need to sync your data.



The Qualys Cloud Platform enforces limits on the API calls subscription users can make. Both API controls are limited per subscription based on your service level. The tasks

generated by this integration may experience slow performance or failures as a result of the enforced API limits. The Qualys API documentation on this topic can be found [here](#).

Step 3a: Configure the Qualys scan probe

You can run the Qualys VMDR integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *QUALYS* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct Qualys credential to *Yes* in the *Credentials* tab.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results.

Step 3b: Configure the Qualys connector

A connector requires you to set a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Qualys-only assets are created.

1. Activate a [connection to Qualys](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Set the [severity and risk levels](#) you want to import (optional).
4. Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
5. Specify the tags to include in the import. This should be a comma-separated list. Only assets that match any of the provided tags will be imported (optional).
6. Specify the network IDs to include in the import. This should be a comma-separated list. Only assets that match any of the provided network IDs will be imported (optional).
7. Enter a name for the task, like *Qualys sync* (optional).
8. Choose the Explorer to perform this connector task from (optional).
9. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
10. Enter a description for the task (optional).
11. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
12. If you want to include assets that have not been assessed for vulnerabilities, switch the **Include unscanned assets** toggle to *Yes*.
13. Enter a number of hours in the *Last update filter* to exclude assets that have not been scanned in longer than that time (optional). Leaving the value blank will use the default of 30 days.
14. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
15. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4: View Qualys assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Qualys assets. These assets will have a Qualys icon listed in the **Source** column.

The Qualys integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to Inventory > [Vulnerabilities](#) to view the vulnerability data provided by Qualys VMDR.

To filter by Qualys assets, consider running the following query:

- [View all Qualys assets:](#)

```
source:Qualys
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Qualys VMDR scan data.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Qualys integration unable to connect?

1. Are you getting any data from the Qualys integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Qualys integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Qualys host if it is on-premises.

Palo Alto Networks Firewall

runZero integrates with Palo Alto Networks Firewall using the [PAN-OS XML API](#) to provide additional network visibility, enhance network context, and improve [reporting](#).

Getting started

To set up the Palo Alto Networks Firewall integration, you'll need to:

- Create or obtain API Keys to use with the Palo Alto Networks Firewall XML API.
- Add the Palo Alto Networks Firewall API key in runZero.
- Perform Palo Alto Networks Firewall synchronization

Requirements

- Before you can set up the Palo Alto Networks Firewall integration, make sure you have an API Key for your PAN OS XML.
- Prior to adding a Palo Alto Networks Firewall credential, scan your Palo Alto Networks Firewall with a runZero Explorer if you want to use trusted authentication (optional).

Step 1: Add the Palo Alto Networks Firewall credential to runZero

1. Go to the [Add credential](#) page in runZero. Provide a name for the credentials, like **PAN-OS Firewall**.
2. Choose **Palo Alto Networks Firewall API Key** from the list of credential types.
3. Provide the following information:
 - **Palo Alto Networks API key** - The API key you want to use with the Palo Alto Networks Firewall integration. Ensure the XML API is enabled by following the steps in this guide: <https://docs.paloaltonetworks.com/ngfw/api/api-authentication-and-security/pan-os-api-authentication>. Once the XML API is enabled, you can generate the API key by following the steps in this guide: <https://docs.paloaltonetworks.com/ngfw/api/api-authentication-and-security/generate-api-key>
 - **Palo Alto Networks insecure** - Set this to **Yes** if you want to attempt authentication without a verified thumbprint.
 - **Palo Alto Networks thumbprints** (optional) - A set of IP[:port]=SHA256:B64HASH or hostname.domain.tld=SHA256:B64HASH pairs to trust for authentication.
 - You will need to scan your Palo Alto Networks firewalls with runZero in order to obtain the TLS thumbprint. The [PAN API thumbprints service attribute report](#) lists all previously seen thumbprints.
 - **CIDR allow list** - Set which IP addresses this API Key will be sent to in the **CIDR allow list**.
4. If you want all other organizations to be able to use this credential, select the **Make this a global credential** option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Palo Alto Networks Firewall.

Step 2: Performing Palo Alto Network synchronization

Once you have defined your Palo Alto Networks Firewall API key credentials, the second step is to enable Palo Alto Networks Firewall synchronization as part of a scan task. Any task which includes scanning the Palo Alto Networks Firewalls can be used to synchronize PAN-OS data.

On the Credentials tab of the scan setup, use the toggle switch to enable the appropriate set of Palo Alto Networks credentials.

When the scan runs, the Explorer will use the credentials to authenticate with any Palo Alto Networks Firewall it finds that the credentials are configured to trust. Data discovered about Palo Alto Networks Firewalls will be imported into runZero automatically, and merged with the other information runZero finds by scanning.

Step 3: View Palo Alto Networks Firewall assets

To filter by Palo Alto Networks Firewall assets, consider running the following queries:

[View all Palo Alto Networks Firewall assets:](#)

```
attribute:pan.api.thumbprint
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by Palo Alto Networks.

Palo Alto Prisma Cloud

runZero integrates with Palo Alto Prisma Cloud by importing data from the [Prisma API](#). This integration allows you to sync data about your cloud assets and vulnerabilities from Prisma to provide better visibility of your cloud assets and security posture. The supported Prisma cloud sources are AWS, Azure, and GCP.

Getting started with Prisma

To set up an integration with Prisma, you'll need to:

1. Create a Palo Alto Prisma Cloud credential in runZero.
2. Choose whether to configure the integration as [a scan probe or connector task](#).
3. Activate the integration to pull your data into runZero.

Step 1: Obtain your Prisma API credentials

1. Follow the [Prisma documentation](#) to create a Prisma cloud user role with sufficient permissions. See [user role descriptions](#) for more information.

2. Obtain your Prisma Cloud API access key and secret key. These will be used to authenticate against the Prisma Cloud API. Follow the [Prisma documentation](#) to configure them properly.
3. Identify your API URL. This will be sent to you from Palo Alto in your fulfillment email. See [possible URL values](#).

Step 2: Add the Prisma credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like Palo Alto Networks Prisma Cloud.
2. Choose **Prisma Client Secret** from the list of credential types.
3. Create your Prisma service account via the settings page in the Prisma portal, and then provide the following information:
 - **Prisma Cloud Access Key** - The access key you obtained from the steps above.
 - **Prisma Cloud Secret Key** - The secret key you obtained from the steps above.
 - **Prisma API URL** - The API Endpoint URL used to access the Prisma API.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Prisma Cloud.

Step 3: Choose how to configure the Prisma integration

The Prisma integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your Prisma credential, you'll need to sync your data from Prisma.

Step 4a: Configure the Prisma integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Prisma-only assets are created.

1. Activate a connection to [Prisma](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Set the **Exclude importing vulnerabilities** toggle if you want vulnerability records to be saved or not.
4. Set the **Exclude importing software** toggle if you want software records to be saved or not.

5. Set the **Exclude assets that can not be merged into an existing asset** toggle if you want no new asset records created, only enrich existing assets.
6. Set the severity and risk levels you want to import.
7. Enter a name for the task, like *Prisma Sync* (optional).
8. In the Run task with section, choose the Explorer or Explorer Group to perform this connector task from (optional).
9. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
10. Enter a description for the task (optional).
11. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
12. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Prisma integration as a scan probe

You can run the Prisma integration as a scan probe so that the runZero Explorer will pull your Prisma assets into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *Prisma* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Optionally, set the severity and risk levels for ingested vulnerability results.
- Set the correct *Prisma* credential to *Yes* in the *Credentials* tab.

Step 5: View Prisma assets, and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Prisma assets. These assets will have a Prisma icon listed in the **Source** column.

The Prisma integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to **Inventory** > **Vulnerabilities** to view the vulnerability data provided by Prisma.

To filter by Prisma assets, consider running the following queries:

- [View all Prisma assets:](#)

```
source:Prisma
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from Prisma Cloud.

Troubleshooting

If you see authentication or access errors when running the integration, it is likely that the API URL set for your credential is incorrect. Refer to the fulfillment email you received from Palo Alto. This will contain the correct API URL to use.

Rapid7

runZero integrates with Rapid7's [InsightVM](#) and [Nexpose](#) to enrich your asset inventory and gain visibility into vulnerabilities detected in your environment.

Asset inventory

There is a column on the asset inventory page showing the count of vulnerabilities detected by Rapid7 for each asset. When a single asset is selected, the vulnerabilities table lists all the results related to that asset. The vulnerability count can be impacted by the type of vulnerability scan as well as the import settings selected.

Vulnerabilities table

The **Vulnerabilities** tab of the inventory lists all vulnerability results that have been imported from Rapid7. The table lists every result, and selecting a result will take you to the page for the impacted asset.

Severity and risk scores

Rapid7 assigns all vulnerabilities a severity rating (Moderate, Severe, or Critical) based on the vulnerability's CVSSv2 score. runZero normalizes the severities shown the vulnerability inventory to be consistent across the runZero Console.

runZero Severity	Rapid7 Severity	CVSS Range
Info	Moderate	0.0
Medium	Moderate	0.1 - 3.4
High	Severe	3.5 - 7.4
Critical	Critical	7.5 - 10.0

runZero will also normalize risk scores assigned by Rapid7. A risk score of 0.0 will be shown as *none* in the runZero Console, and all other risk scores will match the assigned severity level.

Rapid7 InsightVM

runZero integrates with Rapid7 InsightVM by importing data from the [InsightVM API](#).

Rapid7 InsightVM Cloud and on-premises InsightVM are supported.

On-premises InsightVM

For on-premises installation, use the InsightVM connector [as a scan probe](#) from a runZero Explorer which has network access to the InsightVM deployment.

To set up the InsightVM integration, you'll need to:

1. Create or obtain user credentials to use with the InsightVM API.
2. Add the InsightVM API username, password, and API URL in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the InsightVM integration to pull your data into runZero.

Requirements

Before you can set up the InsightVM integration:

- Obtain credentials for an InsightVM user with administrator access to the InsightVM portal.
- Scan your InsightVM with a runZero Explorer if you want to use trusted authentication (optional).

Step 1: Add the InsightVM credentials to runZero

1. [Create a new credential](#) via the runZero [Credentials page](#).
2. Provide a name for the credentials, like `InsightVM`.
3. Choose **InsightVM Username & Password** from the list of credential types.
4. Provide the following information:
 - **InsightVM username** - The username you want to use to connect to the InsightVM API. This account requires the [user role](#) or greater permissions in InsightVM.
 - **InsightVM password** - The password for your InsightVM username.
 - **InsightVM API URL** - The URL of your InsightVM API instance. By default, the InsightVM API uses port 3780. The expected format is `https://ip:3780` or `https://domain.tld:3780`. runZero automatically appends `api/3/sites?size=500` to this URL.
 - **InsightVM insecure** - Set this to `Yes` if you want to attempt authentication without a verified thumbprint.
 - **InsightVM thumbprints** (optional) - A set of `IP=SHA256:B64HASH` or `domain.tld=SHA256:B64HASH` pairs to trust for authentication.
 - You will need to scan your InsightVM instance with runZero in order to obtain the TLS thumbprint. The [TLS fingerprints service attribute report](#) lists all previously seen fingerprints.
 - If *InsightVM insecure* is set to `No` and no thumbprints are provided:
 - With a self-signed certificate, the connection will fail because the certificate chain cannot be verified.
 - With a valid certificate from a public CA, the connection can work without thumbprints.
5. If you want all other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization

basis.

6. Save the credential.

You're now ready to set up and activate the connection to bring in data from InsightVM.

Step 2: Choose how to configure the Rapid7 integration

The Rapid7 InsightVM integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. Scan probes will be the right option for most users. Setting up a connector will only work for if you're self-hosting runZero or your InsightVM instance is publicly accessible.

Step 3: Sync your InsightVM data

After you add your InsightVM credential, you'll need to activate the integration to sync your data.

Step 3a: Configure the InsightVM scan probe

You can run the InsightVM integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *INSIGHTVM* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct InsightVM credential to *Yes* in the *Credentials* tab.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results.
- Enter a Site Name Filter to filter results based on the InsightVM site using regular expressions (optional).

Step 3b: Configure the InsightVM connector

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new InsightVM-only assets are created.

1. Activate a [connection to InsightVM](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Set the [severity and risk levels](#) you want to import (optional).
4. Input a Site Name Filter to filter your results by specific sites (optional). This input uses regular expressions and supports the [re2 syntax](#).
5. Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
6. Enter a name for the task, like *InsightVM sync* (optional).
7. Choose the Explorer to perform this connector task from (optional).
8. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.

9. Enter a description for the task (optional).
10. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
11. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
12. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4: View InsightVM assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your InsightVM assets. These assets will have a Rapid7 icon listed in the **Source** column.

The InsightVM integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to Inventory > [Vulnerabilities](#) to view the vulnerability data provided by InsightVM.

To filter by Rapid7 assets, consider running the following queries:

- [View all Rapid7 assets](#):

```
source:Rapid7
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Rapid7 scan data.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Rapid7 InsightVM integration unable to connect?

1. Are you getting any data from the Rapid7 InsightVM integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Rapid7 InsightVM integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

How can I get a TLS thumbprint for the InsightVM credential?

Here is a set of example commands that would calculate your TLS thumbprint, where is the IP address of your InsightVM instance:

```
$ (echo|openssl s_client -connect <IP address>:3780 -showcerts 2>/dev/null) | openssl x509 -inform PEM -outform DER |
```

An example value returned by this set of commands would look something like this:

X1NWttnkIQprK6zSre/VweKpbRlj7Dt4M6hNfUacytE= Use the following entry for the corresponding TLS thumbprint: 192.168.0.3:3780=SHA256:X1NWttnkIQprK6zSre/VweKpbRlj7Dt4M6hNfUacytE=

InsightVM Cloud

The Rapid7 InsightVM Cloud integration allows you to enrich runZero data with data from the Rapid7 InsightVM Cloud service.

To set up the Rapid7 InsightVM Cloud integration, you'll need to:

1. Create or obtain a Rapid7 Platform API Key to use with the [Insight API](#).
2. Create a credential in runZero with the Rapid7 Platform API Key.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Step 1: Add the Rapid7 Platform API credentials to runZero

1. [Create a new credential](#) via the runZero [Credentials page](#).
2. Provide a name for the credential, like `Rapid7 InsightVM Cloud`.
3. Choose **InsightVM Cloud API Key** from the list of credential types.
4. Provide the following information:
 - **InsightVM API Key** - The API key to access your InsightVM account.
 - **InsightVM Host** - The host of your InsightVM API instance. This is the hostname only, for example, `"us2.api.insight.rapid7.com"`.
5. If you want all other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
6. Save the credential.

You're now ready to set up and activate the connection to bring in data from InsightVM Cloud.

Step 2: Choose how to configure the Rapid7 integration

The Rapid7 InsightVM integration can be configured as either a [scan probe or a connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. Most use cases will work fine as a connector task.

Step 3: Sync your InsightVM data

After you add your InsightVM credential, you'll need to activate the integration to sync your data.

Step 3a: Configure the InsightVM Cloud scan probe

You can run the InsightVM integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *INSIGHTVMCLOUD* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct InsightCloud credential to *Yes* in the *Credentials* tab.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results.

Step 3b: Configure the InsightVM connector

A connector requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new InsightVM-only assets are created.

1. Activate a [connection to InsightVM Cloud](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Set the [severity and risk levels](#) you want to import (optional).
4. Enter a name for the task, like *InsightVM Cloud sync* (optional).
5. Choose the Explorer to perform this connector task from (optional).
6. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
7. Enter a description for the task (optional).
8. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
9. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
10. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Nexpose

runZero integrates with Rapid7 Nexpose by importing files that were exported from your Nexpose instance.

Getting started with Rapid7 Nexpose

To use the Rapid7 Nexpose integration, you'll need to:

1. Download an XML Export or XML Export 2.0 report from Nexpose.
2. Import the Nexpose files through the inventory pages.

Requirements

Before you can set up the Nexpose integration:

- Make sure you have access to the Nexpose portal.

Step 1: Export Nexpose vulnerability scan report

1. Sign in to Nexpose with the account being used for the runZero integration.
2. Go to the Reports page and select *Create a report*.
3. From the Export tab, select either XML Report or XML Report 2.0.
4. Set the scan, asset, asset group, or site scope.
5. Click *Save & Run the Report*.
6. When the report completes, save the report to a local file.

Step 2: Import the Nexpose files into runZero

1. Go to the [Inventory page](#) in runZero.
2. Choose **Import > Nexpose XML Export (.xml)** from the list of import types.
3. On the import data page:
 - Choose the site you want to add your assets to.
 - Set tags to apply to the imported assets (optional).
 - Set the [severity and risk levels](#) to ingest (optional).
 - Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).

Step 3: View Nexpose assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Nexpose assets. These assets will have a Rapid7 icon listed in the **Source** column.

The Nexpose integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to Inventory > [Vulnerabilities](#) to view the vulnerability data provided by Nexpose.

To filter by Rapid7 assets, consider running the following queries:

- [View all Rapid7 assets:](#)

```
source:Rapid7
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Nexpose scan file.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Rapid7 Nexpose integration unable to connect?

1. Are you getting any data from the Rapid7 Nexpose integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Rapid7 Nexpose integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Nexpose host.

SentinelOne

runZero integrates with SentinelOne by importing data from the [SentinelOne API](#). This integration allows you to sync and enrich your asset inventory, import software installed on assets, and import vulnerabilities affecting the installed software. Adding your SentinelOne data to runZero makes it easier to find things like endpoints that are missing required software or identify vulnerable endpoints.



Any IP address reported by SentinelOne will be treated as a secondary address, not a primary address, since these IPs can be stale and may not be associated with a specific network or site.

Getting started

To set up the SentinelOne integration, you'll need to:

1. Configure SentinelOne to allow API access through runZero.
2. Add the SentinelOne API key and SentinelOne base API URL in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the SentinelOne integration to sync your data with runZero.

Requirements

Before you can set up the SentinelOne integration:

- Make sure you have access to the SentinelOne admin portal.

Step 1: Configure SentinelOne to allow API access to runZero

1. Sign in to SentinelOne with the account being used for the runZero integration.
2. Go to **User > My User**.
3. **Generate** the API token, then download or copy it. This API key expires and will need to be regenerated every six months.

Step 2: Add the SentinelOne credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like `SentinelOne`.
2. Choose **SentinelOne API key** from the list of credential types.
3. Provide the following information:
 - **SentinelOne API URL** - Your organization-specific base URL, which will depend on your account type. It will be something like `organization.sentinelone.net`.
 - **SentinelOne API key** - To generate your API key, go to **User > My User** in your SentinelOne portal. From there, a key can be generated, regenerated, or revoked.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from SentinelOne.

Step 3: Choose how to configure the SentinelOne integration

The SentinelOne integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the SentinelOne integration to sync data

After you add your SentinelOne credential, you'll need to set up a connector task or scan probe to sync your data.

Step 4a: Configure the SentinelOne integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new SentinelOne-only assets are created.

1. Activate a [connection to SentinelOne](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).

2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Enter a name for the task, like `SentinelOne sync`.
4. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
5. Under **Task configuration**, choose the site you want to add your assets to.
6. If you do not want to import software, switch the **Import Software** toggle to *No*.
7. If you do not want to import vulnerabilities, switch the **Import Vulnerabilities** toggle to *No*.
8. If the **Import Vulnerabilities** toggle is set to *Yes*, you can set the desired vulnerability severities to import using the **Severities** checkboxes.
9. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
10. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the SentinelOne integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the SentinelOne toggle to *Yes*, and change any of the default options if needed.
4. On the Credentials tab, set the SentinelOne toggle for the credential you wish to use to *Yes*.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 5: View SentinelOne assets and software

After a successful sync, you can [go to your inventory](#) to view your SentinelOne assets. These assets will have a SentinelOne icon listed in the **Source** column.

The SentinelOne integration gathers details about installed software in addition to enriching asset inventory data. Go to Inventory > [Software](#) to view the software data provided by SentinelOne.

To filter by SentinelOne assets, consider running the following queries:

- [View all SentinelOne assets:](#)

```
source:SentinelOne
```

- [Find assets that have a SentinelOne agent installed:](#)

```
edr.name:SentinelOne
```

- [Find Windows assets, excluding servers, that are missing a SentinelOne agent:](#)

```
os:windows and not type:server and not edr.name:SentinelOne
```

Click into each asset to see its individual attributes. runZero will show you the attributes returned by the SentinelOne API, with the exception of policies.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the SentinelOne integration unable to connect?

1. Are you getting any data from the SentinelOne integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the SentinelOne integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the SentinelOne host if it is on-premises.

Shodan

runZero integrates with Shodan by importing data from the [Shodan API](#). This integration allows you to sync data about your externally-facing assets and services from Shodan to provide better visibility of your internet footprint and cyber hygiene.

Getting started

To set up the Shodan integration, you'll need to:

1. Add the Shodan API key in runZero.
2. Choose whether to configure the integration as [a scan probe or connector task](#).
3. Activate the Shodan integration to sync your data with runZero.

Requirements

Before you can set up the Shodan integration:

- Make sure you have a Shodan account with the correct license to meet your needs.

Step 1: Add the Shodan credential to runZero

1. Go to the [new credential page](#) in runZero. Provide a name for the credential, like Shodan.
2. Choose **Shodan Search API key** from the list of credential types.
3. Provide your **Shodan Search API key** - To view your API key, go to your Account page in the Shodan portal. Your API key is available on that page and can be reset if needed.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential. You're now ready to set up and activate the connection to bring in data from Shodan.

Step 2: Choose how to configure the Shodan integration

The Shodan integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 3: Set up and activate the Shodan integration to sync data

After you add your Shodan credential, you'll need to set up a connection or a scan probe to sync your data from Shodan.

Step 3a: Configure the Shodan integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Shodan-only assets are created.

1. Activate a [connection to Shodan](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. You can choose whether to specify a Shodan search using Shodan's search syntax, or to have runZero generate a search for all of the public IP addresses of live assets.
4. Optionally, you can choose to enter a comma-separated list of tags to exclude from Shodan assets by using the "Ignore Shodan tags" input.
5. Enter a name for the task, like *Shodan sync*.
6. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start at the date and time you have set.
7. Under **Task configuration**, choose the site you want to add your assets to.
8. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
9. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [tasks](#) page to see when the next sync will occur.

Step 3b: Configure the Shodan integration as a scan probe

1. Create a new scan task or select a future or recurring scan task from your [Tasks page](#).
2. Add or update the scan parameters based on any additional requirements.
3. On the Probes and SNMP tab, choose which additional probes to include, set the Shodan toggle to **Yes**, and change any of the default options if needed. As with running the integration as a connector task, you can choose to specify a Shodan search string directly, or choose *assets* mode to have runZero generate a search query to look for all public IP addresses of live assets.
4. On the Credentials tab, set the Shodan toggle for the credential you wish to use to **Yes**.
5. Click **Initialize scan** to save the scan task and have it run immediately or at the scheduled time.

Step 4: View Shodan assets and services

After a successful sync, you can [go to your inventory](#) to view your Shodan assets. These assets will have a Shodan icon listed in the **Source** column.

The Shodan integration gathers details about services in addition to enriching asset inventory data. Go to Inventory > [Services](#) to view the service data provided by Shodan.

To filter by Shodan assets or services, consider running the following queries:

- [View all Shodan assets:](#)

```
source:shodan
```

- [View all Shodan services:](#)

```
source:shodan
```

Click into each asset or service to see its individual attributes. runZero will show you the attributes returned by the Shodan Search API.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Shodan integration unable to connect?

1. Are you getting any data from the Shodan integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this

documentation and follow the steps exactly.

3. If the Shodan integration is unable to connect be sure to check the task log for errors.

Some common errors include:

- 500 - server error, unable to connect to the endpoint
- 404 - hitting an unknown endpoint on the server
- 403 - not authorized, likely a credential issue

Tailscale

runZero integrates with Tailscale to import the devices in your tailnet, including their addresses, operating system, hostnames, tags, and the Tailscale account that owns each node. Tailscale is the first integration runZero ships as a platform-packaged Starlark integration: it is built with the same engine as [custom integration scripts](#), but the script is embedded in the product, so it configures like any other connector — no script editing involved.

Getting started

To set up the Tailscale integration, you'll need to:

1. Create a Tailscale API access token, or an OAuth client, in the Tailscale admin console.
2. Add the Tailscale credential to runZero.
3. Set up and activate the connector task to sync your data with runZero.

Requirements

- Access to the [Tailscale admin console](#) with permission to create API access tokens or OAuth clients.
- The runZero custom integrations entitlement.
- When the task runs on an Explorer, the Explorer must be version 5.0.260723.0 or newer. The Tailscale API is reachable from the cloud, so the task can also run without an Explorer.

Step 1: Create a Tailscale API token or OAuth client

The integration accepts either credential form:

- **API access token** — in the Tailscale admin console, go to **Settings > Keys** and generate an API access token (it starts with `tskey-api-`). Tokens expire, so prefer an OAuth client for a scheduled sync.
- **OAuth client** — in the Tailscale admin console, go to **Settings > OAuth clients** and create a client with at least the `devices:core:read` scope. Note the client ID and client secret.

Step 2: Add the Tailscale credential to runZero

1. Go to the [Credentials page](#) in runZero and click **Add Credential**.
2. Choose **Tailscale Settings & Secrets** from the list of credential types.
3. Provide the following information:
 - **Tailscale API URL** — leave the default `https://api.tailscale.com` unless you proxy the API.
 - **Tailnet** — the tailnet ID or name, for example `T1234CNTRL` or `example.com`.
 - **OAuth client ID** — the OAuth client ID, or leave blank to use a plain API key.
 - **API key / OAuth client secret** — the API access token, or the OAuth client secret when a client ID is set.
4. Save the credential.

Step 3: Set up and activate the Tailscale integration

1. Activate a [connection to Tailscale](#). You can also reach it from the Tailscale card on the [integrations page](#).
2. Choose the credential you added earlier. If you don't see the credential listed, make sure the credential has access to the organization you are currently in.
3. Enter a name for the task, like `Tailscale sync`.
4. Schedule the sync, and choose the site newly discovered assets should be created in.
5. To keep the sync from creating assets runZero has not seen through other means, enable **Exclude assets that cannot be merged into an existing asset**.
6. Select an Explorer, or leave the selection empty to run the sync from the runZero cloud.
7. Activate the connection. The sync will run on the defined schedule.

Step 4: View Tailscale assets

After a successful sync, your Tailscale devices appear in [your inventory](#) with the Tailscale icon in the **Source** column, and their attributes are stored under `tailscale.device.*`. The Tailscale account that owns each node is resolved into asset ownership.

To filter by Tailscale assets:

```
source:tailscale
```

Tanium API Gateway

runZero integrates with Tanium by importing data from the [Tanium Gateway API](#). This integration allows you to sync data about your endpoints, applications, and vulnerabilities from Tanium to provide better visibility over your network.

Getting started with Tanium

To set up an integration with Tanium, you'll need to:

1. Generate an API token with the necessary permissions.
2. Configure the Tanium credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Step 1: Generate an API key in Tanium Dashboard

1. Sign in to Tanium and navigate to **Administration > Roles**.
2. Create a role with the necessary permissions:
 1. Search for the **Gateway User** role.
 2. Select it and click the **Clone** button that appears to create a copy of this role.
 3. On the **Clone Role** screen, enable **Platform Content Permissions > Sensor > Read** and add these Content Sets (via the **n+** button beside the green check):
 - Base
 - Comply
 - Comply Reporting
 - Core AD Query Content
 - Core Content
 - Reserved
 - Tanium Data Service
 4. Save the role.
3. Navigate to **Administration > Personas** and click **New Persona** to create a persona using the role you just created:
 1. Name the persona.
 2. Under **Manage Roles**, search for and apply your new role.
 3. Under **Computer Groups**, add the groups you need, or check **Unrestricted Management Rights** to allow access to all Computer Groups.
 4. Assign a user or service account which has the permissions granted to the persona.
 5. Save the persona.
4. Navigate to **Administration > API Tokens** and click **New API Token**.
 1. Enter a name and select a TTL.
 2. Select the persona you just created from the dropdown (you may need to refresh the page for it to appear).
 3. Enter IP addresses to allow requests from:
 - If you will run the integration via an Explorer or CLI, enter the IP addresses or ranges of your host(s);
 - Otherwise, enter 0.0.0.0/0.
 4. Save the API token.

Step 2: Add the Tanium API token to runZero

1. Go to the [Credentials](#) page in runZero.
2. Choose **Tanium API Token** from the list of credential types.
3. Provide a name for the credential, like `Tanium`.
4. Provide the following information:
 - **Tanium API URL** - Your Tanium API Gateway URL. The full URL will be something like `https://<customername>-api.cloud.tanium.com/plugin/products/gateway/graphql`. If the

- path (/plugin/products/gateway/graphql) is omitted, it will be added automatically when the API is called.
- **Tanium API token** - The API token (including the token- prefix) created in step 1.
- **Insecure** - Enable this option to approve authenticating with untrusted endpoints. When enabled, certificate validation is disabled. Use with caution.
- 5. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
- 6. Verify and save the credential.

You're now ready to set up and activate the connection to bring in data from Tanium.

Step 3: Choose how to configure the Tanium integration

The Tanium integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your Tanium credential, you'll need to sync your data from Tanium.

Step 4a: Configure the Tanium integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Tanium-only assets are created.

1. Activate a connection to [Tanium](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Optionally provide a list of computer groups to include in the import. The list must be comma-separated. We will only import data for the computer groups specified.
4. Enter a name for the task, like *Tanium Sync* (optional).
5. Choose the Explorer to perform this connector task from (optional).
6. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
7. Enter a description for the task (optional).
8. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
9. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
10. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Tanium integration as a scan probe

You can run the Tanium integration as a scan probe so that the runZero Explorer will pull your Tanium assets into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *TANIUM* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct *TANIUM* credential to *Yes* in the *Credentials* tab.

Step 5: View Tanium assets

After a successful sync, you can [go to your inventory](#) to view your Tanium assets. These assets will have a Tanium icon listed in the **Source** column.

To filter by Tanium assets, consider running the following queries:

- [View all Tanium assets:](#)

```
source:Tanium
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from Tanium.

Tenable

runZero integrates with Tenable Vulnerability Management (previously Tenable.io), Tenable Nessus, and Tenable Security Center to enrich your asset inventory and gain visibility into vulnerabilities detected in your environment. The [Tenable Vulnerability Management](#), [Nessus Professional](#), and [Tenable Security Center](#) integrations pull data from the Tenable API, while all versions of [Tenable Nessus](#) and Tenable Security Center (previously Tenable.sc) are also supported through Nessus v2 file imports (.nessus).

Note that at this time, only the main [Tenable Vulnerability Management](#) cloud API endpoint at <https://cloud.tenable.com> is supported as an API integration.

Asset inventory

There is a column on the asset inventory page showing the count of vulnerabilities detected by Tenable for each asset. When a single asset is selected, the vulnerabilities table lists all the results related to that asset. The vulnerability count can be impacted by the type of vulnerability scan as well as the import settings selected.

Vulnerabilities table

The **Vulnerabilities** tab of the inventory lists all vulnerability results that have been imported from Tenable. The table lists every result, and selecting a result will take you to the page for the impacted asset.

Severity and risk scores

Tenable uses the third-party Common Vulnerability Scoring System (CVSS) values from the National Vulnerability Database (NVD) to describe severity associated with vulnerabilities. Tenable assigns all vulnerabilities a severity rating (Info, Low, Medium, High, or Critical) based on the vulnerability's static CVSSv2 or CVSSv3 score. By default, Tenable uses CVSSv2, however users can opt to use CVSSv3 instead. runZero displays the CVSSv2-based severity in the Inventory and Asset views.

To supplement the severity ratings, Tenable calculates a dynamic Vulnerability Priority Rating (VPR) for most vulnerabilities. The VPR is a dynamic companion to the data provided by the vulnerability's CVSS score, since Tenable updates the VPR to reflect the current threat landscape. VPR values range from 0.1-10.0, with a higher value representing a higher likelihood of exploit. runZero displays the VPR-based risk in the Inventory and Asset views. For more details about how Tenable handles severity and risk, please refer to their [documentation](#).

Tenable Vulnerability Management

runZero integrates with Tenable Vulnerability Management (previously Tenable.io) by importing data from the Tenable [API](#).

Getting started with Tenable Vulnerability Management

To set up an integration with Tenable Vulnerability Management, you'll need to:

1. Create an Administrator API key in an access group with Can View [permission](#) to Manage Assets. Optionally, this must have the Scan Manager role in order to retrieve agent health data.
2. Configure the Tenable Vulnerability Management credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Requirements

Before you can set up the Tenable Vulnerability Management integration:

- Make sure you have administrator access to the Tenable portal.

Step 1: Create an Administrator API key

1. Sign in to Tenable Vulnerability Management with the Administrator account being used for the runZero integration.
2. Go to **My Profile > My Account > API Keys**.
3. **Generate** the API token, and then download or copy it.

Step 2: Add the Tenable credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like Tenable Vulnerability Management.
2. Choose **Tenable.io Access & Secret** from the list of credential types.
3. Generate your Tenable access and secret keys via your account page in the Tenable portal, and then provide the following information:
 - **Access key** - Your 64-character Tenable access key.
 - **Secret key** - Your 64-character Tenable secret key.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Tenable Vulnerability Management.

Step 3: Choose how to configure the Tenable integration

The Tenable Vulnerability Management integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. Setting up a connector will work if you're self-hosting runZero or integrating with Tenable Vulnerability Management.

Step 4: Set up and activate the integration to sync data

After you add your Tenable credential, you'll need to sync your data from Tenable Vulnerability Management.

Step 4a: Configure the Tenable integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Tenable-only assets are created.

1. Activate a connection to [Tenable Vulnerability Management](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.

3. If you want to exclude assets that have not been scanned by runZero from your integration import, check the **Exclude assets that cannot be merged into an existing asset** checkbox. By default, the integration will include assets that have not been scanned by runZero.
4. If you want to include assets that have not been assessed for vulnerabilities, check the **Include assets that have not been assessed for vulnerabilities** checkbox.
5. If you want to exclude retrieving asset agent health data, check the **Disable importing asset agent data** checkbox.
6. Check the **Disable importing software from the Tenable inventory** checkbox if you want software records to be ingested for fingerprint analysis but not stored in your runZero software inventory (optional).
7. Check the **Disable importing vulnerabilities from the Tenable inventory** checkbox if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
8. Set the [severity and risk levels](#) you want to import (optional). **Note:** Much of the host information provided by Tenable is from Info-level plugins, so if you only import higher levels of severity you may not see much information about assets not scanned by runZero.
9. Optionally provide a list of tags to include in the import. The list should be comma separated and use the format `category:value`. We will import assets that match at least one of the specified tags.
10. To filter by asset source, set the **Filter by asset source** option to either *All supported sources* or a combination of sources from the checklist.
11. Enter a name for the task, like *Tenable Vulnerability Management sync* (optional).
12. Choose the Explorer to perform this connector task from (optional).
13. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
14. Enter a description for the task (optional).
15. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
16. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Tenable integration as a scan probe

You can run the Tenable Vulnerability Management integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *TENABLE* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results.
- Set the correct Tenable credential to *Yes* in the *Credentials* tab.

Step 5: View Tenable assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Tenable assets. These assets will have a Tenable icon listed in the **Source** column.

The Tenable integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to **Inventory** > **Vulnerabilities** to view the vulnerability data provided by Tenable Vulnerability Management.

To filter by Tenable assets, consider running the following queries:

- [View all Tenable assets:](#)

```
source:Tenable
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Tenable scan data.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Tenable Vulnerability Management integration unable to connect?

1. Are you getting any data from the Tenable Vulnerability Management integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the Tenable Vulnerability Management integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Tenable host if it is on-premises.

How do I solve the following error in Tenable Vulnerability Management:

```
"error-message":"no tenable assets match import criteria",
"level":"error","msg":"could not load scan result data to writer"
```

This is an error we have seen intermittently from Tenable. A solution that usually works is to enable the *Include Unscanned Assets* toggle in the Tenable task configuration. This will disable the filters we apply for live assets that were scanned in the last 30 days.

Nessus Professional

runZero integrates with Nessus Professional by importing data from the Tenable [API](#).

Getting started with Nessus Professional

To set up an integration with Nessus Professional, you'll need to:

1. Create an Administrator API key in an access group with *Can View* [permission](#) to *Manage Assets*.
2. Configure the Nessus Professional credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Requirements

Before you can set up the Nessus Professional integration:

- Make sure you have administrator access to the Nessus Professional portal.

Step 1: Create an Administrator API key

1. Sign in to Nessus Professional with the Administrator account being used for the runZero integration.
 2. Go to **My Profile > My Account > API Keys**.
 3. **Generate** the API token, and then download or copy it.
- You will either need to configure the Tenable credential to skip TLS verification, or provide the TLS thumbprint when creating the runZero credential.

Step 2: Add the Nessus Professional credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like *Nessus Professional*.
2. Choose **Nessus Professional Access & Secret** from the list of credential types.
3. Generate your Tenable access and secret keys via your account page in the Tenable portal, and then provide the following information:
 - **Access key** - Your 64-character Tenable access key.
 - **Secret key** - Your 64-character Tenable secret key.
 - **Nessus API URL** - The API URL for your Nessus Professional instance. The expected format is `https://ip:port` or `https://domain.tld:port`. The default port used by Nessus Professional is 8834.
 - **Nessus insecure** - Set this to *Yes* if you want to attempt authentication without a verified thumbprint.
 - **Nessus thumbprints** (optional) - A set of IP=SHA256:B64HASH pairs to trust for authentication.
 - You will need to scan your Nessus instance with runZero in order to obtain the TLS thumbprint. The [TLS fingerprints service attribute report](#) lists all previously seen fingerprints.
 - If *Nessus insecure* is set to *No* and no thumbprints are provided:
 - With a self-signed certificate, the connection will fail because the certificate chain cannot be verified.

- With a valid certificate from a public CA, the connection can work without thumbprints.
- 4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
- 5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Nessus Professional.

Step 3: Choose how to configure the Nessus Professional integration

The Nessus Professional integration can be configured as either a [scan probe](#) or a [connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. Setting up a connector will work if you're self-hosting runZero or your Nessus Professional instance is publicly accessible.

Step 4: Set up and activate the integration to sync data

After you add your Tenable credential, you'll need to sync your data from Nessus Professional.

Step 4a: Configure the Nessus Professional integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Tenable-only assets are created.

1. Activate a connection to [Nessus Professional](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Set the [severity and risk levels](#) you want to import (optional). **Note:** Much of the host information provided by Tenable is from Info-level plugins, so if you only import higher levels of severity you may not see much information about assets not scanned by runZero.
4. Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
5. Enter a name for the task, like *Nessus Professional sync* (optional).
6. Choose the Explorer to perform this connector task from (optional).
7. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
8. Enter a description for the task (optional).
9. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
10. If you want to include assets that have not been assessed for vulnerabilities, switch the **Include unscanned assets** toggle to *Yes*.

11. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
12. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Nessus Professional integration as a scan probe

You can run the Nessus Professional integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *NESSUS* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Set the correct *Nessus* credential to *Yes* in the *Credentials* tab.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results.

Step 5: View Tenable assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Tenable assets. These assets will have a Tenable icon listed in the **Source** column.

The Tenable integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to Inventory > [Vulnerabilities](#) to view the vulnerability data provided by Nessus Professional.

To filter by Tenable assets, consider running the following queries:

- [View all Tenable assets:](#)

```
source:Tenable
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Tenable scan data.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Nessus Professional integration unable to connect?

1. Are you getting any data from the Nessus Professional integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
- 3.

3. If the Nessus Professional integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the Tenable host if it is on-premises.

How do I solve the following error in Nessus Professional:

- "error-message":"no tenable assets match import criteria","level":"error","msg":"could not load scan result data to writer" {#nessuspro-import-error}

This is an error we have seen intermittently from Tenable. A solution that usually works is to enable the *Include Unscanned Assets* toggle in the Tenable task configuration. This will disable the filters we apply for live assets that were scanned in the last 30 days.

Tenable Nessus

runZero integrates with Tenable Nessus using two methods. For all versions of Nessus, runZero can import Nessus files (.nessus) that were exported from your Nessus instance. Exports from Tenable Security Center are also supported. For Nessus Professional users, the runZero integration can pull scan data from the [Nessus Professional API](#).

Getting started with Tenable Nessus

To use the Tenable Nessus integration, you'll need to:

1. Export vulnerability scan results as Nessus files.
2. Import the Nessus files through the inventory pages.

Requirements

Before you can set up the Nessus integration:

- Make sure you have access to the Nessus portal.

Step 1: Export vulnerability scan results

1. Sign in to Nessus with the account being used for the runZero integration.
2. Open the scan results you want to be able to import into runZero.
3. Choose **Export** > **Nessus** to download the scan results.

Step 2: Import the Nessus files into runZero

1. Go to the [Inventory page](#) in runZero.
2. Choose **Import** > **Nessus scan (.nessus)** from the list of import types.
3. On the import data page:
 - Choose the site you want to add your assets to, and

- Set the [severity levels and minimum risk level](#) to ingest. (**Note:** much of the host information provided by Tenable is from Info-level plugins, so if you only import higher levels of severity you may not see much information about assets not scanned by runZero.)

Step 3: View Nessus assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Nessus assets. These assets will have a Tenable icon listed in the **Source** column.

The Nessus integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to Inventory > [Vulnerabilities](#) to view the vulnerability data provided by Nessus.

To filter by Nessus assets, consider running the following queries:

- [View all Nessus assets:](#)

```
source:Tenable
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from the Nessus scan file.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Tenable Nessus integration unable to connect?

1. Are you getting any data from the Tenable Nessus integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.

Tenable Security Center

runZero integrates with Tenable Security Center (previously Tenable.sc) by importing data from the Tenable Security Center [API](#).

Getting started with Tenable Security Center

To set up an integration with Tenable Security Center, you'll need to:

1. Create an API key for a user that has access to view and query vulnerabilities in Tenable Security Center.

2. Configure the Tenable Security Center credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Requirements

Before you can set up the Tenable Security Center integration:

- Make sure you have administrator access to the Tenable Security Center portal.

Step 1: Create an API key

1. Sign in to Tenable Security Center with an Administrator account.
2. Make sure [API key authentication is enabled](#)
3. Go to **Users > Users**.
4. Check the box for the user you want to create an API key for. **Note:** The API key will have the same access as the user you select. Make sure the user has access to view and query vulnerabilities in the desired organization.
5. At the top of the table, click the **API Keys > Generate API Key** option.
6. Click **Generate** to create the API token, and then download or copy it.

Step 2: Add the Tenable Security Center credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like Tenable Security Center.
2. Choose **Tenable Security Center Access & Secret** from the list of credential types.
3. Generate your Tenable Security Center API key as directed in [Step 1](#), and then provide the following information:
 - **Access key** - Your 64-character Tenable Security Center access key.
 - **Secret key** - Your 64-character Tenable Security Center secret key.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Tenable Security Center.

Step 3: Choose how to configure the Tenable Security Center integration

The Tenable Security Center integration can be configured as either a [scan probe or a connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync. If you are integrating with an internal Tenable Security Center instance, we recommend setting up a connector to run from one of your explorers. Otherwise, if you are integrating with an external-facing Tenable Security Center instance, you can set up a connector to run from the cloud. If you are self-hosting runZero, you can run the connector from an explorer or from your runZero host, whichever can reach your Tenable Security Center install.

Step 4: Set up and activate the integration to sync data

After you add your credential, you'll need to sync your data from Tenable Security Center.

Step 4a: Configure the Tenable Security Center integration as a connector task

A connection requires you to specify a schedule which determines when the sync occurs.

1. Activate a connection to [Tenable Security Center](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Configure the *Tenable Security Center query mode* setting (optional).
 - Select *Define filters* to define a filter based on [vulnerability severity and risk level](#).
Note: Much of the host information provided by Tenable is from Info-level plugins, so if you only import higher levels of severity you may not see much information about assets.
 - Select *Use existing query ID* to provide the Tenable Security Center query to use.
Note: The query must be the *Vulnerability* type and use the *Vulnerability Detail List* tool.
4. Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
5. Enter a name for the task, like *Tenable Security Center sync* (optional).
6. Choose the Explorer to perform this connector task from (optional).
7. Choose the site you want to configure the connector for.
8. Enter a description for the task (optional).
9. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
10. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Tenable Security Center integration as a scan probe

You can run the Tenable Security Center integration as a scan probe so that the runZero Explorer will pull your vulnerability data into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *TENABLESECURITYCENTER* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Optionally, set the [severity and risk levels](#) for ingested vulnerability scan results or provide a query ID.
- Set the correct *TenableSecurityCenter* credential to *Yes* in the *Credentials* tab.

Step 5: View Tenable Security Center assets and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Tenable Security Center assets. These assets will have a Tenable icon listed in the **Source** column.

The Tenable Security Center integration gathers details about vulnerabilities detected in addition to enriching asset inventory data. Go to **Inventory** > **Vulnerabilities** to view the vulnerability data provided by Tenable Security Center.

To filter by Tenable Security Center assets, consider running the following queries:

- [View all Tenable Security Center assets:](#)

```
source:tenablesecuritycenter
```

- [View all Tenable Security Center vulnerabilities:](#)

```
source:tenablesecuritycenter
```

Click into each asset or vulnerability to see its individual attributes. runZero will show you the attributes gathered from the Tenable Security Center API.

VMware

runZero Platform supports synchronization of VMware vCenter and ESXi virtual machine inventories.

Setting up VMware credentials

Unlike other APIs, the VMware synchronization process is configured as part of your regular runZero Explorer scans. The first step is to set up a set of VMware credentials.

On the [Scanning with credentials](#) page, click *Add Credential* and choose a credential type of *VMware vCenter/ESXi Username and Password*, and enter the appropriate username and password. The correct username syntax in most cases is `user@domain.com`. The VMware account used requires at least **read-only** access.

The CIDR allow list field can be used to limit which addresses the credentials should be sent to. This helps ensure that they are not passed to unexpected VMware systems that the runZero Explorer encounters on the network.

If runZero has previously found VMware API endpoints, the optional *VMware thumbprints* field will list their IP addresses and TLS fingerprints. You can edit this list to remove any systems you do not want to trust with your VMware credentials. Alternatively, if you do not want to limit authentication to the set list of IP addresses and TLS certificates, you can set *VMware insecure* to *Yes* to approve authenticating with untrusted endpoints.

The organization access for the credentials can be set as for any other stored credentials; see [Scanning with credentials](#).

Performing VMware synchronization

Once you have defined a set of VMware credentials, the second step is to enable VMware synchronization as part of a scan task. Any task which includes scanning the VMware host

systems can be used to synchronize VMware VM data.

The Probes tab of the scan setup has a section for enabling and disabling the VMware probe. The probe must be enabled for VMware synchronization to work; it is enabled by default.

On the Credentials tab of the scan setup, use the toggle switch to enable the appropriate set of VMware vCenter/ESXi credentials.

When the scan runs, the Explorer will use the credentials to authenticate with any VMware ESXi or vCenter hosts it finds that the credentials are configured to trust. Data about assets which are VMware VMs will be imported into runZero automatically, and merged with the other information runZero finds by scanning.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the VMware integration unable to connect?

1. Are you getting any data from the VMware integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
3. If the VMware integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue
4. Verify you are running the integration task from an Explorer with access to the VMware host if it is on-premises.

Wiz

runZero integrates with Wiz by importing data from the [Wiz API](#). This integration allows you to sync data about your cloud assets, software, and vulnerabilities from Wiz to provide better visibility of your cloud assets and security posture.

Getting started with Wiz

To set up an integration with Wiz, you'll need to:

1. Create a Service Account in Wiz with permissions to read graph resources, read reports, and create reports.

2. Configure the Wiz credential in runZero.
3. Choose whether to configure the integration as [a scan probe or connector task](#).
4. Activate the integration to pull your data into runZero.

Requirements

Before you can set up the Wiz integration:

- Make sure you have administrator access to the Wiz portal.

Step 1: Create a Service Account in Wiz

1. Sign in to Wiz with an Administrator account.
2. Go to **Settings > Access Management > Service Accounts > Add Service Account**.
3. Enter a descriptive name in the **Name** field.
4. Select *Custom Integration (GraphQL API)* for the **Type**.
5. Under **Projects**, select the projects the Service Account should have access to, or leave blank to allow access to all projects.
6. Enable `read:resources`, `read:reports`, and `create:reports` for the **API Scopes**.
7. Click **Add Service Account** and copy the **Client ID** and **Client Secret**.
8. Go to **User Settings > Tenant** and note the **API Endpoint URL** in the format: `https://api.{{region}}.app.wiz.io/`.

Step 2: Add the Wiz credential to runZero

1. Go to the [Credentials page](#) in runZero. Provide a name for the credentials, like `wiz`.
2. Choose **Wiz Client Secret** from the list of credential types.
3. Create your Wiz service account via the settings page in the Wiz portal, and then provide the following information:
 - **Wiz Client ID** - The client ID of your Wiz service account.
 - **Wiz Client Secret** - The client secret of your Wiz service account.
 - **Wiz Auth URL** - The URL used to authenticate the Wiz service account.
 - **Wiz API URL** - The API Endpoint URL used to access the Wiz API.
4. If you want other organizations to be able to use this credential, select the *Make this a global credential* option. Otherwise, you can configure access on a per-organization basis.
5. Save the credential.

You're now ready to set up and activate the connection to bring in data from Wiz.

Step 3: Choose how to configure the Wiz integration

The Wiz integration can be configured as either a [scan probe or a connector task](#). Scan probes gather data from integrations during scan tasks. Connector tasks run independently from either the cloud or one of your Explorers, only performing the integration sync.

Step 4: Set up and activate the integration to sync data

After you add your Wiz credential, you'll need to sync your data from Wiz.

Step 4a: Configure the Wiz integration as a connector task

A connection requires you to specify a schedule and choose a site. The schedule determines when the sync occurs, and the site determines where any new Wiz-only assets are created.

1. Activate a connection to [Wiz](#). You can access all available third-party connections from the [integrations page](#), your [inventory](#), or the [tasks page](#).
2. Choose the credentials you added earlier. If you don't see the credentials listed, make sure the credentials have access to the organization you are currently in.
3. Set the severity and risk levels you want to import (optional).
4. Set the **Fingerprint only** toggle to *Yes* if you want vulnerability records to be ingested for fingerprint analysis but not stored in your runZero vulnerability inventory (optional).
5. Enter a name for the task, like *Wiz Sync* (optional).
6. Choose the Explorer to perform this connector task from (optional).
7. Choose the site you want to add your assets to. All newly discovered assets will be stored in this site.
8. Enter a description for the task (optional).
9. If you want to exclude assets that have not been scanned by runZero from your integration import, switch the **Exclude unknown assets** toggle to *Yes*. By default, the integration will include assets that have not been scanned by runZero.
10. If you want to exclude assets that have not been assessed for vulnerabilities, switch the **Include unscanned assets** toggle to *No*.
11. Schedule the sync. A sync can be set to run on a recurring schedule or run once. The schedule will start on the date and time you have set.
12. Activate the connection when you are done. The sync will run on the defined schedule. You can always check the [Scheduled tasks](#) to see when the next sync will occur.

Step 4b: Configure the Wiz integration as a scan probe

You can run the Wiz integration as a scan probe so that the runZero Explorer will pull your Wiz assets into the runZero Console.

In a new or existing scan configuration:

- Ensure that the *WIZ* option is set to *Yes* in the *Probes and SNMP* tab and change any of the default options if needed.
- Optionally, set the severity and risk levels for ingested vulnerability results.
- Set the correct *Wiz* credential to *Yes* in the *Credentials* tab.

Step 5: View Wiz assets, software, and vulnerabilities

After a successful sync, you can [go to your inventory](#) to view your Wiz assets. These assets will have a Wiz icon listed in the **Source** column.

The Wiz integration gathers details about software and vulnerabilities detected in addition to enriching asset inventory data. Go to **Inventory** > **Software** or **Inventory** > **Vulnerabilities** to view the software and vulnerability data provided by Wiz.

To filter by Wiz assets, consider running the following queries:

- [View all Wiz assets:](#)

```
source:Wiz
```

Click into each asset to see its individual attributes. runZero will show you the attributes gathered from Wiz.

Troubleshooting

If you are having trouble using this integration, the questions and answers below may assist in your troubleshooting.

Why is the Wiz integration unable to connect?

1. Are you getting any data from the Wiz integration?
 - Make sure to query the inventory rather than look at the task details to review all the data available from this integration.
 - In some cases, integrations have a configuration set that limits the amount of data that comes into the runZero console.
2. Some integrations require very specific actions that are easy to overlook. If a step is missed when setting up the integration, it may not work correctly. Please review this documentation and follow the steps exactly.
 - Double-check the API Scopes assigned to the Wiz service account. A valid service account that is missing the required permissions will result in a failed import.
3. If the Wiz integration is unable to connect be sure to check the task log for errors. Some common errors include:
 - 500 - server error, unable to connect to the endpoint
 - 404 - hitting an unknown endpoint on the server
 - 403 - not authorized, likely a credential issue

Outbound integrations

Using runZero data to enrich other tools

In addition to being able to enrich your runZero inventory with data from your [other IT and security tools](#), the runZero platform offers [egress integrations with several platforms](#). By leveraging product APIs and export/import functionality, runZero can provide additional asset context in other IT and security tools.

The following integrations are available to send your runZero data into other platforms:

IT service management

- [Atlassian Insight & Jira Service Management](#)
- [Jira \(issue tracking\)](#)
- [ServiceNow CMDB](#)

Detection and investigation

- [Panther](#)
- [Splunk Search](#)
- [Sumo Logic](#)
- [Tines](#)
- [Thinkst Canary](#)

Vulnerabilities and risk

- [SecurityGate.io](#)

Atlassian Insight & Jira Service Management

This page describes the asset-enrichment integration that imports runZero asset data into Jira Service Management's CMDB. If you want to push runZero [issues](#) to Jira as tickets and sync changes back, see the [Jira integration](#) page instead.

All runZero editions integrate with Jira Service Management via an import in Atlassian Insight. runZero asset data is then imported into the CMDB.

Follow these steps to perform a basic import.

Step 1: Export runZero asset data

You can export data using the **Export** button from the runZero inventory or the Export API.

The following are sample commands for the export API that include common export fields but omit the tags field. You must replace the token ETxxx... with your account's export token from the [Inventory export API page](#).

For a CSV export, use this command (one line):

```
curl -o "assets.csv" -H "Authorization: Bearer ETxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
  "https://console.runzero.com:443/api/v1.0/export/org/assets.csv"
```

For a JSON export, use this command (one line):

```
curl -o "assets.json" -H "Authorization: Bearer ETxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx"
  "https://console.runzero.com:443/api/v1.0/export/org/assets.csv"
```

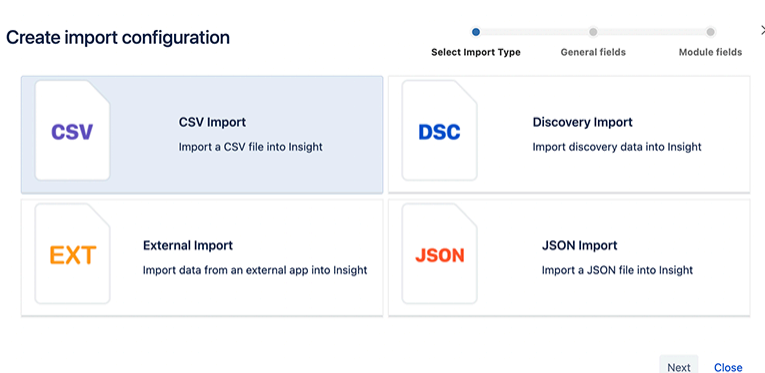


Atlassian Insight does not accept any attribute longer than 255 characters and will produce errors if you are trying to import a file that contains longer fields. If you are seeing such errors, you may need to either trim or omit the fields.

Step 2: Import data into Atlassian Insight

You must use Jira Service Management Premium or Enterprise and have Atlassian Insight installed for the following steps to work.

1. In Jira Service Management, go to the Insight menu and click the + sign to create an object schema.
2. Name the object schema. In our example, we're calling it **Assets**.
3. Click on ... next to the object schema and choose **Configuration**.
4. Go to the **Import** tab.
5. Click **Create Configuration**.



6. Choose CSV or JSON, depending on your previous export and click **Next**.

7. Enter the name for your import.

8. In the **Concatenator** field, enter \s.

Create import configuration

Name *

Rumble CSV import

The name for the configuration

Description

The description for the configuration

Concatenator *

\s

When joining multiple data locators into one Insight attribute, this will be the default concatenator. Enter \s for space-concatenated.

Empty Values *

Remove

Specify if the import should remove or ignore empty values

Unknown Values *

Add

Specify if the import should try to add unknown values to Insight configuration or just ignore them. This is applied on Status and Select attribute types.

9. Scroll down and click **Next**.

10. Choose the file you have previously exported from runZero.

Create import configuration

File *

assets.csv

Select a file to import.

Delimiter *

,

The delimiter used in the file. One character is allowed except for "Tab" where you would enter \t

Encoding *

UTF-8

The encoding used in the file.

Back Save import configuration Close

11. Click **Save Import Configuration**.

12. Click **Create predefined Insight structure** and wait for the task to finish.

13. Click **Create predefined configuration** and wait for the task to finish.

Create Insight structure or Import configuration?

Insight can help you create the insight structure or import configuration to quickly get started importing.

Create predefined structure



You can create a template structure in the current object schema to get started in no time. Insight will try to create the structure depending on the import module.

Object Type*

None

The object type where the structure will be created

Create predefined Insight structure

Create predefined configuration



You can create a template configuration to get started in no time. Insight will look into the object schema to see if any object types are suitable for this import module.

Object Type*

None

The object type where the configuration will be created

Create predefined configuration

⚠ Make sure you have a valid Insight structure (predefined or created manually) before creating the predefined configuration.

Close

14. Close the window. You will see a new entry with your configured import on the page.
Click the settings cog and select **Execute Import**.



15. Click **Import**.

Confirm importing



Confirm that you want to execute import

Confirm that you want to execute import with **Import CSV** configuration which will affect **1** object types. Any already imported objects will be ignored.

Import

Cancel

16. Your import was successful. Close the window.

Finished process in action



Successfully imported objects.

Import took **20** second(s).

Executed as user  **Chris Kirsch**

Execution type **MANUAL**

Assets (#50)

Entries in external source	532

[Close](#)

If you are seeing the following error, one of your fields is longer than 255 characters. Manually trim the field to 255 characters or delete the field to successfully import your assets.

Finished process in action



Error occurred during import.

Import took 1 second(s).

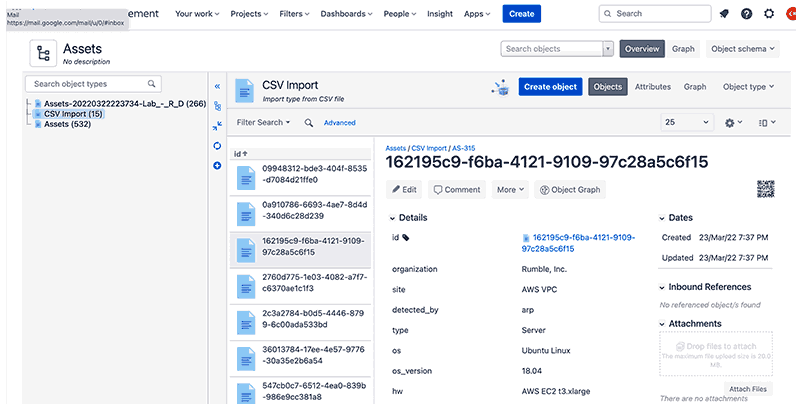
Error messages:**Error(s) for object type: Assets**

- Error: Text attribute (id: 802, name: macs) value must be less than 255 characters long. [Object (Label: 461193c4-2690-466a-8c20-3e30c4fb5777, Id: Unknown)].
- Error: Text attribute (id: 802, name: macs) value must be less than 255 characters long. [Object (Label: d7a87e93-0bf3-43b1-a97d-096b7e5aa235, Id: Unknown)].
- Error: Text attribute (id: 802, name: macs) value must be less than 255 characters long; Text attribute (id: 798, name: addresses_extra) value must be less than 255 characters long. [Object (Label: 32432fa8-46d8-42bc-b858-ecef2df2ed54, Id: Unknown)].
- Error: Text attribute (id: 802, name: macs) value must be less than 255 characters long. [Object (Label: b5c64020-4e68-4c8d-a25d-c8db28b81472, Id: Unknown)].
- Error: Text attribute (id: 804, name: names) value must be less than 255 characters long. [Object (Label: 369d2852-b7b3-4125-8ffa-89851cb8b1e4, Id: Unknown)].
- Error: Text attribute (id: 802, name: macs) value must be less than 255 characters long; Text attribute (id: 798, name: addresses_extra) value

[Close](#)

Step 3: View your imported assets

1. In Jira Service Management, click on the Insights.
2. Click on your object schema.
3. Click on assets to view details.



Jira

The Jira integration connects runZero to a Jira Cloud project for issue-tracking and service-desk operations. Once configured, you can push a runZero [issue](#) out to Jira as a ticket and keep the runZero issue timeline in sync with changes made on the Jira side.

If you are looking for the asset-enrichment integration that imports runZero asset data into Jira Service Management's CMDB, see [Atlassian Insight & Jira Service Management](#) instead. That integration is independent of the issue-tracking sync described here.

How the integration works

- runZero authenticates to Jira Cloud as a regular Atlassian user using HTTP basic auth (email + API token).
- When you create a Jira ticket from a runZero issue, runZero calls the Jira REST API (v3) to create the issue in the project configured on the credential, copies the relevant fields across, and stores the resulting Jira key and link on the runZero issue.
- A configurable Jira webhook posts changelog events back to runZero, where each change becomes a timeline entry on the matching runZero issue. If [state mapping](#) is configured, status and due-date changes also update the runZero issue directly.
- When you change a runZero issue state, runZero calls the Jira Transitions API to move the linked ticket to the corresponding Jira status (if a state mapping is configured).
- When a linked runZero issue has an owner, runZero synchronizes that owner with the Jira assignee when it can match the owner to an external Jira user. Changes to the Jira assignee are synchronized back to the runZero issue.
- All traffic is initiated by runZero (outbound to Jira) or signed and verified with HMAC-SHA256 (inbound from Jira). No other Jira data is written to or read from your

environment.

Prerequisites

Before you start you need:

- An Atlassian Cloud account that can create issues in the target Jira project.
- A Jira project to write tickets to, and that project's **Project key** (for example SCRUM, SEC, or OPS).
- A Jira API token. Create one at <https://id.atlassian.com/manage-profile/security/api-tokens> and copy it somewhere safe — it cannot be viewed again after the dialog is closed.
- A randomly generated **webhook secret** if you plan to enable two-way sync. Any reasonably long random string will do; runZero uses it to verify inbound webhook requests.
- The Jira integration entitlement on your runZero account.

Creating the Jira credential

The Jira integration uses a runZero [credential](#) of type **Jira API Token**. You can create one ahead of time from the Credentials page, or directly from the **Create Jira ticket** dialog the first time you push an issue.

To create the credential ahead of time:

1. Go to [Credentials](#) in the runZero Console.
2. Click **New credential** and choose **Jira API Token**.
3. Fill in the form:
 - **URL** — the base URL of your Jira Cloud site, for example `https://yourcompany.atlassian.net`. Do not include a trailing path.
 - **Email** — the email address of the Atlassian user the API token belongs to. This account must be able to create and transition issues in the target project.
 - **API token** — the API token you created on the Atlassian side.
 - **Webhook secret** — a random string used to authenticate inbound Jira webhooks. Save the value somewhere you can paste it into Jira later.
 - **Jira Project Key** — the key of the Jira project where tickets will be created (for example SEC or OPS). This is required; ticket creation will fail if the field is left empty.
 - **Map Issue States to Jira Values** — optional. A table mapping each runZero issue state to the corresponding Jira status name. See [State mapping and bidirectional sync](#) below for details.
4. Save the credential. runZero validates the URL, email, and token by making an authenticated call to Jira and rejects the credential if the call fails. The validation result is shown next to the credential.

After the credential is saved, open it to edit and you will see a read-only **Jira webhook URL** field. Copy this URL and use it when [configuring the Jira webhook](#).

A single credential can be reused for as many Jira tickets as you like, but if you change the API token in Atlassian you must update the credential in runZero or future calls will fail.

Creating a Jira ticket from an issue

1. Open an issue from the [Issues page](#).
2. In the sidebar,
 - **Credential** — select the desired Jira credential.
 - **Attach snapshot to ticket** — optional. When enabled, runZero uploads a CSV of the issue's instance snapshot as an attachment on the new Jira ticket.
 - Click **Create ticket**.

runZero will:

- Create the Jira issue using the project key from the credential, the summary, description, and the runZero issue's due date if one is set.
- Optionally upload the snapshot CSV as a Jira attachment. CSV upload failures do not abort ticket creation; the ticket is still linked.
- Save the Jira issue key (for example SEC-42), a browse link (`{your-jira-url}/browse/SEC-42`), and the credential and Jira base URL on the runZero issue.
- Append a Created Jira ticket SEC-42 for issue "... " entry to the runZero issue timeline.

Once linked, the issue sidebar shows the Jira key and a button that opens the ticket in Jira.

A runZero issue can be linked to at most one Jira ticket at a time. If you need to relink, [unlink](#) the existing one first.

Unlinking a ticket

From a runZero issue with a linked ticket, use the unlink action in the sidebar to clear the link. Unlinking only affects runZero — the Jira ticket itself is not deleted or modified. After unlinking you can create a new ticket from the same issue if needed.

State mapping and bidirectional sync

The **Map Issue States to Jira Values** table on the Jira credential enables two-way state and due date synchronization between runZero issues and Jira tickets.

To configure state mapping, open the credential and add one row per runZero state you want to sync:

- **Issue State** (key) — one of the four runZero issue states: open, in progress, closed, or resolved.
- **Jira Value** (value) — the exact name of the corresponding Jira status in your project workflow, for example To Do, In Progress, or Done. Matching is case-insensitive for inbound webhook events.

You can map all four states or only a subset. Unmapped states are ignored in both directions.

Outbound sync (runZero → Jira)

When you change a runZero issue state (or when runZero automatically transitions an issue to `resolved`), runZero looks up the new state in the credential's state mapping. If a Jira status name is configured for that state, runZero calls the [Jira Transitions API](#) to move the linked ticket to the target status.

If the Jira workflow does not have an available transition to that status from the ticket's current status, the transition silently fails. The runZero state change is always applied regardless of whether the Jira transition succeeds.

Inbound sync (Jira → runZero)

When Jira sends a webhook with a status change, runZero reverses the state mapping to find the matching runZero state and updates the issue accordingly. Due date changes on the Jira ticket are also propagated back to the runZero issue.

Issue owner synchronization

Issue owners are synchronized with Jira assignees for linked tickets. When an issue is pushed to Jira, runZero attempts to find an external Jira user that matches the runZero owner. For the runZero issue owner you can enter either the email address or username associated with the Jira user. If a matching Jira user is found, runZero assigns that user to the Jira ticket.

When the Jira assignee changes, the Jira webhook updates the owner of the linked runZero issue. For Jira sync, runZero does not allow an owner to be added if it does not match an external Jira user.

Issue ownership uses the built-in `Issue owner` type and supports one owner per issue; custom ownership types are not synchronized. See [Managing Issue ownership](#) and [Issue ownership](#).

NOTE: The casing of the Jira status needs to match the casing set in the Jira workflow

Example mapping:

runZero state	Jira status
Open	To Do
In progress	In Progress
Closed	Closed
Resolved	Done

Configuring the Jira webhook

To enable inbound sync from Jira back to runZero, configure a Jira webhook for the project (or the whole site):

1. In Jira, go to **System** → **WebHooks** and click **Create a WebHook** (administrator access required).
2. **Name** — anything descriptive, for example `runZero issue sync`.
3. **URL** — open your runZero Jira credential and copy the read-only **Jira webhook URL** field. It follows the pattern `https://<your-runzero-console>/api/webhooks/jira/${issue.key}`. The literal text `${issue.key}` is a Jira URL variable; Jira substitutes it with the actual issue key for each event.
4. **Secret** — paste the same webhook secret you saved on the runZero credential. Jira will sign each request with HMAC-SHA256 and send the signature in the `X-Hub-Signature` header.
5. **Events** — enable at least the **issue updated** event. You can add more events; runZero records every changelog item it receives.
6. (Optional) **JQL filter** — restrict the webhook to the project used with this credential, for example `project = SEC`. This prevents unnecessary events from other projects.
7. Save the webhook.

When an event fires, runZero:

1. Looks up the matching runZero issue by Jira issue key + Jira base URL (so multiple Jira sites do not collide).
2. Validates the HMAC-SHA256 signature against the credential's webhook secret using a constant-time comparison.
3. If [state mapping](#) is configured, applies any status or due-date changes to the runZero issue.
4. Appends one timeline line per changelog item. The line uses the form:

```
Jira update (SEC-42): status changed from "In Progress" to "Done"; assignee set to "alice@example.com"
```

If a webhook arrives without a valid signature, for an unknown Jira key, or for a runZero issue that has been deleted, runZero silently returns `200 OK` without writing anything — this prevents the endpoint from leaking which keys are linked.

What gets sent to Jira

At ticket creation time, runZero sends:

- The runZero issue **name** as the Jira **summary**.
- The runZero issue **description** as the Jira **description**.
- The runZero **due date**, if one is set, as the Jira **due date**.
- The **project key** from the credential, used to route the ticket to the right project.
- Optionally, a CSV file containing one row per (vulnerability, asset) instance from the issue snapshot, uploaded as a Jira attachment.

After ticket creation, ongoing sync depends on your [state mapping](#) configuration:

- **runZero → Jira:** state transitions are pushed to Jira via the Transitions API when a runZero issue state changes and a mapping is configured.
- **Jira → runZero:** status and due date changes arrive via the Jira webhook and are applied to the runZero issue (state mapping required for status changes).

runZero does not push comments, description edits, or other field changes to Jira after the initial ticket creation.

Troubleshooting

- **Credential verification fails when saving.** Double-check the URL has no trailing path, the email matches the Atlassian account that owns the API token, and the token has not been revoked. The credential page shows the specific HTTP status returned by Jira.
- **Create Jira ticket returns an error.** The runZero issue must have a non-empty name and description. The selected credential must have a **Jira Project Key** configured; if the field is empty the request will fail with a *project_key is required* error. The project key must exist in Jira and the API user must be allowed to create issues in it. The error message echoes the HTTP status returned by Jira.
- **Snapshot CSV is missing from the Jira ticket.** Snapshot upload is best-effort — the ticket is still created if the upload fails. Check that the API user has permission to add attachments to the project.
- **Jira changes do not appear in the runZero timeline.**
 - Confirm the webhook URL contains the literal `${issue.key}` variable.
 - Confirm the webhook secret in Jira matches the value stored on the runZero credential.
 - Confirm the `X-Hub-Signature` header is being sent by Jira (some proxies strip it).
 - Confirm the runZero issue's stored Jira base URL matches the Jira site sending the webhook. If you migrate Jira sites, unlink and recreate the ticket.
- **State changes are not being pushed to Jira.** Confirm the credential has a state mapping entry for the runZero state that changed, and that the Jira status name matches the exact name in your Jira workflow. The API user must also have permission to transition issues in the project. If the target Jira status is not reachable from the ticket's current status via an available Jira workflow transition, the push will silently fail.
- **Jira status changes are not updating the runZero issue state.** Confirm the Jira status name shown in the webhook event matches the value in the credential's state mapping (matching is case-insensitive). Confirm the webhook secret is correct and that the `X-Hub-Signature` header is reaching runZero.

Panther

runZero data can be imported into your [Panther](#) instance for enhanced logging and alerting.

Requirements

- A Panther account with the required permissions,

- An AWS S3 bucket, and
- Exported .jsonl files from runZero that have been uploaded into your AWS S3 bucket.

Step 1: Adding a custom schema

1. Go to **Configure > Schemas** and select **Create New**.
2. Add a name.
3. Upload a sample log to automatically parse the runZero output schema.

Step 2: Adding a custom log source

1. Go to **Configure > Log Sources** and select **Create New**.
2. Complete the **Basic Information** section.
3. Opt to configure S3 prefixes and schemas now and select the custom schema you created.
4. Configure the IAM role:
 - Opt to configure **Using the AWS Console UI**.
 - Click **Launch Console UI**.
 - Review the stack in AWS, then check the box to approve, and click to deploy the stack.
 - When the deployment completes, navigate to the **Resources** tab and select the **LogProcessingRole** that was created.
 - Copy the ARN from that role into the field on the Panther console.
5. Configure an alarm if logs are not processed (optional).

Once completed, any .jsonl files added to the specified AWS S3 bucket will be automatically ingested and processed by Panther.

SecurityGate.io

All runZero editions integrate with SecurityGate.io to enrich asset visibility in support of your risk assessment program. Setting up the integration requires a few steps in your SecurityGate.io console.

Requirements

- Configuring the SecurityGate.io integration requires a [runZero API key](#).



The SecurityGate.io integration will pull runZero asset data from across all organizations.

Integrate runZero with SecurityGate.io

1. Sign in to your SecurityGate.io console.
2. Go to **My Account** under the **Hello** dropdown menu.

3. Click on **Integration Manager**.
4. Select **Add New Integration**.
5. Choose *Rumble* from the **Integration Partner** dropdown menu.
6. Provide the **Account API Key**, **Server URL**, and **API Version**.
7. Click **Test Connection**. If the test is successful, click **OK** to save the configuration.

Viewing runZero data in SecurityGate.io

After the integration is enabled within SecurityGate.io, runZero data will be available through the Asset Inventory page.

ServiceNow Service Graph

The Service Graph connector for runZero allows you to bring runZero assets into your ServiceNow CMDB as CIs, and optionally periodically update the CIs with fresh information from runZero scans.

The Service Graph Connector fetches and transforms data using ServiceNow IntegrationHub ETL, and passes it through the Identification and Reconciliation Engine (IRE). This allows specific fields and CI class mappings to be fine-tuned from the ServiceNow console. You can also specify a runZero search query to determine which assets get brought in by the connector.

Important notes

- There is no charge from runZero for use of the connector. However, CI resources created in CMDB by the connector will increase ServiceNow Subscription Unit consumption. CIs created by Service Graph Connectors should be charged at a lower rate than CIs created via other means; you will need to consult ServiceNow to confirm this and obtain current pricing.
- If you are a self-hosted runZero customer, you will need to set up a MID Server to enable ServiceNow to connect to your runZero console. If your console uses a self-signed TLS certificate, or a TLS certificate signed by your own internal CA, you will need to configure the MID server to accept the CA or certificate as trusted. For more information, see [ServiceNow Knowledge Base article KB0863673](#).
- While the Service Graph Connector is packaged and available from the ServiceNow Application Store, setting up connectors to integrate with your existing ServiceNow configuration is unlikely to be a one-click operation. You are strongly advised to have a ServiceNow Consulting and Implementation partner available to assist with the process.
- You can find the Scoped Application Installation and Configuration Guide on the [Service Graph Connector for runZero](#) page under Supporting Links and Docs. This has detailed installation instructions with screenshots.

Prerequisites

To use the Service Graph connector for runZero, you need the following:

- A Platform license for runZero.
- A ServiceNow ITOM license including ITOM Discovery and ITOM Visibility.
- Integration Commons for CMDB installed.

The ServiceNow dependencies are checked as part of the connector installation process.

Installation

The first step of the installation is to locate the Service Graph connector for runZero in the ServiceNow Application Store, and follow the usual process to install it into your ServiceNow instance.

The Service Graph connector adds a new menu entry **Service Graph connector for runZero** to the ServiceNow main menu. The module contains entries for **Setup**, **Data Sources**, **Scheduled Imports**, **Support**, and **System Import Sets**. You must have the **admin** role in ServiceNow to configure the connector.

Setup

The **Setup** menu offers a **Guided Setup** process to help you get up and running with the connector. There are three main stages to the setup process:

1. Configure the connection
2. Configure the mappings (optional)
3. Configure the scheduled import (optional)

Step 1: Configure the connection

Configuring the connection consists of two steps: entering a runZero API key as credentials, and configuring connection options.

Entering the API key

The connector can use a runZero **Organization API** or **Export API** key. In either case, you can obtain the key by going to the [Organizations](#) page in runZero and clicking on the organization containing the assets you want to bring into ServiceNow CMDB. Organization tokens begin with OT, export tokens begin with ET.

In the **API Key Credentials** form, paste the runZero token into the field labeled **API Key**, and click **Update**.

Configuring connection options

For runZero cloud users, the default **HTTP connection** options should be correct.

You can optionally add a **Search** string to the **Attributes** for the connection at the bottom of the form. If you do, this search will be passed to runZero to determine the set of assets to bring in to CMDB via the connector. Any search string should be in runZero [search query format](#). You are strongly encouraged to test your search string in your [runZero asset inventory](#).

The **Connection** alias and **Base path** fields should be left as-is. (The connection alias groups together the connection settings and the API key credentials for the connector to use.)

If you are self-hosting runZero, the **connection options** form is where you configure the MID server to use and specify the URL of your runZero console. Your console's fully-qualified hostname should be placed in the **Host** field. The hostname should match the hostname in the console's TLS certificate in order for the MID server to trust it, even if the TLS certificate is trusted by the MID server. Check the **Use MID server** box and select the MID server to use.

Step 2: Configure the mappings (optional)

The second step of setting up the connector allows you to customize the mapping of runZero data to CI classes, using the IntegrationHub ETL Transform Map Assistant.

This stage can be skipped if you do not have special requirements for data mapping.

Step 3: Configure the Scheduled Import (optional)

The third step of setting up the connector is to set up a schedule for data import. Initially, you will want to test the connector setup with a one-off import. Once everything is configured correctly, you can create recurring imports.

Scheduled imports are configured as regular ServiceNow tasks. This step can be skipped if you don't want to set up a scheduled import at this time.

Notes on data mapping

The object data models for ServiceNow CMDB and runZero are not an exact match. Some runZero data does not fit in any standard attributes available in CMDB, and in other cases runZero does not have data that CMDB expects. Review this list to understand how the connector will map runZero attributes to ServiceNow CMDB attributes:

- **IP addresses:** ServiceNow CIs have an attribute which takes a single IP address. The connector places the first IP address in the CI attribute IP Address. The remaining addresses are created as `IP Address` CIs owned by the device CI. Each `IP Address` CI is given a `Description` indicating whether it has been scanned by runZero or not. **Note:** By default, ServiceNow does not show IP addresses owned by a device CI. It only shows IP addresses owned by a network adapter CI which is owned by the device CI. This is a known limitation of ServiceNow's default CMDB forms. runZero doesn't associate IP

addresses with network adapters because the necessary information isn't generally known. In that situation, the CMDB connector development guide states that the IP addresses should be associated with the device CI.

- **Names:** ServiceNow CIs have a single `Name` attribute, while runZero assets can have any number of names with runZero attempting to guess which are the best names and put them at the front of the list. The connector places the first name in the `CI Name` attribute. The first name which looks like a fully-qualified domain name is placed in the `Fully qualified domain name` attribute, and is used to compute a DNS Domain for CIs descended from Network Gear. The full list of names is placed in the `Description` attribute for reference.
- **Serial numbers:** ServiceNow CIs have an attribute which takes a single serial number. The connector picks the first serial number for the `Serial Number` attribute, and then also performs a Serial Number Lookup operation to add the complete list of serial numbers with their associated type.
- **Tags:** runZero assets can have any number of tags, which are either single tags such as `tag_name`, or tags with values in the format `tag=value`. These are placed in the `Key Value` class, and this class is then associated with the `Hardware Class`.
- **MAC addresses/Network Adapters:** As with IP addresses, ServiceNow has both a single-value `MAC Address` attribute, and an option to look up named MAC addresses to create Network Adapter objects. A runZero asset can have any number of MAC addresses, but since runZero does not run agent software on the scanned systems, it has no way to know the interface name associated with a given MAC address. The first MAC address is imported as the `MAC Address` attribute, and the entire set is imported via lookup as `Network Adapter` objects, with each being given a name made up by the MAC address and IP address (e.g. `00:00:00:00:00:00-127.0.0.1`) or the MAC address and hostname if the IP address is not available (e.g. `00:00:00:00:00:00-hostname`). If neither the IP address or hostname are available, the MAC address alone is used as the network adapter name. In some cases, runZero may not know the MAC address(es) of an asset. In this case, a zero MAC address (`00:00:00:00:00:00`) is added to satisfy ServiceNow requirements for some CI classes.
- **Sites:** runZero sites represent distinct networks, which may or may not correspond to physical sites. They are imported as `CMDB Network Site` objects.
- **Organizations:** The runZero organization has no obvious corresponding attribute in network hardware CIs. The organization name is incorporated into the `Description` of CIs.
- **SNMP data:** The `snmp.sysDesc` attribute from runZero, if it exists, is also placed in the `Description`.

Splunk Search

runZero integrates with Splunk using a dedicated [Splunk Addon](#), compatible with Splunk 7, Splunk 8, and Splunk Cloud. With this add-on, you'll be able to pull new or updated hosts into a Splunk index, where you'll be able to analyze, visualize, and monitor them there.

This add-on uses the [Splunk API](#) from the runZero Network Discovery platform. It supports syncing assets into Splunk, with multiple inputs supported, global API key management, and optional search filters for each input. For example, you can track new assets as one input, and SMBv1 enabled assets as another input.

To set up this add-on, you'll need an Export API or Organization API key, which you can generate from your Organization page in the runZero Console.

Get the runZero add-on for Splunk

1. Sign in to Splunk.
2. Go to **Find More Apps**.
3. Search for `runZero Network Discovery`.
4. Install the add-on for runZero.
5. Splunk will prompt you to sign in again. After you log back in again, the add-on will be installed. You'll be able to open the runZero Asset Sync app. Splunk might also prompt you to restart your server.

Asset sync modes

Two asset sync modes are available: New Assets Only and All Updated Assets. You can export asset inventory that contains newly discovered assets or updated assets, since the last poll, in a sync-friendly format for Splunk. You can leverage the same capabilities from the Asset Sync API to pull data in Splunk, such as search filters, fields, and time-based checkpoints.

Once data is pulled into Splunk, you can create Splunk inputs with filters. This allows you to sync specific assets with a certain protocol, discovery date, or open service.

Sumo Logic

All runZero editions integrate with Sumo Logic to enrich asset visibility and help you visualize your asset data. Setting up the integration requires a few steps in your Sumo Logic console. The integration can be set up to support two distinct purposes:

- [Complete asset visibility](#)
- [Targeted alerting and visualization](#)

Requirements

- A Sumo Logic account
- A runZero account and [API key](#).

Sumo Logic asset export

runZero integrates with Sumo Logic to make your asset inventory available directly in Sumo Logic. This article will show you how to export your runZero inventory into Sumo Logic for use within the SIEM.

Integrating runZero with Sumo Logic

Setting up the connection between Sumo Logic and runZero has three options with different configuration steps.

Option A: Local script

1. Create a Sumo Logic HTTP Source.
2. Configure your host to run the provided script.

Option B: AWS Lambda function

1. Create a Sumo Logic HTTP Source.
2. Configure the AWS Lambda function to run the provided script.

Option C: Sumo Logic script source

1. Install a Sumo Logic collector.
2. Create a Sumo Logic script source.

Once your data is flowing into Sumo Logic, you can [start using](#) the data in Sumo Logic.

Option A: Local script**Step 1: Create a Sumo Logic HTTP Source**

1. After logging in to Sumo Logic, navigate to **Manage Data > Collection**.
2. Click **Add Collector** then select **Hosted Collector**.
 - Provide a name, such as runZero Collector and click **Save**.
3. If prompted to add a data source, click **OK**. Otherwise, find your Collector in the list and click **Add Source**.
4. Select the **HTTP Logs and Metrics** source.
 - Provide a name, such as runZero Assets, then click **Save**.
5. Copy the URL provided to use in [step 2](#).

Step 2: Configure your host to run the provided script

1. Identify the host you would like to run the script from.
2. Ensure the host has Python3 and Pipenv installed.
3. Save the script below to the host it will be run from.

```
#!/usr/bin/env python3
import json
import requests
import os

# RUNZERO CONF
RUNZERO_EXPORT_TOKEN = os.environ["RUNZERO_EXPORT_TOKEN"]
HEADERS = {"Authorization": f"Bearer {RUNZERO_EXPORT_TOKEN}"}
BASE_URL = "https://console.runZero.com/api/v1.0"

# SUMO LOGIC CONF
HTTP_ENDPOINT = os.environ["SUMO_HTTP_ENDPOINT"]
```

```
def main():
    url = BASE_URL + "/export/org/assets.json"
    assets = requests.get(url, headers=HEADERS)
    batchsize = 500
    if len(assets.json()) > 0 and assets.status_code == 200:
        for i in range(0, len(assets.json()), batchsize):
            batch = assets.json()[i:i+batchsize]
            f = open("upload.txt", "w")
            f.truncate(0)
            for a in batch:
                json.dump(a, f)
                f.write("\n")
            f.close()
            r = open("upload.txt")
            requests.post(HTTP_ENDPOINT, data=r.read())
            r.close()
    else:
        print(f"No assets found - status code from runZero API: {assets.status_code}")

if __name__ == "__main__":
    main()
```

4. Create your environment variables by running the following commands:

- export RUNZERO_EXPORT_TOKEN=XXX: Use your runZero export API token, which can be obtained in your runZero console on an [organization detail page](#). Select the organization you wish to export data from, then click **Edit organization** to view the export API token.
- export SUMO_HTTP_ENDPOINT=XXX: Use the Sumo Logic token obtained in [step 1](#).

5. Create your virtual environment to run the script by running `pipenv --python /path/to/python3`.

6. Install the `requests` library in your virtual environment for making API calls:

- `pipenv shell`
- `pip install requests`

7. Test the script by running your script from the virtual environment.

- Use the location from the `pipenv` output to start.
- Append `/bin/python3` to use Python in the virtual environment.
- Use the full path to the script.

```
my-server:~/ $ /home/user/.local/share/virtualenvs/runZero-scripts-mVQtFLD0/bin/python3 \
/home/user/scripts/script.py
```

8. Configure a crontab task to run at the desired cadence.

- On the hour: `0 * * * * RUNZERO_EXPORT_TOKEN=XXX SUMO_HTTP_ENDPOINT=XXX /path/to/virtual/env/python3 /path/to/script.py`
- Daily at midnight: `0 0 * * * RUNZERO_EXPORT_TOKEN=XXX SUMO_HTTP_ENDPOINT=XXX /path/to/virtual/env/python3 /path/to/script.py`

- Weekly at midnight on Monday: `0 0 * * 1 RUNZERO_EXPORT_TOKEN=XXX SUMO_HTTP_ENDPOINT=XXX /path/to/virtual/env/python3 /path/to/script.py`

Option B: AWS Lambda function

Step 1: Create a Sumo Logic HTTP Source

1. After logging in to Sumo Logic, go to **Manage Data > Collection**.
2. Click **Add Collector** then select **Hosted Collector**.
 - Provide a name, such as `runZero Collector` and click **Save**.
3. If prompted to add a data source, click **OK**. Otherwise, find your Collector in the list and click **Add Source**.
4. Select the **HTTP Logs and Metrics** source.
 - Provide a name, such as `runZero Assets`, then click **Save**.
5. Copy the URL provided to use in [step 2](#).

Step 2: Configuring the AWS Lambda function to run the provided script

1. Go to your AWS Console and navigate to the [Lambda page](#).
2. Click **Create a function**.
3. Give your function a **name**.
4. Select *Python 3.9* as the **runtime**.
5. Everything else can be left with the default setting. Click **Create function** to move to the next page.
6. Click **Add Trigger** to set up a cron job.
7. Select *EventBridge* to set up a schedule.
8. Use an existing rule or select **Create new rule**.
 - Give it a name and set **Rule type** to *Schedule expression*.
 - Use one of these options or create your own based on desired cadence:
 - Daily: `rate(1 day)`
 - Every 12 hours: `rate(12 hours)`
 - Every 3 hours: `rate(3 hours)`
 - Click **Add** to return to the main Lambda configuration page.
9. Under **Configuration** select *Environment variables*.
10. Enter these two environment variables:
 - `RUNZERO_EXPORT_TOKEN` which can be obtained in your runZero console on an [organization detail page](#). Select the organization you wish to export data from, then click **Edit organization** to view the export API token.
 - `SUMO_HTTP_ENDPOINT` which was obtained in [step 1](#).
11. Click **Save** to return to the main Lambda configuration page.

12. Click the *Code* tab and replace the default code with this script.

```
import json
import urllib3
import os

# RUNZERO CONF
RUNZERO_EXPORT_TOKEN = os.environ["RUNZERO_EXPORT_TOKEN"]
HEADERS = {"Authorization": f"Bearer {RUNZERO_EXPORT_TOKEN}"}
BASE_URL = "https://console.runZero.com/api/v1.0"

# SUMO LOGIC CONF
HTTP_ENDPOINT = os.environ["SUMO_HTTP_ENDPOINT"]

def lambda_handler(event, context):
    http = urllib3.PoolManager()
    url = BASE_URL + "/export/org/assets.json"
    response = http.request("GET", url, headers=HEADERS)
    data = response.data
    assets = json.loads(data)
    batchsize = 500
    if len(assets.json()) > 0 and assets.status_code == 200:
        for i in range(0, len(assets.json()), batchsize):
            batch = assets.json()[i : i + batchsize]
            f = open("upload.txt", "w")
            f.truncate(0)
            for a in batch:
                json.dump(a, f)
                f.write("\n")
            f.close()
            r = open("upload.txt")
            http.request("POST", HTTP_ENDPOINT, data=r.read())
            r.close()
    else:
        print(f"No assets found - status code from runZero API: {assets.status_code}")
```

13. Click **Deploy** to update the code.

14. Click **Test** to verify the code works.

Your asset data export will now be posted to Sumo Logic at the cadence you configured.

Option C: Sumo Logic script source

Step 1: Installing a Sumo Logic collector

Follow the [Sumo Logic documentation](#) in order to install a collector.

Step 2: Creating a Sumo Logic script source

Sumo Logic has documentation on [script sources](#) as well. Here are the steps to follow to set up the script source once your collector is installed.

1. Navigate to the [Collection](#) page in Sumo Logic.
2. Find your collector and click **Add > Add Source**.
3. Select `Script` as the **source type**.
4. Input a Name and Source Category.
5. Select a Frequency.
6. Select **Command** type `/usr/bin/python`.
7. Add the following script in the **Script** field.

```
#!/usr/bin/python
import json
import requests
import os

# RUNZERO CONF
RUNZERO_EXPORT_TOKEN = os.environ['RUNZERO_EXPORT_TOKEN']
HEADERS = {'Authorization': 'Bearer ' + RUNZERO_EXPORT_TOKEN}
BASE_URL = 'https://console.runZero.com/api/v1.0'

def main():
    url = BASE_URL + '/export/org/assets.json'
    assets = requests.get(url, headers=HEADERS)
    if assets.status_code == 200:
        for a in assets.json():
            print(json.dumps(a))
    else:
        print(f"No assets found - status code from runZero API: {assets.status_code}")

if __name__ == '__main__':
    main()
```

8. Click **Save** to allow the source to start working.

Working with the asset data in Sumo Logic

Once your asset data is in Sumo Logic, you can use it in any way you would use any other log source. Here are some sample searches that you could use to create scheduled searches and dashboards.

Search distinct assets

```
_sourceCategory="runzero"
| json field=_raw "id"
| count_distinct(id) as distinct_assets
```

Search assets with more than 3 services running

```
_sourceCategory="runzero"
| json field=_raw "addresses_extra"
| json field=_raw "addresses"
| json field=_raw "id"
| concat("https://console.runzero.com/inventory/", id) as runzero_link
| json field=_raw "service_count"
| where service_count > 3
| count addresses, addresses_extra, service_count, runzero_link
```

Determine counts of different operating systems

```
_sourceCategory="runzero"
| json field=_raw "os"
| where !isEmpty(os)
| json field=_raw "id"
| count os, id
| count os
```

Sumo Logic alerting

runZero integrates with Sumo Logic to help you visualize your asset data. This helps you track your progress on reducing risk in your asset inventory over time.

Setting up the connection between Sumo Logic and runZero requires:

1. [Creating a Sumo Logic HTTP Source](#)
2. [Creating a runZero alert template](#)
3. [Creating a rule in runZero](#)
4. [Handling runZero data in Sumo Logic](#)
5. [Creating a Sumo Logic dashboard \(optional\)](#)

Step 1: Create a Sumo Logic HTTP Source

1. After logging in to Sumo Logic, navigate to **Manage Data > Collection**.
2. Click **Add Collector** select **Hosted Collector**, provide a name, such as runZero Collector and click **save**.
3. If prompted to add a data source, click **OK**. Otherwise, find your Collector in the list and click **Add Source**.
4. Select the **HTTP Logs and Metrics** source, provide a name, such as runZero Alerts, and then click **save**.
5. Copy the URL provided to use in [step 2](#).

Step 2: Create a runZero alert template

1. Create an [alert template](#) in runZero and provide the following details:
 - **Name:** Name for template
 - **Template type:** JSON
 - **Subject line for message:** Leave empty
 - **Body of message:** The following JSON example will include the rule name and the search URL in the alert message body

```
{ "rule_name": "{{rule.name}}", "search_url": "{{search.url}}", "found": "{{search.found}}",
  "assets_new": "{{scan.assets_new}}"
```

2. Create an [alert channel](#) in runZero and provide the following details:
 - **Name:** Name for alert channel
 - **Channel type:** Webhook
 - **Webhook URL:** The webhook URL you copied from Sumo Logic

Step 3: Create a rule in runZero

Now that you have your alert template and channel created, you will want to identify the triggers to alert on. Some common examples are:

- Asset query results - When there is a match on a query in runZero after a scan completes
- New assets found - When a scan completes with new assets
- Agent offline - When your runZero Explorer stops checking in to the console
- Task failed - When a task fails for any reason.

We will use the **asset query results** selection as an example for the rest of the steps. Review [example queries](#) for ideas on what queries you could create.

1. Create a [new alert rule](#).
2. Select `asset-query-results` and click **Configure rule**.
3. Input values for the rule:
 - **Name:** name of the rule.
 - **Conditions:** optional parameters that will trigger the alert when all conditions match.
 - **Query:** The query the assets must match.
 - **Number of matches:** The numeric comparison logic for the **value**.
 - **Value:** The threshold of matches to trigger the rule.
 - **Limit to organization:** Allows you to limit the alert to a specific organization.
 - **Limit to site:** Allows you to limit the alert to a specific site.
 - **Action:** Notify
 - **Notification channel:** Name of the alert channel you created in step 2.
 - **Notification template:** Name of the alert template you created in step 2.

Step 4: Handle runZero data in Sumo Logic

This search will display the raw runZero data

```
_source="runZero Alerts" and _collector="runZero"
```

This search will show alerts matching the runZero rule name

```
_source="runZero Alerts" and _collector="runZero"
| json field=_raw "found" nodrop
| json field=_raw "rule_name" nodrop
| where rule_name = "<RULE NAME>"
```

This search will create a graph of the data matching the runZero rule name

```
_source="runZero Alerts" and _collector="runZero"
| json field=_raw "assets_new" nodrop
| toLong(assets_new)
| json field=_raw "found" nodrop
| json field=_raw "search_url" nodrop
| json field=_raw "rule_name" nodrop
| where rule_name = "<RULE NAME>"
| timeslice 1m
| sum(found) by _timeslice
| order by _timeslice
```

Step 5: Create a dashboard in Sumo Logic (optional)

Now that you know how to look at the data in Sumo Logic and make a graph, you can follow these steps to create a dashboard. You will first create four rules in runZero, then you will import the [Sample Sumo Logic Dashboard](#) below.

Create the runZero rules

You will follow the same actions from step 3 to create each of these rules using the form inputs provided.

Assets running a TLS service

- **Rule type:** asset-query-results
- **Name:** Assets running a TLS service - Sumo
- **Query:** alive:t protocol:tls
- **Number of matches:** is greater than 0
- **Notification channel:** alert channel created in [Step 2](#)
- **Notification template:** alert template created in [Step 2](#)

Multihomed assets - Sumo

- **Rule type:** asset-query-results
- **Name:** Multihomed assets - Sumo

- **Query:** alive:t AND multi_home:t
- **Number of matches:** is greater than 0
- **Notification channel:** alert channel created in [Step 2](#)
- **Notification template:** alert template created in [Step 2](#)

Assets with OpenSSL - Sumo

- **Rule type:** asset-query-results
- **Name:** Assets with OpenSSL - Sumo
- **Query:** alive:t product:openssl
- **Number of matches:** is greater than 0
- **Notification channel:** alert channel created in [Step 2](#)
- **Notification template:** alert template created in [Step 2](#)

New assets to Sumo Logic

- **Rule type:** new-assets-found
- **Name:** New assets to Sumo Logic
- **Number of matches:** is greater than 0
- **Notification channel:** alert channel created in [Step 2](#)
- **Notification template:** alert template created in [Step 2](#)

Sample Sumo Logic Dashboard

1. Navigate to the library in Sumo Logic.
2. Click the **options** button on the folder you'd like to import to and click Import.
 - **Name:** runZero Alert Metrics
 - **JSON:** Copy from the sample below
3. Click Import to see this dashboard under the folder it was imported to.

```
{
  "type": "DashboardV2SyncDefinition",
  "name": "runZero Asset Metrics",
  "description": "",
  "title": "runZero Asset Metrics",
  "theme": "Dark",
  "topologyLabelMap": {
    "data": {}
  },
  "refreshInterval": 0,
  "timeRange": {
    "type": "BeginBoundedTimeRange",
    "from": {
      "type": "RelativeTimeRangeBoundary",
      "relativeTime": "-3d"
    },
    "to": null
  },
  "layout": {
    "layoutType": "Grid",
    "layoutStructures": [
```

```

"key": "panelPANE-AC8FB3DCBD32DA48",
  "structure": "{ \"height\":6, \"width\":12, \"x\":0, \"y\":0}"
},
{
  "key": "panel3D084A3284252A4E",
  "structure": "{ \"height\":6, \"width\":12, \"x\":12, \"y\":0}"
},
{
  "key": "panelPANE-4389DBF794B13B44",
  "structure": "{ \"height\":6, \"width\":12, \"x\":0, \"y\":6}"
},
{
  "key": "panelPANE-FBE08549B2123A4A",
  "structure": "{ \"height\":6, \"width\":12, \"x\":12, \"y\":6}"
}
]
},
"panels": [
  {
    "id": null,
    "key": "panelPANE-AC8FB3DCBD32DA48",
    "title": "New assets found",
    "visualSettings": "{ \"general\":{ \"mode\": \"timeSeries\", \"type\": \"line\", \"displayType\": \"default\", \"markerSize\":5, \"lineDashType\": \"solid\", \"markerType\": \"none\", \"lineThickness\":1}, \"title\":{ \"fontSize\":14}, \"axes\":{ \"axisX\":{ \"titleFontSize\":12, \"labelFontSize\":12}, \"axisY\":{ \"titleFontSize\":12, \"labelFontSize\":12, \"logarithmic\":false}}, \"legend\":{ \"enabled\":true, \"verticalAlign\": \"bottom\", \"fontSize\":12, \"maxHeight\":50, \"showAsTable\":false, \"wrap\":true}, \"color\":{ \"family\": \"Categorical Default\"}, \"series\":{ }, \"overrides\":[ ]}",
    "keepVisualSettingsConsistentWithParent": true,
    "panelType": "SumoSearchPanel",
    "queries": [
      {
        "transient": false,
        "queryString": "_source=\"runZero Alerts\" and _collector=\"runZero\"\\n| json field=_raw \"assets_new\" nodrop\\n| json field=_raw \"found\" nodrop\\n| json field=_raw \"search_url\" nodrop\\n| json field=_raw \"rule_name\" nodrop\\n| where rule_name = \"New Assets to Sumo Logic\"\\n| timeslice 1m\\n| sum(assets_new) by _timeslice\\n| order by _timeslice",
        "queryType": "Logs",
        "queryKey": "A",
        "metricsQueryMode": null,
        "metricsQueryData": null,
        "tracesQueryData": null,
        "spansQueryData": null,
        "parseMode": "Auto",
        "timeSource": "Message",
        "outputCardinalityLimit": 1000
      }
    ],
    "description": "",
    "timeRange": null,
    "coloringRules": null,
    "linkedDashboards": []
  },

```

```

"id": null,
  "key": "panel3D084A3284252A4E",
  "title": "Multihomed assets found",
  "visualSettings": "{\n  \"general\": {\n    \"mode\": \"timeSeries\",\n    \"type\": \"line\",\n    \"displayType\": \"default\",\n    \"markerSize\": 5,\n    \"lineDashType\": \"solid\",\n    \"markerType\": \"none\",\n    \"lineThickness\": 1,\n    \"title\": {\n      \"fontSize\": 14,\n      \"axes\": {\n        \"axisX\": {\n          \"titleFontSize\": 12,\n          \"labelFontSize\": 12,\n          \"axisY\": {\n            \"titleFontSize\": 12,\n            \"labelFontSize\": 12,\n            \"logarithmic\": false,\n            \"legend\": {\n              \"enabled\": true,\n              \"verticalAlign\": \"bottom\",\n              \"fontSize\": 12,\n              \"maxHeight\": 50,\n              \"showAsTable\": false,\n              \"wrap\": true,\n              \"color\": {\n                \"family\": \"Categorical Default\",\n                \"series\": {},\n                \"overrides\": []\n              }\n            }\n          }\n        }\n      }\n    }\n  },\n  \"keepVisualSettingsConsistentWithParent\": true,\n  \"panelType\": \"SumoSearchPanel\",\n  \"queries\": [\n    {\n      \"transient\": false,\n      \"queryString\": \"_source=\\\"runZero Alerts\\\" and _collector=\\\"runZero\\\"\\n| json field=_raw \\\"assets_new\\\" nodrop\\n| toLong(assets_new)\\n| json field=_raw \\\"found\\\" nodrop\\n| json field=_raw \\\"search_url\\\" nodrop\\n| json field=_raw \\\"rule_name\\\" nodrop\\n| where rule_name = \\\"Multihomed Assets\\\" or rule_name = \\\"Multihomed Assets - Sumo\\\"\\n| timeslice 1m\\n| sum(found) by _timeslice\\\" order by _timeslice\",\n      \"queryType\": \"Logs\",\n      \"queryKey\": \"A\",\n      \"metricsQueryMode\": null,\n      \"metricsQueryData\": null,\n      \"tracesQueryData\": null,\n      \"spansQueryData\": null,\n      \"parseMode\": \"Auto\",\n      \"timeSource\": \"Message\",\n      \"outputCardinalityLimit\": 1000\n    }\n  ],\n  \"description\": \"\",\n  \"timeRange\": null,\n  \"coloringRules\": null,\n  \"linkedDashboards\": []\n},\n{\n  \"id\": null,\n  \"key\": \"panelPANE-4389DBF794B13B44\",\n  \"title\": \"Assets with a TLS service\",\n  \"visualSettings\": \"{\\n  \\\"general\\\": {\\n    \\\"mode\\\": \\\"timeSeries\\\",\\n    \\\"type\\\": \\\"line\\\",\\n    \\\"displayType\\\": \\\"default\\\",\\n    \\\"markerSize\\\": 5,\\n    \\\"lineDashType\\\": \\\"solid\\\",\\n    \\\"markerType\\\": \\\"none\\\",\\n    \\\"lineThickness\\\": 1,\\n    \\\"title\\\": {\\n      \\\"fontSize\\\": 14,\\n      \\\"axes\\\": {\\n        \\\"axisX\\\": {\\n          \\\"titleFontSize\\\": 12,\\n          \\\"labelFontSize\\\": 12,\\n          \\\"axisY\\\": {\\n            \\\"titleFontSize\\\": 12,\\n            \\\"labelFontSize\\\": 12,\\n            \\\"logarithmic\\\": false,\\n            \\\"legend\\\": {\\n              \\\"enabled\\\": true,\\n              \\\"verticalAlign\\\": \\\"bottom\\\",\\n              \\\"fontSize\\\": 12,\\n              \\\"maxHeight\\\": 50,\\n              \\\"showAsTable\\\": false,\\n              \\\"wrap\\\": true,\\n              \\\"color\\\": {\\n                \\\"family\\\": \\\"Categorical Default\\\",\\n                \\\"series\\\": {},\\n                \\\"overrides\\\": []\n              }\n            }\n          }\n        }\n      }\n    }\n  },\n  \\\"keepVisualSettingsConsistentWithParent\\\": true,\n  \\\"panelType\\\": \\\"SumoSearchPanel\\\",\\n  \\\"queries\\\": [\n    {\n      \\\"transient\\\": false,\n      \\\"queryString\\\": \"_source=\\\"runZero Alerts\\\" and _collector=\\\"runZero\\\"\\n| json field=_raw \\\"assets_new\\\" nodrop\\n| toLong(assets_new)\\n| json field=_raw \\\"found\\\" nodrop\\n| json field=_raw \\\"search_url\\\" nodrop\\n| json field=_raw \\\"rule_name\\\" nodrop\\n| where rule_name = \\\"Assets running a TLS service\\\" or rule_name = \\\"Assets running a TLS service - Sumo\\\"\\n| timeslice 1m\\n| sum(found)

```

```

by _timeslice\n| order by _timeslice",
    "queryType": "Logs",
    "queryKey": "A",
    "metricsQueryMode": null,
    "metricsQueryData": null,
    "tracesQueryData": null,
    "spansQueryData": null,
    "parseMode": "Auto",
    "timeSource": "Message",
    "outputCardinalityLimit": 1000
  }
],
"description": "",
"timeRange": null,
"coloringRules": null,
"linkedDashboards": []
},
{
  "id": null,
  "key": "panelPANE-FBE08549B2123A4A",
  "title": "Assets running OpenSSL",
  "visualSettings": "{\n  \"general\": {\n    \"mode\": \"timeSeries\",\n    \"type\": \"line\",\n    \"displayType\": \"default\",\n    \"markerSize\": 5,\n    \"lineDashType\": \"solid\",\n    \"markerType\": \"none\",\n    \"lineThickness\": 1,\n    \"title\": {\n      \"fontSize\": 14,\n    },\n    \"axes\": {\n      \"axisX\": {\n        \"titleFontSize\": 12,\n        \"labelFontSize\": 12,\n      },\n      \"axisY\": {\n        \"titleFontSize\": 12,\n        \"labelFontSize\": 12,\n        \"logarithmic\": false,\n      },\n    },\n    \"legend\": {\n      \"enabled\": true,\n      \"verticalAlign\": \"bottom\",\n      \"fontSize\": 12,\n      \"maxHeight\": 50,\n      \"showAsTable\": false,\n      \"wrap\": true,\n    },\n    \"color\": {\n      \"family\": \"Categorical Default\",\n    },\n    \"series\": {},\n    \"overrides\": [],\n  },\n  \"keepVisualSettingsConsistentWithParent\": true,\n  \"panelType\": \"SumoSearchPanel\",\n  \"queries\": [\n    {\n      \"transient\": false,\n      \"queryString\": \"_source=\\\"runZero Alerts\\\" and _collector=\\\"runZero\\\"\\n| json field=_raw \\\"assets_new\\\" nodrop\\n| toLong(assets_new)\\n| json field=_raw \\\"found\\\" nodrop\\n| json field=_raw \\\"search_url\\\" nodrop\\n| json field=_raw \\\"rule_name\\\" nodrop\\n| where rule_name = \\\"Assets with OpenSSL\\\" or rule_name = \\\"Assets with OpenSSL - Sumo\\\"\\n| timeslice 1m\\n| sum(found) by _timeslice\\n| order by _timeslice\",\n      \"queryType\": \"Logs\",\n      \"queryKey\": \"A\",\n      \"metricsQueryMode\": null,\n      \"metricsQueryData\": null,\n      \"tracesQueryData\": null,\n      \"spansQueryData\": null,\n      \"parseMode\": \"Auto\",\n      \"timeSource\": \"Message\",\n      \"outputCardinalityLimit\": 1000\n    }\n  ],\n  \"description\": \"\",\n  \"timeRange\": null,\n  \"coloringRules\": null,\n  \"linkedDashboards\": []\n}
}

```

```
"variables": [],
  "coloringRules": []
}
```

Tines

runZero integrates with Tines to help you automate workflows related to your asset data. This helps teams leverage runZero to the fullest while optimizing the team's workflows with automation. A [video demo](#) is available to show the final outcome of these instructions.

Requirements

- A Tines account
- runZero Export API and Organization API tokens

There are two ways to integrate runZero and Tines:

- Follow the steps to create a [custom story in Tines](#), or
- Use the [runZero sample story](#) to begin with a story outline.

Tines custom story

A Tines story is a collection of actions that work together towards a specific goal, like a playbook. Tines has a Story Library that contains ready-made automated playbooks, or you can create a your own custom story if they don't have one that matches your needs. That's what we'll need to do here.

Step 1: Creating a Tines story and adding runZero API credentials

1. After logging in to Tines, create a new story.
2. Use the + in the **credentials** section to add a [text credential](#).
 - **Name:** specify a name for the credential, for example `runzero_export_token`.
 - **Value:** your [runZero export API token](#), which can be obtained from the desired [runZero organization](#) page. Export API tokens start with ET.
3. Use the + in the **credentials** section to add another [text credential](#).
 - **Name:** specify a name for the credential, for example `runzero_org_token`.
 - **Value:** your [runZero organization API token](#), which can be obtained from the desired [runZero organization](#) page. Organization API tokens start with OT.

Step 2: Creating a Tines webhook action

1. Add a [Webhook](#) action to your story.
2. Click the **Webhook** action and copy the webhook URL, which will look like `https://<tenant-name>.tines.com/webhook/<guid>`.

Step 3: Creating a runZero alert template

1. Create an [alert template](#)

in runZero:

- **Name:** name for template
 - **Template type:** JSON
 - **Subject line for message:** leave empty
 - **Body of message:** the following JSON example will include the rule name and the search URL in the alert message body: `{"rule_name":"{{rule.name}}","search_url":"{{search.url}}"}"`
2. Create an [alert channel](#) in runZero:
 - **Name:** name for alert channel
 - **Channel type:** webhook
 - **Webhook URL:** the webhook URL you copied from Tines

Step 4: Creating a rule in runZero

Now that you have your template ready to go, you will want to identify which triggers to alert on. Some common examples are:

- **Asset query results** - When there is a match on a query in runZero after a scan completes
- **New assets found** - When a scan completes with new assets
- **Agent offline** - When your runZero Explorer stops checking in to the console
- **Task failed** - When a task fails for any reason

We will use the **asset query results** selection for the rest of the steps. Our example will be any asset that has an open Telnet port `port:23`. You can [see more example queries](#).

1. Create an [alert rule](#).
2. Select `asset-query-results` and click *Configure rule*.
3. Input values for the rule:
 - **Name:** name of the rule.
 - **Conditions:** optional parameters that will trigger the alert when all conditions match.
 - **Query:** query the assets must match, such as `port:23` for our example.
 - **Number of matches:** the numeric comparison logic for the **value**.
 - **Value:** the threshold of matches to trigger the rule.
 - **Limit to organization:** allows you to limit the alert to a specific organization in runZero
 - **Limit to site:** allows you to limit the alert to a specific site in runZero
 - **Action:** notify
 - **Notification channel:** name of the alert channel you created in step 2.
 - **Notification template:** name of the alert template you created in step 2.

Step 5: Handling the data in Tines

1. The [Webhook action](#) created in [Step 2](#) will be the entry point for the runZero alert in Tines.
2. Add an [Event Transform](#) to parse the search from the URL provided in the runZero alert. If you connect the **Webhook**

action to this one, some parameters will automatically populate.

- Mode: extract
 - Matchers:
 - Path: path_to_alert_search, for example receive_alerts_from_runzero.body.search_url if your **Webhook** action is named "RECEIVE alerts from runZero"
 - Regex: \?.*
 - Extract to: search
3. Add an **Event Transform** that takes the output of the previous step. If you connect the previous **Event Transform** action with this one, some parameters will automatically populate.
- Mode: explode
 - Matchers:
 - Path: path_to_extraction_transform, for example get_search_from_url.search if your previous **Event Transform** action is named "GET search from URL"
 - To: individual_item
4. Add an **HTTP Request** to make an API call to the runZero Export API.
- URL: the path_to_explosion_transform added to the end of the runZero API URL endpoint, for example
https://console.runzero.com/api/v1.0/export/org/assets.json<<get_search_from_regex.individual_item>>
if your previous **Event Transform** action is named "GET search from regex"
 - Content Type: JSON
 - Method: get
 - Use the **+ Option** button to add **Headers**:
 - Change header to Authorization
 - Change value to Bearer CREDENTIAL.runzero_export_token
5. Add an **Event Transform** that takes the runZero Export API output and loops through each value.
- Mode: explode
 - Path: path_to_HTTP_request, for example get_assets_from_runzero.body if your **HTTP Request** action is named "GET assets from runZero"
 - To: individual_item

Step 6: Personalized automation in Tines

Now that you have received the alert, parsed the search, and obtained the assets from the runZero Export API, it's time to add your own flare. While you will likely want to do something more crafty, the sample story provided includes these two actions as examples.

1. **Send an email** to the destination of your choice for each asset. The sample JSON shows how you might customize it to put the context you'd like in the Subject and Body of the email.
2. The **HTTP Request** action shows how you might reach back into runZero to add tags to the assets after other data is gathered.

runZero sample story

The [sample story below](#) can be imported to Tines to do all of the actions outlined in the steps above. Simply save a JSON file with the contents below, and use the **Import** button in Tines to

upload it. After importing, you will need to complete the following steps.

Step 1: Update credentials in Tines

1. Replace the `runzero_export_token` credential:
 - **Name:** specify a name for the credential, such as `runzero_export_token`
 - **Type:** text
 - **Value:** your [runZero export API token](#), which can be obtained from the desired [runZero organization](#) page. Export API tokens start with `ET`.
2. Replace the `runzero_org_token` credential:
 - **Name:** specify a name for the credential, such as `runzero_org_token`
 - **Type:** text
 - **Value:** your [runZero organization API token](#), which can be obtained from the desired [runZero organization](#) page. Organization API tokens start with `OT`.

Step 2: Create runZero alert template

1. Create an [alert template](#) in runZero:
 - **Name:** name for template
 - **Template type:** JSON
 - **Subject line for message:** leave empty
 - **Body of message:** the following JSON example will include the rule name and the search URL in the alert message body: `{"rule_name":"{{rule.name}}","search_url":"{{search.url}}"}"`
2. Create an [alert channel](#) in runZero:
 - **Name:** name for alert channel
 - **Channel type:** webhook
 - **Webhook URL:** the webhook URL you copied from Tines

Step 3: Create the rule in runZero

Now that you have your template ready to go, you will want to identify which triggers to alert on. Some common examples are:

- Asset query results: when there is a match on a query in runZero after a scan completes.
- New assets found: when a scan completes with new assets.
- Agent offline: when your runZero Explorer stops checking in to the console.
- Task failed: when a task fails for any reason.

We will use the **asset query results** selection for the rest of the steps. Our example will be any asset that has an open Telnet port `port:23`. You can [see more example queries](#).

1. Create a [new alert rule](#).
2. Select `asset-query-results` and click `Configure rule`.
3. Input values for the rule:
 - **Name:** name of the rule.
 - **Conditions:** optional parameters that will trigger the alert when all conditions match.
 - **Query:** query the assets must match, such as `port:23` for our example.

- **Number of matches:** the numeric comparison logic for the **value**.
- **Value:** the threshold of matches to trigger the rule.
- **Limit to organization:** allows you to limit the alert to a specific organization in runZero
- **Limit to site:** allows you to limit the alert to a specific site in runZero
- **Action:** notify
 - **Notification channel:** name of the alert channel you created in step 2.
 - **Notification template:** name of the alert template you created in step 2.

runZero sample story JSON

```
{
  "schema_version": 4,
  "standard_lib_version": 6,
  "name": "runZero Sample Story",
  "description": null,
  "guid": "7706b502e51b4c68f0dbe9721c88d665",
  "slug": "runzero_sample_story",
  "exported_at": "2022-10-27T22:22:37Z",
  "agents": [
    {
      "type": "Agents::WebhookAgent",
      "name": "RECEIVE alerts from runZero",
      "disabled": false,
      "guid": "ab1f5bf7be49c943d893ab993ade9421",
      "options": {
        "path": "da6e90c0b276bdda97e6bfa31ad50787",
        "secret": "5e290305e0a483bf843f1213f0f21dda",
        "verbs": "get,post"
      },
      "reporting": {
        "time_saved_value": 0,
        "time_saved_unit": "minutes"
      },
      "monitoring": {
        "monitor_all_events": false,
        "monitor_failures": false,
        "monitor_no_events_emitted": null
      },
      "width": null
    },
    {
      "type": "Agents::HTTPRequestAgent",
      "name": "GET assets from runZero",
      "disabled": false,
      "guid": "d54d8f3f5b0ea857308f45f61a224547",
      "options": {
        "url": "https://console.runZero.com/api/v1.0/export/org/assets.json<<get_search_from_regex.individual_item>>",
        "content_type": "application_json",
        "method": "get",

```

```

"Authorization": "Bearer <<CREDENTIAL.runzero_export_token>>"
  }
},
"reporting": {
  "time_saved_value": 0,
  "time_saved_unit": "minutes"
},
"monitoring": {
  "monitor_all_events": false,
  "monitor_failures": false,
  "monitor_no_events_emitted": null
},
"width": null,
"schedule": []
},
{
  "type": "Agents::EventTransformationAgent",
  "name": "GET search from URL",
  "disabled": false,
  "guid": "d62ddd189bb9080d778d900f2292504d",
  "options": {
    "mode": "extract",
    "matchers": [
      {
        "path": "=receive_alerts_from_runzero.body.search_url",
        "regexp": "\\?.*",
        "to": "search"
      }
    ]
  },
  "reporting": {
    "time_saved_value": 0,
    "time_saved_unit": "minutes"
  },
  "monitoring": {
    "monitor_all_events": false,
    "monitor_failures": false,
    "monitor_no_events_emitted": null
  },
  "width": null,
  "schedule": null
},
{
  "type": "Agents::EventTransformationAgent",
  "name": "GET search from regex",
  "disabled": false,
  "guid": "41afc5c4d4c0fdb694d14e1eb380688a",
  "options": {
    "mode": "explode",
    "path": "=get_search_from_url.search",
    "to": "individual_item"
  },
  "reporting": {

```

```

"time_saved_unit": "minutes"
  },
  "monitoring": {
    "monitor_all_events": false,
    "monitor_failures": false,
    "monitor_no_events_emitted": null
  },
  "width": null,
  "schedule": null
},
{
  "type": "Agents::EventTransformationAgent",
  "name": "LOOP through assets for follow up actions",
  "disabled": false,
  "guid": "b9ada61162bdf006f09d264845ebd304",
  "options": {
    "mode": "explode",
    "path": "=get_assets_from_runzero.body",
    "to": "individual_item"
  },
  "reporting": {
    "time_saved_value": 0,
    "time_saved_unit": "minutes"
  },
  "monitoring": {
    "monitor_all_events": false,
    "monitor_failures": false,
    "monitor_no_events_emitted": null
  },
  "width": null,
  "schedule": null
},
{
  "type": "Agents::EmailAgent",
  "name": "SEND Email Action",
  "disabled": false,
  "guid": "0e9647b114f315d480f1f85c58e481d3",
  "options": {
    "recipients": "youremail@email.com",
    "reply_to": "youremail@email.com",
    "sender_name": "Your Name",
    "subject": "<<receive_alerts_from_runzero.body.rule_name>>",
    "body": "Alert: <<receive_alerts_from_runzero.body.rule_name>>\n\\<br />\n\\<br />\nLink to as
set: https://console.runzero.com/inventory/<<loop_through_assets_for_follow_up_actions.individual_ite
m.id>>\n\\<br />\n\\<br />\nAddresses: <<loop_through_assets_for_follow_up_actions.individual_item.add
resses>>\n\\<br />\n\\<br />\nServices: \n<<NEWLINE_TO_BR(NEAT_JSON(loop_through_assets_for_follow_up_a
ctions.individual_item.services))>>"
  },
  "reporting": {
    "time_saved_value": 0,
    "time_saved_unit": "minutes"
  },
  "monitoring": {

```

```

"monitor_failures": false,
  "monitor_no_events_emitted": null
},
"width": null,
"schedule": null
},
{
  "type": "Agents::HTTPRequestAgent",
  "name": "TAG assets in runZero",
  "disabled": false,
  "guid": "c3171fed91ec40d39572260295a63ae0",
  "options": {
    "url": "https://console.runZero.com/api/v1.0/org/assets/<<loop_through_assets_for_follow_up_actions.individual_item.id>>/tags",
    "content_type": "application_json",
    "method": "patch",
    "headers": {
      "Authorization": "Bearer <<CREDENTIAL.runzero_org_token>>"
    },
    "payload": {
      "tags": "hello=from_tines"
    }
  },
  "reporting": {
    "time_saved_value": 0,
    "time_saved_unit": "minutes"
  },
  "monitoring": {
    "monitor_all_events": false,
    "monitor_failures": false,
    "monitor_no_events_emitted": null
  },
  "width": null,
  "schedule": []
}
],
"diagram_notes": [
  {
    "content": "Hello! Thanks for importing the runZero Sample Story. \n\nThis story is a basic example of what you can do with runZero and Tines. \n\nPrerequisites:\n1. Create a text [credential](https://www.tines.com/docs/credentials/text) called `runzero_export_token`. Set the value as your [runZero export token](docs/leveraging-the-api.md), which can be obtained from the desired [runZero organization](https://console.runzero.com/organizations) page. Export tokens start with `ET`. \n2. Create a text [credential](https://www.tines.com/docs/credentials/text) called `runzero_org_token`. Set the value as your [organization API token](docs/leveraging-the-api.md), which can be obtained from the desired [runZero organization](https://console.runzero.com/organizations). Organization API tokens start with `OT`. \n\nThe steps are as follows:\n1. **RECEIVE alerts** provides the webhook destination for your runZero alerts \n2. **GET search from URL** parses out the search string from the URL provided in the runZero alert \n3. **GET search from regex** takes the list provided in step 2 and allows you to use the individual value \n4. **GET assets from runZero** uses the runZero Export API to get the list of assets related to the alert \n5. **LOOP through assets for follow up actions** takes the list of assets and sends each to the next steps individually \n6. **NOTE**: this is the step that you could implement more custom logic in most cases. All of the initial runZero and Tines data transfer is done, but it's your chance to customize this story to fit your use case. \n7. **SEND email action**
  }
]

```

simply sends an email to the destination of your choice \n\n8. **TAG assets in runZero** adds a tag in runZero to each asset showing an example of how you might reach back into runZero after doing other automated activities ",

```

    "position": [
      165.0,
      180.0
    ],
    "guid": "0e4b9b331dc65316849f7b089bf7bde2",
    "width": 375
  }
],
"links": [
  {
    "source": 0,
    "receiver": 2
  },
  {
    "source": 1,
    "receiver": 4
  },
  {
    "source": 2,
    "receiver": 3
  },
  {
    "source": 3,
    "receiver": 1
  },
  {
    "source": 4,
    "receiver": 6
  },
  {
    "source": 4,
    "receiver": 5
  }
],
"diagram_layout": "{\"ab1f5bf7be49c943d893ab993ade9421\": [555, 180], \"d54d8f3f5b0ea857308f45f61a224547\": [555, 450], \"d62ddd189bb9080d778d900f2292504d\": [555, 270], \"41afc5c4d4c0fdb694d14e1eb380688a\": [555, 360], \"b9ada61162bdf006f09d264845ebd304\": [555, 555], \"0e9647b114f315d480f1f85c58e481d3\": [555, 690], \"c3171fed91ec40d39572260295a63ae0\": [780, 690]}\",
"send_to_story_enabled": false,
"entry_agent_guid": null,
"exit_agent_guids": [],
"exit_agent_guid": null,
"keep_events_for": 604800,
"reporting_status": true,
"send_to_story_access": null,
"send_to_stories": [],
"form": null,
"forms": []
}
```

Thinkst Canary

All runZero editions integrate with Thinkst Canary by providing quick access from the Canary console to your asset data in the runZero Console. Setting up the integration is as simple as one change to your Canary settings.

Integrate runZero with Thinkst Canary

1. Sign in to your runZero console.
2. Sign in to your Canary console.
3. Go to **Global Settings** under the gear icon.
4. Click on **Integrations**.
5. Toggle the runZero switch.

Accessing runZero data from Canary

After the integration is enabled within the Canary settings, runZero data will be available through any incident. When viewing an incident, click the magnifying glass icon under `source IP` or `reverse IP` lookup to open a search in your runZero console.

Exposure management

runZero redefines exposure management with unrivaled visibility across your entire internal and external attack surface — covering IT, OT, IoT, mobile, and cloud. Uncover the unknown and unmanageable, reveal elusive exposures, and target the true risks other approaches miss. No agents, no authentication, no appliances. And most importantly, no blind spots.

Comprehensive asset inventory

runZero provides a comprehensive and unified asset inventory by combining [active scanning](#), [passive traffic sampling](#), and [integrations](#) with deep fingerprinting and world-class correlation capabilities. Assets are normalized, deduplicated, and tracked as they move across your environment.

Total attack surface management

runZero offers comprehensive attack surface management for external, internal, cloud, IoT, and OT environments.

External & Cloud

runZero offers hosted scan engines for external monitoring. In addition to CIDRs, hostnames, and IPs, the runZero scan scope enables automatic target detection through specific keywords:

- `domain:runzero.com`: This keyword will find all hosts associated with runZero.com and then scan both the IPb4 and IPv6 addresses of the results.
- `asn4:1233`: This keyword will resolve all CIDRs associated with ASN 1233 and then scan these.
- `country4:us`: This keyword can be used to scan entire countries, but more often it is used in the Excludes field to limit external discovery to a particular regions.
- `defaults`: This keyword expands to all CIDRs and hostnames registered in the associated site.

These keywords can be combined and used for both Scope and Exclusions. When the `Initialize` action is take in the scan page, a preview of the expanded targets is shown, along with an estimate of the scan time based on address count and full scope.

Integrations are also used to identify and monitor the external attack surface. Assets imported from cloud providers like AWS, Azure, and GCP also include their external IPs and hostnames.

Internal

runZero enables comprehensive internal network attack surface enumeration using active scans, passive discovery, and integrations.

The runZero scanner supports subnet sampling, which allows for incredibly fast discovery of massive internal ranges. Subnet sampling works by trickling packets into the most commonly used octets of each /24 and only scanning subnets where at least one reply was observed. This method also parses ICMP error messages to identify the network ranges used by the network equipment itself. runZero's RFC1918 report can be used to identify blind spots and launch scans to close the gaps.

In addition to scans, runZero can import PCAP files directly into the web interface, as well as turn any deployed Explorer into a passive network sensor. Passive discovery uses up to one CPU core of resources to parse all traffic passing through the interface, including broadcast, SPAN port data, and encapsulated traffic (VLAN, VXLAN, GRE, etc.).

Lastly, runZero's extensive set of integrations makes it easy to determine what internal endpoints are missing a particular security control. For example, if you use CrowdStrike as your EDR, runZero can flag all Windows systems missing this agent. This also works for finding gaps in vulnerability management scopes.

IoT & OT

runZero was built for fast, unauthenticated, and safe scanning of all devices in all environments. This results in comprehensive coverage of internet of things (IoT) devices and reliable fingerprinting of operational technology (OT) equipment. runZero worked with the [US Department of Energy](#) to ensure that our active discovery is safe and accurate for OT environments.

Users who are unable to deploy active scanning can still take advantage of passive discovery, PCAP imports, and integrations with OT and IoT management tools.

Inside-Out

runZero also offers a unique capability called [Inside Out Attack Surface Management](#) (IOASM). IOASM works by comparing the unique fingerprints of internal assets with a global database of public endpoints. This feature is enabled by default for customers using a runZero-hosted platform and is optionally available for self-hosted customers through additional configuration. IOASM can quickly tell you if an internal device is publicly exposed through an unexpected IP, including port forwards, VPNs, IPv6-tunnels, and more. Additional information can be found in this [presentation](#).

IOASM reports four distinct vulnerabilities based on heuristics and detection type:

- (TLS|SSH) Private Key is Public
- (TLS|SSH) Private Key is Widely Shared
- Potential External Access to Internal Asset (SSH, TLS)
- Potential External Access to Internal Asset (MAC Address)

The reported risk also varies based on heuristics. A publicly service attached to a Remote Desktop or Secure Shell system is considered higher risk than reuse of a public TLS key for a web application.

IOASM also makes use of the [BadKeys](#) compromised key database.

Vulnerability detection

runZero identifies, imports, and manages vulnerabilities across your total attack surface using a combination of [query-based](#) detection logic, [active scans](#), and API-based [integrations](#) with leading endpoint management and vulnerability management platforms.

Findings provide a comprehensive view of all detected security issues, misconfigurations, and other exposures across your assets. See [Findings](#) for more details.

runZero's [Rapid Response](#) program drives real-time detection and notification for zero-hour exposures.

You can find the complete list of runZero-provided queries in the [Query Library](#). Custom queries can be created to report vulnerabilities on matching assets and services.

Scan-based vulnerabilities are identified using an embedded version of [Nuclei](#), the leading open source vulnerability scanner, in conjunction with [curated templates](#) and runZero's best-in-class network discovery and fingerprinting engine. Vulnerability categories are configured as part of the scan configuration and specific templates are chosen dynamically by precisely matching assets and services. Out-of-band testing relies on runZero-hosted [Interactsh](#) services, hosted at `reflect-us.rumble.network` and `reflect-eu.rumble.network`.

runZero recalculates vulnerabilities, findings, and asset risk as part of task processing. The resulting vulnerabilities and findings are shown in the respective product sections.

Risk prioritization

runZero normalizes and assigns risk scores to all assets. These scores are influenced by a combination of threat intelligence, vulnerability information, and exposure measurement. Asset criticality can be set through automated rules or manually through the product interface. Assets can be assigned to specific owners for remediation. Tags are imported from API integrations and can be managed natively within the interface, including rule-based tagging. Exported assets include the risk, criticality, and tag information set through the product interface.

Continuous monitoring

runZero continuously monitors your organization for changes to exposure, at a per-asset and per-service level. Recurring active scans, background passive traffic sampling, and regular sync with your existing infrastructure enable quick detection and reporting of new risks. Alerts can be managed in-product, sent by email, or delivered by webhook to the platform of your choice. All asset data can be synced to external platforms, including popular SIEMs and data lakes.

Rapid responses

runZero's Rapid Response program provides immediate detection and notification of emerging threats. Older entries are migrated to standalone [queries](#) or [templates](#).

Rapid responses omitted from PDF. See help.runzero.com for the full list.

Each Rapid Response includes a query to find matching assets, a trigger to analyze all inventories for exposure, and a corresponding blog post with the details of the issue. This program focuses on helping customers mitigate exposures before compromise.

Vulnerabilities covered by the Rapid Response program are replaced by more specific coverage as mitigations become available.

Query library

Query library omitted from PDF. See help.runzero.com for the full list.

runZero includes a substantial library of pre-built queries. These queries can be used to detect vulnerabilities, trigger alerts, and apply changes to assets, such as tags and ownership. These queries are categorized by use case and risk level. Custom queries can also be configured to report vulnerabilities on matching assets and services.

Templates

The table below lists the [Nuclei](#) vulnerability templates available for scans. The full set of tuned templates can be found in our [nuclei-templates](#) repository.

Vulnerability templates omitted from PDF. See help.runzero.com for the full list.

In addition to [query-based](#) vulnerability reporting, runZero natively detects exposures using an embedded version of the open-source [Nuclei](#) vulnerability scanner and its YAML-based vulnerability check templates. To maintain fast scan times and minimize network disruption, runZero dynamically selects appropriate templates based on the scan's configured categories and precise asset and service fingerprinting.

Findings

Findings are groups of significant exposures with the same root cause.

You can promote one or more vulnerabilities from a finding into a runZero [issue](#) for tracking and remediation, and optionally push the issue to Jira as an external ticket.

Findings summary omitted from PDF. See help.runzero.com for the full list.

Findings are created through three sources:

1. **Query-based:** These findings are identified through specific queries defined within the system (see list below).
2. **Nuclei-generated:** These findings result from scans where default credentials and vulnerability checks are enabled, leveraging Nuclei templates (see [templates](#)).
3. **KEV (Known Exploited Vulnerabilities):** These findings are triggered when a discovered vulnerability is present on the CISA Known Exploited Vulnerabilities (KEV) catalog or VulnCheck KEV.

Issues

Issues let you group one or more vulnerabilities together for tracking, prioritization, and remediation. An issue captures a point-in-time **snapshot** of every (vulnerability, asset) pair it contains so that the original context is preserved even if the asset or vulnerability later changes.

Use issues to:

- Roll several related vulnerability instances up into a single unit of work.
- Assign a risk level, state, and due date and track progress over time.
- Record decisions and discussion in a per-issue timeline.
- Push the issue out to Jira as an external ticket and keep the runZero timeline in sync via a [Jira webhook](#).

What is an issue

Each issue has the following fields:

- **Name** — a short label that identifies the issue.
- **Description** — longer free-form context about the issue.
- **Remediation** — remediation guidance for the people doing the work.
- **Risk** — one of none, low, medium, high, or critical. The default is the highest risk found across the issue's vulnerabilities at creation time.
- **State** — one of open, in progress, closed, or resolved. As long as the state is not closed, resolved is set automatically by runZero when all instances have been remediated (see [Automatic remediation tracking](#) below), or it can be set via a [Jira webhook state mapping](#).
- **Due date** — optional target date for resolving the issue.
- **Owner** — the single user responsible for the issue. Issues use the built-in `Issue owner` ownership type and cannot be assigned custom ownership types. See [Issue ownership](#) for details.
- **Instances** — the (vulnerability, asset) pairs captured in the snapshot.

Creating an issue

You can create issues from anywhere in the runZero Console that shows vulnerabilities or groups of vulnerabilities:

- The [vulnerability groups inventory](#)
- The [findings inventory](#)
- The [vulnerabilities by asset inventory](#) table.
- The [findings]
- The vulnerabilities card on an [asset](#) details page.
- The vulnerabilities table on a [findings](#) details page.
- The vulnerabilities table on a [certificate](#) details page.
- A [vulnerability](#) details page.

Select one or more vulnerabilities and click **Create issue**. You will be taken to a draft issue where you can edit the name, description, solution, risk, and due date before saving. Once saved, the issue is added to the [Issues page](#) for the current organization.

A snapshot of each selected vulnerability and its asset is stored on the issue at creation time. This means the issue keeps showing the original instance details even if the underlying asset is later modified, deleted, or rescanned.

Working with an issue

The issue details page is split into three sections:

- **Header** — name plus quick-edit risk, state, and due date.
- **Summary** — description and remediation guidance.
- **Instances** — the table of vulnerabilities captured in the snapshot, with a slideout with vulnerability and snapshotted asset details. Instances whose vulnerability has since been remediated are flagged.
- **Comments** and **History** tabs — the issue timeline (see below).
- **Sidebar** — risk, state, due date, and the linked Jira ticket, if one has been created.

Comments and history

The issue timeline records three kinds of entries:

- **User events** including changes to the issue fields and comments authored by a runZero user from the **Comments** tab.
- **System events** that runZero generates automatically (state changes, ticket creation, and similar actions).
- **Jira events** that arrive via the [Jira webhook](#), one timeline line per changelog item on the linked ticket.

The **Comments** tab shows comments only; the **History** tab shows everything in the timeline.

Issue ownership

An issue can have one owner. Issue ownership is separate from the ownership types used for assets and vulnerability records: every issue uses the built-in `Issue owner` type, and custom ownership types cannot be added to an issue. When an issue is linked to Jira, runZero can synchronize its owner with the Jira assignee when a matching external user is found. See [Issue owner synchronization](#).

Automatic remediation tracking

Whenever a scan, import, or integration import finishes for an organization, runZero recomputes the **remediated instances** counter for every non-closed issue in that organization. An instance is considered remediated when the original vulnerability is no longer present on the original asset.

If the state has been set to `closed`, runZero will not recompute the remediated instances and will not make any automated changes to the state.

When every instance on an issue has been remediated, runZero automatically transitions the issue to the `resolved` state.

If an issue is in the `resolved` state and one or more of its vulnerabilities are later detected on the original asset again, runZero automatically moves the issue back to `in progress` to alert you to the regression.

Linking a Jira ticket

You can push an issue out to Jira as an external ticket by first creating a [Jira credential](#). The Issues page sidebar shows a **Create Jira ticket** action that allows you to select the Jira credential and optionally upload a CSV of the issue's instance snapshot as a Jira attachment. After the ticket is created, the runZero issue stores the Jira issue key, a deep link to the ticket, and the credential and Jira base URL used. See the [Jira integration](#) page for full details.

Permissions

- **Viewing** issues requires read access to the organization the issue belongs to.
- **Creating, editing, and deleting** issues requires write access to that organization.
- Issues are not visible from the **All organizations** view; switch into a specific organization to use the Issues page.

Data analysis

Reviewing results

Task details

After each discovery task completes, the task details page will list a summary of how many assets were updated. To understand the numbers, it's important to remember that runZero will correlate assets across IPs and data sources, which can result in different results than IP-based matching alone.

The change summary on the task details page includes the following statistics:

- Asset changes:
 - **Newly discovered assets** are devices that were found during the task for which no device with matching fingerprints was previously seen.
 - **Assets marked offline** are assets that runZero has previously seen on the scanned network, but that didn't respond on any of the IP addresses during this scan. When this happens, the asset is marked offline. The offline status is a flag on the asset, and doesn't count as a change to the asset. Assets may be marked offline because the device was powered down or disconnected, or because of network problems.
 - **Assets back online** are assets that were marked offline at some point in the past, but the runZero Explorer got a response from them during this scan. The online status is a flag on the asset, and doesn't count as a change to the asset.
 - **Assets changed** is the number of assets where some property of the asset was modified, other than its online status. Examples include changes to the device's IP addresses or hostname, or responses from new ports or protocols.
 - **Assets unchanged** is the number of assets that were seen exactly where runZero found them in the last scan, with no changes to their responses.
 - **Assets ignored** is the number of occasions where the Explorer got a response from probing an IP address, but it turned out to be bogus in some way. This typically happens when a web proxy, stateful firewall, or SIP gateway responds as if it is the asset at every address on a subnet.
 - **Total assets seen by task** is the total of **Assets changed** plus **Assets unchanged**. It indicates the number of asset records that are now up-to-date.
- User changes:
 - **Newly discovered users** are users that were seen for the first time during the integration sync.
 - **Users changed** are users that had attributes change during the integration sync.
 - **Users unchanged** are users that did not change during the integration sync.
 - **Users updated by task** is the total number of **Users changed** and **Users unchanged**, indicating how many user records are now up-to-date.
- Group changes:
 - **Newly discovered groups** are groups that were seen for the first time during the integration sync.

- **Groups changed** are groups that had attributes change during the integration sync.
- **Groups unchanged** are groups that did not change during the integration sync.
- **Groups updated by task** is the total number of **Groups changed** and **Groups unchanged**, indicating how many group records are now up-to-date.

Dashboard & inventory views

The dashboard will be populated with results after the first scan completes. The dashboard provides trend data and insights that will help you assess how your inventory is changing over time. You can select a time period and site for the trend data using the selectors at the top right of the dashboard page.

The main asset trends graph shows the number of assets in each of the four main states – live, offline, scanned and unscanned. Beneath the graph are additional asset breakdowns, each of which shows a top 10 of an asset category – asset type, operating system, hardware and tags.

The service trends graph shows how many total services were found in your asset inventory, along with breakdowns for ARP, ICMP, TCP and UDP. Below the service trends graph are breakdowns of the top 10 TCP ports, UDP ports, protocols and products detected.

Clicking the menu button at the top right of each table and selecting “View more” shows a more detailed inventory by category.

Insights from queries

Queries and reports can help you gain valuable insights, but you may wonder where to get started. We recommend trying the pre-built queries in the [Query Library](#) first. Some of these queries are a result of runZero’s [Rapid Response](#) to emerging threats and are described on our [blog](#).

runZero’s query language allows you to search and filter your asset inventory based on asset fields and value pairs. See the [documentation](#) about querying your data. Once you are familiar with the query language you can write your own queries.

You can set queries to run automatically by opening the query and setting the “Automatically track query results on the dashboard”. The query will run when scans complete, and you will be notified of any resulting insights on the dashboard page.

Sample Queries

Asset inventory

- Equipment that is likely 8+ years old: `alive:t mac_age:>8years`
- Assets with end-of-life OS: `os_eol:<now`
- Virtual machines: `has:virtual`
- Devices acting as a router: `router:true`

- Devices that may be bridging: `has_public:t` and `has_private:t`

Service inventory

- Protocol on a non-standard port example: `protocol:ssh not port:22`
- Publicly addressed assets running RDP or VNC: `has_public:t` and `(protocol:rdp or protocol:vnc)`
- Authenticated web services that are not encrypted: `(_asset.protocol:http AND not _asset.protocol:tls) AND ( html.inputs:"password:" OR last.html.inputs:"password:" OR has:http.head.wwwAuthenticate OR has:last.http.head.wwwAuthenticate )`
- Older TLS versions in use: `alive:t AND protocol:"=tls" AND ( tls.versionName:"=TLS 1.0" OR tls.versionName:"=TLS 1.1")`

Some other sample queries are described in our blog entries:

- [Finding duplicate SSH host keys](#)
- [Identifying rogue remote access solutions](#)
- [Finding device serial numbers](#)

Reports

After viewing the dashboard and inventory, your next stop should be the runZero [Reports](#) page.

Switch topology

This report uses SNMP information to map how the switches on your network are connected. Each switch displays its IP address, name, and the number of assets connected to it. If runZero detected MAC addresses that were not found as part of the scan scope, you will see a number of unmapped assets indicated below the switch.

You can click on a switch to see a pop-up with the number of identified and unmapped assets. From there, you can click to view the unmapped assets, and be taken to a table of unmapped MACs by switch port.

Double-clicking on a switch will expand that part of the diagram and show the individual assets connected to the switch.

The switch topology report won't always be entirely accurate as it's based on which switch claims to have seen each MAC address, and this may not always be the nearest access switch. Our algorithm looks for the switch port with the least number of shared MACs to find the best match, but this may not give the answer you expect, depending on switch cache timeouts and how the switches were scanned.

Subnet utilization

The subnet utilization report lists the subnets scanned on your network, and what percentage of each is in use. For example, if you have scanned 10.0.1.0/24 and found 25 assets, the report

will show that 10% of the available IP addresses in the subnet are in use.

Network bridges

The network bridges report is a way to find devices that bridge multiple network segments. It can be useful to locate unintentional bridging between your internal networks and the Internet.

The report shows your internal networks in green, and external networks in red. It then shows you the multihomed assets which bridge an internal network to an external one.

RFC 1918 coverage

The RFC 1918 coverage report is a way to view how much of the private internal network address space has been scanned for assets. It can help you discover rogue assets, unscanned subnets, and secondary interfaces on scanned devices. More information is in the section on [coverage reports](#).

Unmapped MACs

This report uses SNMP information to list MAC addresses runZero found evidence for, but which weren't encountered as addresses of assets during the network scan. The MAC addresses are grouped by the switch that reported them, along with information about the vendor, manufacture date and switch port of the possible asset, to help identify them.

Outliers

The outliers reports allow you to obtain a summary of how often different values occur in specific attributes of assets and services. The values are sorted from most frequent to least frequent.

For example, the HTTP servers outliers report will list all of the HTTP servers encountered by runZero, starting with the most common.

As well as the one-click outliers reports, you can produce an outliers report for any asset or service attribute.

- Switch topology to identify how your assets are connected and find "unmapped" MAC addresses (in red) that were not included in your scan scope (a summary of which is in the Unmapped MACs report)
- Bridging to visualize what hosts may have both public and private connections
- RFC 1918 coverage that can identify potential blindspots on your network like missing (unscanned) subnets, rogue devices, and "hinted" IPs that are secondary interfaces on unscanned network ranges.
- See the "View all" button at the top right for a list of other reports to investigate outliers

Domain membership report

The domain membership report lists the Active Directory domains encountered by runZero, and lists how many assets are in each.

Analysis reports

Analysis reports are more advanced reports. They may run as tasks, rather than being generated on-the-fly.

The first analysis report is [Compare Sites](#), which generates searchable reports of the differences between two sites.

The Outlier Overview Report analyzes assets across the organization and summarizes the most unusual values for an assortment of key attributes such as hardware type and SNMP enterprise ID.

The Specific Outlier Report allows you to select an attribute which has outlying values and get a detailed breakdown of those values.

The Organization Overview Report builds a high level summary report of the entire organization. It can optionally include lists of assets found.

Alerts

As well as manually generated reports and queries, runZero also supports [automatic alerts](#) to designated channels for post-scan inventory queries, asset changes, Explorer and scan issues, security operations, or API events.

Available channels are internal notifications in the runZero web console, email, or webhooks that can enable integration with services such as Slack or Mattermost. Alerts use the same query language as the sample queries above, so this is a good way to automate proactive notification for critical events.

Using dashboards

Dashboards provide customizable, visual views into your asset inventory and can be [created](#) to serve different use cases such as compliance, vulnerability remediation, or asset visibility.

A variety of visualization [widgets](#) are available that show operational information, trends, insights, goals, sources, and most and least seen graphs. You can also create your own [custom widget](#) based on queries to get the exact data you are looking to surface, displayed either as a trend line or latest count.

You can filter the dashboard data by site and time buckets based on your needs by using the dropdown buttons at the top of the dashboard and selecting the timeframe and/or site you wish to filter by.

You can also [customize](#) the arrangement and size of each widget on the dashboard to suit your particular needs using Edit Mode, and access the [widget library](#) to add or remove widgets as desired.

Dashboard selection

The dashboard selector button at the top of the dashboard displays the dashboard's name by default, and when clicked will open up a menu that displays runZero-managed dashboards, any shared dashboards, as well as your personal customized dashboards. To change dashboards, simply click the name of the dashboard you wish to view.

Creating dashboards

To create a new personal dashboard, click the dashboard selector and click the "Create new dashboard" button, which will create the new dashboard and automatically take you into Edit Mode where you can begin adding, resizing and arranging widgets.

Customizing dashboards

Personal dashboards can be customized by clicking the "Edit" button found in the dashboard header, which will put the dashboard into "Edit Mode" and allow resizing, rearranging, or adding and removing widgets.

While in Edit Mode, each of the widgets on the dashboard can be resized according to your needs by clicking and dragging the bottom right corner of the widget.

Widgets can also be rearranged within the dashboard by clicking and dragging on the widget heading area as indicated by the "cross-hair" mouse cursor icon.

When satisfied with the configuration, save your customization settings by clicking the "Save" button so that you can quickly get to the information you'd like to see any time you visit the dashboard.

If you wish to revert the dashboard to its original state, click the "Cancel" button to destroy your changes. This will reset the visibility, arrangement, and size of all widgets to reflect the current view of the dashboard prior to customizing it.

Sharing dashboards

A dashboard may be shared with any number of organizations by clicking the "Share" button. In the dialog, select the organizations to share the dashboard with and click "Done" to save the settings.

When a dashboard has been shared with your organization by another user, the shared dashboard will appear under the "Shared with me" heading in the dashboard selector.

Dashboards that you share with others will still appear within the "My dashboards" section, and can only be modified by you, but may be duplicated by other organization users if desired.

To share a read-only snapshot of a dashboard with someone outside of runZero, use [external reporting](#) instead. The Share button on shareable dashboards creates an email-gated public link, optionally on a recurring schedule.

Duplicating dashboards

When viewing a runZero-managed dashboard or one shared with your organization, it cannot be modified directly, but you may make a copy of it to customize as desired by clicking the "Duplicate" button.

Dashboards can also be duplicated by clicking the menu button at the top of the dashboard and selecting "Duplicate dashboard". In this case, a dialog will open that allows you to set the name of the new dashboard to distinguish it from the original version if desired.

Note: sharing settings from the original dashboard will not be copied to the new dashboard.

Default dashboards

A user may choose to set any dashboard as their preferred dashboard by clicking the dashboard menu button and selecting "Set as my preferred dashboard".

An organization admin can also set the default dashboard for all users of the selected organization by selecting the "Set dashboard as organization default" option. Within the dialog, select the "Override user-preferred dashboard" checkbox to override the user's preferred default dashboard if desired. When configured as such, users will only see their preferred dashboard if the organization doesn't have a default dashboard set to override their preference.

Deleting dashboards

A custom dashboard can be deleted by clicking the menu button at the top of the dashboard, selecting “Delete dashboard” and confirming the intention in the dialog that opens. This cannot be undone.

Widget library

The dashboard widget library is used to add and remove widgets on the dashboard, and can be accessed by clicking the “Widgets” button at the top right of the dashboard.

When you open the widget library, a list of all the available widgets is presented for you to select from. You may also choose to create your own [custom widget](#) based on queries.

Widgets included on the dashboard will be marked with a checkmark icon. When clicked, the widget will be deselected and removed from the dashboard when you click the “Save” button.

Conversely, widgets that are not currently visible on the dashboard can be added by clicking on the widget to select it, and then saving the changes.

Widget types

Operational information

- Live assets: number of assets currently alive based on the latest scans
- Active scans: number of scans currently in progress
- Explorers online: number of Explorers reporting as healthy
- Accounts: number of users with access to your console

Trends

- Asset trends: timeline view of asset counts broken out by live, recent, and a couple other states
- Service trends: timeline view of network services seen in your environment

Insights

- Latest insights: automated queries that were ran after your latest scan with result counts

Most and least seen graphs

A variety of asset views by different asset properties. All of these graphs can be toggled between most and least seen based on the counts.

- Asset types
- Operating systems
- Hardware
- RRT latency
- MAC vendors
- Newest MAC age
- TCP ports
- UDP ports
- Products
- Protocols
- Address count
- Extra address count
- Risk
- Criticality
- Tags

Goals

- Goals overview: summary view of all goals that are pinned to display on the dashboard

Asset count and trend tables

A view into the latest count and historical timeline for your assets related to ownership coverage, source types, and custom integrations.

- Asset ownership: view of all ownership types including assets missing owners
- Asset sources: view of all sources of asset data from supported integrations
- Custom integrations: view of all custom sources of asset data from API integrations

Bookmarks

A customizable list of frequently visited pages, favorite reports, or external links. After adding the widget to your dashboard, you can manage the list by using the menu button at the top of the widget and selecting the appropriate menu item.

Bookmarks can be edited or deleted by selecting the "Edit bookmarks" menu item, selecting the bookmark to be edited from the list shown in the dialog, updating the name or URL below, and saving the change. Similarly, a bookmark can be removed by clicking the trash can icon within the list if desired.

Additional bookmarks can be added by selecting the "Add bookmark" menu item, entering a name and URL, and then saving the change. Bookmarks to external sites will open in a new window, while internal console URLs will open in the current window.

When viewing individual reports or browsing the list of reports, you can easily add any report to your bookmarks by clicking the bookmark ribbon icon. After a report is bookmarked, you can remove it by clicking the ribbon icon again, which will open the editing dialog and allow you to remove the bookmark.

Custom widgets

Several widget types can be created and customized using the “Create widget” button in the widget library. These widgets fall into 4 types:

- **Single match count** - Displays the most recent match count for a given query.
- **Single match trend** - Displays a trend chart detailing the history of a given query.
- **Multi-query match count** - Displays the most recent match count for 2 or more given queries, displayed similar to the *Organization overview* widget.
- **Multi-query trend line** - Displays a trend chart detailing the history of 2 or more given queries.
- **Custom text widget** - Displays custom Markdown-formatted text content, which can be used to provide additional context or information on the dashboard.
- **Goal progress overview** - Displays a summary and current progress towards completion for a single goal.

Custom widgets can be customized in several ways. For single-query widgets, you can choose a name, query, and a color to style the results with. In the case of a multi-query widget, you can choose a name for the widget itself, and you can choose a name and color for each individual query as well.

Any custom widgets that display as a trend chart can be exported to JPG, PNG, or SVG formats via the menu button in the top right of the widget on your dashboard.

After creation, custom widget templates can be managed in the widget library by selecting the appropriate icon at the bottom of each widget’s preview image. When updating a custom widget, you will be prompted to also update the widget on the dashboard if desired. Similarly, when deleting a custom widget, you will be prompted to also delete the widget from the dashboard if desired.

Drill down

Except for timeline-based charts, each widget’s distinct data points can be viewed in detail by clicking on the relevant table row links or buttons, or by clicking on individual segments of donut, bar or column charts.

Printing and exporting

All chart-based widgets on the dashboard support exporting to SVG, PNG, and JPEG file formats, and table-based widgets can be exported to CSV format, by using the menu button at the top of the widget and selecting the file format you wish to export.

The entire dashboard and all of its widgets can be printed by selecting the printer icon at the top of the dashboard. For best results, use a Chromium-based browser or Firefox with the 'Print backgrounds' option enabled to print the dashboard.

Recalculate metrics

Dashboard widgets, query results, and saved-query counts are powered by metrics that runZero recomputes after every scan, integration sync, and scheduled metrics task. If you have just imported data, edited tags (including `geo*` location tags or manual `category` overrides), or changed asset criticality and want the dashboard, [World Map](#), and saved-query counts to reflect those changes immediately, click the dashboard menu button and select **Recalculate metrics**. A one-shot metrics task is queued and the dashboard will refresh once it completes.

Themes

The runZero console ships with three themes — **Classic**, **Light**, and **Dark**. Choose the one you prefer from your **Account** → **Preferences** page. Theme selection is per-user and applies to every dashboard, report, inventory view, and the Network and World Maps.

Display mode

The dashboard supports displaying in theater (or kiosk) mode, where navigation menus are hidden and only the dashboard content is shown. To enable theater mode, click the menu button at the top of the dashboard and select the "Enter theater mode" option.

When theater mode is active, the dashboard also supports fullscreen display by clicking the *Fullscreen* button at the top of the page. In order to exit fullscreen mode, press the `Escape` key or click the *Exit fullscreen* button to return to theater mode.

To exit theater mode, click the X icon to return to the normal browsing experience.

AI Threads

The AI section of the console lets you ask questions about your inventory, build reports, export data, draw charts and diagrams, and [write custom integration scripts](#) in a conversation. It lives at [AI threads](#) in the console navigation, and has three tabs: **Threads** (your conversations), **Artifacts** (the saved output), and **Reporting** (guided report templates).

An **AI thread** is a private conversation between one user and the AI: no other user can see your threads, ever. What you choose to share is the [artifact](#) — the report, table, script, or other output a thread produces — never the conversation itself.

Requirements

- The account must have the AI entitlement, and AI must be enabled in the account (or organization) AI configuration. When AI is turned off, the composer is locked and the page says so; superusers are linked to the [AI configuration](#).
- An organization must be selected in the organization picker. A thread records the organization scope that was active when it was created, and the AI's searches never reach beyond that scope. The My Organizations options does work with AI threads, but keep in mind that anyone you share an artifact with must have the same accessible organization permissions.
- Runs are also subject to the account's AI token budget; when a daily budget or token cap is exhausted, new runs are refused until it resets.

Threads

Starting a thread

A thread starts from any of these places:

- the **New thread** composer on the threads page;
- the **New AI thread** button in the top bar;
- the command palette (**Cmd/Ctrl+K**), which can start a thread carrying the context of the page you are on;
- **Ask AI** handoffs from inventory pages — asset details, vulnerability and software groups, and similar views — which start the thread scoped to that object. A **Context** chip on the composer shows the scope, and can be removed before the first question to ask about the whole organization;
- the **Reporting** tab, where picking a guided report and answering its scoping questions starts a thread that generates the report;
- [generating a custom integration script](#) — each authoring run appears as a live turn on its own thread.

In the composer, **Enter** sends and **Shift+Enter** inserts a newline. A new thread is titled from your prompt, and after the first answer the AI replaces that with a short generated title; renaming a thread yourself stops the automatic titling.

The thread list

Threads are grouped by recency — Today, Yesterday, Earlier this week, Earlier this month, Older — and sorted by last activity. Each row shows a running indicator when a turn is in flight, an icon for the thread's newest artifact, the relative time of the last activity, when the thread [expires](#), and the organizations it spans. The search box matches thread titles, prompts, and responses; press / to focus it.

The list shows the threads whose organization scope overlaps your current organization selection. When you have threads outside that scope, a footer note says so and offers to switch to **My Organizations**.

Reading a thread

Every turn shows your prompt, a **work log** of the steps the AI took — searches it ran, data it read — and a summary line with the elapsed time, token usage, tool calls, and the artifacts it produced. The **Normal** view folds the work log away; **Verbose** expands every step. The answer itself is saved as an artifact and shown as a linked card; the newest one opens automatically in the artifact panel beside the conversation.

Background execution

A run is independent of your browser: closing the tab, navigating away, or losing the network does not stop it. Reopening the thread re-attaches to the run and replays everything that streamed while you were away, and the command palette lists running AI jobs so you can jump back to them. Only **Stop** ends a run early — it returns your prompt to the composer — and a chat turn otherwise runs up to 5 minutes, a report turn up to 10.

Renaming and deleting

Rename a thread with the pencil beside its title (titles are capped at 120 characters). Deleting a thread permanently removes the conversation and every **unshared** artifact it produced, along with their attached files; shared artifacts are kept. A thread with a running turn cannot be deleted until the run finishes or is stopped. There is no way to share a thread — share its artifacts instead.

Organization scope

A thread can only be continued by its owner, with an organization selection that still contains every organization the thread was created against. Switching to a different organization, or losing access to one of the thread's organizations, blocks new turns until the selection (or access) is restored; the existing conversation stays readable.

Tools

Every turn works through an available set of read-only tools, scoped to the thread's organizations, and the logs which ones are called. The same tools are exposed over [MCP](#), so an external MCP client and an in-product AI thread can ask the same questions of the same data.

Inventory search. `query_assets` and `query_asset_counts` for assets, `query_services`, `query_certificates`, and `query_tasks` for their own inventories, and `query_inventory` for everything else — software, software groups, vulnerabilities, vulnerability groups, wireless, findings, events, directory users and groups, tasks, issues, goals, credentials, saved queries, and sites. `get_attributes` pulls selected host attributes for a batch of assets, and `list_organizations` lists what the thread can reach.

Writing queries. Answers are grounded in real [search syntax](#). `get_field_definitions` returns the searchable fields for an entity or integration source with descriptions and example values; `get_query_syntax` returns the operator reference, [worked examples](#), and that entity's fields; and `translate_query` turns a plain-language phrase into runZero syntax — asking for “cisco devices with ssh exposed” yields `(os:cisco or hw:cisco) and protocol:ssh`. The AI tries to cite data use live queries into the inventory search.

Detection coverage. `search_vuln_coverage` and `search_scan_checks` search the vulnerability match rules and active scan checks runZero ships, so “can runZero detect this?” is answered from platform coverage rather than from your findings. `list_rapid_response` lists the [Rapid Response](#) feed with the query that finds affected assets.

Organization analytics. A set of snapshot tools covering the whole organization at once: overview distributions, subnets, outliers, exposed assets, datastores, security-control signals, critical systems, unmanaged assets, end-of-life assets, Explorers, recent tasks, metrics, and goals.

A turn spends at most 8 tool calls on an ordinary question. Asking for a report, a deep investigation, a comparison, a chart, or a trend raises the budget to 20. Guided reports are not capped this way; they are bounded by the 10-minute turn limit instead.

Artifacts

Everything a thread produces is saved as an **artifact**: the answer document itself, plus any tables, charts, diagrams, or scripts it generated along the way. Artifacts are the durable, shareable output of the AI. Artifacts are private by default and expire with their parent thread unless shared. Once shared, they stay active indefinitely and can be manually deleted as necessary.

The **Artifacts** tab your private artifacts and any artifacts shared by team members to their respective organizations. These are searchable by title, kind, and author, with **Created by me** and **Created by other people** filters. Selecting one shows the content and its provenance: who created it, when, from which organizations, with which model, its size, and when it expires. Both threads and artifacts can span multiple organizations and all viewers of an artifact need overlapping access to every organization whose data was included.

Artifact kinds

Kind	Notes
Markdown response	The primary answer document of a chat turn.
Report	The output of a guided report, named for the report.
CSV / TSV / JSON	Data exports, rendered as read-only source and downloadable as files.
SVG graphic	Rendered graphics, embedded inline where the answer references them.
Mermaid diagram	Diagram source , rendered in place and downloadable as a .mmd file.
HTML / HTML application	Self-contained documents, rendered in a hardened sandbox: no scripts from the network, no outbound requests.
Graphviz diagram	DOT source, shown read-only and downloadable as a .dot file.
Shell script / PowerShell / Starlark script	Code, rendered read-only with syntax highlighting.

Large inline tables and code blocks inside a report are carved into **attached files** — downloadable resources listed with the artifact — so a big export does not bloat the document. A single turn can produce up to 100 artifacts, each up to 512 KiB.

Charts and diagrams

When an answer requires a chart — a distribution, a trend, a flow, a path, a timeline, or a comparison — the AI draws it using [Mermaid](#). These charts are written in plain text and support bar and line charts, pie, flowchart, sequence, state, class, entity-relationship, timeline, journey, gantt, quadrant, sankey, mindmap, kanban, radar, treemap, venn, block, packet, architecture, C4 context, requirement, git graph, and fishbone. Standalone SVGs are reserved for the rare graphic no Mermaid form fits.

A diagram that stands on its own — one you asked for, or one worth sharing separately — is saved as its own **Mermaid diagram** artifact. It still renders in place in the answer, and it is separately downloadable as a .mmd file. Diagrams also render in the Markdown artifact's HTML and PDF exports. Inside the body of an HTML report the diagram is a link to the artifact rather than a picture, because the report sandbox runs no scripts; graphics that must appear inside an HTML report body are drawn as SVG instead.

Viewing and exporting

Every artifact has a **Full screen** view, which fills the window and returns with **Esc**. HTML reports carry their own toolbar with **Print / PDF**, an **HTML** download, and **Open** (full screen in

a new tab). Prose artifacts offer Markdown, HTML, and PDF exports; file artifacts (data and scripts) download as their original file. Every artifact can be renamed by its creator (up to 160 characters).

Sharing and privacy

Every artifact starts **private** to its creator. Sharing one makes it visible to every user who can read **all** of the organizations it was created from; **Copy link** (available once shared) produces a link that opens it in the Artifacts tab. Unsharing returns it to private.

The creator can always delete their own artifact. A shared artifact can also be deleted by a superuser or by an administrator of every organization in its scope. Deletion is permanent, removes attached files, and removes the artifact for everyone.

Use in custom integration

A Starlark script artifact — typically the product of the [AI script generator](#) — has a **Use in custom integration** action that loads the script back into the integration it was written for, or into a new one. It requires the custom integration entitlements in addition to AI.

Reporting

The **Reporting** tab is a gallery of guided report templates — such as End-of-Life OS Analysis, Rogue & Unmanaged Assets, Critical Systems & Blast Radius, and Internet Exposure — each with optional scoping questions. **Start report thread** opens a new thread that generates the report, so the result arrives as an artifact you can review before opening or sharing it.

Limits

- At most 4 AI runs per user and 12 per account may run at once, and one per thread.
- A single request (a prompt and its context) is capped at 64 KiB.
- A chat turn runs up to 5 minutes; a guided report turn up to 10 minutes.
- A chat turn makes at most 8 tool calls, or 20 when the question asks for a report, deep investigation, comparison, chart, or trend.
- One turn can produce at most 100 artifacts, 512 KiB each and 2 MiB in total.

Retention

AI data has fixed retention windows, separate from the [organization-level data retention settings](#):

- **Threads are removed 180 days after their last use.** Every update pushes the clock back 180 days, so an active thread never expires and an abandoned one is removed half a year after it was abandoned. Renaming a thread does not reset the clock. Thread rows show the countdown — expires in N days within 30 days, a date beyond that.

- **Unshared artifacts expire with their thread.** When a thread expires or is deleted, the unshared artifacts it produced are removed with it, attached files included.
- **Shared artifacts are kept indefinitely** while shared. Unsharing one restores the thread-derived expiry; deleting it removes it immediately.
- **Run records are temporary.** The job record and streamed event log behind a run are kept about 15 minutes while running and 24 hours after the run ends; the durable results are the thread and its artifacts.

Cleanup runs daily, and expired items disappear from the console at their expiry time. These windows are not configurable; you can delete individual threads and artifacts at any time.

Using the inventory

The [inventory](#) page is the heart of runZero Network Discovery and the key to understanding what is on your network. The inventory displays all assets within the Organization and can be sorted, filtered, and exported to obtain specific views of the environment.

Understanding assets

An [asset](#) within runZero is defined as a unique network entity. Assets may have multiple IP addresses and MAC addresses and these addresses may change as the environment is updated. runZero tracks assets based on several heuristics, including MAC address, IP address, hostnames, and fingerprint results for the operating system and running services.

In most cases, runZero can accurately follow assets over time in environments using DHCP, even across remote subnets. For external networks, scans that are initiated with fully qualified hostnames will consolidate assets based on the hostname, which allows for consistent asset tracking for cloud-based external systems with dynamic IP addresses.

Within an organization, assets are isolated by site, and each site can have address space that overlaps with other sites. Sorting the Inventory view based on the site column can help in these scenarios, as can filtering the Inventory based on a specific site name.

The search field allows the inventory to be filtered based on the specified criteria. Please see the [search query syntax](#) documentation for specific details.

In addition to viewing assets, the Inventory page provides [data export](#) functionality, along with the ability to select assets, and specify the comments field. The *Rescan* action can be used to selectively rescan specific systems from the inventory, while the *Remove Assets* and *Purge Assets* can be used to permanently remove data from the inventory view.

An inventory query can also be shared as a read-only snapshot with people outside of runZero using [external reporting](#).

The *Reports* button provides quick access to key reports from the runZero [reports page](#).

Supported Asset Types

IT

- ADSL Modem
- ADSL Router
- ATM
- ATM DSL Unit
- BMC
- Broadband Router
- Cable Modem
- Desktop

- Docking Station
- DOCSIS Cable Modem
- DSL Modem
- DSLAM
- DSU/CSU
- Ethernet Adapter
- Fax Server
- Firewall
- Frame Relay
- Hypervisor
- IDS
- IPS
- JTAG Adapter
- KVM
- Laptop
- Lights Out Management
- Load Balancer
- Mainframe
- Management Console
- Management Processor
- Media Gateway
- Media Server
- Multiplexer
- NAC
- NAS
- Network Appliance
- Network Management
- Network Scanner
- Onboard Administrator
- Optical Line Terminal
- PABX
- Paging Terminal
- PBX
- Power Device
- Powerline
- Print Server
- Printer
- Remote Terminal
- Router
- SD-WAN Appliance
- Security Appliance
- Security Gateway
- SIP Gateway
- SSL-VPN
- Storage
- Storage Appliance
- Tablet
- Tape Library
- Terminal Server
- Thin Client

- USB Server
- VoIP Gateway
- VoIP Router
- VoIP Server
- VoIP Switch
- VPN
- WAF
- WAN Accelerator
- WAP
- Web Proxy
- Wireless Controller
- Wireless Dock
- Wireless Ethernet Bridge
- Wireless LAN Controller
- WLAN Repeater

OT

- AC Drive Device
- BACnet Router
- Building Automation
- Data Logger
- DCS
- Device Server
- Engineering Workstation
- Ethernet IO Module
- Frequency Converter
- General Purpose Analog I/O
- General Purpose Discrete I/O
- High-Speed Counter
- Historian
- HMI
- HMI Controller
- IED
- Industrial Control
- Industrial Robot
- Motion Control
- Motion Controller
- Motor Drive
- PLC
- Position Controller
- Power Relay
- Process Controller
- Protocol Gateway
- PXI Switch Module
- Redundancy Module
- Relay Controller
- Remote Access Server
- RTU
- Safety Controller

- Safety PLC
- SCADA
- Servo Controller
- Servo Drive
- SIS
- Soft Starter
- Specialty I/O
- Traffic Control
- Variable Frequency Drive
- VFD

IoT

- 3D Printer
- Access Control
- Access Controller
- Air Quality Monitor
- Alarm Panel
- Amazon Device
- Audio Encoder
- Audio/Video Switch
- Automobile
- AV Appliance
- AV Controller
- AV Receiver
- Barcode Scanner
- Check Scanner
- Clock
- Copier
- Dishwasher
- Display Controller
- DVR
- Environment Control
- Exercise Equipment
- Game Console
- Handheld Blood Analyzer
- Handheld Scanner
- Home Appliance
- IoT
- IP Camera
- IP Phone
- IPTV
- IR Network Adapter
- Irrigation Control
- Lab Equipment
- Laboratory Instrument
- Light Bulb
- Light Switch
- Mattress
- Media Player

- Media Receiver
- Multifunction Device
- Musical Instrument
- Network Audio
- NVR
- PDU
- Point of Sale
- Power Meter
- Presenter
- Projector
- Radio
- Room Controller
- Scale
- Security System
- Sensor
- Set-Top Box
- SIP Device
- Smart TV
- Smart Watch
- Thermal Imager
- UPS
- Vacuum Cleaner
- Video Conferencing
- Video Decoder
- Video Encoder
- Voice Appliance
- Voice Assistant
- VoIP Intercom
- VR Headset
- Water Meter
- Whiteboard
- Wireless Presenter

Loading assets

Data is loaded into the inventory using the **Scan** and **Import** buttons. The results are analyzed and merged, updating asset information as necessary.

The **Scan** button has two options: Standard Scan and Full RFC 1918 Discovery. The latter is an easy way to set up a fast scan of all private range IP addresses. You can then use the [coverage reports](#) to check for assets in unexpected private address ranges.

The **Import** button has two options. Importing runZero scan data allows you to import data. This means you can scan networks that have no connectivity to the internet, and still view the results in the runZero console. It's also useful for reprocessing old scan data so that you can use the [site compare](#) feature to see how assets have changed over time.

Bulk asset update

The bulk asset update feature allows you to modify assets by [exporting a CSV](#) using the **Export** button, making changes to the data in a spreadsheet program or text editor, and then importing the result back into runZero with the **Import** button. This feature will update existing assets that have a matching `id` value in the organization.

The fields listed below can be updated through the bulk asset update:

- Type
- Operating system
- OS version
- Hardware
- Comments
- Tags
- Owner
- Names
- Domains

The `type`, `os`, `os_version`, and `hardware` fields only accept a single value. The `comments`, `tags`, `owner`, `names`, and `domains` fields each accept multiple values, and a space-delimited list of `field=value` pairs is the standard syntax. The `tags` field can also be specified without `tag=` as just a space-delimited list of values.

Only modifications to the `tags`, `comments`, and `owners` fields will be retained through subsequent scans, any changes to the other supported fields will be overwritten by the latest scan data.

Removing tags from assets

To remove a tag from an asset or group of assets, first select the asset or assets in the inventory using the **Select** button in the toolbar. Then click **Modify** and choose **Set asset tags**. In the tag editor, prefix the tag you want to remove with a minus sign (-). This removes the matching tag while leaving any other tags on the asset in place.

```
-tag_name
-tag_name=value
```

For example, enter `-env` to remove the `env` tag, including tags such as `env=prod`. To remove only a specific tag value, enter `-env=prod`; other `env` tags remain in place.

This pattern can also be used in bulk updates when modifying multiple assets at once. If you are editing the `tags` field in a bulk asset update CSV before using **Import**, use the same `-tag` syntax to remove tags from the selected assets.

To remove all tags from a single asset, open the asset details page, click **Manage** in the top-right corner, and choose **Unmerge asset**. This deletes the current asset record and re-processes it, which removes stale tags and keeps the valid tags from the most recent tasks.

Connecting to other systems

The **Connect** button lets you connect runZero to other systems. The [integrations](#) you're able to connect depends on your license level, but may include tools like cloud and virtualization platforms, endpoint protection solutions, identity and access management tools, and vulnerability and risk platforms. These inbound integrations can also be configured as [scan probes](#) if required.

Viewing services

The Inventory page has a submenu labeled *Services*. This changes the table of data from an asset-focused view to a service-focused view. For each asset, you will see one row for each service runZero detected.

Like the main asset view, the services view has a full search interface. You can filter services by protocol, port, and [many other criteria](#), using the [runZero search language](#).

Note

runZero does not enumerate Windows services through the Service Control Manager because active scanning is unauthenticated. In some cases, you may still see limited Windows service-related details reported from unauthenticated DCE/RPC Endpoint Mapper (EPM) data, which is used primarily to identify WiFi and WLAN service availability.

Viewing screenshots

If the runZero Explorer has access to Google Chrome, it will attempt to take screenshots of web pages it finds while scanning your network. (This feature can be disabled in the scan options when setting up the scan.)

You can view the screenshots for all of your assets via the *Screenshots* submenu, and click through to the asset records for full details.

Viewing software

The inventory page has a submenu labeled *Software*. This flips the table of data from an asset-focused view to a software-focused view. For each asset, you will see one row for each software detected by runZero or a supported integration.

Like the main asset view, the software view has a full search interface. You can filter software by vendor, product, and [many other criteria](#), using the [runZero search language](#).

Viewing vulnerabilities

The inventory page has a submenu labeled *Vulnerabilities*. This flips the table of data from an asset-focused view to a vulnerability-focused view. For each asset, you will see one row for each vulnerability detected by a supported integration.

Like the main asset view, the vulnerability view has a full search interface. You can filter vulnerabilities by CVSS score, name, CVE, and [many other criteria](#), using the [runZero search language](#).

Viewing certificates

The *Certificates* submenu provides a searchable, sortable [certificates inventory](#) of all of the encryption certificates the runZero Explorer encountered while scanning.

Like the other views, the certificates view has a search interface and can be sorted.

Viewing wireless networks

If the machine running the runZero Explorer has a working WiFi adapter and appropriate system tools installed, the Explorer will attempt to scan for nearby wireless networks. The *Wireless* submenu will show the results of the scan.

The tools required are:

- Windows: netsh.exe (part of modern Windows releases)
- macOS: Airport Utility
- Linux: iwlist, often available via the `wireless-tools` package.

Viewing users and groups

The inventory pages for users and groups contain data imported from directory services such as Active Directory. The user directory and group directory can be populated using third party integrations listed under *Directory services* in the *Integrate* drop-down menu.

Understanding assets

runZero treats assets as unique network entities from the perspective of the system running the Explorer. An asset may have multiple IP addresses, MAC addresses, and hostnames and it may move around the network as these attributes are updated. runZero tries hard to follow assets by correlating new scan data with the existing inventory, using multiple attributes.

An asset is always associated with a single site. If the same system happens to be covered by multiple sites, these will be treated as different assets, and will only be correlated against assets within their respective site. This separation by site allows the same network to be scanned from multiple perspectives and compared in a single view within the organization.

After each scan, all assets within the corresponding site are updated. If a system is identified that doesn't match an existing asset, a new asset will be created. If an asset is part of the site and it is not found during a scan, it will be marked as offline. If an asset is not correlated, due to substantial changes to the fingerprint (for example, a new network adapter was installed and the firewall was enabled), the previous asset will be marked as offline, and a new asset will be created to track the new configuration. This can lead to some level of duplication within a site, but these duplicates are usually marked as offline, and can be safely ignored or removed from the inventory by hand.

Asset fields

The following asset fields are available.

Primary addresses

runZero will report at least one and often multiple primary IP addresses for a given asset. These addresses can encompass multiple network interfaces but will only be displayed as a primary address if runZero has scanned it. This requires that the address is within the scan scope of one or more runZero scans.

Secondary addresses

runZero may report one or more secondary addresses, based on network response probes. These are IP addresses that were detected on the asset but were not within the scan scope. Secondary address detection is critical when trying to identify systems that bridge networks that should be isolated.

Hostnames

runZero may report one or more hostnames. These names can be obtained from the initial DNS lookup (when hostnames are provided in the scan scope), from DNS PTR lookups during the scan, and by extracting names advertised within network probe responses.

Operating System (OS)

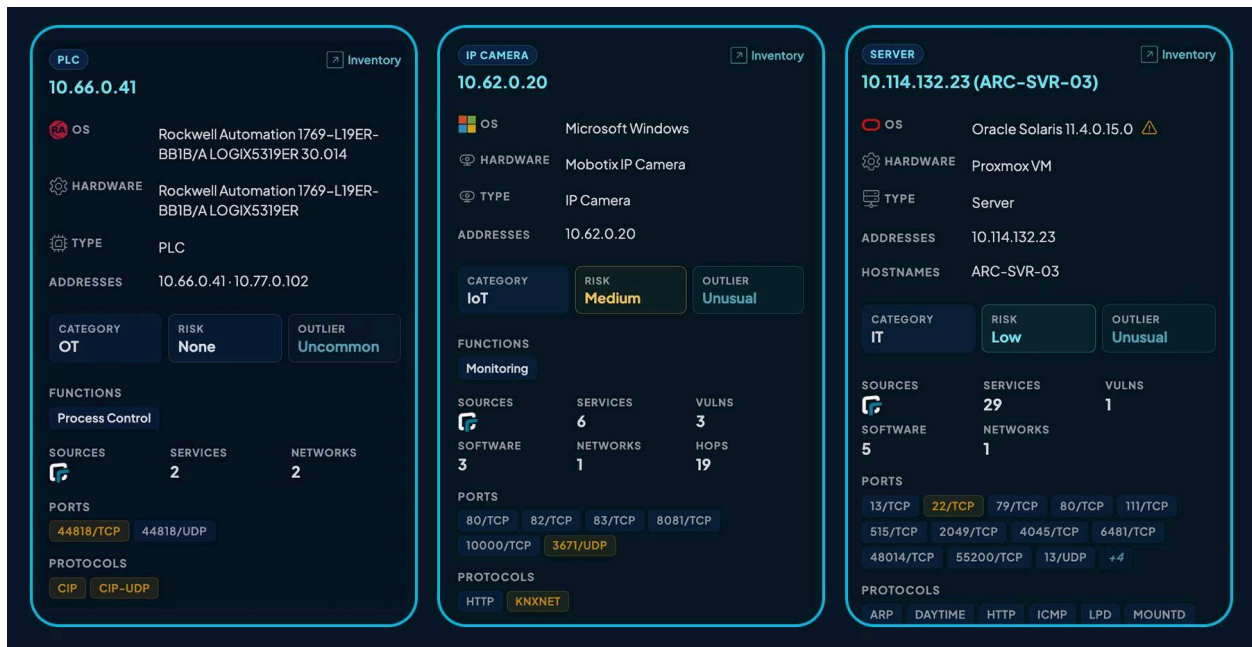
runZero attempts to fingerprint, and failing that, guess at the operating system running on each asset. If limited information is available, this field may be empty.

Type

runZero attempts to determine the general device type through analysis of fingerprints and running services.

Category

runZero classifies every asset into a high-level asset category so you can quickly separate traditional computing infrastructure from operational technology and connected devices. The category is derived from the device type, vendor, and service protocols (for example, an asset speaking Modbus, CIP, or S7Comm is automatically placed in OT).



Value	Meaning
IT	Traditional information-technology assets: servers, workstations, laptops, network gear, storage, hypervisors.
OT	Operational-technology assets: PLCs, RTUs, IEDs, DCS, SCADA, HMIs, drives, protocol gateways, industrial robots.
IoT	Connected devices that are neither IT nor OT: IP cameras, building-automation sensors, printers, smart-TVs, medical devices, point-of-sale, access control.

Query with `category:OT`, `category:IT`, or `category:IoT`. The category can be overridden manually by setting the `category` asset attribute to one of these three values.

Functions

In addition to a category, OT and OT-adjacent assets are tagged with one or more asset functions describing what the device does. Functions are multi-valued — a single device can carry several at once (for example, a substation IED is both `Process Control` and `Monitoring`).

Function	Typical assets
Process Control	PLCs, RTUs, IEDs, DCS, SCADA controllers, supervisory-control software.
Safety	Safety controllers, safety PLCs, Safety Instrumented Systems (SIS).
Human Interface	HMIs, operator panels, touchscreens, control-room displays.
Engineering	Engineering workstations and laptops used to program, configure, or maintain OT devices.
Monitoring	Sensors, meters, condition-monitoring units, tank monitors, environmental sensors used for observation only.
Environmental	Building-automation, HVAC, lighting, refrigeration, room and climate controllers.
Communications	Field-bus gateways, protocol translators, serial-to-IP adapters, modems used inside OT segments.
Networking	Switches, routers, firewalls, and data diodes that segment or interconnect OT zones.
Remote Access	Jump servers, VPN concentrators, and remote-access servers used to reach OT environments.
Data Management	Historians, data loggers, and other systems whose primary role is storing operational data.
Power Management	UPS units, PDUs, power meters, transfer switches, generators.
Physical Security	Access-control panels, door controllers, alarm panels, surveillance NVRs.
Cyber Security	OT-resident IDS/IPS, security gateways, and other dedicated cyber-security appliances.
Management	Management appliances and consoles used to administer OT devices and networks.

Query with `function:"Process Control"`, `function:Safety`, `has:functions`, etc. Function values can also be set manually by editing the asset's `functions` attribute.

Hardware

runZero attempts to determine the physical (or virtual) hardware if enough information is present.

MAC addresses

runZero may be able to enumerate one or more MAC addresses from the asset. MAC addresses are pulled from ARP if available, but also several network services that can return MAC address information across routed segments.

Services

runZero tries to detect approximately 100 TCP services by default, along with several useful UDP services. These services are in addition to ARP and ICMP. The services field contains a list of the most recently recorded services for the asset.

Round Trip Time (RTT)

runZero records the amount of time certain probes take in order to get a rough sense of the latency between the Explorer and the asset.

Detected by

runZero records which probe was used to identify an asset. For assets that are on remote subnets and have firewalls in place, this field indicates what service was used to obtain a response.

Alive status

runZero tracks whether a given asset was found during the most recent scan where its site was in scope. If the asset was not found, it will be marked as offline until a following scan detects it again.

First seen

runZero tracks the initial timestamp when an asset was first identified.

Last seen

runZero tracks the last timestamp when an asset responded to a probe during a scan.

Explorer

runZero tracks the most recent Explorer to detect an asset through active scanning or passive discovery. The Explorer is not typically updated for third-party connections using an Explorer, because the asset was detected by the third-party and not the Explorer.

Outlier score

runZero computes an outlier score for all assets in your inventory. The outlier score has a value from 0 to 5 (inclusive). It is a heuristic that aims to indicate how unusual the asset is, compared to all of the others in the inventory.

Outlier scores are computed by examining key properties of the asset and its services, working out which values are unusual (infrequent) across the organization, and then computing how many unusual properties each asset has. The more unusual properties, the higher the asset's outlier score will be.

Understanding findings

Findings simplify vulnerabilities, misconfigurations and best practices into a prioritized, curated and aggregated list that helps you identify and remediate the most critical risk in your environment. runZero Findings are available from the *Findings* menu, and from the Risk Management dashboard.

What are findings?

Findings highlight the risks attackers are most likely to target. This enables security teams to focus remediation efforts on risks with real operational impact.

Findings group similar vulnerabilities and misconfigurations together, providing a more holistic view of the risk they represent. Each finding contains a description of the risk, remediation steps, risk rankings, and an individual list of all assets and entities affected by the risk.

Finding categories

Each finding is placed into a category based on the type of risk it represents. Categories include:

- Internet Exposure: identifies assets and services that are potentially unintentionally exposed to the internet
- Certificates: expired or soon-to-expire certificates as well as widely shared private keys
- Vulnerability: actively exploited vulnerabilities or critical vulnerabilities that runZero believes are critical to address
- End-of-Life: operating systems, hardware and applications that have reached End-of-Life (EOL) or End-of-Service (EOS), and are no longer supported by the vendor
- Open Access: network services such as unauthenticated databases and sensitive applications that are accessible without authentication
- Compliance: assets and services that violate security best practices
- Best Practice: general best practices that cover insecure authentication, service misconfiguration and obsolete protocols
- Rapid Response: emerging and novel threats covered in detail by runZero Rapid Response blog posts

Findings list

Findings can be found by using the main navigation menu directly below the Inventory item. They can be [searched](#), sorted and exported similarly to other views within the console.

Finding details

When clicking on an individual finding's name to see its details, you will see an overview of the risks discovered, any associated external resources, and pertinent remediation information to

help you address the risk. Below the finding summary, you will find a list of all the asset instances that the finding applies to, so that you can quickly prioritize remediation.

How are Findings different from Vulnerabilities?

A vulnerability is a specific security issue, usually closely tied to a specific CVE or security advisory. Vulnerabilities are found by the runZero Explorer or imported via one of the many supported integrations. There can be hundreds of thousands of vulnerabilities for any given asset, and even more across your entire environment.

A finding is a curated, aggregated and prioritized list of risks that are most likely to be targeted by attackers. A single finding may group together several similar vulnerabilities. Findings are not always tied to a CVE and may include misconfigurations, best practices, and other security issues that may not make sense as a vulnerability.

Asset risk and criticality

runZero is able to help users assign and evaluate risk and criticality levels to the assets in their inventory. This can help prioritize risk mitigation or vulnerability remediation efforts by allowing users to quickly identify the assets in their organization with the highest levels of risk or criticality.

Defining risk and criticality

The **risk** level assigned automatically to assets in your inventory is inferred from the risk associated with vulnerabilities or risky configurations on that asset and defaults to the value `none`. Vulnerability risk level may be defined by the vulnerability management solution the vulnerability records are ingested from, or by the risk level assigned to a query vulnerability. The risk level can be overridden, in which case the override is retained until the asset or vulnerability is deleted. For vulnerabilities ingested from integrations, this may occur when the source no longer reports the vulnerability on that asset.

The **criticality** level is assigned manually and defaults to the value `unset`. This value is intended to be used to denote the criticality or importance of an asset to your organization. As an example, you may choose to assign business-critical systems such as database and web servers a `critical` level, but normal enduser systems a `medium` level.

Assigning asset risk and criticality

Both asset risk and criticality can be assigned via the asset inventory. Asset criticality can also be assigned with alert rules.

Superusers, administrators, and users can add or modify asset risk and criticality levels, and can reset risk assignment or remove criticality assignment from assets.

Risk and criticality in the asset inventory

Follow these steps to set risk and criticality through the asset inventory:

1. Select all the assets you wish to update, applying a query filter if needed.
2. Click the *Modify asset risk* or *Modify asset criticality* button to open the relevant popup.
3. Select the level of risk or criticality you wish to apply to the asset(s).
4. Click *Override risk* or *Set criticality* to apply your changes.

Applying criticality with rules

To automatically apply asset criticality values to assets after a scan, [create an alert rule](#) by going to **Alerts > Rules** and clicking the *Create rule* button:

1. Select an inventory query you wish to use, such as the `asset-query-results` rule type, then click *Configure rule*.

2. Configure any desired settings.
3. Set the **Action** to *Modify asset*.
4. Select an option from the *Asset criticality* menu.
5. Save the rule.

This rule will now add the specified asset criticality level to all assets that match the rule when a scan completes.

Asset risk report

The **Asset risk report** provides visibility into the risk and criticality levels across your asset inventory. To run the Asset risk report, go to **Reports > Asset risk report** and click the **Asset risk report** button. Configure the following fields:

1. **Sites:** Select a site of assets to include in the report, or leave the default *All Sites*.
2. **Minimum risk:** Choose the minimum asset risk level to include.
3. **Minimum criticality:** Choose the minimum asset criticality to include.
4. **Top vulnerabilities per asset:** Set this field to an integer between 0 and 20. If the value is set to an integer between 1 and 20, the report will list up to that number of the top vulnerabilities detected on each asset. The top vulnerabilities are identified by sorting the vulnerability results for each asset by risk rank, then risk score, then severity rank, then severity score.
5. Click **Create report** to generate the results.

The resulting report is grouped by asset criticality level and then sorted by risk level. The results can be exported as JSON Lines (.jsonl), a JSON document (.json), or CSV (.csv).

Certificates

runZero automatically collects TLS certificates as part active scans and passive network monitoring. These are stored in the Certificates Inventory, which you can access from the *Inventory* menu, *Certificates* sub-menu. Although the most sources of TLS certificates are web servers, certificates are also acquired from SMTP, RDP, and other TLS-aware services.

Certificates have a validity period. In the inventory view, the *Valid from* and *Valid until* values are color coded to show if the certificate is not valid yet, has expired, or is soon to expire.

A certificate has an *issuer*, the authority which signed the certificate. The issuer may be the same as the subject, in which case the certificate will be marked as self-signed certificate.

Certificates rely on two kinds of algorithms:

- The public key algorithm, used for decryption and verification.
- The signature algorithm, used to produce a content hash that is signed using the public key algorithm.

The security of a public key algorithm depends on the key size. This is shown in bits. How large a key is considered secure will depend on the algorithm. For example, for RSA a key size should be 2048 bits or larger to be considered secure; whereas for ed25519, all keys are 256 bits, and that's considered secure.

The certificate inventory view has a button for each certificate to hide it from the inventory. This can be useful for embedded devices that have certificates you can't change, that you don't want to see in the inventory. If you later decide you want to see which certificates you've hidden, you can use the search query `hidden:true`.

TLS certificates

TLS certificates have three sets of names for their hosts:

- The Subject, an X.509 distinguished name (DN) made up of one or more attributes, each with a value.
- The Subject Alternative Name lists. These provide the values web browsers check against when connecting to a server. SANs are stored in a certificate in separate fields by type (DNS, IP, email or URI).
- The Common Name, which was once used by web browsers but is now ignored by them. It is typically in X.509 format.

The Names shown in runZero for a TLS certificate are assembled from the four different kinds of SANs, plus the Subject DN.

Each TLS certificate has a subject key ID used to identify the subject of the certificate, and an authority key ID used to identify the signing authority. These are used to link certificates together in a chain — the authority key ID of each certificate is the subject key ID of the next certificate in the chain.

The issuer for a TLS certificate is included as a DN in the certificate.

TLS certificates have a flag to say if they are intended for use as a certificate authority or (CA). The flag doesn't mean that they *are* a CA, or that they are trusted; just that the certificate has that flag set. CA certificates are usually self-signed.

The values shown to identify TLS certificate algorithms are mostly taken from RFC 5280, and should correspond closely to the values used by OpenSSL.

The self-signed value indicates whether a certificate appears to be self-signed, based on its issuer, subject, and (if present) their IDs. A certificate may be flagged as apparently self-signed but not have a valid signature according to some software, as different software may implement more or less strict checks.

The certificate details page has a button to download the certificate in PEM format. You can also download the entire chain of certificates known to runZero as a single PEM file.

Geolocation

runZero resolves a geographic location for every asset where it has enough information to do so, and uses those locations to power the [World Map](#), inventory search, and exports. This page describes the different sources runZero uses for asset geolocation, how user-supplied location tags are interpreted, and the order in which sources are applied.

Where locations come from

runZero builds asset locations from five independent sources during each analysis pass. Every location is recorded with provenance, so each pin on the World Map can be traced back to the specific attribute, integration, or tag that produced it.

Source	Description
attribute-coord	Paired numeric latitude/longitude attributes from integrations or imports.
attribute-ip	Egress / public IP addresses resolved through MaxMind GeoLite2.
attribute-text	Discrete country / city attributes, plus free-form <code>snmp.sysLocation</code> -style strings.
cloud-region	AWS, Azure, GCP, OCI, IBM Cloud, and DigitalOcean region codes resolved to a region centroid.
geolite-isp	GeolP lookup of the asset's primary scanned IP through MaxMind GeoLite2.
manual	User-supplied geo* asset tags .

An asset can carry locations from more than one source at a time. For example, a cloud workload may have a `cloud-region` location from its AWS region, an `attribute-ip` location from its public NAT address, and a `manual` location set by a tag — all three appear as separate pins on the [World Map](#), each labelled with its source.

When multiple sources agree on the same coordinate the entry is de-duplicated.

GeolP lookups against MaxMind

runZero ships with the MaxMind GeoLite2 City and ASN databases. These are used to resolve any public IP address that runZero observes for an asset into a latitude, longitude, country, subdivision (state/province), city, postal code, timezone, continent, ASN number, and ASN organization name.

Private RFC1918 and RFC6598 addresses are rejected by the lookup layer, so a private IP never produces a location.

Primary and hinted external IPs

When an asset is scanned, runZero records the public IP it observed for that asset. The primary scanned address is looked up directly against GeoLite2 and produces a location with source `geolite-isp`.

When passive sources or integrations report additional public addresses for the same asset — for example a secondary NAT address, or an alternate egress on a multi-homed device — those are also looked up. The resulting locations are recorded alongside the primary so that an asset visible at two different egress points appears in both places on the [World Map](#).

Egress and public-IP attributes from integrations

Many integrations report the public / external / NAT IP address an asset was observed from. runZero ingests these and resolves each through GeoLite2. The lookup produces both a city-level location and an ASN annotation for the pin.

The attributes consumed by the egress IP pipeline are:

Attribute	Set by
<code>externalIP</code>	CrowdStrike, SentinelOne
<code>publicIP</code>	Azure (VM/database/load balancer), Microsoft Defender for Endpoint, AWS
<code>publicIPs</code>	AWS
<code>natIP</code>	Google Cloud Platform
<code>wanIPAddresses</code>	Google Workspace
<code>discover.externalIP</code>	CrowdStrike

The full set of unique public IPs seen across these attributes is also written back to the asset as an `egressIPs` attribute, so you can query the underlying values directly.

Cloud region codes

When an integration provides a cloud region code — for example AWS `us-east-1`, Azure `eastus`, or GCP `eu-west-4` — runZero maps the code to the published centroid of that region. This gives the asset a country, subdivision (where applicable), city, and continent without requiring a GeoIP lookup.

The cloud region pipeline reads from attributes including `region`, `cloud.region`, `aws.region`, `azure.region`, `azure.location`, `gcp.region`, `gcp.zone`, `oci.region`, `availabilityZone`, `availability_zone`, `zone`, `placement.region`, and `placement.availabilityZone`. Unknown values (for example a generic `region=Northeast` tag) are ignored.

Discrete country and city attributes

Some integrations report location as separate `country` (or `countryCode`) and `city` fields. runZero pairs these up and resolves the combination to the matching GeoNames city centroid. If only a city is present, the global highest-population match is used.

The country fields consumed are: `country`, `countryCode`, `country_code`, `host.country`, `host.countryCode`, `host.location.country`, `host.location.countryCode`, `discover.country`, `discover.countryCode`, `custom.country`, and `site.country`.

The city fields consumed are: `city`, `host.city`, `host.location.city`, `discover.city`, `custom.city`, and `site.city`.

Free-form location strings

Free-form location strings from SNMP and integrations are parsed best-effort. The parser strips address noise — house numbers, suite/floor markers, PO boxes, postal codes, street suffixes like `St`, `Ave`, `Blvd` — and matches the remaining trailing tokens against the bundled GeoNames tables.

The free-form attributes consumed are: `location`, `host.location`, `site.location`, `custom.location`, `location.address`, `snmp.sysLocation`, and `shodan.snmp.location`.

This is best-effort by design. SNMP `sysLocation` has no standardized format, and many devices report a campus or rack identifier rather than an address. When the parser cannot anchor the string to a known city, region, or country, no location is recorded.

Manually setting an asset location

You can pin an asset to a specific geographic location by setting one of the `geo*` tags below. Each tag has a fixed shape, and the value tells runZero where the asset is. The location appears on the [World Map](#), in exports, and in queries the next time metrics are recalculated for the asset.



Manual `geo*` tags do not take effect immediately. They are picked up the next time runZero recalculates metrics for the asset, which happens automatically as part of normal task processing (scans, integration syncs, and scheduled metrics tasks). To force the change to apply right away, go to **Dashboard** → **Recalculate metrics**.

TL;DR — set one tag. Use `/` to separate fields.

```
geoCity = Austin/TX/US
```

```
geoCity = Springfield/IL/US
```

```
geoLatLng = 30.2672/-97.7431
```

Available tags

Tag	Shape	Example
geoLatLng	lat/lng	geoLatLng = 30.2672/-97.7431
geoContinent	continent name or two-letter alias (EU, NA, ...)	geoContinent = Europe
geoCountry	country (see Country values)	geoCountry = US
geoRegionCountry	region/country	geoRegionCountry = TX/US
geoRegion	region/country or a single token (state, country, or continent)	geoRegion = Texas
geoState	alias of geoRegion	geoState = TX/US
geoProvince	alias of geoRegion	geoProvince = Ontario
geoCity	city/country OR city/region/country	geoCity = Austin/US

You only need to set **one** tag per asset. If you set more than one, runZero uses whichever tags match their shape. Tags whose values do not match their expected shape are silently ignored — runZero will not guess.

When to use which tag

- Use **geoLatLng** if you have exact coordinates from a CMDB, building floor plan, or GPS source.
- Use **geoContinent** when you only know the continent.
- Use **geoCountry** when you only know the country.
- Use **geoRegionCountry** when you know the country and the state/province/region but not the city.
- Use **geoRegion** (or its synonyms **geoState** / **geoProvince**) when you want runZero to accept any plausible region-ish value: a region/country pair, a unique state name (Texas, Ontario, Bavaria), a country code, or a continent.
- Use **geoCity** in every other case. The two-field form is enough for major cities; use the three-field form to disambiguate names like Springfield, Portland, OR Cambridge.

Field separator

The separator between fields is always a forward slash (/). The characters -, ., ,, and _ are **not** field separators because they appear inside real city and region names — Winston-Salem, St. Louis, Stratford-upon-Avon, St-Lawrence.

```
geoCity = Austin/US      ✓ city/country
geoCity = Austin TX US   ✗ no separators
```

`geoCity = Austin-TX-US` ✗ hyphen is part of names, not a separator
`geoCity = Austin,TX,US` ✗ comma is part of names, not a separator

If you set a tag through the UI or API and the value contains spaces, runZero stores those spaces as underscores. The parser treats `_` and a space identically inside a field, so `geoCity = San Francisco/CA/US` and `geoCity = San_Francisco/CA/US` parse identically.

Country values

Any of these forms work for the country field:

Form	Examples
ISO 3166-1 alpha-2	US, DE, GB, JP
ISO 3166-1 alpha-3	USA, DEU, GBR, JPN
English name	United States, Germany, Japan
Common alias	UK and Britain (both → GB)

Country values are case-insensitive. When an alias is recognized, runZero normalizes it to the ISO alpha-2 code shown on the map.

Region values

Region values are state, province, or admin region names. runZero recognizes both short codes and full names for the countries below; for other countries, runZero accepts the [GeoNames admin1 code](#) directly (often a numeric value like 02 or 16).

Country	Examples of accepted region values
United States	TX, Texas, CA, California
Canada	ON, Ontario, BC, British Columbia, Québec
Australia	NSW, New South Wales, VIC, Victoria
United Kingdom	England, Scotland, Wales, Northern Ireland
Ireland	Connacht, Leinster, Munster, Ulster
New Zealand	Auckland, Canterbury, Wellington

If runZero does not recognize the region, the asset still resolves at the country level — the region tag is informative but never blocking.

City matching and GeoNames

City and region matches are performed against the bundled GeoNames data. The matcher is fuzzy: `St. Louis`, `St Louis`, `St-Louis`, `Saint Louis`, and `saint_louis` all resolve to the same record. The

canonical GeoNames spelling is what shows on the map and in exports, so your tag values do not need to be perfectly formatted to look right.

When a city name is ambiguous (Springfield, Cambridge, Portland), use the three-field form to disambiguate:

<code>geoCity = Springfield/IL/US</code>	✓ disambiguated
<code>geoCity = Springfield/MO/US</code>	✓ different city
<code>geoCity = Springfield/US</code>	△ resolves to whichever Springfield is largest

Examples

Pinning a corporate office:

```
geoCity = Austin/TX/US
```

A device with exact coordinates:

```
geoLatLng = 30.2672/-97.7431
```

A subnet known to be in Germany but unknown city:

```
geoCountry = DE
```

A device in a region but unknown city:

```
geoRegionCountry = Bayern/DE
```

What happens if the value is wrong?

Each tag has a fixed shape and a strict matcher. When the value does not fit the shape — or names a place that runZero's bundled GeoNames data does not recognize — the tag produces no manual location, and no manual pin is shown on the map for that asset.

This is intentional. To debug:

1. Confirm the field separator is /, not -, ., or ..
2. Confirm the country value resolves (try the ISO alpha-2 code).
3. For `geoCity`, try the three-field form to disambiguate the city.
4. As a last resort, use `geoLatLng` with explicit coordinates.

Searching by location in the asset inventory

The asset inventory supports a `geo` keyword (with synonyms `geo_within` and `geo_radius`) that matches assets whose geolocation falls inside a circular area:

```
geo:<lat>,<lon>[,<radius>]
```

- `<lat>` and `<lon>` are decimal degrees (30.2672, -97.7431).

- `<radius>` is optional and defaults to **50 km**. Supply a number followed by `km`, `mi`, or `m` (for example `25km`, `15mi`, `5000m`). Plain numbers are treated as kilometres.
- The radius is clamped between 1 metre and 1,000 km.

Examples:

```
geo:30.2672,-97.7431      # within 50 km of downtown Austin
geo:30.2672,-97.7431,25km  # within 25 km
geo:52.5200,13.4050,100mi  # within 100 miles of Berlin
geo_within:51.5074,-0.1278,5km  # synonym
```

Match is true if **any** location source on the asset falls inside the circle. This means a workload with both a cloud region centroid and a public IP egress in the same metro area matches the same `geo: query`.

The same `geo:` keyword is available on the [services inventory](#).

When locations are refreshed

Asset locations are recalculated on every analysis pass. Each source's previously recorded locations are replaced by the new ones, so stale data — for example a `cloud-region` attribute that an integration stopped reporting — is removed automatically on the next pass.

If you change a `geo*` tag value, the new location will appear on the next analysis pass and not before.

Managing ownership

runZero is able to help users track ownership with the ability to configure different types of owners and assign owners to runZero assets and vulnerability records. Ownership coverage can also be tracked as a [goal](#).

Ownership types

Superusers can manage the available types of ownership on the [Account > Ownership types](#) page. Custom ownership types can be configured to meet your needs. Some common ownership types may include **Security owner**, **IT owner**, or **Compliance owner**.

The ownership type requires configuring three fields:

- **Name:** the name of the ownership type.
- **Reference:** whether the ownership type should be correlated with the user inventory, group inventory, or neither.
- **Visibility:** whether the ownership type is visible through the asset inventory and asset details pages.

The default `Asset Owner` ownership type, when visible, will be automatically populated with ownership-related data that runZero can glean from your configured integrations. The name of this ownership type can be changed by a superuser.

The list of ownership types can be prioritized by dragging the types into the preferred order. This will dictate the order in which the types are displayed in the inventory and asset details pages. Only types marked `visible` will be displayed.

Issue ownership

Issues use a separate, built-in `Issue owner` ownership type. Each issue can have one owner, and custom ownership types cannot be assigned to issues. For details about assigning an issue owner and synchronizing it with Jira, see [Issue ownership](#) and [Jira synchronization](#).

Default asset owner

Integrations populate the default asset ownership type. This is a prioritized mapping showing which traits from each integration would update that value. If an asset has more than one of these attributes, the first match in this table would populate.

Integration Source	Device/Object Type	Attribute Name	Reference Type
Google Workspace	Mobile	owner	User
LDAP	Computer	managedBy	User
LDAP	Computer	manager	User

Google Workspace	ChromeOS	orgUnitPath	Group
Defender 365	Device	rbacGroupName	Group
SentinelOne	Device	groupName	Group
Google Workspace	ChromeOS	recentUsers.names	User
Google Workspace	Mobile	email.names	User
Google Workspace	Endpoint	email.names	User
Miradore	Device	user.name	User
Intune	Device	userDisplayName	User
CrowdStrike	Falcon Device	email	User
Google Workspace	ChromeOS	annotatedUser	User
Google Workspace	ChromeOS	recentUsers	User
Google Workspace	Mobile	email	User
Google Workspace	Endpoint	email	User
Miradore	Device	user.email	User
Intune	Device	emailAddress	User
Intune	Device	userPrincipalName	User
SentinelOne	Device	lastLoggedInUserName	User

Assigning owners to assets and vulnerabilities

Once created, custom owners can be assigned via the inventory or through an alert rule.

Superusers, administrators, and users can add or modify owner values, and can remove owners from assets or vulnerability records. Annotators can only add owner values, but cannot modify or remove owners.

Ownership in the inventory

Follow these steps to assign owners through the asset or vulnerability inventory:

1. Select all the assets or vulnerability records you wish to update, applying a query filter if needed.
2. Click the *Manage asset ownership* or *Manage vulnerability ownership* button to open the ownership popup. **Note:** Ownership values applied to an asset will be inherited by unowned vulnerability records on that asset. Vulnerability records with owners defined will not inherit the ownership value assigned to the asset.
3. Click *Add ownership type* and choose which type(s) of owner you wish to apply to the selected assets or vulnerability records.

4. Add the owner value to the field.
5. Click *Save* to apply your changes.

Applying owners with rules

To automatically apply ownership values to assets after a scan, [create an alert rule](#) by going to **Alerts > Rules** and clicking the *Create rule* button:

1. Select an inventory query you wish to use, such as the `asset-query-results` rule type, then click *Configure rule*.
2. Configure any desired settings.
3. Set the **Action** to *Modify asset*.
4. Specify a value for the *Set [ownership type]* field for the ownership type(s) you wish to apply. **Note:** Ownership values applied to an asset will be inherited by unowned vulnerability records on that asset. Vulnerability records with owners defined will not inherit the ownership value assigned to the asset.
5. Save the rule.

This rule will now add the specified owner type and value to all assets that match the rule when a scan completes.

Managing tasks

You can view and manage discovery scans and other background actions from the [Tasks overview page](#). The *Active* and *Completed* task sections will show standard tasks, such as scans and imports, along with their current progress and summarized results. You can [search](#) or [filter](#) the tasks using different attributes.

Task status values

Tasks can have the following status values:

- **New:** The task has been created and is waiting to be picked up by a scanner or connector.
- **Active:** A scan task is in progress and the Explorer is scanning the network.
- **Scanned:** The network scan part of a scan task has completed.
- **Connecting:** A connector task is connecting to the remote system and downloading data.
- **Connected:** A connector task has finished downloading and the data is waiting to be processed.
- **Processing, queued:** Task data has been collected from a scan or a connector task, and is queued for processing.
- **Processing:** Task data is being processed.
- **Processed:** Task data has been processed and runZero data updated.
- **Stopping:** The task was requested to stop and is in the process of doing so.
- **Stopped:** The task successfully stopped.
- **Canceled:** An error occurred which meant that the task could not continue.
- **Paused:** A repeating task has been paused.

Tabs

Tabs on the Tasks overview page allow you to view a filtered subset of active, processing, scheduled, failed, completed, or recurring tasks. Each tab includes a search bar so that you can [search your complete task history](#).

- **Active** shows all tasks currently scanning or connecting
- **Processing** shows all tasks currently processing
- **Scheduled** shows all non-recurring tasks that are scheduled to run
- **Failed** includes all tasks that were unable to finish due to an error or cancellation
- **Completed** lists all completed tasks in chronological order based on when the task finished processing
- **Recurring** shows all recurring tasks

Task details

Clicking on a task will open a card at the top of the page with details. The concentric circles show the progress of the task, with the outer circle tracking the scan or connection progress

and the inner circle tracking the processing progress. The card also includes relevant timestamps and user creation data, as well as site, Explorer, and hosted zone settings. Completed tasks will also include the change summary, and recurring tasks will include a list of past runs.

Scheduled tasks

Scheduled tasks are one-off actions that will be started in the future, while recurring tasks are actions that happen on a regular basis. Recurring scan tasks generate a new standard task on each iteration of their schedule. Both scheduled and recurring scans will only launch if their associated Explorer is online and no other scan tasks are running.

The Tasks page allows *Scheduled* and *Recurring* tasks to be removed and *Active* tasks to be stopped. Please note that stopping an active *Scan* task may take a few moments, as the status of the task is not updated until the Explorer confirms that the scan was terminated.

System tasks

runZero has system tasks that you may notice while searching through your tasks. Here are some examples you might see:

- **Report-based:** a few reports will create tasks when initialized, including:
 - Organization overview report
 - Site and organization comparison
 - External assets report
- **Outlier calculation:** this generates the outlier score seen in the asset details page and will show up with the name `Outlier calculation`
- **Query:** all queries that are pinned to the Dashboard as Insights will show up with the name `Query`

Dismissing failed tasks

Tasks that were unable to finish due to an error or cancellation can be removed from the runZero Console by viewing the task details, clicking the “Dismiss” button and confirming the desired action. Dismissing failed tasks will archive the tasks so that they are no longer accessible.

Reprocessing tasks

You can use the Reprocess Tasks tool on the tasks overview to delete existing assets in the current organization and repopulate asset data from previous scan, import, and integration tasks using runZero’s latest algorithms. This can potentially fix duplicated or incorrectly merged assets. To reprocess tasks:

1. On the Tasks Overview page, open the hamburger menu at the top right of the table and select “Reprocess tasks”;
2. Review the warnings in the modal that opens;

3. Enter a number of days to reprocess (this defaults to 7);
4. Enter the confirmation text;
5. Click "Continue".

The process will take some time to delete the organization's assets and queue tasks that were created within the number of days specified for reprocessing. When it has finished queueing tasks, it will display the number of tasks to be reprocessed. You will then be able to monitor their progress in the console as normal.

Please note:

All assets in the organization will be deleted.

Any tags, risk, criticality, comments, or other manually applied changes will be removed.

During reprocessing, asset information may be incomplete and alert rules may be triggered.

If your organization has a very large number of assets, it is possible for the request to time out before completing asset deletion. In this case, you can retry the request.

The entire process may take several hours to complete.

Tracking goal progress

With runZero goals, users are able to create and monitor progress toward achieving security initiatives. All goal types are supported by the robust query language on the backend. [All types of inventory queries](#) are supported by the goal tracking feature.

There are two types of goals:

- **Saved query** - a goal based on the results of a user-defined or runZero system query against all inventory.
- **Baseline** - a goal based on the results of a user-defined query against a subset of inventory, specified by another user-defined query.

Goal creation

New goals can be created by users whose [default role](#) is user or greater from the [Goals](#) page in the console. Users with viewer-level or greater access will be able to view the goals page and see the goals that apply to organizations they have access to.

To create a new goal, click the [New goal](#) button on the [Goals](#) page in the console.

Creating saved query goals

1. On the *Goal type* tab, select the **Saved query** type.
2. On the *Build goal* tab, complete any desired fields:
 - **Name:** Provide a name for your goal.
 - **Description:** Provide a description for the goal.
 - **Notes:** Add relevant notes about this goal.
 - **Organization access:** By default, a goal will be created for the currently selected organization, but you may select additional organizations as desired. Goals are editable by any user with User role access to every organization associated with the goal, and can be read by any user with Viewer role access to at least one organization.
 - **Query:** Determine the query you want to use to track progress for this goal by selecting the query from the table. If needed, use the search field to find the query you wish to use.
 - **Threshold:** Select whether goal progress will be calculated as *Greater than or equal* or *Less than or equal*, whether goal progress will be calculated as a percent or fixed number, and then set the desired threshold value.
 - **Target date:** Provide an optional target date for achieving this goal.
3. To see a preview of the goal's current state, click *Preview*.
4. Click *Save* when you're ready, or make changes as needed.

Creating baseline goals

1. On the *Goal type* tab, select the **Baseline** type.

2. On the *Build goal* tab, complete any desired fields:
 - **Name:** Provide a name for your goal.
 - **Description:** Provide a description for the goal.
 - **Notes:** Add relevant notes about this goal.
 - **Organization access:** By default, a goal will be created for the currently selected organization, but you may select additional organizations as desired. Goals are editable by any user with User role access to every organization associated with the goal, and can be read by any user with Viewer role access to at least one organization.
 - **Inventory dataset:** Select which inventory type should be searched by the target and baseline queries.
 - **Target query:** Compose the query for what the goal is targeting. The count this query returns becomes the numerator of the fraction representing the state of the goal, which the threshold is then compared to in order to determine goal progress.
 - **Target label:** Provide a label for the target query to appear in the goal breakdown. if a label is not provided, a default will be used.
 - **Baseline query:** Compose the query that defines the scope of the goal. The count this query returns becomes the denominator of the fraction representing the state of the goal, which the threshold is then compared to in order to determine goal progress.
 - **Baseline label:** Provide a label for the baseline query to appear in the goal breakdown. if a label is not provided, a default will be used.
 - **Threshold:** Select whether goal progress will be calculated as *Greater than or equal* or *Less than or equal*, whether goal progress will be calculated as a percent or fixed number, and then set the desired threshold value.
 - **Target date:** Provide an optional target date for achieving this goal.
3. To see a preview of the goal's current state, click *Preview*.
4. Click *Save* when you're ready, or make changes as needed.

Goal progress calculation

Once a goal has been created, the progress will be calculated once a day and when tasks complete. As a result, you may see a different number of metric calculations in the historical line chart depending on the number of tasks that are completed on a given day.

The *overall status* noted in the *Goal details* card on a goal details page reflects the progress toward goal completion for all the configured organizations that you have access to. This may differ for a user with more or less per-organization access.

Note: When a goal applies to multiple organizations, progress is calculated per-organization and extrapolated across all applicable organizations.

Goal events and notifications

The events system includes the following events related to goals:

- **goal-created:** This event fires when a goal is created and saved.
- **goal-updated:** This event fires when a goal is modified and saved.

- `goal-completed`: This event fires when the progress of a goal crosses the completion threshold.
- `goal-lapsed`: This event fires when the progress for a previously completed goal falls below the completion threshold.
- `goal-removed`: This event fires when a goal is deleted.

These events can be used with [channels](#) and [rules](#) to receive notifications about a change in goal status. After a [notification channel](#) has been created, [create a rule](#) for the desired event and select the channel you wish to receive the notification through.

Understanding suppression

runZero supports suppressing records for findings, vulnerabilities and vulnerability groups so that they are not shown in the platform if desired. This can be useful for filtering out false positives for example, or hiding any vulnerabilities that are not relevant to your environment for some reason. Suppressed objects are not deleted, but remain stored within the platform with metadata about when and why they were suppressed, and by whom.

How suppression works

Suppression works differently depending on the type of object being suppressed. When a vulnerability instance is suppressed, it becomes hidden from view on the vulnerabilities by asset inventory. When a finding is suppressed, it becomes hidden from view on the findings list, and any vulnerabilities associated with the finding will also be suppressed. Similarly, when a vulnerability group is suppressed, it becomes hidden from view on the vulnerability groups inventory, and any vulnerabilities associated with the group will also be suppressed.

When a vulnerability, finding, or vulnerability group is suppressed, it will not be considered when displaying an associated asset's risk score or be included in dashboard metrics data from that point forward.

Note: when vulnerability groups or findings are suppressed, and new associated vulnerability instances are found by future scans, those new vulnerabilities will also be suppressed automatically.

If all existing vulnerability instances associated with a finding or vulnerability group are suppressed, but the finding or vulnerability group itself is not suppressed, then the finding or vulnerability group will still be shown in the platform, but it will show a count of zero associated vulnerabilities until new associated vulnerability instances are detected by future scans.

Suppressing and unsuppressing objects

Vulnerabilities, findings, and vulnerability groups can be suppressed or unsuppressed from multiple locations within the runZero platform. Individual findings and vulnerabilities can be suppressed from their respective detail pages using the "Suppress" button found in the page header, and bulk actions can be performed from data grid views such as the vulnerability inventory by asset, vulnerability groups inventory, or findings list.

When choosing to suppress an object, a modal will appear, asking for a reason and an optional comment. Select the most appropriate reason from the dropdown menu, or select "Other" to provide a custom reason in the comment box. Click the "Suppress" button to confirm the desired action, and if successful, the modal will then close and the object will be marked as suppressed.

When a finding or vulnerability is suppressed, a banner appears at the top of its detail page, and suppression can be removed by clicking the "Remove suppression" button and

confirming your intent.

When using a data grid view such as the vulnerability inventory by asset, vulnerability groups inventory, or findings list, multiple rows can be selected using the checkboxes in each row, and then by clicking the "Modify" button and selecting either the "Suppress" or "Remove suppression" option from the dropdown menu. This can be used to quickly suppress or remove suppression from many objects at once. Action buttons within each row can also be used to quickly suppress or unsuppress individual rows.

Searching for suppressed objects

When using the vulnerability inventory by asset, vulnerability groups inventory or findings list, a boolean search query `suppressed` can be used to filter for suppressed or unsuppressed items. For example, to find all suppressed items, use the following query: `suppressed:true`. To display both suppressed and unsuppressed items, a query of `suppressed:any` can be used.

Additionally, each data grid view has a "Suppression" quick filter button with menu that can be used to toggle the view between showing all items regardless of suppression state, only suppressed items, or only unsuppressed items.

Dashboard widget

The "Suppression overview" dashboard widget provides a count for suppressed findings, vulnerabilities, and vulnerability groups at a glance, and can be added to any custom dashboard via the [widget library](#) modal.

Suppression events

A number of runZero system events are triggered when suppression actions are taken. These events can be used to power rules within the [rules engine](#) that take action when suppression events occur. The following events are used for suppression actions:

- `vulnerabilities-suppressed`: fires when a vulnerability is suppressed.
- `vulnerabilities-unsuppressed`: fires when a vulnerability's suppression is removed.
- `findings-suppressed`: fires when a finding is suppressed.
- `findings-unsuppressed`: fires when a finding's suppression is removed.
- `vulnerability-groups-suppressed`: fires when a vulnerability group is suppressed.
- `vulnerability-groups-unsuppressed`: fires when a vulnerability group's suppression is removed

Understanding network segmentation

runZero multi-homed asset detection

Network segmentation is a critical security control for many businesses, but verifying that segmentation is working correctly can be challenging, especially across large and complex environments. Common techniques to validate segmentation, such as reviewing firewall rules and spot testing from individual systems can only go so far, and comprehensive testing, such as running full network scans from every segment to every segment, can be time intensive and are hard to justify on a regular basis.

For businesses subject to the PCI DSS requirements, validating cardholder data environment (CDE) segmentation is an important part of the security audit process. The [PCI guidance on scoping and segmentation](#) describes a common CDE administration model.

The network bridge detection in runZero is opportunistic and far from perfect, but it may highlight areas where segmentation is broken, and can cut down on the number of surprises encountered in a future security audit.

Using the bridge report

The bridge report shows external networks in red and internal networks in green. This view is not a typical network map, but instead shows possible paths that can be taken through the network by traversing multi-homed assets. Assets where runZero only detected a single IP address are not shown in order to keep the graph readable.

Zooming in will show asset and subnet details. Clicking a bridged node once will highlight the networks it is connected to, and clicking it a second time will either take you to the asset page. Clicking a network once will highlight the connections to bridged nodes, and clicking a second time will perform a CIDR-based inventory search.

Bridge detection is useful when validating network segmentation and ensuring that an attacker can't reach a sensitive network from an untrusted network or asset. Examples of this include laptops plugged into the internal corporate network that are also connected to a guest wireless segment and systems connected to an untrusted network, such as a coffee shop's wireless network that also have an active VPN connection to the corporate network.

runZero detects network bridges by looking for extra IP addresses in responses to common network probes (NetBIOS, SNMP, MDNS, UPnP, and others) and only reports bridges when there is at least one asset identified with multiple IP addresses. Typical hardening steps, such as desktop firewalls and disabled network services will usually prevent multi-homed assets from being detected by runZero. The screenshot below shows how to search for multi-homed assets in the runZero inventory.

Using the asset route pathing report

Network segmentation is a foundational security control that can be easily undermined by network misconfigurations and multi-homed machines. runZero Platform users can now visualize potential network paths between any two assets in an organization using the asset route pathing report.

This report generates a graph of multiple potential paths by analyzing IPv4 and IPv6 traceroute data in combination with subnet analysis of detected multi-homed assets—without requiring access to the hosts or network equipment. This unique methodology identifies surprising and unexpected paths between assets that may not be accounted for by existing security controls or reviews.

With a view of potential paths, security professionals can verify whether a low-trust asset, such as a machine on a wireless guest network, can reach a high-value target, such as a database server within a cardholder data environment (CDE). The new feature highlights potential network segmentation violations and opportunities for an attacker to move laterally from one segment to another.

Protocol gateways

A protocol gateway is a device that bridges an IP network to a separate, sometimes non-IP fieldbus, backplane, or serial bus — for example, a Modbus/TCP-to-Modbus-RTU gateway, an EtherNet/IP CIP rack with backplane modules, a BACnet/IP router fronting an MS/TP segment, or a HART-IP multiplexer in front of a loop of HART field instruments. From the IP side, only the gateway is reachable. The PLCs, RTUs, IEDs, meters, sensors, and CNC modules behind it are invisible to a typical port scanner.

runZero unmask those downstream devices. When a scan identifies a supported gateway, the scanner issues protocol-native, read-only enumeration queries to the gateway and reports each downstream device as its own sub-asset in the inventory. Sub-assets are linked back to the gateway so you can see, search, and report on the full field-level topology — without credentials, agents, or active control commands.

How it works

1. **Identify the gateway.** The standard service probe identifies a supported industrial protocol on the gateway's IP address and port (for example, Modbus/TCP on TCP/502, CIP on TCP/44818, BACnet/IP on UDP/47808, HART-IP on TCP/5094).
2. **Enumerate downstream devices.** The probe issues protocol-specific identification requests against each address space the gateway exposes — Modbus unit IDs, CIP backplane slots, BACnet device instances and routed networks, KNX individual addresses, HART sub-device indices, S7 racks/slots, and so on. All requests are read-only.
3. **Emit a sub-asset per downstream device.** Each enumerated device is reported with a stable foreign ID of the form `<protocol>/<port>/<address>`, plus whatever identity the protocol exposes — vendor, model, firmware, serial, name, location.
4. **Link sub-assets to the gateway.** Each sub-asset carries `parent.id`, `parent.addresses`, and protocol-specific parent attributes that point back to the gateway asset. The console uses these to show downstream devices on the gateway's detail page and to render gateway → sub-asset edges on the [Network Map](#).
5. **Score the gateway as a choke point.** Because the gateway is the only path to its sub-assets, the [Network Map](#) treats it as a protocol-gateway pivot and ranks it at the top of the choke-point panel.

Sub-assets are full assets in the inventory: they appear in search, in queries, in exports, and in dashboards. They are matched and merged with assets discovered through other sources (for example, a PLC enumerated through CIP and also seen directly via SNMP), so the same physical device is not duplicated.

Supported protocols

The protocols below produce sub-assets when enumeration is enabled. The `gateway_kind` column is the value used by the `gateway_kind:` and `child_kind:` search keywords in both the asset inventory and the [Network Map](#).

Protocol	Default ports	gateway_kind	What runZero enumerates
BACnet/IP	UDP 47808 (+ range)	bacnet	Device instances and routed networks reachable through the BACnet router, including BBMD / FDT entries.
EtherNet/IP CIP	TCP/UDP 44818	cip	Modules in the controller backplane (CPU, scanner, communications, I/O, drives) via List Identity.
Modbus/TCP	TCP 502	modbus	Connected unit IDs reachable through the gateway, with vendor / product / firmware where exposed.
KNXnet/IP	UDP 3671	knxnet	KNX devices behind the IP interface, keyed by KNX individual address.
Siemens S7Comm	TCP 102	s7comm	Backplane rack/slot modules (CPU, CP, IM, signal modules) for S7-300/400/1200/1500.
DNP3	TCP 20000	dnp3	Outstation addresses behind a DNP3 master / data concentrator.
PROFINET	UDP 34962-34964	profinet	Slot/subslot module list reported by PROFINET I/O devices.
EtherCAT	UDP 34980	ethercat	Slaves discovered on the EtherCAT segment behind the master.
Omron FINS	UDP 9600	omronfins	CPU modules and connected units reachable through the FINS gateway.
Mitsubishi MELSEC-Q	TCP 5006-5007	melsecq	CPU modules on the MELSEC backplane, including CPU model and type code.
HART-IP	TCP 5094	hartip	HART field instruments behind the multiplexer (Cmd 84 walk of sub-device indices).
IEC 60870-5-104	TCP 2404	iec104	Outstations addressed through the IEC-104 master gateway.
IEC 61850 MMS	TCP 102	mms	Logical devices and IEDs reported through the substation MMS gateway.
Beckhoff ADS	TCP 48898	ads	ADS sub-devices behind the Beckhoff TwinCAT runtime.
IEEE C37.118 Synchrophasor	TCP 4712	c37118	PMUs reported by the Phasor Data Concentrator.

Other industrial protocols — including Emerson BSAP/IP, ANSI C12.22, Allen-Bradley CSPv4 / PCCC, Diagnostics over IP (DoIP), Fanuc FOCAS, SEMI HSMS / SECS-GEM, M-Bus over TCP, and OPC UA — return downstream-device summaries as gateway attributes when probed, but

currently do not promote those entries to standalone sub-assets. See [Supported protocols](#) for the full list of probes and what each one returns.

Enabling enumeration

Sub-asset enumeration runs as part of the standard scan when the relevant protocol is enabled and the matching probe is selected in the scan configuration. Most OT probes are off by default to keep scans of IT-only networks lean — see the [Scanning OT networks](#) playbook for the recommended enable-list and probe options.

When you enable a gateway protocol, the scanner runs both the standard fingerprint probe **and** the downstream enumeration step. There is no separate “enumerate sub-assets” toggle; enabling the protocol turns on both.

Finding gateways and sub-assets

Use the asset inventory search to pivot between gateways and the devices behind them. The topology keywords below are derived directly from the `edges` table runZero builds during enumeration, so the same queries work in **Inventory** → **Assets**, in any report that has an asset search box, and in the [Network Map](#) search bar.

Goal	Query
All protocol gateways	<code>is:gateway</code>
Layer-2 switches	<code>is:switch</code>
Modbus/TCP gateways	<code>gateway_kind:modbus</code>
EtherNet/IP CIP gateways	<code>gateway_kind:cip</code>
BACnet/IP routers	<code>gateway_kind:bacnet</code>
Siemens S7Comm racks	<code>gateway_kind:s7comm</code>
All sub-assets discovered behind any protocol gateway	<code>is:child</code>
Sub-assets behind a specific gateway protocol	<code>child_kind:modbus (OR bacnet, cip, s7comm, dnp3, profinet, ethercat, omronfins, melsecq, hartip, c37118, knxnet)</code>
Gateways fronting more than 10 downstream devices	<code>is:gateway AND downlink_count:>10</code>
Public-facing protocol gateways	<code>is:gateway AND has_public:t</code>
OT assets within 2 hops of a public-facing asset	<code>category:OT AND hops_from_public:<=2</code>
Devices in the OT category	<code>category:OT</code>

The topology keywords accept the OT protocol names from the `gateway_kind` column in the [Supported protocols](#) table. They are not limited to OT — `is:switch` and the existing `has:uplink` /

has:downlink / has:switch keywords cover layer-2 switch relationships.

Each sub-asset's detail page shows the parent gateway under **Source data** → **Parent**, and the gateway's detail page lists every sub-asset enumerated through it.

Where sub-assets show up

- **Inventory** — Sub-assets are regular assets. They appear in **Inventory** → **Assets**, in saved queries, and in CSV exports.
- **Asset details** — A sub-asset's page links back to its parent gateway. The gateway's page lists all sub-assets discovered through it.
- **Network Map** — Gateways render as protocol-gateway nodes with edges out to each sub-asset on their backplane. The **Proto Gateways** category in the Asset Index counts BACnet, EtherNet/IP, Modbus, KNXnet/IP, C37.118, and the other supported gateway kinds.
- **Choke points** — Protocol gateways are always promoted into the Network Map's **Choke Points** panel, with a +1000 Pivot Risk Score boost and a count of the assets they front. If the gateway is also EOL or carries known vulnerabilities, an additional +200 boost is applied.
- **Categories and functions** — Sub-assets are placed in the **OT Category** and assigned **OT Functions** (typically Process Control, With Safety, Monitoring, Communications, Or Human Interface added when the protocol exposes enough detail).

Safety

All gateway enumeration uses read-only protocol requests — identification reads, list-identity, who-is, sub-device indexing. The scanner never sends control or write commands, never issues operator overrides, and never attempts to authenticate. Probes are throttled by the standard scan rate and per-protocol pacing.

The runZero scan engine has been independently tested against a range of OT and industrial environments, including the U.S. Department of Energy's National Renewable Energy Laboratory (NREL) CECA evaluation, which validated that active scanning is safe for operational technology. See the [Scanning OT networks](#) playbook for additional safety guidance, recommended pacing, and per-protocol options.

See also

- [Network Map](#) — Visualize gateways, sub-assets, and the choke points they create.
- [Scanning OT networks](#) — How to enable OT probes and tune them for production environments.
- [Supported protocols](#) — Full list of protocols runZero probes, including which ones return backplane / sub-device summaries.
- [Understanding assets](#) — Categories, functions, and how runZero classifies devices.

Understanding fingerprints

runZero automatically builds **fingerprints** for every asset it discovers. Fingerprints describe how runZero identified a device, service, or operating system based on collected evidence. Each fingerprint includes a set of attributes that show *what was matched* and *where it came from*.

Fingerprints are a core part of how runZero normalizes your inventory, enabling accurate correlation and deduplication across multiple discovery sources.

How fingerprints work

During discovery, runZero gathers a variety of clues—such as TCP banners, mDNS names, SMB negotiation data, TLS certificates, and more. Each clue is analyzed and classified into one or more fingerprint categories.

Fingerprints are stored within the asset record using keys that follow the pattern:

```
fp.<category>.<attribute>
```

For example:

- `fp.os.source` — identifies which source provided the operating system fingerprint
- `fp.os.match` — shows the string or probe result matched to identify the OS
- `fp.hw.vendor` — lists the hardware vendor name

These values are visible in exports, the API, and internal datasets.

Refingerprinting and submitting improvements

If a fingerprint appears incorrect or incomplete, you can:

- **Refingerprint the asset** — re-run the fingerprinting process for that specific asset using the latest detection logic. This can update OS, hardware, type, category, and function fields.
- **Refingerprint in bulk** — select one or more assets in the **Asset Inventory** and choose **Modify** → **Refingerprint assets** to re-run fingerprinting against every selected asset using the latest detection logic.
- **Submit a fingerprint improvement** — provide feedback to help enhance fingerprinting accuracy. When submitting an improvement, you can specify:
 - **Type** (e.g., *Server, Router*)
 - **Operating System** (e.g., *Windows 10, Ubuntu 22.04*)
 - **Hardware** (e.g., *VMware VM, Synology NAS*)
 - Optional **comments** for additional context.

These improvements help refine fingerprinting logic in future releases and improve identification accuracy for similar devices.

Fingerprint categories

Each fingerprint category corresponds to a logical area of asset identification. Common categories include:

Category	Description	Example sources
os	Vendor, operating system, and version detected	TCP banners, SMB, mDNS
hw	Hardware and vendor identification	SNMP, MAC lookup, integrations
type	Asset type classification	runZero service probe

Common fingerprint attributes

The table below lists fingerprint attributes observed in exports and the API.

Attribute	Meaning	Sample Values
fp.build	Internal fingerprinting engine build version	v4.0.251103.0-2fd02762da549a3e87ffb5b0649e1621011872dd
fp.hw.certainty	Confidence level of hardware fingerprint	0.8, 1.0
fp.hw.cpe23	CPE 2.3 identifier for the hardware	cpe:2.3:h:dell:optiplex_7080:-:*:*:*:*:*:*
fp.hw.device	Detected hardware device type	Laptop, Server, Switch
fp.hw.match	Raw hardware identification string	Dell Inc. OptiPlex 7080, HP ProLiant DL360
fp.hw.product	Hardware product name	OptiPlex 7080, ProLiant DL360, MacBookPro18,3
fp.hw.serialNumber	Serial number extracted from hardware probe	ABC123XYZ, VMware-564d4a4a
fp.hw.source	Source that provided the hardware fingerprint	SNMP, Intune, CrowdStrike, runZero
fp.hw.vendor	Vendor identified from hardware data	Dell, HP, Apple, Lenovo
fp.hw.version	Hardware or firmware version string	1.0.3, rev A00

fp.os.build	OS build number parsed from source data	22621, 19045, 22D68
fp.os.certainty	Confidence level of OS fingerprint	0.95, 1.0
fp.os.cpe23	CPE 2.3 identifier for the operating system	cpe:2.3:o:microsoft:windows_10:19045:*:*:*:*:*
fp.os.device	Detected OS device type	Desktop, Server, Mobile
fp.os.edition	OS edition or variant	Pro, Enterprise, Home
fp.os.match	String matched to infer the OS	Windows 10.0, Ubuntu 22.04, macOS 14.1
fp.os.product	OS product name	Windows 10, Ubuntu, macOS
fp.os.source	Source that provided OS identification	runZero, MDNS, Intune, AzureAD
fp.os.vendor	Vendor identified from OS data	Microsoft, Canonical, Apple
fp.os.version	OS version parsed from matched data	10.0.19045, 22.04, 14.1
fp.type.source	Source that provided type classification	runZero, Integration, Hostname, AzureAD

Using fingerprints in queries

During asset processing, these fp.x.y values are normalized to os, os_version, hw, and type attributes.

These are some sample searches using the normalized attributes:

- Windows os assets

```
os:windows
```

- Ubuntu assets running os_version less than 20.04

```
os_version:<20.04 os:ubuntu
```

- Assets with Dell hardware

```
hw:dell
```

- Assets with the type server

```
type:server
```

Troubleshooting fingerprints

If a fingerprint appears incorrect or incomplete:

1. Check which `fp.*.source` values contributed to the identification. Multiple integrations or probes may overlap.
2. Review recent discovery runs to confirm whether an updated probe result is available.
3. Consider enriching with integration data sources (e.g., Intune, CrowdStrike) to improve fingerprint accuracy.

By understanding and using fingerprint data, you can verify discovery accuracy, correlate results across integrations, and build more precise asset views within runZero.

Managing alerts

runZero can trigger automatic alerts when certain events occur through a combination of Channels and Rules.

runZero currently supports Internal, Email, Email runZero Users, and Webhook channel types.

Internal channels store events within the [Alerts](#) list within the runZero Console. Internal alerts support explicit acknowledgement. Internal alerts can be bulk acknowledged and cleared from within the runZero Console.

Email channels can be configured to deliver mail to one or more recipients. These email messages contain a summary of the alert and a link to the specifics within the runZero Console. Email is sent from the runZero infrastructure using the Sendgrid service.

Webhook channels allow runZero to post alerts to internet-reachable web services. The post request contains a standard text message for use with platforms like Slack and Mattermost, but also additional fields containing the full alert details. Webhooks are a great way to tie runZero alerts into third-party platforms.

To trigger an alert on a channel, a [Rule](#) must be created. Rules define which events lead to alert on which channels. The name of the rule will be included in the alert content and should describe the type of event that it monitors.

The following are some example event types that can be used to create rules:

- Scan completed
- New assets found
- Assets back online
- Assets now offline
- Assets changed

Scan completion and *assets changed* rules can be noisy but may be useful to keep a running log of network changes over time. For a typical monitoring use case, a rule would be created to trigger on *Assets now offline*, *Assets back online*, and *New assets found*, automatically alerting an email alias or a Slack channel.

Alert rules, when combined with recurring scans, can be a simple way to track network changes over time.

Using the rules engine

The Rules Engine is an automation framework for monitoring, alerts, and workflow management. You can use the Rules Engine to customize alerts for the events that matter most to your organization and automate repetitive tasks. At the heart of the Rule Engine are rules. A rule defines the action that is taken based on a set of conditions. You can create rules to proactively alert your team when there are changes to things like Explorers, assets, scans, organizations, and sites. You can also automate tagging and modification of asset fields based on the results of a query.

Some ways you can use the Rules Engine to help automate your workflow:

- Alert your team when new policy violations are identified.
- Modify asset fields when the assets match specific criteria.
- Bulk tag assets that match a specific query.
- Get a Slack notification when a query returns new results.
- Monitor when an Explorer goes offline in the runZero console.
- Know when there are changes to organizations, sites, and users.

Key concepts

Rules can help you stay on top of events as they happen and get better visibility across your network, assets, and your runZero deployment. To build a rule, you need to define four things: events, organization access, conditions, and actions. A rule determines that when a specific event happens, and certain conditions are met, the system will automatically perform the configured action.

Events

Each rule begins with an event. The event sets off the trigger and puts your rule into motion. An event can be based on a query or a system-defined event. runZero offers a library of system-defined events you can use to create your rules. See the [runZero events](#) catalog for the full list. Choosing any of these events will show the conditions and actions available.

Organization access

Each rule can be used by any number of organizations. Rules are editable by any user with User role access to every organization associated with the rule, and can be read by any user with Viewer role access to at least one organization. By default, a rule will be triggered for the currently selected organization.

Setting organizational access on templates and channels only specifies who can view and edit, not which rules can use them. You can set up a rule for an organization with a template and/or channel whose access does not include that organization, as long as you have at least Viewer role access to one or more organizations in the template or channel's organization access list.

Conditions

A condition narrows the scope of your rule. Unless the condition is met, the rule will not execute the action. You will only see conditions that apply for the event you have chosen. Generally, conditions specify sites, organizations, and asset attributes for the event.

Actions

An action executes your rule, if the event occurs, and the conditions meet all the criteria. An action can be a notification to a channel, or it can be a modification to an asset. What you will need to configure depends on the action type. For notifications, you'll need to specify the

notification channel and template. For asset modification, you can edit fields like the OS vendor, OS product, OS version, hardware vendor, hardware product, hardware version, asset tags, and asset type.

Channels

A channel provides a way for you to communicate when a specific event has occurred. You can create multiple channels to support different types of communication needs. For example, you may want to create a Slack channel for one team, and an email list for another. It depends on what communication channels you prefer, and who you are trying to reach.

Much like rules, channels can also be utilized by any number of organizations, and managed by any user with User role access to all associated organizations. Similarly, any user with Viewer role access to one or more organizations will be able to view the channel details only, and may not edit or create new channels.

The body of the message uses default text from runZero. Customizations for messaging is currently unavailable.

Create a rule

Rules set the criteria for actions to take place. To create a rule, you need to choose an event, define the conditions, and choose a resulting action.

Step 1: Open the Rules Engine

- From the Alerts menu, select the Rules submenu.
- Click **Create Rule** to open the editor.

Step 2: Choose an event type and configure organization access

- Provide a descriptive name for the rule. Something that quickly tells you what the rule does.
- Choose an event you want to use as your trigger.
- You can browse the list of available predefined events. Use the left-hand categories to narrow down the list, or the search field to quickly filter by keyword.
- Choosing 'asset-query-results' or 'service-query-results' will allow you to modify the fields for the resulting assets.
- After you've chosen an event and configured organization access, click **Next**.

Step 3: Define the conditions

- The conditions you can configure depend on the event you have selected.
- If you have an asset or service based query selected, you'll need to [provide a query](#) for the rule. This query will run against the site after the scan completes. Note that assets with data from non-runZero sources must be recent (seen in the last 30 days) to be included in the scope of the search, and runZero-scanned assets must be live.
- You may also set the scope to a specific site or Explorer, and sometimes, depending on the event, minimum asset counts or task type.

Step 4: Choose an action, and optionally select a specific channel or template

- Actions can execute a notification to the channel of your choice or modify assets. For example, you can choose to send notifications via email when orphaned devices are found.

Step 5: Turn on and save the rule

- Turn the rule on if you want to activate it immediately by selecting the “Enable this rule” checkbox. Otherwise, you can save the rule and turn it on later.
- Save the rule when you’re done.

Keep in mind

Using scan and asset event types can be noisy, but they are useful for tracking network changes over time. To help you focus on the events that matter most, track assets that go offline, assets that come back online, and newly discovered assets.

Monitoring the status of rules

The rules submenu of the Alerts page displays a list of all rules that have been created. For each rule, you can see:

- Whether the rule is enabled.
- The event that triggers processing of the rule.
- The organizations the rule applies to, if the rule has been limited to specific organizations.
- When the rule was last triggered.
- Whether the rule resulted in an action being processed or not.
- When the rule was created and the username that created it.

A status of “skipped” means that last time the rule was processed, its preconditions weren’t met, so no action was taken. A status of “processed” means that the rule’s preconditions were met, and its action has been processed.

If there is an error processing a rule or sending a notification, the action status of the rule will be set to “error”. The error message can be seen as a tooltip on the error status.

Creating alert templates

With the [Rules Engine](#), you can define rules that alert you on specific events, such as changes to scans, assets, and Explorers. To customize the alert messages, you can create custom templates to standardize and format alerts triggered from rules. With custom templates, you can include more context and data for your alerts.

Templates can output as raw HTML, a runZero HTML template, JSON, and text for use in emails, internal notifications, or webhooks. You can customize the contents of these templates as needed.

Like rules and channels, templates can also be utilized by any number of organizations, and managed by any user with User role access to all associated organizations. Similarly, any user with Viewer role access to one or more organizations will be able to view the template details only, and may not edit or create new templates.

Template building basics

You can define the contents of an alert message using the [Mustache](#) templating language. As long as you know a little bit about how the [Mustache syntax](#) works, you can build custom HTML and JSON templates to reference and pull in runZero data.

- For Slack notifications, you can use Slack's [Block Kit](#) and their interactive [Block Kit Builder](#) to construct a rich message in JSON format, and then use Mustache to insert the relevant data.
- For Microsoft Teams, you can use the [Adaptive Cards](#) format to build rich messages, and again insert data via Mustache.

Our templates have two fields for template data: subject and body. Both subject and body can be customized using the Mustache syntax.

Inserting a data in a template

A standard set of objects is passed to the `{{template engine}}`, you just have to indicate the fields from the objects you want to insert into your template. Use the Mustache template syntax, `{{variable}}`, to include alert values when a rule matches certain conditions.

For HTML and JSON templates, values inserted using `{{ }}` are automatically escaped according to the appropriate rules. To avoid escaping a particular value, use triple curly braces, like this: `{{{task.name}}}`.

For JSON templates, the special self variable `{{.}}` at the top level will output all available variables and their values as JSON.

Special rules for JSON

For JSON templates, variables are rendered differently depending on whether they are single values or multiple values:

- A string or number gets rendered to the raw value. A string will have any embedded quotes escaped, but it won't be wrapped in quotes. This is so you can put multiple strings or numbers into a single JSON string in your template.
- An object or array gets rendered to the full JSON representation of the object or array. This is for convenience, so that you can dump arrays and objects to JSON without

having to loop through values. However, it means that if you try to put an array value into a JSON string by surrounding it with quotes, the result won't be valid JSON.

Consider the following template running on an asset with multiple IP addresses:

```
{
"addresses_1": {{addresses}},
"addresses_2": "{{#addresses}}{{.}} {{/addresses}}",
"addresses_3": "{{addresses}}"
}
```

Here's the result:

```
{
"addresses_1": ["10.0.1.4","10.1.7.5"],
"addresses_2": "10.0.1.4 10.1.7.5 ",
"addresses_3": "["10.0.1.4","10.1.7.5"]"
}
```

The `addresses_1` substitution works as you might want — `{{addresses}}` is an array, so it gets replaced with a proper JSON array containing the addresses.

The second substitution also works. In that case, the template loops through each address in the array, and puts the values into a single string. The values don't get put in quotes, but they will have any quotes inside them escaped.

The third substitution doesn't work properly — the addresses get turned into a JSON array of strings, but that gets surrounded by the quotes, resulting in invalid JSON.

Utilizing the runZero HTML template

When creating an alert template, the runZero HTML template option will wrap the provided alert template body with a runZero-provided HTML template. This option has been added to make it much easier to create consistent rich-content notification emails, and includes a base set of CSS classes you can apply to the contents of your template as you see fit.

Class name	Description
padding-sm	Applies a small amount of padding in each direction.
padding-md	Applies a medium amount of padding in each direction.
color-yellow	Applies a yellow text color to the target text
preamble	Creates a "preamble" block intended to go below the header. <code>p</code> tags within this block have a slightly larger font size and line height.
card	Creates a block that has a thin border, slightly rounded corners, and a slightly lighter background color than the main background color.
section	Similar to the card block, but with a darker background and a small amount of bottom margin. <code>section</code> is intended to be used within <code>card</code> .

preserve-newlines	Preserves newline characters in the contained text. preserve-newlines is intended to be paired with section, but can be used anywhere.
button	Creates a button with styling similar to the standard button seen in the runZero console, with a light blue background, white text color, and slightly rounded corners. When paired with size-lg, the button is given a slightly larger font size and a minimum width.

Additionally, the following elements have specific styles assigned to them:

Tag name	Description
h1	Bold font-weight, white text color, a moderate amount of bottom margin, considerably larger font size, and center-aligned text.
h2	Bold font-weight, white text color, a moderate amount of bottom margin, and a moderately larger font size.
h3	Normal font-weight, white text color, a small amount of bottom margin.
p	8 pixel bottom margin
hr	Thin light blue top border, with a considerable amount of margin above and below.
a	Light blue text color.

Using this template type and these classes, here is an example of a template intended for use with the site-created event:

```
<div class="preamble padding-md">
  <h1>Site created</h1>
  <p>A new site was created in an organization</p>
</div>

<div class="card padding-md">
  <h3>{{site.name}} attributes</h3>
  <div class="section padding-sm">
    <p>Created at: {{event.created_at}}</p>
    <p>ID: {{site.id}}</p>
    <p>Organization: {{organization.name}}</p>
    <p>Created by {{event.source_type}}: {{event.source_name}}</p>
  </div>
  <a class="button size-lg" href="{{console}}/sites/{{site.id}}/?_oid={{organization.id}}">View site</a>
</div>
```

While we strive to maintain backwards compatibility for templates utilizing this template type, the available classes and predefined styling may be changed without warning.

Inserting a value

To insert a value, put the variable name in double curly brackets, like this {{variablename}}.

The following example shows how to insert the console address:

The runZero Console is at `{{console}}`.

Inserting a value from an object

To insert a value from a specific object, separate the object name and field name with a dot, like this `{{object.fieldname}}`.

The following example shows how to insert the organization name:

The organization name is `{{organization.name}}`.

Inserting multiple values from an object

To insert multiple values from an object easily, use a section. You will need to start the section with `{{#objectname}}` and close it with a matching `{{/objectname}}`. The following example shows how to insert the results from a scan that include total assets and number of assets changed:

Here are the results:

```
{{#scan}}
Scan found {{assets_total}} assets and changed {{assets_changed}} of them.
{{/scan}}
```

What happens if a field contains multiple values?

If a section refers to a field which contains multiple values, the template engine will loop through the values in the field, processing the section inside for each individual value.

The following example loops through all of the assets in `report.new`, and for each one, outputs its names and addresses fields. If there is nothing stored in `report.new`, the section between the tags will not be rendered.

```
{{#report.new}}
{{names}} {{addresses}}
{{/report.new}}
```

Note that within a loop, you can still refer to values from the outer object. If a named value isn't found in the current loop object, the template engine will check the outer object. For example, this can be useful for referring to the `{{console}}` variable, which provides the root runZero console URL.

Using boolean values

You can use boolean values with the `{{#field}}{{/field}}` tags. If the value of the field is false, the section between the tags is not rendered.

For example:

```
{{#query.truncated}}
(Additional results were found but not included in this report)
{{/query.truncated}}
```

Objects and fields reference

To include runZero data and details in your alerts, you can build your template using the following objects and fields.

globals

Field	Contents	Example
console	The base URL of the runZero web console.	https://console.runzero.com

event

The following fields are available in the `event` object:

Field	Contents	Example
action	The action which triggered the event	task-completed
created_at	When the event was created	2021-04-02 12:50:26 -0500
id	The UUID of the event	b4b871db-bdf1-4a42-b82d-18ae99972228
source_name	Name of the thing which caused the event	Weekly security scan
source_type	Type of thing which caused the event	task
success	Whether the event succeeded	true
target_name	Name of the object targeted by the event	Head Office
target_type	The type of the object targeted by the event	site

task

For events triggered by a task, the following fields are available in the `task` object:

Field	Contents	Example
created_at	When the task was created	2021-04-02 12:50:26 -0500
created_by	The user who created the task	user@example.com
description	The description of the task	Weekly scan of main network
error	The text of any error message for the task	explorer unavailable after 4h

id	The UUID of the task	b4b871db-bdf1-4a42-b82d-18ae99972228
name	The name of the task	Weekly Scan
start_time	When the task started	2021-04-02 12:50:26 -0500
status	The status of the task	processed
type	The type of task	scan
updated_at	When the task was last updated	2021-04-02 12:50:26 -0500
url	A URL linking to the task details	

organization

The following fields are available in the `organization` object:

Field	Contents	Example
id	The UUID of the organization	86f12ee1-f0f1-419a-8799-63fff555777a
name	The name of the organization	IT Dept.

site

The following fields are available in the `site` object:

Field	Contents	Example
id	The UUID of the site	49f9323a-fea1-4afc-b490-2414c3aaae
name	The name of the site	Head Office

rule

The following fields are available in the `rule` object:

Field	Contents	Example
action	The action the rule said to take	notify
created_at	When the rule was created	2021-03-08 12:43:59 -0600
created_by	The user who created the rule	user@example.com
event	The event triggering the rule	scan-completed
id	The UUID of the rule	b2269a2c-69a1-4652-bcdf-899938886c17
name	The name of the Rule	Alert on scan
updated_at	When the rule was last updated	2021-04-01 17:09:40 -0500

channel

The following fields are available in the `channel` object:

Field	Contents	Example
id	The UUID of the channel	7c9f0d5a-0f8e-4b3b-8a2e-1d9b5f1c2c3d
name	The name of the channel	Notify auditors
type	The channel type	email

scan

For events triggered by a scan task, the following additional fields are available in a `scan` object:

Field	Contents	Example
explorer_id	The UUID of the runZero Explorer which carried out the scan	0fd44a62-d827-41c0-b26c-4837222d8888
assets_changed	The number of assets changed as a result of the scan	9
assets_ignored	The number of assets ignored by the scan	2
assets_new	The number of new assets detected by the scan	2
assets_offline	The number of assets which were previously online but now offline	1
assets_online	The number of assets previously offline but now online again	2
assets_total	The total number of assets for the site scanned (including offline)	11
assets_unchanged	The number of assets unchanged by the scan	1
assets_updated	The total number of assets up-to-date as a result of the scan (changed + unchanged)	15
duration	Duration of the scan in seconds	26
end_time	When the scan ended	2021-04-02 12:50:26 -0500
excludes	Any IP addresses excluded from the scan	10.0.1.123
id	The UUID of the scan task	894a112c-3fb9-4301-8da7-8ce7fffb4443
name	The name of the scan task	Weekly security scan

rate	The scan rate	1000
recv_bytes	How many bytes were received during the scan	45176
recv_error	How many receive errors were detected	1
recv_packets	How many data packets were received during the scan	555
scheduled_time	When the scan was scheduled to run	2021-04-02 12:48:00-0500
sent_bytes	How many bytes were sent during the scan	44740
sent_error	How many send errors were detected	0
sent_packets	How many data packets were sent during the scan	577
start_time	When the scan actually started	2021-04-02 12:49:55-0500
tags	An array of tags associated with the scan task	
targets	The CIDR address ranges scanned	10.0.1.0/24
type	The type of operation	scan

explorer

For events triggered by a scan task, the following additional fields are available in an explorer object (runZero 2.1.8+):

Field	Contents	Example
id	The UUID of the runZero Explorer which carried out the scan	0fd44a62-d827-41c0-b26c-4837222d8888
name	The name of the Explorer which carried out the scan	MM34B-2
internal_ip	The internal IP address of the Explorer which carried out the scan	10.0.1.200

search

For events triggered by a search query rule, the following additional fields are available in a search object:

Field	Contents	Example
url	A link to perform the same search	https://console.runzero.com/inventory/?search=...

found	The number of matches found	3
comparator	The operation used to compare the number of matches	>=
value	The value the number of matches was compared against	1

report

For “scan completed” events, a `report` object contains the following results from the scan:

Field	Contents
truncated	Whether the set of objects was truncated due to large numbers of assets
changed	An array of changed assets (see below)
new	An array of new assets (see below)
offline	An array of assets now offline (see below)
online	An array of assets now online (see below)

The limit on the number of objects passed to the template is 25 for email notifications, 10 for Webhook notifications.

asset (from a scan report)

Each asset returned as part of a scan report has the following fields:

Field	Contents	
addresses	The IP address(es) of the asset	10.0.1.123
alive	Whether the asset responded to probes	true
created_at	The timestamp when the asset record was created	2021-04-02 12:50:26 -0500
criticality_rank	The criticality rank from (0-5)	1
detected_by	The method by which the asset was detected	arp
domains	Any domains the asset was found in	WORKGROUP
first_seen	The timestamp when the asset was first seen	2021-04-02 12:50:26 -0500
hw	A summary of the asset hardware	HP LaserJet Pro
id	The UUID of the asset	b38295fb-bef1-fa42-b82d-18ae99972228
last_seen	The timestamp when the asset was most recently seen	2021-04-02 12:50:26 -0500

macs	Any MAC addresses detected for the asset	00:56:55:00:91:04
modified_risk_rank	The risk rank after applying user overrides	2
names	Any names detected for the asset	LP538N
os	A summary of the asset's operating system	Linux
outlier_score	The normalized outlier score (0-5)	2
risk_rank	The normalized risk rank (0-4)	4
service_count	How many services the asset is running	4
software_count	How many software entries were identified on the asset	10
type	The type of asset	Router
updated_at	The timestamp when the asset record was last updated	2021-04-02 12:50:26-0500
vulnerability_count	How many vulnerabilities were identified on the asset	4

query

For query events, the following data is provided in the `query` variable:

Field	Contents
count	The number of rows matching the search query
assets	If it was an asset search, the array of assets matching the query
services	If it was a service search, the array of services matching the query
wlans	If it was a wireless network search, the array of wireless networks matching the query
truncated	Whether the set of assets was truncated due to the query returning a large number

The limit on the number of assets passed to the template from a query is 25 for email notifications, 10 for Webhook notifications.

asset (from a query)

For assets returned from a query rule, the following fields are available:

Field	Contents	
addresses	The IP address(es) of the asset	10.0.1.123
alive	Whether the asset responded to probes	true

comments	Any comments set in the asset record	
created_at	The timestamp when the asset record was created	2021-04-02 12:50:26 -0500
detected_by	The method by which the asset was detected	arp
domains	Any domains the asset was found in	WORKGROUP
first_seen	The timestamp when the asset was first seen	2021-04-02 12:50:26 -0500
hw	A summary of the asset hardware	ThinkPad X1
id	The UUID of the asset	b38295fb-bef1-fa42-b82d-18ae99972228
last_seen	The timestamp when the asset was most recently seen	2021-04-02 12:50:26 -0500
macs	The list of MAC addresses associated with the asset	[F4:F5:E8:89:92:31, 00:D0:2D:9F:47:77]
names	Any names detected for the asset	laptop.local
organization	The organization the asset belongs to	IT
os	A summary of the asset's operating system	Windows 10
service_count	How many services the asset is running	4
site	The site the asset was detected at	New York
tags	An array of tags set on the asset	
type	The type of asset	Thermostat
updated_at	The timestamp when the asset record was last updated	2021-04-02 12:50:26 -0500

service (from a query)

For services returned from a query rule, the following fields are available, some of which are taken from the associated asset:

Field	Contents	Example
id	The UUID of the service (not the asset)	b38efadb-61f1-f332-b92d-18ae99972228
created_at	When the service record was created	2021-04-02 12:50:26 -0500
summary	A summary of the service	ciscoSystems
port	The TCP/UDP port the service is on	53

vhost	The vhost of the service	ftp.example.com
address	The TCP/IP address of the service	192.168.33.44
transport	The transport	udp
protocol	The name of the protocol, if known	ssh
organization	The name of the organization the service's asset belongs to	HR
site	The name of the site the service's asset belongs to	Lab
alive	Whether the asset offering the service was alive	true
last_seen	When the asset was last seen	2021-04-02 12:50:26 -0500
first_seen	When the asset was first seen	2021-02-11 09:38:17 -0500
type	The asset type offering the service	Laptop
os	The OS offering the service	Linux
hw	The hardware offering the service	APC UPS
addresses	A list of other IP addresses associated with the asset offering the service	[192.168.0.2, 192.168.0.3]
macs	The list of MAC addresses associated with the asset	[F4:F5:E8:89:92:31, 00:D0:2D:9F:47:77]
names	Any names associated with the asset	[fw-3, fw-3a]
tags	Tags set on the asset	
domains	Any domain associated with the asset	LOCAL
service_count	A count of how many services the asset offers	4
comments	Any comments set on the asset	

certificates (from a query)

For certificates returned from a query rule, the following fields are available:

Field	Contents	Example
id	The UUID of the certificate	ff6a8817-828a-43a3-b2cb-596dff5f0bd5
type	The type of certificate	x509
self_signed	Whether the certificate is self-signed	true

names	All names associated with the certificate	["example.com", "www.example.com"]
serial	The serial number of the certificate	0A014A1D4D0F861EC4C6EFD6E13A2F
validity_start	When the certificate becomes valid	2025-11-06 15:58:04 -0500
validity_end	When the certificate expires	2025-11-08 15:58:04 -0500
validity_period	The total validity period in seconds	172800
public_key_algorithm	The algorithm used for the public key	rsaEncryption
public_key_bits	The size of the public key in bits	2048
signature_algorithm	The algorithm used for the signature	sha256WithRSAEncryption
fp_bkhash	The BKHash fingerprint of the certificate	d2c7e8f9a1b2c3d4e5f6
fp_sha1	The SHA1 fingerprint of the certificate	3c482c71d874763e560c617129793dec76e91196
fp_sha256	The SHA256 fingerprint of the certificate	19vJPgdyh3BdGgTiSSavQsKx133IzFkJhu51n6fzG+k=
cn	The common name from the certificate subject	example.com
subject	The subject distinguished name	CN=example.com,O=Example Corp,C=US
issuer	The issuer distinguished name	CN=Example CA,O=Example Corp,C=US

is_ca	Whether the certificate is a certificate authority	false
key_usage	The key usage extension values	digitalSignature
ext_key_usage	The extended key usage extension values	serverAuthentication
san_dns_names	Subject Alternative Name DNS entries	["example.com", "www.example.com"]
san_ip_addresses	Subject Alternative Name IP entries	["192.168.1.1"]
san_email_addresses	Subject Alternative Name email entries	["admin@example.com"]
san_uris	Subject Alternative Name URI entries	["spiffe://example.com/service"]
public_key_insecure	Whether the public key is considered insecure	false
signature_insecure	Whether the signature algorithm is considered insecure	false
public_key_parameters	Parameters for the public key, if any	map[rsa_exponent:65537 rsa_modulus:244...]
name_constraints	Name constraints extension, if present	map[]
comments	Any comments set on the certificate	"Issued for legacy system"
tags	Tags applied to the certificate	["production", "web"]
risk_rank	The calculated risk rank (-1..5)	2
modified_risk_rank	The modified risk rank if set	1
risk	Either modified risk rank if set or risk_rank if unset as a name	"info"
vulnerability_count	Number of vulnerabilities associated	3
service_address	The address of the associated service	192.168.1.10
service_port	The port of the associated service	443
service_transport	The transport protocol of the service	tcp

asset_site_id	The UUID of the associated site	49f9323a-fea1-4afc-b490-2414c3aaaaee
asset_site_name	The name of the associated site	"Head Office"
asset_id	The UUID(s) of associated assets	b38295fb-bef1-fa42-b82d-18ae99972228

wlan (wireless LAN, from a query)

For wireless lans (wlans) returned from a query rule, the following fields are available:

Field	Contents	Example
id	The UUID of the wireless LAN in runZero's database	f938934b-ae23-f112-b23d-18ae99972228
last_seen	When the wlan was last seen	2021-04-02 12:50:26 -0500
ssid	The ESSID of the network	Free WiFi
bssid	The ESSID of the network	c4:41:1e:99:88:77
type	The type of wireless network	infrastructure
authentication	The authentication used to access the network	WPA2-PSK
encryption	The encryption used to protect data	AES
signal	The signal strength as a percentage	86
channels	The channel of the network	11
organization	The name of the organization the service's asset belongs to	HR
site	The name of the site the service's asset belongs to	Lab

findings_with_instances

Field	Contents	Example
finding_category	The category of the finding	Open Access
finding_code	The code of the finding	rz-finding-open-access-default-credentials
finding_id	The ID of the finding	b1f8a2e7-3c4d-4e6a-9c2f-1a5e7d8b9c0a
finding_name	The name of the finding	Service Accessible With Default Credentials

finding_risk_rank	The normalized risk rank of the finding (0-4)	3
finding_vulnerability_count	How many vulnerabilities are associated with the finding; the same value as new_vulnerability_count	8
new_vulnerability_count	How many vulnerabilities are associated with the finding after the task was run	8
old_vulnerability_count	How many vulnerabilities were associated with the finding the last time the task was run	5
organization_id	The ID of the organization	e2d3c4b5-6a7b-8c9d-0e1f-2a3b4c5d6e7f
site_id	The ID of the site	f3e4d5c6-b7a8-9c0d-1e2f-3a4b5c6d7e8f
site_name	The name of the site	Primary
task_id	The ID of the task	a4b5c6d7-e8f9-0a1b-2c3d-4e5f6a7b8c9d
task_name	The name of the task	Metrics
task_start_time	The start time of the task	2026-03-02 11:16:59-0500
task_type	The type of the task	analysis

rapid_response_published

For rapid-response-published events, the `rapid_response_published` object contains `new`, `removed` and `updated` lists. The objects in these lists have the following fields available:

Field	Contents	Example
id	The UUID of the new rapid response query	f938934b-ae23-f112-b23d-18ae99972228
name	The name of the query	Rapid Response: Assets with multiple RCE exploits
description	Plaintext description of the query	Certain assets can be exploited
solution	Plaintext description of how to resolve the issue exposed by this query	Update the Chipland Exploitable NVR to v3.2 or newer
query	The query contents	hw:="Chipland Exploitable NVR"

effective_query	Query contents that are used to match assets. Includes any system predicates, such as alive:t.	alive:t AND (hw:"Chipland Exploitable NVR")
finding_code	The finding code for the query, if assigned	rz-finding-rapid-response-assets
exploitable	Whether or not the rapid response is known exploitable	true
cve	Newline-separated list of CVEs that may apply to this query	CVE-2025-12345\nCVE-2025-12346
console_inventory_path	The path to the console inventory page for the query	/inventory
attributes	Various miscellaneous attributes, such as reference URLs and runZero blog links.	See example below table

Example attributes object:

```
{
  "links": {
    "rz_blog": [
      {
        "title": "How to find Chipland Exploitable NVR on your network",
        "url": "https://www.runzero.com/blog/"
      }
    ],
    "urls": [
      {
        "url": "https://example.local/1.html"
      },
      {
        "url": "https://example.local/2.html"
      }
    ]
  }
}
```

rapid_response_with_matches

For rapid-response-with-matches events, the following fields are available:

Field	Contents	Example
id	The UUID of the rapid response query	f938934b-ae23-f112-b23d-18ae99972228
query	Various attributes relating to the query. See rapid_response_published for the fields available.	See rapid_response_published fields

old_count	How many resources matched last time this query was run.	0
new_count	How many resources matched the most recent time this query was run.	2

Example: Alert when scan completes

Let's take an example of something you might want to get alerted on: completed scans. We can create a rule that emails us when a scan completes, and provides us with some details, such as the number of new, online, offline, and modified assets. We'll build these details into our template.

Step 1. Create a template.

You can create a template from the [Alerts](#) page.

- Name the template something like Email the team when a scan completes.
- For the template type, choose HTML, since we want to use this template for emails.
- For the subject line, enter something that's descriptive, like `runZero scan {{scan.name}}` completed at `{{scan.end_time}}`. You can use the Mustache syntax for the subject.

Step 2. Create the body message

Now, let's create the email body. We want the email to tell us how many new, online, offline, and modified assets there are, as well as give us details on the new assets discovered.

The body looks like this:

```
<h1>{{site.name}}</h1>

<h2>Scan Results</h2>
{{#scan}}
<ul>
<li>{{assets_new}} new assets</li>
<li>{{assets_online}} online assets</li>
<li>{{assets_offline}} offline assets</li>
<li>{{assets_changed}} modified assets</li>
</ul>
{{/scan}}

<h2>New assets</h2>
<ul>
{{#report.new}}
<li>{{names}} {{addresses}} {{os}}</li>
{{/report.new}}
{{^report.new}}
<li>No new assets were discovered.</li>
{{/report.new}}
</ul>

<p><a href="{{task.url}}">View the scan results</a></p>
```

See that caret (^)? It represents inverted sections. These sections only render when the list is empty or the value is empty or false.

Step 3. Save the template and create a rule

The template is now available for you to choose when you [create a rule](#).

Sample JSON templates

A JSON array of new assets found by a scan

```
[
  {{#report.new}}
  {
    "addresses": "{{addresses}}",
    "alive": "{{alive}}",
    "detected_by": "{{detected_by}}",
    "domains": "{{domains}}",
    "first_seen": "{{first_seen}}",
    "hw": "{{hw}}",
    "names": "{{names}}",
    "os": "{{os}}",
    "type": "{{type}}"
  },
  {{/report.new}}
  null
]
```

The trailing `null` in the array is so that the array is valid JSON, because JSON doesn't allow trailing commas.

All available attributes

Output all available variables and their attributes as a JSON object:

```
{{.}}
```

Alternatively, you can build your own JSON array with all attributes:

```
{
  "event": {
    "action": "{{event.action}}",
    "created_at": "{{event.created_at}}",
    "id": "{{event.id}}",
    "source_name": "{{event.source_name}}",
    "source_type": "{{event.source_type}}",
    "success": "{{event.success}}",
    "target_name": "{{event.target_name}}",
    "target_type": "{{event.target_type}}"
  }
}
```

```

"organization": {
  "name": "{{organization.name}}",
  "id": "{{organization.id}}"
},
"site": {
  "name": "{{site.name}}",
  "id": "{{site.id}}"
},
"report": {
  "truncated": "{{report.truncated}}",
  "changed": "{{report.changed}}",
  "new": "{{report.new}}",
  "offline": "{{report.offline}}",
  "online": "{{report.online}}"
},
"rule": {
  "action": "{{rule.action}}",
  "created_at": "{{rule.created_at}}",
  "created_by": "{{rule.created_by}}",
  "event": "{{rule.event}}",
  "id": "{{rule.id}}",
  "name": "{{rule.name}}",
  "updated_at": "{{rule.updated_at}}"
},
"scan": {
  "explorer_id": "{{scan.explorer_id}}",
  "assets_changed": "{{scan.assets_changed}}",
  "assets_ignored": "{{scan.assets_ignored}}",
  "assets_new": "{{scan.assets_new}}",
  "assets_offline": "{{scan.assets_offline}}",
  "assets_online": "{{scan.assets_online}}",
  "assets_total": "{{scan.assets_total}}",
  "assets_unchanged": "{{scan.assets_unchanged}}",
  "assets_updated": "{{scan.assets_updated}}",
  "duration": "{{scan.duration}}",
  "end_time": "{{scan.end_time}}",
  "excludes": "{{scan.excludes}}",
  "id": "{{scan.id}}",
  "name": "{{scan.name}}",
  "rate": "{{scan.rate}}",
  "recv_bytes": "{{scan.recv_bytes}}",
  "recv_error": "{{scan.recv_error}}",
  "recv_packets": "{{scan.recv_packets}}",
  "scheduled_time": "{{scan.scheduled_time}}",
  "sent_bytes": "{{scan.sent_bytes}}",
  "sent_error": "{{scan.sent_error}}",
  "sent_packets": "{{scan.sent_packets}}",
  "start_time": "{{scan.start_time}}",
  "tags": "{{scan.tags}}",
  "targets": "{{scan.targets}}",
  "type": "{{scan.type}}"
},
"explorer": {

```

```

"name": "{{explorer.name}}",
  "internal_ip": "{{explorer.internal_ip}}"
},
"search": {
  "url": "{{search.url}}",
  "found": "{{search.found}}",
  "comparator": "{{search.comparator}}",
  "value": "{{search.value}}"
},
"query": {
  "count": "{{query.count}}",
  "assets": "{{query.assets}}",
  "services": "{{query.services}}",
  "wlans": "{{query.wlans}}",
  "truncated": "{{query.truncated}}"
}
}

```

Event attributes only

```

{
  "event": {{event}}
}

```

or

```

{
  "action": "{{event.action}}",
  "created_at": "{{event.created_at}}",
  "id": "{{event.id}}",
  "source_name": "{{event.ource_name}}",
  "source_type": "{{event.source_type}}",
  "success": "{{event.success}}",
  "target_name": "{{event.target_name}}",
  "target_type": "{{event.target_type}}"
}

```

Organization attributes only

```

{
  "organization": {{organization}}
}

```

or

```

{
  "name": "{{organization.action}}",
  "id": "{{organization.id}}"
}

```

Site attributes only

```
{
  "site": {{site}}
}
```

or

```
{
  "name": "{{site.action}}",
  "id": "{{site.id}}"
}
```

Report attributes only

```
{
  "report": {{report}}
}
```

or

```
{
  "truncated": "{{report.truncated}}",
  "changed": "{{report.changed}}",
  "new": "{{report.new}}",
  "offline": "{{report.offline}}",
  "online": "{{report.online}}"
}
```

Rule attributes only

```
{
  "rule": {{rule}}
}
```

or

```
{
  "action": "{{rule.action}}",
  "created_at": "{{rule.created_at}}",
  "created_by": "{{rule.created_by}}",
  "event": "{{rule.event}}",
  "id": "{{rule.id}}",
  "name": "{{rule.name}}",
  "updated_at": "{{rule.updated_at}}"
}
```

Scan attributes only

```
{
  "scan": {{scan}}
}
```

```
}
```

or

```
{
  "explorer_id": "{{scan.explorer_id}}",
  "assets_changed": "{{scan.assets_changed}}",
  "assets_ignored": "{{scan.assets_ignored}}",
  "assets_new": "{{scan.assets_new}}",
  "assets_offline": "{{assets_offline}}",
  "assets_online": "{{scan.assets_online}}",
  "assets_total": "{{scan.assets_total}}",
  "assets_unchanged": "{{scan.assets_unchanged}}",
  "assets_updated": "{{scan.assets_updated}}",
  "duration": "{{scan.duration}}",
  "end_time": "{{scan.end_time}}",
  "excludes": "{{scan.excludes}}",
  "id": "{{scan.id}}",
  "name": "{{scan.name}}",
  "rate": "{{scan.rate}}",
  "recv_bytes": "{{scan.recv_bytes}}",
  "recv_error": "{{scan.recv_error}}",
  "recv_packets": "{{scan.recv_packets}}",
  "scheduled_time": "{{scan.scheduled_time}}",
  "sent_bytes": "{{scan.sent_bytes}}",
  "sent_error": "{{scan.sent_error}}",
  "sent_packets": "{{scan.sent_packets}}",
  "start_time": "{{scan.start_time}}",
  "tags": "{{scan.tags}}",
  "targets": "{{scan.targets}}",
  "type": "{{scan.type}}"
}
```

Explorer attributes only

```
{
  "explorer": {{explorer}}
}
```

or

```
{
  "id": "{{explorer.id}}",
  "name": "{{explorer.name}}",
  "internal_ip": "{{explorer.internal_ip}}"
}
```

Search attributes only

```
{
  "search": {{search}}
}
```

or

```
{
  "url": "{{search.url}}",
  "found": "{{search.found}}",
  "comparator": "{{search.comparator}}",
  "value": "{{search.value}}"
}
```

Query attributes only

```
{
  "query": {{query}}
}
```

or

```
{
  "count": "{{query.count}}",
  "assets": "{{query.assets}}",
  "services": "{{query.services}}",
  "wlans": "{{query.wlans}}",
  "truncated": "{{query.truncated}}"
}
```

Asset, service, or wireless query alert to SIEM or SOAR

```
{
  "organization": {
    "name": "{{organization.name}}",
    "id": "{{organization.id}}"
  },
  "site": {
    "name": "{{site.name}}",
    "id": "{{site.id}}"
  },
  "rule": {
    "action": "{{rule.action}}",
    "created_at": "{{rule.created_at}}",
    "created_by": "{{rule.created_by}}",
    "event": "{{rule.event}}",
    "id": "{{rule.id}}",
    "name": "{{rule.name}}",
    "updated_at": "{{rule.updated_at}}"
  },
  "search": {
    "url": "{{search.url}}",
    "found": "{{search.found}}",
    "comparator": "{{search.comparator}}",
    "value": "{{search.value}}"
  }
}
```


Asset, service, or wireless query alert to Slack

```
{
  "blocks": [
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": ":red_circle: *runZero Alert* - {{rule.name}}"
      }
    },
    {
      "type": "divider"
    },
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "*Rule information*\n\n_Name_: {{rule.name}}\n_Type_: {{rule.event}}\n_Link_: https://console.runzero.com/alerts/rules/{{rule.id}}"
      }
    },
    {
      "type": "section",
      "text": {
        "type": "mrkdwn",
        "text": "*Match information*\n\n_Organization_: {{organization.name}}\n_Site_: {{site.name}}\n_Match count_: {{search.found}}\n_Search_: {{search.value}}\n_Link_: {{search.url}}"
      }
    }
  ]
}
```

Asset, service, or wireless query alert to Microsoft Teams

```
{
  "type": "message",
  "attachments": [
    {
      "contentType": "application/vnd.microsoft.card.adaptive",
      "contentUrl": null,
      "content": {
        "$schema": "http://adaptivecards.io/schemas/adaptive-card.json",
        "type": "AdaptiveCard",
        "version": "1.5",
        "body": [
          {
            "type": "TextBlock",
            "text": "runZero Alert - {{rule.name}}"
          },
          {

```

```

"text": "***Rule information**\n\nName_: {{rule.name}}\n\nType_: {{rule.event}}\n\nLink_: [Rule](http://console.runzero.com/alerts/rules/{{rule.id}})",
  "wrap": true
},
{
  "type": "TextBlock",
  "text": "***Match information**\n\nOrganization_: {{organization.name}}\n\nSite_: {{site.name}}\n\nMatch count_: {{search.found}}\n\nSearch_: {{search.value}}\n\nLink_: [Search]({{search.url}})",
  "wrap": true,
  "spacing": "Medium"
}
]
}
]
}

```

Data type accepted by each channel

When you create a rule, the channel you select must accept the data type used by the template. For example, if you want to send a Slack notification, the template must be in plain text or JSON.

Here are the data types accepted by each channel:

Channel	Data type
Email	Plain text, HTML, JSON
Webhook	Plain text, JSON
Internal notification	Plain text

Managing templates

Go to [Alerts > Templates](#). From this page, you can create, edit, and delete templates. Before deleting templates, make sure they are not in use by any rules. If you delete a template that is in use by a rule, the rule will revert to the default template.

runZero events

runZero records system events for administrative actions, alert rule processing, scans, Explorers, API activity, and more. Use an event action below as the trigger for an [alert rule](#), or search for it in the [audit log](#).

Event catalog omitted from PDF. See help.runzero.com for the full list.

Querying your data

runZero provides many ways to query your data. Generally, queries can be broken into two concepts:

- Filters or parameters used in the search bars on pages across the console, or
- System and custom queries for which match metrics are calculated as tasks complete.

Both allow you to leverage the extensive query language to quickly find the information you're looking for.

Filtering and searching data

The various inventory pages are likely the main place you'd look to use these queries, but many other pages include the same type of search bar that can be used to filter results. The following documentation pages will help you craft a query that meets your needs:

- [Syntax for crafting compound queries](#)
- [Examples to guide query development](#)
- [Keywords that can be used across the inventory pages](#)
- [Keywords that can be used across other components of the console](#)

System and custom queries

The [Queries page](#) serves as an inventory of all your saved queries. Saved queries can either be system queries published by runZero or custom queries created by you or your team.

Standard query attributes

Queries have the following standard attributes:

- **Name:** A name for the saved query.
- **Description:** A description of the saved query.
- **Search query:** The query or parameters to search for matches to.
- **Search live assets only:** When toggled to *Yes* (default), `alive:t` is added to the search query to only include assets marked alive.
- **Type:** The inventory the query will search.
- **Category:** The category the query falls under.
- **Severity:** A severity level for the query.
- **Automatically track query results on the dashboard:** When toggled to *Yes*, the query and count of matches will be included in a component on the dashboard.

Vulnerability record attributes

System and custom queries can also be used to create and associate vulnerability records with matching assets. This will be enabled by default on some system queries. To have a

query create vulnerability records, switch the **Apply a vulnerability record to matching assets** toggle to **Yes**, then complete the following fields:

- **Vulnerability ID:** Choose a unique ID to track this vulnerability within runZero.
- **CVEs:** Include a list of CVEs relevant to this vulnerability record (optional).
- **Solution:** Provide context for how this vulnerability could be remediated on assets (optional).
- **Risk:** Select a risk level to associate with the vulnerability. This impacts the [asset risk](#).
- **Exploitable:** Specify whether an exploit is available for the vulnerability.
- **CVSS v3 base score:** The CVSS v3 base score (0.00 to 10.00).
- **CVSS v3 temporal score:** The CVSS v3 temporal score (0.00 to 10.00).
- **CVSS v2 base score:** The CVSS v2 base score (0.00 to 10.00).
- **CVSS v2 temporal score:** The CVSS v2 temporal score (0.00 to 10.00).
- **CPE 2.3 identifier:** Specify a relevant Common Platform Enumeration identifier in URI format (v2.3) to associate with the reported vulnerability (optional).

Creating and editing queries

Custom queries can be created by users with the default role of administrator or higher from the [Queries page](#) by clicking the **New query** button. Once created, custom queries can be edited and copied. System queries cannot be edited directly, but can be copied if you wish to make changes.

Search query syntax

runZero supports deep searching across the Asset, Service, and Wireless Inventory, across organizations and sites, and through the Query Library. The runZero Export API uses the same inventory search syntax to filter results.

Query syntax

Boolean operators

Search queries can be combined through **AND** and **OR** operators and be grouped using parenthesis.

AND

For example, a Asset Inventory query of `os:"Windows 10" AND protocols:http AND protocols:smb2` will show only those assets where Windows 10 was identified and both SMB and a web server were discovered. Search values that contain spaces must be placed in double quotes.

OR

By contrast, the example query of `os:"Windows 10" AND protocols:http OR protocols:smb2` will search for Windows 10 running a web server **or** any assets with the SMB service exposed. In addition to **AND** and **OR**, the **NOT** operator can be used to filter a query. For example, the query `os:"Windows 10" AND NOT protocols:http` will show Windows 10 systems without a web server. If the negation

should happen as the first term the `AND` should be dropped. The query `NOT protocol:http AND os:"Windows 10"` is equivalent to the previous search, with the terms reversed.

Wildcard and fuzzy searches

Most keywords are a fuzzy match by default. To force an exact match, prefix match, or suffix match, the `=` prefix can be applied to the search term, with the `%` character used as a wildcard. To search an operating system name of just `Windows`, the Asset Inventory query would be `os:="Windows"`, while to specify a prefix match of `Ubuntu Linux`, the query `os:="Ubuntu Linux%"` can be used.

Single-character wildcard

The `%` wildcard matches any number of characters. To match exactly one character, use the `_` single-character wildcard. For example, `os:="Window%"` will match both `Windows` and `Windows 10`, while `os:="Window_"` will match `Windows` but not `Windows 10`.

Time and date values

Time and date (timestamp) fields can be searched using `<` (less than) and `>` (greater than) operators to compare against the current time. You can also use `-` to compare to a relative time in the past, for example `<-3years` would look for timestamps that occurred before three years ago. Supported units:

- hours
- minutes
- seconds
- months
- years

A special value of `now` can also be used.

For example, an asset search of `first_seen:<1year` would search for assets first detected this year. Other examples:

```
first_seen:<3days
```

```
first_seen:>2019-08-01
```

```
first_seen:>8/1/2019
```

```
last_seen:<1week
```

```
last_seen:<2months
```

```
last_seen:<1year
```

```
created_at:>2weeks
```

```
created_at:<30minutes
```

```
updated_at:>1year
```

`updated_at:<12hours`

`os_eol:<now`

`os_eol:>4weeks`

`os_eol:<-2years`

`os_eol_extended:>now`

`os_eol_extended:>90days`

Empty values

To search for an empty value, the `=` prefix can be used with no value after. For example, the query `os:=` will find assets with no identified operating system.

Note that this only works for single-valued attributes such as `os` and `type`; it won't work for multi-value attributes such as `names` OR `addresses`.

Asset and service inventory searches

Asset and Service attributes support two special search types in addition to the documented keywords:

- Asset Inventory searches treat unknown keywords as filters against individual Asset **attributes**.
- Service Inventory searches treat unknown keywords as filters against individual Service **data** values.

In situations where an Asset keyword conflicts with a Service data key, or an Asset attribute conflicts with a Service keyword, the prefixes `_asset.` and `_service.` can be used to disambiguate.

Searches are handled slightly differently. Service queries can filter against Asset attributes (`os:linux`) and Service attributes (`banner:Password`), but the Asset queries are limited to summary information about services (`protocol:ssh`).

Query examples

There are endless ways to combine terms and operators into effective queries, and the examples below can be used as-is or adjusted to meet your needs.

Network configurations and access

- Multihomed assets with public and private IP addresses:

`alive:t AND has_public:t AND has_private:t`

- Multihomed assets connected only to private networks

multi_home:t AND has_public:f

- Default SSH configuration using passwords for authentication:

alive:t AND protocol:"ssh" AND ssh.authMethods:"=password"

- Microsoft FTP servers:

alive:t AND protocol:"ftp" AND banner:"=Microsoft FTP%"

- Remote access services/protocols:

protocol:rdp OR protocol:vnc OR protocol:teamviewer

- Assets with public IPs running remote access services:

has_public:t OR has_public:t AND alive:t AND (protocol:rdp OR protocol:vnc OR protocol:teamviewer)

- Open ports associated with cleartext protocols:

port:21 OR port:23 OR port:80 OR port:443 OR port:139 OR port:445 OR port:3306 OR port:1433 OR port:161 OR port:8080 OR port:3389 OR port:5900

- Telnet on nondefault ports:

protocol:telnet AND NOT port:23

- Windows assets offering SMB services:

os:windows AND protocol:smb1 OR protocol:smb2

- Switch assets accepting Username and Password authentication:

type:switch AND (_asset.protocol:http AND NOT _asset.protocol:tls) AND (html.inputs:"password:" OR last.html.inputs:"password:" OR has:http.head.wwwAuthenticate OR has:last.http.head.wwwAuthenticate)

- Assets more than 8 hops away:

attribute:"ip.ttl.hops" AND ip.ttl.hops:>"8

Asset lifecycle and hardware

- Assets created as a result of arbitrary responses:

has_mac:f AND has_name:f AND os:= AND hardware:= AND detected_by:icmp AND service_count:<2

- End of Life assets:

os_eol:<now

- Assets where both OS support and extended support are expired:

os_eol:<now AND os_eol_extended:<now

- Assets where OS support is EOL but still covered by extended support:

os_eol:<now AND os_eol_extended:>now

- EOL Linux operating systems:

os:linux AND os_eol:<now

- EOL Windows operating systems:

os:windows AND os_eol:<now

- Assets discovered within the past two weeks:

first_seen:"<2weeks"

- All available serial number sources

protocol:snmp has:snmp.serialNumbers OR hw.serialNumber:t OR ilo.serialNumber:t

- Asset serial numbers from SNMP:

protocol:snmp has:snmp.serialNumbers

- Older Windows OSes:

os:"Windows Server 2012" OR os:"Windows 7"

- Older Linux OSes:

OS:linux AND os_eol:<now

- BACnet devices:

type:bacnet

- Hikvision DVRs:

type:dvr AND os:hikvision

- IoT Devices:

type:"IP Camera" OR type:"thermostat" OR type:"Amazon Device" OR hw:"Google Chromecast" OR type:"Game Console" OR type:"Robotic Cleaner" OR type:"Nest Device" OR type:"Network Audio" OR type:"Smart TV" OR type:"VR Headset" OR type:"Voice Assistant"

- Video-related assets:

type:"IP Camera" OR type:"DVR" OR type:"Video Encoder"

Misconfigurations

- SMBv1:

protocol:"smb1"

- Remote access with common services:

protocol:rdp OR protocol:vnc OR protocol:teamviewer OR protocol:spice OR protocol:pca

- Switches with default configurations for web access:

type:switch AND (_asset.protocol:http AND NOT _asset.protocol:tls) AND (html.inputs:"password:" OR last.html.inputs:"password:" OR has:http.head.wwwAuthenticate OR has:last.http.head.wwwAuthenticate)

- Default SSH configurations using passwords for authentication:

alive:t AND protocol:"ssh" AND ssh.authMethods:"=password"

- Switches using Telnet or HTTP for remote access:

type:switch AND protocol:telnet OR protocol:http

- Microsoft FTP servers:

alive:t AND protocol:"ftp" AND banner:"=Microsoft FTP%"

- Virtual machines that are not syncing time with the host:

@vmware.vm.config.tools.syncTimeWithHost:"False"

Weak configurations

- Telnet (vs. SSH):

protocol:telnet

- FTP on ports 10-21 (vs. FTPS on port 990):

protocol:ftp

- FTP on ports 20-21 (vs. SCP on port 22):

protocol:ftp

- HTTP on port 80 (vs. HTTPS on port 443):

protocol:http

- SSH versions < 2.0:

protocol:ssh AND NOT banner:"SSH-2.0"

- TLS:

tls.versionName:"=TLSv1.3" OR tls.versionName:"=TLSv1.2" OR tls.versionName:"=TLSv1.1" OR
tls.versionName:"=TLSv1.0"

- LDAP on port 389 (vs. LDAPS on port 636):

protocol:ldap OR port:389

- Wireless access points without WPA authentication:

not authentication:WPA

- Online assets with SSH accepting password authentication:

alive:t AND has:"ssh.authMethods" AND protocol:"ssh" AND (ssh.authMethods:"=password" OR
ssh.authMethods:"=password%publickey")

- Detect OpenSSL version 3.0 - 3.0.6:

product:openssl AND version:3.0

EDR / MDM

- CrowdStrike coverage gaps:

not edr.name:crowdstrike AND (type:server OR type:desktop OR type:laptop)

- Assets with CrowdStrike Agent status "Not Provisioned":

@crowdstrike.dev.provisionStatus:"NotProvisioned"

- Assets with CrowdStrike Agent mode "Reduced Functionality":

@crowdstrike.dev.reducedFunctionalityMode:"yes"

- Assets with CrowdStrike Agent status "Normal":

@crowdstrike.dev.status:"normal"

- SentinelOne coverage gaps:

not edr.name:Sentinelone AND (type:server OR type:desktop OR type:laptop)

- Assets with SentinelOne Agent requiring patch:

(alive:t OR scanned:f) AND has:"@sentinelone.dev.appsVulnerabilityStatus" AND
@sentinelone.dev.appsVulnerabilityStatus:"=patch_required"

- Assets missing either CrowdStrike or SentinelOne EDR agents:

NOT edr.name:crowdstrike AND (type:server OR type:desktop OR type:laptop) OR NOT edr.name:sentinelone AND (type:server OR type:desktop OR type:laptop)

- Miradore coverage gaps:

not source:Miradore AND (os:google android OR os:apple ios) AND type:mobile

- Microsoft Defender coverage gaps:

not edr.name:"Defender" AND os:Windows

- Assets not managed by a Microsoft product:

source:runzero AND NOT (source:ms365defender OR source:intune OR source:azuread)

- Find mobile devices on the network:

(os:google ANDroid OR os:apple ios) AND type:mobile

- Known FCC security threats, like Kaspersky:

alive:t AND edr.name:Kaspersky

Virtual machine configurations

- Virtual machines with less than 8 GB of memory:

@vmware.vm.config.hardware.memoryMB:<"8192"

- VMs with less than 16GB of memory:

@vmware.vm.runtime.maxMemoryUsage:"16384"

- Virtual machines that are not syncing time with the host:

@vmware.vm.config.tools.syncTimeWithHost:"False"

- Virtual machines that are configured with floppy drives:

@vmware.vm.config.extra.floppy0.autodetect:"true"

- Virtual machines running VMware tools:

@vmware.vm.config.extra.guestinfo.vmtools.versionString:"_"

- Virtual machines running Windows:

source:VMware AND os:Windows

- Virtual machines running Linux:

source:VMware AND os:Linux

Vulnerability concerns

- Rapid7 - fails PCI compliance:

```
test.pciComplianceStatus:"fail"
```

- Tenable - High and Critical severity vulnerabilities that are on CISA's Known Exploited list:

```
plugin.xrefs.type:"CISA-KNOWN-EXPLOITED" AND (severity:high OR severity:critical)
```

- Tenable - Critical severity vulnerabilities where exploits are available:

```
plugin.exploitabilityEase:"Exploits are available" AND severity:critical
```

- Tenable - High and Critical severity vulnerabilities where exploits are not required

```
plugin.exploitabilityEase:"No exploit is required" AND (severity:critical OR severity:high)
```

Wireless results

- Search ESSID for authentication exceptions:

```
ssid:"<ESSID>" AND NOT authentication:"wpa2-enterprise"
```

- Find unknown BSSIDs broadcasting known ESSID (exclude known BSSIDs in query for gap analysis)

```
ssid:="<ESSID>" AND NOT bssid:"<MAC address>"
```

Inventory keywords

The data across your runZero inventories can be queried and filtered using the [search syntax](#) in conjunction with the available inventory keywords. Keywords and example values are documented for the following inventories:

- [Assets](#)
- [Services](#)
- [Software](#)
- [Vulnerabilities](#)
- [Certificates](#)
- [Wireless networks](#)
- [Users](#)
- [Groups](#)

Asset inventory

When viewing assets, you can use the following keywords to search and filter.

User-specified fields

Comments

Use the syntax `comment:<text>` to search comments on an asset.

```
comment:"contractor laptop"
```

```
comment:"imaging server"
```

Tags

Use the syntax `tag:<term>` to search tags added to an asset. The term can be the tag name, or the tag name followed by an equal sign and the tag value. Tag value matches must be exact. The tag value can be empty, in which case only tags without a value will match.

```
tag:"group"
```

```
tag:"group="
```

```
tag:"group=production"
```

Organization name or ID

Use the syntax `organization:<term>` to filter by organization name or ID.

```
organization:runZero
```

```
organization:"Temporary Project"
```

```
organization:f1c3ef6d-cb41-4d55-8887-6ed3cfb3d42d
```

Site name or ID

Use the syntax `site:<term>` to filter by site name or ID.

```
site:Primary
```

```
site:"Branch Office"
```

```
site:ad67d649-041b-439d-af59-f200053a8899
```

Explorer name or ID

Use the syntax `explorer:<term>` to filter by Explorer name or ID.

```
explorer:DESKTOP-AB451F
```

```
explorer:8b927a8e-d405-40e9-aa47-d6afc9bff237
```

Hosted zone

Use the syntax `hosted_zone:<zone name>` to filter by the hosted runZero Explorer that found the asset. Using this filter after a hosted scan can be a good way to locate externally facing assets.

Owner

Use the syntax `owner:<term>` to filter by owner name.

```
owner:user@runzero.com
owner:"Security Team"
```

Ownership status

Use the syntax `owner_count:<number>` to filter by owner count. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
owner_count:>0
owner_count:0
```

Use the syntax `has_owner:<boolean>` to find assets with owners or assets that are missing owners.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
has_owner:t
has_owner:f
```

Use the syntax `ownership_type:<term>` to filter assets by ownership type name. This will return assets that have an owner assigned for the specified ownership type.

```
ownership_type:"Asset Owner"
ownership_type:"Security Owner"
```

Asset fields

Asset ID

The ID field is the unique identifier for a given asset, written as a UUID. Use the syntax `id:<uuid>` to filter by ID field.

```
id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

Operating system

The operating system field is a string describing the detected operating system software. This field is searched using the syntax `os:<text>`. The OS version, if available, can be searched

using os_version:<number>.

os:"Windows"

os:"Ubuntu Linux"

os_version:8

OS CPE

The operating system Common Platform Enumeration (CPE) field is a string describing the detected operating system software aligned to the CPE naming scheme. This field is searched using the syntax `os.cpe23:<text>`. In cases where runZero was able to fingerprint the operating system but the NIST database does not contain an official matching entry, an unofficial CPE will be generated and include `r0_unofficial` in the other field of the CPE.

os.cpe23:"ubuntu"

os.cpe23:"cpe:/o:canonical:ubuntu_linux:22.04.1"

os.cpe23:"cpe:/o:alma:linux:-::~~~~~r0_unofficial"

Type

The type field is a string describing the detected system type, such as Desktop, Laptop, Server, BMC, or Mobile. Use the syntax `type:<text>` to search this field.

type:Desktop

type:BMC

type:"Game Console"

Hardware

The hardware field is a string describing the detected physical hardware, such as `macMini` or `Nintendo Switch`. Use the syntax `hardware:<text>` to search this field.

hardware:Switch

hardware:macMini

Hostnames

The hostnames associated with an asset are obtained from DNS and exposed services. Use the syntax `name:<text>` to search these names.

name:"www"

name:"TV"

To search an asset where any asset has a specific prefix or suffix, use the `:=` exact match operator, and use `%` as a wildcard:

```
name:="FTP.%"
```

```
name:="%-09"
```

Use the syntax `name_count:<number>` to search the hostname count. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
name_count:>1
```

Use the syntax `name_overlap:<boolean>` to find assets sharing the same name. The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
name_overlap:t
```

Domains

The domains associated with an asset are obtained from DNS and exposed services. Use the syntax `domain:<domainname>` to search the domain names.

```
domain:"amazon.com"
```

```
domain:"corp.lan"
```

```
domain:"WORKGROUP"
```

The domain count can be searched using the syntax `domain_count:<number>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
domain_count:>1
```

Addresses

Use the syntax `address:<ip>` to search the addresses (both primary and secondary) associated with an asset, `primary_address:<ip>` to search only the primary addresses associated with an asset, or `secondary_address:<ip>` to search only the secondary addresses associated with an asset. These keywords also allow for CIDR mask matching, as well as wildcard matches using `'%'`. A comma-separated list of addresses will be used as an efficient multiple-match.

```
address:192.168.0.1
```

```
address:10.0.0
```

```
address:10.1.2.0/24
```

```
address:%.0.1
```

```
address:10.%.254
```

```
address:10.0.0.1,10.0.0.2,10.0.0.3
```


Use the syntax `address_count:<term>` and `address_extra_count:<number>` to search address primary and secondary counts. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`address_extra_count:0`

Use the syntax `address_overlap:<boolean>` to find assets sharing primary IP addresses. This can be further filtered to single sites using the `site` keyword. The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`address_overlap:t`

Use the syntax `address_extra_overlap:<boolean>` to find assets sharing secondary IP addresses. This can be further filtered to single sites using the `site` keyword. The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`address_extra_overlap:t`

Networks

Use the syntax `net:<cidr>` to search the addresses (both primary and secondary) associated with an asset by CIDR mask.

`net:192.168.0.0/24`

Default SNMP communities

Use the syntax `has:snmp.v2DefaultCommunities` to search for assets with a default SNMP community (public, private, and other defaults).

`has:snmp.v2DefaultCommunities`
`snmp.v2DefaultCommunities:public`

Public address

Use the keyword `has_public` and syntax `has_public:<boolean>` to locate any asset with a non-reserved IP address. This often corresponds to public-facing systems, though public IPs can also be used internally behind a firewall. Note that public IPv6 addresses are included by this filter; to search for only public IPv4 addresses, you can use `has_public_v4`.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`has_public:true`

Private address

Use the keyword `has_private` and syntax `has_private:<boolean>` to locate any asset with a private IP address.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`has_private:false`

IPv6 address

Use the keyword `has_ipv6` and the syntax `has_ipv6:<boolean>` to locate any asset with an identified IPv6 address.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`has_ipv6:false`

Link-local IPv6 address

Use the keyword `has_link_local` and syntax `has_link_local:<boolean>` to locate any asset with an identified IPv6 link local (`fe80::`) address.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`has_link_local:true`

MAC address

Use the syntax `mac:<term>` to search MAC addresses associated with an asset.

`mac:00:5c:04`

`mac:00:00:1c`

Use the syntax `mac_count:<number>` to search the MAC address count. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`mac_count:>2`

If you use exact search (`:=`) you can also search for full MAC addresses in Cisco format or dash-separated format:

`mac:=00-10-fa-c2-bf-d5`

```
mac:=0010.fac2.bfd5
```

Use the syntax `mac_overlap:<boolean>` to find assets sharing the same MAC address. The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
mac_overlap:t
```

MAC address vendors

The vendor associated with the MAC addresses of an asset can be searched using the syntax `mac_vendor:<text>`.

```
mac_vendor:Apple
```

```
mac_vendor:"Intel Corporate"
```

To search only the vendor associated with the newest MAC address, use the syntax

```
newest_mac_vendor:<text>
```

```
newest_mac_vendor:Apple
```

The MAC address vendor count can be searched using the syntax `mac_vendor_count:<number>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
mac_vendor_count:0
```

MAC address age

Use the syntax `mac_age:<term>` to search the allocation date of the newest MAC address associated with an asset. The term supports the standard runZero [time comparison syntax] [time].

```
mac_age:>1year
```

```
mac_age:<6months
```

```
mac_age:2019-12-31
```

Outlier score

Use the syntax `outlier_score:<value>` to search the calculated outlier score of assets. The outlier score is in the range 0 to 5 inclusive. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
outlier_score:>2
```

```
outlier_score:0
```

Upstream switch IP address

Use the syntax `switch.ip:<address>` to search the IP address of the upstream switch assets are connected to.

```
switch.ip:192.168.1.1
```

```
switch.ip:fe80::81f2:1c9d:5ac9:5420
```

Upstream switch name

Use the syntax `switch.name:<hostname>` to search the hostname of the upstream switch assets are connected to.

```
switch.name:"SWITCH-1"
```

```
switch.name:office
```

Upstream switch port

Use the syntax `switch.port:<address>-<port number string>` to search the port on the upstream switch assets that are connected to.

```
switch.port:192.168.1.1-25
```

```
switch.port:10.1.2.3-0/1/2
```

Upstream switch shared port

Use the syntax `attribute:switch.portShared` to find assets which connect to a switch port that reports multiple MAC addresses.

```
attribute:switch.portShared
```

Attributes

Use the syntax `attribute:<term>` to search the asset attribute fields, such as the port used to detect the TTL.

```
attribute:"ip.ttl.port"
```

```
attribute:"cpe:/a:isc:bind:9.11.3"
```

```
attribute:"9.11.3"
```

To determine if an asset has any attribute defined, use the `has:<attribute-name>` keyword. The `has` keyword can be inverted to find missing fields with `not has:<term>`.

```
has:"ip.ttl.port"
```

```
not has:"rdns.names"
```

In addition to the standard fields, the following special attributes are available:

- `has:screenshot` returns assets where at least one screenshot was obtained.
- `has:icons` returns assets where at least one icon was obtained (HTTP, UPnP, or similar).
- `has:uplink` returns assets seen in the CAM table of a network switch.
- `has:switch` is an alias of `has:uplink`.
- `has:downlink` returns assets where the CAM table was queried at least one other asset was connected.
- `has:unmapped` returns assets where the CAM table was queried at least one other asset was connected but not identified by IP.

See [Topology role](#) below for the related `is:`, `gateway_kind:`, `child_kind:`, `downlink_count:`, and `hops_from_public:` keywords.

The attribute can be specified as a term directly. If the attribute name conflicts with an existing term, the prefix `_asset.` can be specified to disambiguate the query.

```
ip.ttl.port:80
```

```
rdns.names:"router"
```

```
_asset.ip.ttl.hops:"1"
```

Foreign attributes from third-party inbound integrations can be queried using the syntax `@<integration>.<source>.<attribute>:<term>`. The table below includes the correct prefix for each integration.

Integration	Prefix
Miradore	@miradore.dev.
AWS EC2	@aws.ec2.
AWS ELB & ELBv2	@aws.elb.
AWS RDS	@aws.rds.
CrowdStrike	@crowdstrike.dev.
Azure Load Balancer	@azure.vm.
Azure VM	@azure.vm.
Azure Scale Set VM	@azure.vms.
Censys	@censys.host.
VMWare	@vmware.vm.
GCP Load Balancer	@gcp.lb.
GCP E2-Micro VM	@gcp.vm.
GCP CloudSQL	@gcp.cloudsql.
SentinelOne	@sentinelone.dev.

Tenable.io & Nessus	@tenable.dev.
Rapid7 Nexpose & InsightVM	@rapid7.dev.
Qualys VMDR	@qualys.dev.
Shodan	@shodan.dev.
Azure AD	azuread
Active Directory (LDAP)	@ldap.computer.
Microsoft 365 Defender	@ms365defender.dev.
Microsoft Intune	@intune.dev.
Google Workspace ChromeOS	@googleworkspace.chromeos.
Google Workspace Endpoint	@googleworkspace.endpoint.
Google Workspace Mobile	@googleworkspace.mobile.

```
@aws.ec2.region:"us-east-2"
```

```
@crowdstrike.dev.agentVersion:"6.49.16201.0"
```

```
@googleworkspace.chromeos.model:"HP Chromebook"
```

Foreign attributes support semver-style comparison operators (>, >=, <, <=, =). Include at least the major and minor version number to use this search option. Any additional non-numerical text included in the prefix or suffix will be ignored in version comparison.

```
@jamf.custom.osVersion:>6.0
```

Topology role

runZero builds a topology graph from layer-2 switch CAM tables and from sub-asset enumeration through [protocol gateways](#). The keywords below filter assets based on that graph and work in **Inventory** → **Assets**, in any asset search box, and in the [Network Map](#).

Keyword	Matches
is:switch	Asset is a layer-2 switch with one or more observed CAM-table entries (downstream ports).
is:gateway	Asset is a protocol gateway fronting one or more enumerated sub-assets.
is:child	Asset is a sub-asset enumerated through a protocol gateway.
gateway_kind: <proto>	Asset is a gateway for the given protocol — bacnet, cip, modbus, s7comm, dnp3, profinet, ethercat, omronfins, melsecq, hartip, c37118, knxnet.
child_kind: <proto>	Asset is a sub-asset behind a gateway of the given protocol (same vocabulary as gateway_kind).

downlink_count:<n>	Number of downstream links (switch ports + OT gateway children). Supports comparisons: >10, <=0, >=1, ranges.
hops_from_public:<n>	Minimum topology hops to a public-facing asset (0 = the asset itself is public-facing). Supports comparisons.
has:uplink	Asset is connected to an upstream switch port (alias has:switch).
has:downlink	Asset is a switch with at least one downstream port observed.

Examples:

```
is:gateway
```

```
gateway_kind:modbus
```

```
is:child AND child_kind:cip
```

```
is:switch AND downlink_count:>10
```

```
category:OT AND hops_from_public:<=2
```

Asset services**Service ports**

The TCP and UDP services associated with an asset can be searched by port number using the syntax `port:<number>`.

```
port:80
```

```
port:161
```

Service TCP ports

Use the syntax `tcp:<number>` to search the TCP services associated with an asset by port number.

```
tcp:443
```

To search for assets with a specific list of TCP ports open, you can use the syntax `service_ports_tcp:=<list>`. Values should be in ascending numerical order, and separated by commas.

```
service_ports_tcp:=80,443
```

Service UDP ports

Use the syntax `udp:<number>` to search UDP services associated with an asset by port number.

```
udp:53
```

To search for assets with a specific list of UDP ports open, you can use the syntax `service_ports_udp:=<list>`. Values should be in ascending numerical order, and separated by commas.

```
service_ports_udp:=53,123
```

Service protocols

Use the syntax `service_protocols:<term>` (or `protocol:<term>` for short) to search the identified service protocols associated with an asset.

```
protocol:http
```

```
service_protocol:telnet
```

The protocol count can be searched using the syntax `protocol_count:<number>`. This search supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
protocol_count:>1
```

Service products

Use the syntax `service_products:<term>` (or `product:<term>` for short) to search for the identified service products associated with an asset.

```
product:openssh
```

```
service_products:nginx
```

The product count can be searched using the syntax `product_count:<number>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
product_count:>3
```

Service counts

Use the following keywords to search the number of services associated with an asset can be searched by port number:

- `service_count_tcp:<number>`
- `service_count_udp:<number>`
- `service_count_icmp:<number>`
- `service_count_arp:<number>`

These keywords support numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

Examples include:

```
service_count_tcp:>=5
```

```
service_count_arp:0
```

```
service_count_udp:<=1
```


Asset tracking fields

Timestamps

Use the following syntaxes to search the asset timestamp fields (`first_seen`, `last_seen`, `created_at`, `updated_at`, `os_eol`, `os_eol_extended`):

- `first_seen:<term>`
- `last_seen:<term>`
- `created_at:<term>`
- `updated_at:<term>`
- `os_eol:<term>`
- `os_eol_extended:<term>`

The term supports the standard runZero [time comparison syntax][time].

`first_seen:<3days`

`first_seen:>2019-08-01`

`first_seen:>8/1/2019`

`last_seen:<1week`

`last_seen:<2months`

`last_seen:<1year`

`created_at:>2weeks`

`created_at:<30minutes`

`updated_at:>1year`

`updated_at:<12hours`

`os_eol:<now`

`os_eol:>4weeks`

`os_eol_extended:>now`

`os_eol_extended:>90days`

Online status

Use the syntax `online:<boolean>` or the inverse syntax `offline:<boolean>` to search the online status of an asset.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

online:t

online:1

offline:0

Operating system support status

The syntax `os_eol_expired:<boolean>` can be used to find identify assets based on whether their operating systems are End of Life (EOL). This field evaluates both the `os_eol` and `os_eol_extended` values to only return assets with expired coverage.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

os_eol_expired:t

os_eol_expired:1

os_eol_expired:no

Detection method

The detected by attribute of an asset can be searched using the syntax `det:<term>` or `detected_by:<term>`. The term is one of `arp`, `icmp`, `<portnumber>-tcp`, or `<portnumber>-udp`. In the case of multiple detections, the priority goes `arp`, `icmp`, and then the first detected service.

det:arp

detected_by:80-tcp

det:53-udp

Time to Live (TTL) comparisons

Use the syntax `ttl:<term>` and `lowest_ttl:<term>` to search the lowest TTL of an asset. TTL is the estimated number of hops between the scan source and the asset.

This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

lowest_ttl:>3

Round Trip Time (RTT) comparisons

Use the syntax `rtt:<term>` and `lowest_rtt:<term>` to search the lowest RTT for an asset. RTT is the round-trip response time of a given probe measured in nanoseconds (1,000,000 == 1ms).

This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

lowest_rtt:>500000000

Multiple MAC address status

Use the syntax `multi_mac:<boolean>` to determine if an asset has multiple MAC addresses.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
multi_mac:t
```

Any MAC address status

Use the syntax `has_mac:<boolean>` to find assets with any MAC addresses.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
has_mac:yes
```

```
has_mac:f
```

Multiple IP address status

Use the syntax `multi_home:<boolean>` to determine if an asset has multiple IP addresses.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
multi_home:t
```

Multiple hostname status

Use the syntax `multi_name:<boolean>` to find assets with multiple hostnames.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
multi_name:yes
```

```
multi_name:false
```

Software installations

Use the syntax `software:<term>>` to find assets with associated software.

The term has three forms:

- `software:<product>` will look for any assets with a software product that matches the term.
- `software:<product>/<version>` will look for any assets with a software product and version that matches the term exactly.
- `software:<vendor>/<product>/<version>` will look for any assets with a software vendor, product, and version that matches the term exactly.

All three forms allow the use of % as a wildcard (beginning, middle, or end of the term).

Version terms can be specified with comparators to search for versions that are less than or greater than a specific version. For example `software:"Google/Google Chrome/>=135"` will return all assets with Google Chrome version 135 or greater.

```
software:IIS
```

```
software:Microsoft/IIS/10.0
```

Certificate fields

Certificate-related fields can be searched using the following keywords.

Certificate ID

Use the syntax `certificate_id:<uuid>` to filter by certificate ID.

```
certificate_id:4e3a2b1c-5d6f-7a8b-9c0d-1e2f3a4b5c6d
```

Certificate type

Use the syntax `certificate_type:<term>` to search by certificate type.

```
certificate_type:"x509"
```

Certificate serial

Use the syntax `certificate_serial:<term>` to search by certificate serial number.

```
certificate_serial:"01A23B45C"
```

Certificate public key

Use the syntax `certificate_public_key:<term>` to search by certificate public key.

```
certificate_public_key:"MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A..."
```

Certificate public key algorithm

Use the syntax `certificate_public_key_algorithm:<term>` to search by certificate public key algorithm.

```
certificate_public_key_algorithm:"ecPublicKey"
```

Certificate signature

Use the syntax `certificate_signature:<term>` to search by certificate signature.

```
certificate_signature:"abcdef1234567890"
```

Certificate signature algorithm

Use the syntax `certificate_signature_algorithm:<term>` to search by certificate signature algorithm.

```
certificate_signature_algorithm:"rsaEncryption"
```

Certificate fingerprint (bkh hash)

Use the syntax `certificate_fp_bkhash:<term>` to search by certificate fingerprint (bkh hash).

```
certificate_fp_bkhash:"d2c7e8f9..."
```

Certificate fingerprint (SHA1)

Use the syntax `certificate_fp_sha1:<term>` to search by certificate SHA1 fingerprint.

```
certificate_fp_sha1:"349eb7db55bbad9d7deabc5effdfd5521ed984a0"
```

Certificate fingerprint (SHA256)

Use the syntax `certificate_fp_sha256:<term>` to search by certificate SHA256 fingerprint.

```
certificate_fp_sha256:"SHA256:Sc0pLUCv1NaGtu4Xy2f0Te1A6cC+KyU3x7xuN8+aLtA="
```

Certificate subject

Use the syntax `certificate_subject:<term>` to search by certificate subject.

```
certificate_subject:"CN=example.com,O=ExampleCorp,C=US"
```

Certificate common name (CN)

Use the syntax `certificate_cn:<term>` to search by certificate common name.

```
certificate_cn:"example.com"
```

Certificate issuer

Use the syntax `certificate_issuer:<term>` to search by certificate issuing authority.

```
certificate_issuer:"CN=Example CA,O=Example Corp,C=US"
```

Certificate subject key ID

Use the syntax `certificate_subject_key_id:<term>` to search by certificate subject key ID.

```
certificate_subject_key_id:"1234567890abcdef"
```

Certificate authority key ID

Use the syntax `certificate_authority_key_id:<term>` to search by certificate issuing authority key ID.

```
certificate_authority_key_id:"abcdef1234567890"
```

Service inventory

When viewing services, you can use the keywords in this section to search and filter.

Ports

The TCP and UDP services associated with a service can be searched by port number using the syntax `port:<number>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
port:<=25
```

TCP ports

Use the syntax `tcp:<number>` to search TCP service associated with a service by port number.

```
tcp:53
```

To search for all services on assets with a specific list of TCP ports open, you can use the syntax `service_ports_tcp:=<list>`. Values should be in ascending numerical order, and separated by commas.

```
service_ports_tcp:=80,443
```

UDP ports

Use the `udp:<number>` syntax to search UDP services associated with a service by port number.

```
udp:443
```

To search for all services on assets with a specific list of UDP ports open, you can use the syntax `service_ports_udp:=<list>`. Values should be in ascending numerical order, and separated by commas.

```
service_ports_udp:=53,123
```

Transport

Use the syntax `transport:<term>` to search the transport associated with a service by name.

```
transport:tcp
```

```
transport:udp
```

```
transport:icmp
```

Protocol

Use the syntax `service_protocols:<term>` (or `protocol:<term>` for short) to search the protocols associated with services.

```
protocol:http
```

```
protocol:telnet
```

Assets with product

Use the syntax `service_products:<term>` (or `product:<term>` for short) to search for the identified service products associated with an asset, and return all services for the matching assets.

```
product:openssh
```

```
service_products:nginx
```

Virtual Host (vHost)

Use the syntax `vhost:<text>` to search for virtual hosts associated with a service by name .

```
vhost:"www"
```

Address

Use the keyword `service_address` to match against the service IP address.

```
service_address:192.168.0.1
```

Public address

Use the keyword `service_has_public` and syntax `service_has_public:<boolean>` to locate any service with a non-reserved I address.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
service_has_public:true
```

Private address

Use the keyword `service_has_private` and syntax `service_has_private:<boolean>` to locate any service with a private IP address.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

```
service_has_private:false
```

IPv6 address

Use the keyword `service_has_ipv6` and the syntax `service_has_ipv6:<boolean>` to locate any service with an identified IPv6 address.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

```
service_has_ipv6:false
```

Link-local IPv6 address

Use the keyword `service_has_link_local` and syntax `service_has_link_local:<boolean>` to locate any service with an identified IPv6 link local (fe80::) address.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

```
service_has_link_local:true
```

Assets with outlier score

You can use the syntax `outlier_score:<value>` to search the calculated outlier score of assets, and return all services on those assets. The outlier score is in the range 0 to 5 inclusive. This search term supports numerical comparison operators (>, >=, <, <=, =).

```
outlier_score:>2
```

```
outlier_score:0
```

Assets with MAC address vendors

To search the vendors associated with the MAC addresses of an asset, and return all services on those assets, use the syntax `mac_vendor:<text>`.

```
mac_vendor:Apple
```

```
mac_vendor:"Intel Corporate"
```


To search only the vendor associated with the newest MAC address, use the syntax

```
newest_mac_vendor:<text>
```

```
newest_mac_vendor:Apple
```

Assets with MAC address age

To search the ages of the newest MAC addresses associated with each asset, and return all services associated with those assets, use the syntax `mac_age:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
mac_age:>1year
```

```
mac_age:<6months
```

```
mac_age:2019-12-31
```

Attributes

You can search all service attributes with the syntax `<attribute>:<term>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

If the attribute name conflicts with an existing term, the prefix `_service.` can be added to disambiguate the query.

Note that service attributes can be slow and it is often better to prefix `_asset.protocol:<term>` filter in front of the service attribute query. For example, to search for SSH banners, use the syntax `_asset.protocol:ssh AND banner:<term>`.

```
banner:password
```

```
service.product:"OpenSSH"
```

```
html.title:"Apache2 Ubuntu Default Page"
```

```
http.code:>=500
```

```
screenshot.image.size:>=100000
```

```
_service.arp.macVendor:Xerox
```

To determine if a service has an attribute at all, use the `has` keyword. The `has` keyword can be inverted to find missing fields, with `not has:<term>`.

```
has:"http.head.server"
```

```
not has:"html.title"
```

Certificate fields

Certificates related to the service can be searched as well.

Certificate ID

Use the syntax `certificate_id:<uuid>` to filter by certificate ID.

```
certificate_id:4e3a2b1c-5d6f-7a8b-9c0d-1e2f3a4b5c6d
```

Certificate fingerprint (bkh hash)

Use the syntax `fp_bkhash:=<text>` to search for services with certificates that match the specified bkhash fingerprint anywhere in the certificate chain.

```
fp_bkhash:=d2c7e8f9a1b2c3d4e5f6
```

Certificate fingerprint (SHA256)

Use the syntax `fp_sha256:=<text>` to search for services with certificates that match the specified SHA256 fingerprint anywhere in the certificate chain.

```
fp_sha256:=SHA256:19vJPGdyh3BdGgTiSSavQsKx133lzFkJhu51n6fzG+k=
```

Certificate fingerprint (SHA1)

Use the syntax `fp_sha1:=<text>` to search for services with certificates that match the specified SHA1 fingerprint anywhere in the certificate chain.

```
fp_sha1:=a1b2c3d4e5f6g7h8i9j0
```

Certificate validity fields

Use the following syntax to search details about when the certificate found on the service was created. These fields support the standard runZero [time comparison syntax][time].

```
certificate_valid_from:>2025-01-01
certificate_valid_until:<2026-01-01
certificate_created_at:<1year
certificate_updated_at:>2024-12-01
certificate_last_seen:<2months
```

Certificate authority key ID

Use the syntax `certificate_authority_key_id:<text>` to search by certificate authority key ID.

```
certificate_authority_key_id:"c03152cd5a50c3827c7471cecbe99cf97aeb82e2"
```

Certificate common name (CN)

Use the syntax `certificate_cn:<text>` to search by certificate common name.

```
certificate_cn:"example.com"
```

Certificate public key algorithm

Use the syntax `certificate_public_key_algorithm:<text>` to search by public key algorithm.

`certificate_public_key_algorithm:rsaEncryption`

Certificate serial number

Use the syntax `certificate_serial:<text>` to search by certificate serial number.

`certificate_serial:123456`

Certificate signature algorithm

Use the syntax `certificate_signature_algorithm:<text>` to search by signature algorithm.

`certificate_signature_algorithm:sha256WithRSAEncryption`

Certificate subject key ID

Use the syntax `certificate_subject_key_id:<text>` to search by subject key ID.

`certificate_subject_key_id:"c769916e10f850397928f62bd34ca6ec39d8e00a"`

Certificate type

Use the syntax `certificate_type:<text>` to search by certificate type.

`certificate_type:x509`

Certificate key usage

Use the syntax `certificate_key_usage:<text>` to search by key usage.

`certificate_key_usage:"digitalSignature"`

Certificate extended key usage

Use the syntax `certificate_ext_key_usage:<text>` to search by extended key usage.

`certificate_ext_key_usage:"serverAuth"`

Certificate OCSP server

Use the syntax `certificate_ocsp_server:<text>` to search by OCSP server.

`certificate_ocsp_server:http://ocsp.example.com`

Certificate CRL distribution points

Use the syntax `certificate_crl_distribution_points:<text>` to search by CRL distribution point.

certificate_crl_distribution_points:http://crl.example.com/

Certificate issuing certificate URL

Use the syntax `certificate_issuing_certificate_url:<text>` to search by issuing certificate URL.

certificate_issuing_certificate_url:http://issuer.example.com/

Certificate signature

Use the syntax `certificate_signature:<text>` to search by certificate signature.

certificate_signature:Bden73ipj8B2xb10zy5n0vIytCktGrht5xL7ZfFlalIBQxbG05Iuf6Y1yICcEpYqsgSJS6JKCdw5dujmPmGRwBZfVhIbSRb0

Certificate public key

Use the syntax `certificate_public_key:<text>` to search by certificate public key.

certificate_public_key:MIGfMA0GCsQGIb3DQEBAQUAA4GNADCBiQKBgQDJ899KGnqHjVuBekYqosp2l8zWbiyu2I62CzaqaouLtqn1nXaQLMdruh1

Certificate comments

Use the syntax `certificate_comments:<text>` to search comments on a certificate.

certificate_comments:"our self signed wilddcard"

Certificate self-signed

Use the syntax `certificate_self_signed:<boolean>` to search for self-signed certificates.

certificate_self_signed:true

Certificate hidden

Use the syntax `certificate_hidden:<boolean>` to search for certificates that have been hidden from the inventory.

certificate_hidden:true

Certificate signature algorithm insecure

Use the syntax `certificate_signature_algorithm_insecure:<boolean>` to search for certificates with insecure signature algorithms.

certificate_signature_algorithm_insecure:true

Certificate public key insecure

Use the syntax `certificate_public_key_insecure:<boolean>` to search for certificates with insecure public key algorithm and key size combinations.

certificate_public_key_insecure:true

Certificate is CA

Use the syntax `certificate_is_ca:<boolean>` to search for certificates that are certificate authorities (CAs).

```
certificate_is_ca:true
```

Software inventory

When viewing software groups, you can use the keywords in this section to search and filter.

Vendor

The vendor associated with a software can be searched by name using the syntax `vendor:<name>`.

```
vendor:oracle
```

Product

The product associated with a software can be searched by name using the syntax `product:<name>`.

```
product:java
```

Version

The version associated with a software can be searched by name using the syntax `version:<name>`.

```
version:1.2.3
```

Software instance inventory

When viewing software instances on assets, you can use the keywords in this section to search and filter.

Source

The source reporting the software installed can be searched or filtered by name using the syntax `source:<name>`.

```
source:runzero
```

Vendor

The vendor associated with a software can be searched by name using the syntax `vendor:<name>`.

```
vendor:oracle
```

Product

The product associated with a software can be searched by name using the syntax `product:<name>`.

```
product:java
```

Vulnerabilities inventory

When viewing vulnerability groups, you can use the keywords in this section to search and filter.

Name

The name field can be searched using the syntax `name:<term>`.

```
name:"Cisco IOS Software DHCP Remote Code Execution Vulnerability"
```

```
name:"PHP < 5.3.12 / 5.4.2 CGI Query String Code Execution"
```

CVE

The CVE field can be searched using the syntax `cve:<term>`.

```
cve:CVE-2021-44228
```

```
cve:CVE-2016-2183
```

KEV

Membership in a Known Exploited Vulnerability (KEV) list can be searched using the syntax `kev:<term>`.

```
kev:t
```

will search for vulnerabilities that appear on a KEV list.

Specific KEV lists can be searched by name.

- `kev:cisa` will search for vulnerabilities listed as actively exploited in the [CISA Known Exploited Vulnerabilities Catalog](#).
- `kev:vulncheck` will search for vulnerabilities listed as actively exploited in the [VulnCheck Catalog](#).
- `kev:true` will search for vulnerabilities in either of the above lists.

Severity

The severity field can be searched using the syntax `severity:<term>`.

```
severity:info
```

```
severity:medium
```

Risk

The Risk and Risk Score fields can be searched using either numeric or keyword values. Risk score is an integer from zero through four, where 0 is Info level risk and 4 indicates Critical risk.

```
risk:"Critical"
```

```
risk:2
```

Vulnerability instance count

The Asset count field can be searched using the syntax `count:<text>`.

```
count:>0
```

Site name or ID

Use the syntax `site:<term>` to filter by site name or ID.

```
site:Primary
```

EPSS score

The EPSS score can be searched using the syntax `epss_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
epss_score:>0.5
```

```
epss_score:<=0.1
```

```
epss_score:=0.9
```

Timestamps

Use the following syntaxes to search the vulnerability group timestamp fields (`created_at`, `suppressed_at`):

- `created_at:<term>`
- `suppressed_at:<term>`

The term supports the standard runZero `[time comparison syntax][time]`.

`created_at:>2weeks`

`created_at:<30minutes`

`suppressed_at:<1day`

Suppression

The current suppressed or unsuppressed state of vulnerability groups can be searched using the syntax `suppressed:<term>`.

`suppressed:t`

will search for vulnerability groups that are suppressed. Inversely,

`suppressed:f`

will search for vulnerability groups that are not suppressed.

A special value of `any` can be used to display all records, regardless of suppression settings:

`suppressed:any`

will display both suppressed and unsuppressed vulnerability groups.

Additionally, the user who performed the suppression can be found using the following syntaxes:

`suppressed_by:<username>`

Vulnerability instance inventory

When viewing vulnerability instances on assets, you can use the following keywords to search and filter information.

Vulnerability ID

The ID field is the unique identifier for a given vulnerability, written as a UUID. Use the syntax `id:<uuid>` to filter by the ID field.

`id:a124a141-e518-4735-9878-8e89c575b1d2`

Source

The source reporting the vulnerability detected can be searched or filtered by name using the syntax `source:<name>`.

`source:tenable`

Severity

The severity field can be searched using the syntax `severity:<term>`.

```
severity:info
```

```
severity:medium
```

Severity score

The severity score can be searched using the syntax `severity_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
severity_score:<5.0
```

```
severity_score:>=9.0
```

Risk

The risk field can be searched using the syntax `risk:<term>`.

```
risk:none
```

```
risk:critical
```

Risk score

The risk score can be searched using the syntax `risk_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
risk_score:>7.0
```

```
risk_score:=10.0
```

Modified risk

Vulnerabilities whose risk has been modified, either manually or by way of alert rule trigger

```
has_modified_risk:<boolean>
```

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
has_modified_risk:true
```

```
has_modified_risk:1
```

```
has_modified_risk:no
```

Category

The category field can be searched using the syntax `category:<term>`.

`category:Local`

`category:Remote`

Name

The name field can be searched using the syntax `name:<term>`.

`name:"Cisco IOS Software DHCP Remote Code Execution Vulnerability"`

`name:"PHP < 5.3.12 / 5.4.2 CGI Query String Code Execution"`

Description

The description field can be searched using the syntax `description:<term>`.

`description:"The remote device is missing a vendor-supplied security patch."`

`description:"remote code execution"`

Solution

The solution field can be searched using the syntax `solution:<term>`.

`solution:patch`

`solution:upgrade`

CVE

The CVE field can be searched using the syntax `cve:<term>`.

`cve:CVE-2021-44228`

`cve:CVE-2016-2183`

KEV

Membership in a Known Exploited Vulnerability (KEV) list can be searched using the syntax

`kev:<term>`.

`kev:t`

will search for vulnerabilities that appear on a KEV list.

Specific KEV lists can be searched by name.

- `kev:cisa` will search for vulnerabilities listed as actively exploited in the [CISA Known Exploited Vulnerabilities Catalog](#).
- `kev:vulncheck` will search for vulnerabilities listed as actively exploited in the [VulnCheck Catalog](#).
- `kev:true` will search for vulnerabilities in either of the above lists.

Exploitable

Vulnerabilities that are exploitable can be searched using the syntax `exploitable:<boolean>`.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`exploitable:true`

`exploitable:1`

`exploitable:no`

CVSSv2 score

The CVSSv2 fields, `cvss2_base_score` and `cvss2_temporal_score`, can be searched using the syntax `cvss2_base_score:<term>` and `cvss2_temporal_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`cvss2_base_score:>6.5`

`cvss2_base_score:<=3.0`

`cvss2_temporal_score:=10.0`

`cvss2_temporal_score:<5.0`

CVSSv3 score

The CVSSv3 fields, `cvss3_base_score` and `cvss3_temporal_score`, can be searched using the syntax `cvss3_base_score:<term>` and `cvss3_temporal_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

`cvss3_base_score:>6.5`

`cvss3_base_score:<=3.0`

`cvss3_temporal_score:=10.0`

`cvss3_temporal_score:<5.0`

EPSS score

The EPSS score can be searched using the syntax `epss_score:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
epss_score:>0.5
```

```
epss_score:<=0.1
```

```
epss_score:=0.9
```

Address

The address field can be searched using the syntax `address:<term>`.

```
address:192.168.0.1
```

Transport

The transport field can be searched using the syntax `transport:<term>`.

```
transport:tcp
```

```
transport:udp
```

Port

The port can be searched using the syntax `port:<term>`. The term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
port:22
```

```
port:443
```

Operating system support status

The syntax `os_eol_expired:<boolean>` can be used to find identify vulnerabilities on assets based on whether their operating systems are End of Life (EOL). This field evaluates both the `os_eol` and `os_eol_extended` values to only return vulnerabilities on assets with expired coverage.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

```
os_eol_expired:t
```

```
os_eol_expired:1
```

```
os_eol_expired:no
```

Finding code

The finding code field is the unique identifier for a given finding. Use the syntax `finding_code:<term>` to filter by the code field.

```
finding_code:rz-finding-internet-exposed-database
```

Finding name

Use the syntax `finding_name:<text>` to search by finding name.

```
finding_name:"Internet Exposed Database"
```

Finding risk

The Finding Risk and Finding Risk Score fields can be searched using either numeric or keyword values. Risk score is an integer from zero through four, where 0 is Info level risk and 4 indicates Critical risk.

```
finding_risk:"Critical"
```

```
finding_risk:2
```

Detection timestamps (first detected at, last detected at)

The timestamp fields, `first_detected_at` and `last_detected_at`, can be searched using the syntax `first_detected_at:<term>` and `last_detected_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
first_detected_at:>2weeks
```

```
first_detected_at:<30minutes
```

```
last_detected_at:>1month
```

```
last_detected_at:2hours
```

Publication timestamps (published at)

The timestamp field, `published_at`, can be searched using the syntax `published_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
published_at:>2weeks
```

```
published_at:<30minutes
```

Timestamps (created at, updated at)

Use the following syntaxes to search the vulnerability timestamp fields (`created_at`, `updated_at`, `suppressed_at`):

- `created_at:<term>`
- `updated_at:<term>`
- `suppressed_at:<term>`

The term supports the standard runZero `[time comparison syntax][time]`.

`created_at:>2weeks`

`created_at:<30minutes`

`updated_at:>1month`

`updated_at:2hours`

`suppressed_at:<1day`

Attributes

You can search all vulnerability attributes with the syntax `<attribute>:<term>`. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

If the attribute name conflicts with an existing term, the prefix `_vulnerability.` can be added to disambiguate the query.

`plugin.hasPatch:true`

`severityID:3`

`_vulnerability.state:REOPENED`

To determine if a vulnerability has any attribute defined, use the `has:<term>` keyword. The `has` keyword can be inverted to find missing fields with `not has:<term>`.

`has:plugin.vpr.score`

`not has:output`

Certificate fields

Certificate-related fields can be searched using the following keywords.

Certificate ID

Use the syntax `certificate_id:<uuid>` to filter by certificate ID.

`certificate_id:4e3a2b1c-5d6f-7a8b-9c0d-1e2f3a4b5c6d`

Certificate type

Use the syntax `certificate_type:<term>` to search by certificate type.

`certificate_type:"x509"`

Certificate serial

Use the syntax `certificate_serial:<term>` to search by certificate serial number.

`certificate_serial:"01A23B45C"`

Certificate public key

Use the syntax `certificate_public_key:<term>` to search by certificate public key.

`certificate_public_key:"MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8A..."`

Certificate public key algorithm

Use the syntax `certificate_public_key_algorithm:<term>` to search by certificate public key algorithm.

`certificate_public_key_algorithm:"ecPublicKey"`

Certificate signature

Use the syntax `certificate_signature:<term>` to search by certificate signature.

`certificate_signature:"abcdef1234567890"`

Certificate signature algorithm

Use the syntax `certificate_signature_algorithm:<term>` to search by certificate signature algorithm.

`certificate_signature_algorithm:"rsaEncryption"`

Certificate fingerprint (bkh hash)

Use the syntax `certificate_fp_bkhash:<term>` to search by certificate fingerprint (bkh hash).

`certificate_fp_bkhash:"d2c7e8f9..."`

Certificate fingerprint (SHA1)

Use the syntax `certificate_fp_sha1:<term>` to search by certificate SHA1 fingerprint.

`certificate_fp_sha1:"349eb7db55bbad9d7deabc5effdfd5521ed984a0"`

Certificate fingerprint (SHA256)

Use the syntax `certificate_fp_sha256:<term>` to search by certificate SHA256 fingerprint.

```
certificate_fp_sha256:"SHA256:Sc0pLUCv1NaGtu4Xy2f0Te1A6cC+KyU3x7xuN8+aLtA="
```

Certificate subject

Use the syntax `certificate_subject:<term>` to search by certificate subject.

```
certificate_subject:"CN=example.com,O=ExampleCorp,C=US"
```

Certificate common name (CN)

Use the syntax `certificate_cn:<term>` to search by certificate common name.

```
certificate_cn:"example.com"
```

Certificate issuer

Use the syntax `certificate_issuer:<term>` to search by certificate issuing authority.

```
certificate_issuer:"CN=Example CA,O=Example Corp,C=US"
```

Certificate subject key ID

Use the syntax `certificate_subject_key_id:<term>` to search by certificate subject key ID.

```
certificate_subject_key_id:"1234567890abcdef"
```

Certificate authority key ID

Use the syntax `certificate_authority_key_id:<term>` to search by certificate issuing authority key ID.

```
certificate_authority_key_id:"abcdef1234567890"
```

Suppression

The current suppressed or unsuppressed state of vulnerabilities can be searched using the syntax `suppressed:<term>`.

```
suppressed:t
```

will search for vulnerabilities that are suppressed. Inversely,

```
suppressed:f
```

will search for vulnerabilities that are not suppressed.

A special value of `any` can be used to display all records, regardless of suppression settings:

suppressed:any

will display both suppressed and unsuppressed vulnerabilities.

Additionally, the user who performed the suppression can be found using the following syntaxes:

suppressed_by:<username>

Certificate inventory

When viewing certificates, you can use the following keywords to search and filter.

General certificate fields

Certificate ID

The ID field is the unique identifier for a given certificate, written as a UUID. Use the syntax `id:<uuid>` to filter by ID field.

`id:21e5252d-a6a5-467e-83ed-683657412dff`

Certificate type

Use the syntax `type:<text>` to search for certificates by type.

`type:x509`

Name

Use the syntax `name:<text>` to search for certificates by name.

`name:example.com`

Validity

Use `valid_from:<time>` and `valid_until:<time>` to search for certificates by when they are valid.

`valid_from:>2025-01-01`

`valid_until:<2026-01-01`

Public key

Use the syntax `public_key:<text>` to search for certificates by public key.

`public_key:MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDJ899KGnqHjVuBekYqosp2l8zWbiyu2I62CzaqaouLtqn1nXaQLMdruh1NN9ShCPfCM2JA`

Public key algorithm

Use `pk_algorithm:<text>` or `public_key_algorithm:<text>` to search for certificates by public key algorithm.

`pk_algorithm:rsaEncryption`

Public key size

Use `pk_size:<number>` to search for certificates by public key size. You will usually want to specify the public key algorithm as well, as different algorithms have different key size ranges.

`pk_algorithm:rsaEncryption` and `pk_size:<2048`

RSA modulus

Use `rsa_modulus:<number>` to search for certificates using RSA encryption by modulus.

`rsa_modulus:24103124269210325885520760221975660748569505485024599426541169488887185854621`

RSA exponent

Use `rsa_exponent:<number>` to search for certificates using RSA encryption by exponent.

`rsa_exponent:65537`

DSA p

Use `dsa_p:<number>` to search for certificates using DSA encryption by parameter p.

`dsa_p:17715385462748585523709179925166512392868113546704423498760231332951635604832634171839704441561927811385837663709`

DSA q

Use `dsa_q:<number>` to search for certificates using DSA encryption by parameter q.

`dsa_q:23882561345098730106466767...`

DSA g

Use `dsa_g:<number>` to search for certificates using DSA encryption by parameter g.

`dsa_g:11506196528065909918480452874980083486869300271555091075530405860582486808935940564845326748921808017076639122038`

ECDSA curve

Use `ecdsa_curve:<text>` to search for certificates using ECDSA encryption by curve.

`ecdsa_curve:P-256`

Insecure public key

Use the syntax `public_key_insecure:true` to search for certificates with insecure public key algorithm and key size combinations.

Signature

Use the syntax `signature:<text>` to search for certificates by signature.

```
signature:Bden73ipj8B2xb10zy5n0vIytCktGrht5xL7ZfFlalIBQxbG05Iuf6Y1yICcEpYqsgSJS6JKCdw5dujmPmGRwBZfVhIbSRb0exFQ4BVp82WtL
```

Signature algorithm

Use `sig_algorithm:<text>` OR `signature_algorithm:<text>` to search for certificates by signature algorithm.

```
sig_algorithm:sha512WithRSAEncryption
```

Insecure signature

Use the syntax `signature_algorithm_insecure:true` to search for certificates with insecure signatures.

Self-signed

Use `self_signed:true` to search for self-signed certificates.

Certificate authority

Use `is_ca:true` to search for certificates that are certificate authorities (CAs).

Subject

Use `subject:<text>` to search for certificates by X.509 subject DN.

```
subject:"CN=Server Name/O=Company Name"
```

Common name

You can search for certificates by common name using `cn:<name>`. This is equivalent to searching the subject DN for just the CN field.

```
cn:"Server Name"
```

Subject alternative name

You can search the four sets of Subject Alternative Names (SANs) using the following keywords:

```
san_dns_name:example.com
san_ip_address:10.0.1.23
```

san_email_address:postmaster@example.com
san_uri:https://example.com

Issuer

Use issuer:<text> to search for certificates by X.509 issuer DN.

issuer:"CN=Certificate Authority Name"

Subject key ID

To search by X.509 subject key ID, use subject_key_id:<text>. Values are accepted with or without colons in.

subject_key_id:"12:90:EF:DD:E1:27:A4:47:3E:32:57:AF:44:75:92:8E:8C:C2:0A:C0"
subject_key_id:1290EFDDE127A4473E3257AF4475928E8CC20AC0

Authority key ID

To search by X.509 authority key ID, use authority_key_id:<text>. Values are accepted with or without colons in.

authority_key_id:"12:90:EF:DD:E1:27:A4:47:3E:32:57:AF:44:75:92:8E:8C:C2:0A:C0"
authority_key_id:1290EFDDE127A4473E3257AF4475928E8CC20AC0

OCSP server

Use ocsp_server:<text> to search for certificates by OCSP server.

ocsp_server:http://ocsp.test.com

CRL distribution point

Use crl_distribution_points:<text> OR crl_distribution_point:<text> to search for certificates by CRL distribution point.

crl_distribution_points:http://x1.c.lencr.org/

Issuing certificate URL

Use issuing_certificate_url:<text> to search for certificates by Issuing certificate URL.

issuing_certificate_url:http://x1.i.lencr.org/

Hash

You can find certificates based on their MD5, SHA1, SHA256 or BK hash values.

md5:<hash value>
sha1:<hash value>
sha256:<hash value>
bkhash:<hash value>

Serial number

Use `serial_number:<text>` to search for certificates by serial number.

```
serial_number:123456
```

Timestamps

Use the following syntaxes to search certificate inventory timestamp fields:

- `created_at:<term>`
- `updated_at:<term>`
- `last_seen:<term>`

The term supports the standard runZero time comparison syntax `[time comparison][time]`, for example:

```
last_seen:<1week
```

```
last_seen:<2months
```

```
last_seen:<1year
```

Associate services

Use `associate_services:<number>` or `service_count:<number>` to search for certificates by the number of associated services.

```
associated_services:>10
```

Hidden Certificates

Use `hidden:true` to search for certificates that have been hidden from the inventory.

Version

Use `version:<text>` to search for certificates by version, such as x509 version.

```
version:3
```

Tags

Use the syntax `tag:<term>` to search tags added to a certificate. The term can be the tag name, or the tag name followed by an equal sign and the tag value. Tag value matches must be exact.

```
tag:"group"
```

```
tag:"group=production"
```

Comments

Use the syntax `comment:<text>` to search comments on a certificate.

```
comment:"contractor laptop"
```

```
comment:"imaging server"
```

```
### Vulnerability name {#certificates-vulnerability-name}
```

Use the syntax ``vulnerability_name:<text>`` to search for certificates by associated vulnerability name.

```
```plaintext
vulnerability_name:"Expired Certificate On TLS Service"
```

## Vulnerability Count

Use the syntax `vulnerability_count:<number>` to search for certificates by the number of associated vulnerabilities. This field supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
vulnerability_count:>2
```

## Has vulnerability

Use the syntax `has_vulnerability:<boolean>` to search for certificates that have (or do not have) associated vulnerabilities.

```
has_vulnerability:true
```

## Wireless inventory

When viewing WiFi networks, you can use the keywords in this section to search and filter.

### SSID and ESSID

The SSID/ESSID field can be searched using the syntax `ssid:<text>`.

```
ssid:"Guest Network"
```

```
ssid:"Corporate"
```

### BSSID (MAC)

The BSSID field can be searched using the syntax `bssid:<text>` or `mac:<text>`.

```
bssid:"00:01:02:03:04:05"
```

```
mac:"00:01:%"
```

## Vendor

The vendor field can be searched using the syntax `mac_vendor:<text>`.

```
mac_vendor:"Google"
```

```
mac_vendor:"Netgear"
```

```
mac_vendor:"Cisco"
```

## Family

The family field can be searched using the syntax `family:<term>`.

```
family:"010304"
```

## Channels

The channels field can be searched using the syntax `channel:<term>`.

```
channel:"11"
```

## Type

The network type field can be searched using the syntax `type:<text>`.

```
type:"infrastructure"
```

## Interface

The network interface field can be searched using the syntax `interface:<text>`.

```
interface:"wlan0"
```

## Encryption

The encryption field can be searched using the syntax `encryption:<term>`.

```
encryption:"aes"
```

```
encryption:"none"
```

## Authentication

The authentication field can be searched using the syntax `authentication:<term>`.

```
authentication:"wpa2-psk"
```

```
authentication:"open"
```

## Timestamps

The timestamp fields (`first_seen`, `last_seen`, `created_at`) timestamps can be searched using the syntax `first_seen:<term>`, `last_seen:<term>` and `created_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
first_seen:<30seconds
```

```
first_seen:>2019-08-01
```

```
last_seen:<1week
```

```
last_seen:<2months
```

```
created_at:>2weeks
```

```
created_at:<30minutes
```

## Signal

The signal field can be searched using the syntax `signal:<number>` or `sig:<number>`. The term can include the operators `>`, `=`, `<=`, and `=`. The default operator is `=`.

```
signal:">75"
```

```
signal:"<=25"
```

```
signal:99
```

## Organization name or ID

Use the syntax `organization:<term>` to filter by organization name or ID.

```
organization:runZero
```

```
organization:"Temporary Project"
```

```
organization:f1c3ef6d-cb41-4d55-8887-6ed3cfb3d42d
```

## Site name or ID

The site name or ID can be used as a filter with the syntax `site:<term>`

```
site:Primary
```

```
site:"Branch Office"
```

```
site:ad67d649-041b-439d-af59-f200053a8899
```

## Explorer name or ID

The Explorer name or ID can be used as a filter with the syntax `explorer:<term>`



```
explorer:DESKTOP-AB451F
```

```
explorer:8b927a8e-d405-40e9-aa47-d6afc9bff237
```

## Wireless ID

The ID field is the unique identifier for a given wireless network, written as a UUID. This field is searched using the syntax `id:<uuid>`.

```
id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

## Last task ID

The Last Task ID field defines which task most recently reported the wireless network and is written as a UUID. This field is searched using the syntax `task:<uuid>`.

```
task:39ab0e71-3cf1-4176-b6b0-4ed495288229
```

## Wireless attributes

All wireless attributes can be searched using the syntax `<attribute>:<term>`.

```
radio_type:"802.11n"
```

## Users inventory

When viewing the Users inventory, you can use the following keywords to search and filter users.

### Source

The source reporting the users can be searched or filtered by name using the syntax `source:<name>`.

```
source:ldap
```

### Name fields

There are multiple name fields found in the user attributes that can be searched or filtered using the same syntax. Use the syntax `<name_field>:<text>` to search a field for matches.

The following name fields can be searched this way:

- name
- display\_name
- first\_name
- last\_name

```
name:j
```

```
display_name:"john doe"
```

```
first_name:john
```

```
last_name:doe
```

## Description

Use the syntax `description:<text>` to search the results in this field.

```
description:shared
```

## Groups

The `group_id` field is the unique identifier for a given group, written as a UUID. To search for users that are part of a group based on the group's ID, use the syntax `group_id:<uuid>`.

```
group_id:f8a26321-cbce-4fb5-a9ca-ffa809489dd5
```

## Email

Use the syntax `email:<address>` to search for users by email address.

```
email:john@example.com
```

## Phone

Use the syntax `phone:<phone number>` to search for users by phone number.

```
phone:888-555-1234
```

## Title

Use the syntax `title:<text>` to search for users by title.

```
title:CISO
```

## Location

Use the syntax `location:<text>` to search for users by location.

```
location:TX
```

## Last logon

Use the syntax `last_logon_at:<date>` to search for users by last logon time. This field is compatible with the [date comparison operators](#).

```
last_logon_at:<1year
```

## First seen

Use the syntax `first_seen_at:<date>` to search for users by when they were first seen in the organization. This field is compatible with the [date comparison operators](#).

```
first_seen_at:<1day
```

## Last seen

Use the syntax `last_seen_at:<date>` to search for users by when they were last seen in the organization. This field is compatible with the [date comparison operators](#).

```
last_seen_at:>2years
```

## Site name or ID

Use the syntax `site:<term>` to filter by site name or ID.

```
site:Primary
```

```
site:"Branch Office"
```

```
site:ad67d649-041b-439d-af59-f200053a8899
```

## Organization name or ID

Use the syntax `organization:<term>` to filter by organization name or ID.

```
organization:runZero
```

```
organization:"Temporary Project"
```

```
organization:f1c3ef6d-cb41-4d55-8887-6ed3cfb3d42d
```

## Groups inventory

When viewing the Groups inventory, you can use the following keywords to search and filter groups.

### Source

The source reporting the groups can be searched or filtered by name using the syntax `source:<name>`.

```
source:ldap
```

## Name fields

There are two name fields found in the group attributes that can be searched or filtered using the same syntax. Use the syntax `<name_field>:<text>` to search a field for matches.

The following name fields can be searched this way:

- name
- display\_name

name:admin

display\_name:"Domain Administrators"

## Description

Use the syntax `description:<text>` to search the results in this field.

description:admin

## Users

The `user_id` field is the unique identifier for a given user, written as a UUID. To search for groups that have a specific member based on the user's ID, use the syntax `user_id:<uuid>`.

user\_id:f8a26321-cbce-4fb5-a9ca-ffa809489dd5

## Email

Use the syntax `email:<address>` to search for users by email address.

email:john@example.com

## First seen

Use the syntax `first_seen_at:<date>` to search for users by when they were first seen in the organization. This field is compatible with the [date comparison operators](#).

first\_seen\_at:<1day

## Last seen

Use the syntax `last_seen_at:<date>` to search for users by when they were last seen in the organization. This field is compatible with the [date comparison operators](#).

last\_seen\_at:>2years

## Site name or ID

Use the syntax `site:<term>` to filter by site name or ID.

site:Primary

site:"Branch Office"

site:ad67d649-041b-439d-af59-f200053a8899

## Organization name or ID

Use the syntax `organization:<term>` to filter by organization name or ID.

organization:runZero

organization:"Temporary Project"

organization:f1c3ef6d-cb41-4d55-8887-6ed3cfb3d42d

## Interface keywords

The data across your runZero account can be queried and filtered using the [search syntax](#) in conjunction with the available interface-specific keywords. Keywords and example values are documented for the following sections of the runZero Console:

- [Scan templates](#)
- [Findings](#)
- [Tasks](#)
- [Analysis reports](#)
- [Explorers](#)
- [runZero users and groups](#)
- [Sites and organizations](#)
- [Credentials](#)
- [Queries](#)
- [Events](#)

## Scan templates

When viewing scan templates, you can use the keywords in this section to search and filter.

### ID

The ID field is the unique identifier for a given template, written as a UUID. Use the syntax `id:<uuid>` to filter by ID field.

id:cdb084f9-4811-445c-8ea1-3ea9cf88d536

### Name

Use the syntax `name:<text>` to search by scan template name.

name:WiFi

```
name:"Data Center"
```

## Timestamps

Use the following syntaxes to search the scan template timestamp fields (`created_at`, `updated_at`):

- `created_at:<term>`
- `updated_at:<term>`

The term supports the standard runZero [time comparison syntax][time].

```
created_at:>2weeks
```

```
created_at:<30minutes
```

```
updated_at:>1year
```

```
updated_at:<12hours
```

## Created by

The email address for the user that created the template can be searched using the syntax `created_by_email:<term>`.

```
created_by_email:user@example.com
```

## Findings

When viewing findings, you can use the keywords in this section to search and filter.

### Finding code

The finding code field is the unique identifier for a given finding. Use the syntax `finding_code:<uuid>` to filter by the code field.

```
finding_code:rz-finding-internet-exposed-database
```

### Name

Use the syntax `name:<text>` to search by finding name.

```
name:"Internet Exposed Database"
```

### Description

The Description field can be searched using the syntax `description:<text>`.

```
description:"indicated databases"
```

## Solution

The Solution field can be searched using the syntax `solution:<text>`.

```
solution:"indicated databases"
```

## Risk

The Risk / Risk Rank value can be searched using either numeric or keyword values. Risk rank is an integer from zero through four, where 0 is Info level risk and 4 indicates Critical risk.

```
risk:"Critical"
```

```
risk_rank:>2
```

## Category

The finding Category field can be searched using the syntax `category:<text>`.

```
category:"End-of-Life"
```

## Vulnerability instance count

The Instance field can be searched using the syntax `vulnerability_count:<text>`.

```
vulnerability_count:>0
```

## Organization and site names

The names of organizations or sites affected can be searched using the following search terms:

- `organization_name:<text>`
- `site_name:<text>`

The IDs are unique and are written as UUIDs.

```
organization_id:0eacf412-6e69-11ec-88b9-f875a414a63a
```

## Organization and site IDs

The IDs of organizations or sites affected can be searched using the following search terms:

- `organization_id:<uuid>`
- `site_id:<uuid>`

The IDs are unique and are written as UUIDs.

```
organization_id:0eacf412-6e69-11ec-88b9-f875a414a63a
```

## Timestamps

Use the following syntaxes to search the finding timestamp fields (`last_detected_at`, `created_at`, `updated_at`, `suppressed_at`):

- `created_at:<term>`
- `updated_at:<term>`
- `last_detected_at:<term>`
- `suppressed_at:<term>`

The term supports the standard runZero `[time comparison syntax][time]`.

`last_detected_at:>2weeks`

`created_at:<30minutes`

`updated_at:>1year`

`updated_at:<12hours`

`suppressed_at:<1day`

## Suppression

The current suppressed or unsuppressed state of findings can be searched using the syntax `suppressed:<term>`.

`suppressed:t`

will search for findings that are suppressed. Inversely,

`suppressed:f`

will search for findings that are not suppressed.

A special value of `any` can be used to display all records, regardless of suppression settings:

`suppressed:any`

will display both suppressed and unsuppressed findings.

Additionally, the user who performed the suppression can be found using the following syntaxes:

`suppressed_by:<username>`

## Tasks

When viewing all tasks, you can use the keywords in this section to search and filter them.



## Name

The Name field can be searched using the syntax `name:<text>`.

```
name:"test scan"
```

## Description

The Description field can be searched using the syntax `description:<text>`

```
description:"full scan"
```

## Created by

The Created By field can be searched using the syntax `created_by:<term>`.

```
created_by:"admin"
```

## Type

The task type can be searched using `type:<text>`.

```
type:scan
```

## Status

The task status can be searched using `status:<text>`.

```
status:error
```

## Error

The task error message can be searched using `error:<text>`.

```
error:"no disk space"
```

## Recurrence frequency

The frequency tasks recur at (the "Freq" column) can be searched using `recur_frequency:<text>` or `freq:<text>`. The term `recurring:<boolean>` or `recur:<boolean>` can be used to search based on whether tasks recur at all.

```
recur_frequency:hourly
```

```
freq:daily
```

```
freq:continuous
```

```
recur:true
```

To search for tasks with a frequency of Nth Weekday of Month, you can use (for example) `freq:nth_weekday,2` `freq:monday` to find tasks which repeat on the second monday of each month.

## Timestamps (created at, updated at)

The timestamp fields, `created_at` and `updated_at`, can be searched using the syntax `created_at:<term>` and `updated_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

`created_at:>2weeks`

`created_at:<30minutes`

`updated_at:>1month`

`updated_at:2hours`

## Next/last run time

You can search by next recurrence and last recurrence using the terms `recur_last:<term>` and `recur_next:<term>`. The term supports the standard runZero [time comparison syntax][time].

`recur_last:<2hours`

`recur_next:>1day`

## Start time

You can search by start time using the syntax `start_time:<term>`. The term supports the standard runZero [time comparison syntax][time].

`start_time:<2hour`

## Grace period

The [grace period](#) can be searched using the syntax `grace_period:<term>` or just `grace:<term>`. The term supports the standard runZero [time comparison syntax][time].

`grace:<2hour`

## Site name or ID

Use the syntax `site:<term>` to filter by site name or ID.

`site:Primary`

`site:"Branch Office"`

`site:ad67d649-041b-439d-af59-f200053a8899`

## Template ID

Use the syntax `template_id:<term>` to filter by scan template ID.

`template_id:de657459-041b-439d-af59-ff1f153a7722`

## Source

The data source for tasks can be searched using the term `source:<text>` OR `source_id:<number>`.

`source:censys`

Sources are:

| ID | Name                  | Description                   |
|----|-----------------------|-------------------------------|
| 1  | runzero               | runZero                       |
| 2  | miradore              | Miradore                      |
| 3  | aws                   | Amazon Web Services           |
| 4  | crowdstrike           | CrowdStrike                   |
| 5  | azure                 | Microsoft Azure               |
| 6  | censys                | Censys                        |
| 7  | vmware                | VMWare                        |
| 8  | gcp                   | Google Cloud Platform         |
| 9  | sentinelone           | SentinelOne                   |
| 10 | tenable               | Tenable.io & Nessus           |
| 12 | rapid7                | Rapid7 Nexpose & InsightVM    |
| 14 | qualys                | Qualys VMDR                   |
| 15 | shodan                | Shodan                        |
| 16 | azuread               | Azure AD                      |
| 17 | ldap                  | Active Directory (LDAP)       |
| 18 | ms365defender         | Microsoft 365 Defender        |
| 19 | intune                | Microsoft Intune              |
| 20 | googleworkspace       | Google Workspace              |
| 21 | sample                | runZero traffic sampling      |
| 22 | tenablesecuritycenter | Tenable Security Center       |
| 23 | packet                | runZero packet capture import |

## Credential ID

You can search for tasks that use a specific set of credentials using `credential_id:<id>`.

```
credential_id:d7931a68-6e56-11ec-ad72-f875a414a63a
```

## Parameters

Tasks can be searched for task parameters using `params:<text>`. This can be useful for searching for scan tasks that had specific probes enabled.

```
params:bacnet
```

## Asset counts

Completed tasks can be searched by the asset counts found in their results. The available search terms are:

- New assets: `assets_new:<number>`
- Assets back online: `assets_back_online:<number>`
- Assets marked offline: `assets_marked_offline:<number>`
- Assets changed: `assets_changed:<number>`
- Assets unchanged: `assets_unchanged:<number>`
- Assets ignored: `assets_ignored:<number>`
- Assets updated: `assets_updated:<number>`

These terms support numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).

```
assets_new:>0
```

```
assets_unchanged:>=1
```

## Analysis reports

When viewing generated analysis reports, you can use the keywords in this section to search and filter.

### Name

The Name field can be searched using the syntax `name:<text>`.

```
name:"main"
```

### Description

The Description field can be searched using the syntax `description:<text>`

```
description:"compare secondary"
```

## Type

The report type can be searched using the syntax `type:<text>`

```
type:outliers
```

## Report ID

The ID field is the unique identifier for a given analysis report, written as a UUID. This field is searched using the syntax `id:<uuid>`.

```
id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

## Timestamps

The timestamp when a report was generated can be searched using the syntax `created_at:.`

The term supports the standard runZero [time comparison syntax][time].

```
created_at:>2019-08-01
```

```
created_at:<1week
```

## Created by

The Created By field can be searched using the syntax `created_by:<term>`.

```
created_by:jsmith
```

## Explorers

When viewing deployed Explorers, you can use the keywords in this section to search and filter.

### Name

The Name field can be searched using the syntax `name:<text>`.

```
name:"main"
```

### Site

The site can be searched using the syntax `site:<text>`.

```
site:Primary
```

## Up

Whether the Explorer is up can be searched using the syntax `up:<boolean>`.

`up:true`

## Address

The IP address(es) the Explorer is deployed on can be searched using the syntax `address:<IP address>`.

`address:10.0.1.200`

## Version

The software version of Explorers can be searched using `version:<text>`.

`version:2.9.7`

## Npcap version

The version of the npcap library for Windows Explorers can be searched using `npcap_version:<text>`.

`npcap_version:1.60`

## Architecture

The machine architecture Explorers are deployed on can be searched using `architecture:<text>`.

`architecture:amd64`

## OS

The operating system Explorers are deployed on can be searched using `os:<text>`. Note that macOS is recorded as `darwin`, the underlying Unix core of macOS.

`os:windows`

`os:darwin`

## Capability

The capabilities of the Explorers can be searched using the syntax `capability:<keyword>`. Two keywords are supported:

- `screenshot` for Explorers which can screenshot web pages
- `ec2` for Explorers which can describe AWS EC2 instances

Example:

capability:screenshot

## Tags

Use the syntax `tag:<term>` to search tags added to an Explorer. The term can be the tag name, or the tag name followed by an equal sign and the tag value. Tag value matches must be exact.

`tag:"admin"`

`tag:"group=cloud"`

## runZero users and groups

### User search keywords

When viewing users, you can use the keywords in this section to search and filter.

#### Email

Use the syntax `email:<address>` to search for someone by email address.

`email:john@example.com`

#### Name

Use the syntax `name:<text>` to search for someone by name.

`name:john`

`name:"John Smith"`

#### Superuser

To search for people based on whether they have superuser access, use the term `superuser:<boolean>`.

`superuser:true`

`superuser:f`

#### Access

Use the syntax `access:<term>` to search for users with a specific access level. Possible access levels are `admin`, `user`, `annotator`, `viewer`, `billing` and `none`.

`access:admin`

## Status

To search for users by invitation status, use the term `status:<text>`. Possible status values are `activated`, `pending` and `expired`.

```
status:pending
```

## SSO

To search for people based on whether they can only sign in via SSO, use the term `sso:<boolean>`.

```
sso:true
```

## MFA

To search for people based on whether they have enrolled an MFA token, use the term `mfa:<boolean>`.

```
mfa:f
```

## Group ID

The `group_id` field is the unique identifier for a given group, written as a UUID. To search for users that are part of a group based on the group's ID, use the syntax `group_id:<uuid>`.

```
group_id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

## Group name

To search for users that are part of a group based on the group's name, use the syntax `group_name:<text>`.

```
group_name:administrators
```

```
group_name:"Temp annotators"
```

## User groups

To search for users based on whether they are part of a group or not, use the term `has_group:<boolean>`. Use boolean value `t` or `true` to show all users who are members of a group.

```
has_group:t
```

Conversely, use boolean value `f` or `false` to show users who are not members of a group.

```
has_group:f
```

## Group count

Use the syntax `group_count:<number>` to search the group membership count. This search term supports numerical comparison operators (`>`, `>=`, `<`, `<=`, `=`).



```
group_count:>1
```

```
group_count:=0
```

## Group search keywords

When viewing your groups, you can use the keywords in this section to search and filter.

### ID

The ID field is the unique identifier for a given group, written as a UUID. Use the syntax `id:<uuid>` to filter by ID field.

```
id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

### Name

Use the syntax `name:<text>` to search by group name.

```
name:administrators
```

```
name:"Temp annotators"
```

### Access

Use the syntax `access:<term>` to search for groups with a specific access level. Possible access levels are `admin`, `user`, `annotator`, `viewer`, `billing` and `none`.

```
access:admin
```

### Timestamps (created at, updated at)

Filter groups by their timestamp fields, `created_at` and `updated_at`, using the syntax `created_at:<term>` and `updated_at:<term>`. The terms support the standard runZero [time comparison syntax][time].

```
created_at:<30days
```

```
updated_at:<1week
```

### Expiration

Filter groups by their expiration timestamp, `expires_at`, using the syntax `expires_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
expires_at:<30days
```

```
expires_at:>8/1/2019
```

The `expired` property describes whether or not a group has expired. Search this property using the `expired:<boolean>` syntax.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

expired:true

expired:0

Use the syntax `has_expiration:<term>` to find any assets with an expiration date.

The term is a boolean value:

- true, t, 1, and yes represent *true*
- false, f, 0, and no represent *false*

has\_expiration:true

has\_expiration:0

## Email

The `created_by_email` property holds the email address for the user that created the group. It can be searched using the syntax `created_by_email:<term>`.

created\_by\_email:user@runzero.com

## Group mapping search keywords

When viewing your SSO group mappings, you can use the keywords in this section to search and filter.

### ID

The ID field is the unique identifier for a given group mapping, written as a UUID. Use the syntax `id:<uuid>` to filter by ID field.

id:cdb084f9-4811-445c-8ea1-3ea9cf88d536

### SSO attribute

The `sso_attribute` is the name of the attribute field to check for matching values. Use the syntax `sso_attribute:<text>` to search by `sso_attribute`.

sso\_attribute:department

### SSO value

The `sso_value` is the value or comma-separated list of values to match. Use the syntax `sso_value:<text>` to search by `sso_value`.

sso\_value:security

```
sso_value:"admins, administrators"
```

## Group ID

The `group_id` field is the unique identifier for a given group, written as a UUID. To search for group mappings related to a group based on the group's ID, use the syntax `group_id:<uuid>`.

```
group_id:cdb084f9-4811-445c-8ea1-3ea9cf88d536
```

## Group name

To search for group mappings related to a group based on the group's name, use the syntax `group_name:<text>`.

```
group_name:administrators
```

```
group_name:"Temp annotators"
```

## Timestamps (created at, updated at)

Filter group mappings by their timestamp fields, `created_at` and `updated_at`, using the syntax `created_at:<term>` and `updated_at:<term>`. The terms support the standard runZero [time comparison syntax][time].

```
created_at:<30days
```

```
updated_at:<1week
```

## Email

The `created_by_email` property holds the email address for the user that created the group. It can be searched using the syntax `created_by_email:<term>`.

```
created_by_email:user@runzero.com
```

# Sites and organizations

## Site search keywords

When viewing sites, you can use the keywords in this section to search and filter.

### Name

The Name field can be searched using the syntax `name:<text>`.

```
name:"Primary"
```

### Description

The Description field can be searched using the syntax `description:<text>`.

```
description:"wireless"
```

```
description:"vlan 50"
```

## Scope

The Scope field can be searched using the syntax `scope:<term>` .

```
scope:"10.10.10."
```

## Excludes

The Excludes field can be searched using the syntax `excludes:<term>` .

```
excludes:"192.168.0."
```

## Timestamps (created at, updated at)

The timestamp fields (`created_at`, `updated_at`) timestamps can be searched using the syntax `created_at:<term>` and `updated_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

```
created_at:>2weeks
```

```
created_at:<30minutes
```

```
updated_at:>1month
```

```
updated_at:2hours
```

## Organization search keywords

### Name

The Name field can be searched using the syntax `name:<text>`.

```
name:"main"
```

### Description

The Description field can be searched using the syntax `description:<text>`

```
description:"branch office"
```

```
description:"pci"
```

## Timestamps (created at, updated at)

The timestamp fields (`created_at`, `updated_at`) timestamps can be searched using the syntax `created_at:<term>` and `updated_at:<term>`. The term supports the standard runZero [time comparison syntax][time].

created\_at:>2weeks

created\_at:<30minutes

updated\_at:>1month

updated\_at:2hours

## Credentials

When viewing saved credentials, you can use the keywords in this section to search and filter.

### Credential fields

#### Credential ID

The ID field is the unique identifier for a given credential, written as a UUID. This field is searched using the syntax `id:<uuid>`.

`id:cdb084f9-4811-445c-8ea1-3ea9cf88d536`

#### Credential name

The credential name can be searched using the syntax `name:<text>`.

`name:"AWS read-only account"`

`name:"Miradore API key"`

#### Credential type

The credential type can be searched using the syntax `name:<text>`.

`type:aws_access_secret`

`type:miradore_api_key_v1`

#### Credential global property

The global property describes the level of access for all organizations. If a credential is global, all organizations have access to it. The global property can be searched using the syntax `global:<boolean>`.

The term is a boolean value:

- `true`, `t`, `1`, and `yes` represent *true*
- `false`, `f`, `0`, and `no` represent *false*

`global:true`

`global:0`

## Credential timestamps

Credential timestamp fields (`created_at` and `last_used_at`) can be searched using the syntax:

- `created_at:<term>`
- `last_used_at:<term>`

The term supports the standard runZero `[time comparison syntax][time]`.

`created_at:<3days`

`created_at:>2019-08-01`

`created_at:>8/1/2019`

`created_at:<1week`

`created_at:<2months`

`last_used_at:<1year`

`last_used_at:>2weeks`

`last_used_at:<30minutes`

`last_used_at:>1year`

`last_used_at:<12hours`

`last_used_at:0`

## Credential created by

The `created_by_email` holds the email address for the user that created the credential. It can be searched using the syntax `created_by_email:<term>`.

`created_by_email:user@example.com`

## Queries

When viewing saved queries, you can use the keywords in this section to search and filter.

### Name

The Name field can be searched using the syntax `name:<text>`.

`name:"smb2"`

### Description

The Description field can be searched using the syntax `description:<text>`.

```
description:"smb version 1"
```

```
description:"wep"
```

## Type

The Type field can be searched using the syntax `type:<term>` .

```
type:"services"
```

## Category

The Category field can be searched using the syntax `category:<term>`.

```
category:"security"
```

```
category:"audit"
```

## Severity

The Severity field can be searched using the syntax `severity:<term>`.

```
severity:"info"
```

```
severity:"critical"
```

## Created by

The Created By field can be searched using the syntax `created_by:<term>`.

```
created_by:"runzero"
```

## Timestamps (created at, updated at)

The timestamp fields, `created_at` and `updated_at`, can be searched using the syntax `created_at:<term>` and `updated_at:<term>`. The term supports the standard runZero [time comparison syntax] [time].

```
created_at:>2weeks
```

```
created_at:<30minutes
```

```
updated_at:>1month
```

```
updated_at:2hours
```

## Events

When viewing system events under alerts, you can use the keywords in this section to search and filter.

Note that event records are retained for one year.

## Action

Use the syntax `action:<text>` to search by the action which caused the event.

```
action:agent-reconnected
```

## Created timestamp

The timestamp fields `created_at` can be searched using the syntax `created_at:<term>`. The term supports the standard runZero [time comparison syntax](#).

```
created_at:>2weeks
```

```
created_at:<30minutes
```

```
updated_at:>1month
```

```
updated_at:2hours
```

## Details

The details in the event record can be searched using the syntax `details:<text>`. This can be useful for searching for IP addresses.

```
details:192.168.0.1
```

## Source and target name

The source (`src`) column can be searched using the syntax `src:<text>` or `source:<text>`. The target (`tgt`) column can be searched using `tgt:<text>` or `target:<text>`.

```
src:crowdstrike
```

```
target:primary
```

## Source and target type

The source type (shown at the start of the `src` column) can be searched using the syntax `src_type:<text>` or `source_type:<text>`.

Similarly, the target type can be searched using `tgt_type:<text>` or `target_type:<text>`.

```
src_type:task
```

```
target_type:site
```



## Organization, site, source and target IDs

The IDs of organizations, sites, sources and targets mentioned in event details can be searched using the following search terms:

- `organization_id:<uuid>`
- `site_id:<uuid>`
- `source_id:<uuid>` OR `src_id:<uuid>`
- `target_id:<uuid>` OR `tgt_id:<uuid>`

The IDs are unique and are written as UUIDs.

`organization_id:0eacf412-6e69-11ec-88b9-f875a414a63a`

## Automating queries

runZero's query language allows you to search and filter your asset inventory, based on asset fields and values. runZero includes a query library of prebuilt searches which can be browsed from the Queries page. You can apply these queries after a scan to investigate discovery findings.

In addition to a flexible query language, the same search syntax can be used to track and monitor events across your network, based on any combination of fields. You can save your custom queries to reuse over and over again. Review the [query syntax](#) documentation for a refresher on how to build a query.

If there are certain queries you always want to run after a scan, you can turn on the **Auto** option. After the query runs, you will be able to view its results in the [Queries table](#).

From the Queries table, any filter that has checkmark in the *Auto* column means that it is active. The *Matches* column is next to it. You can click on the numbered result in the *Matches* column to view the results of the query. If there were no matches, the filter will show 0.

## Turn on automatic search queries

1. Select Queries from the left navigator.
2. Open or create the query you want to turn on.
3. Turn on the **Automatically track query results on the dashboard** option.
4. Save the query.

# Data export

## Exporting asset data

The [inventory](#) view provides a few ways to export asset data. The `Export` menu offers `Export` All options in both CSV and JSON format, and when a [search query](#) has been provided, options to export just the search results as both CSV and JSON.

The CSV format can be opened with tools like Microsoft Excel and easily imported into other applications but does not contain the full details of certain fields, such as Services. The JSON format contains a complete export but may take additional processing to use with other tools.

If you have a specific export/import scenario in mind, please contact [support](#) and let us know. The [Export API](#) leverages both the CSV and JSON formats and supports arbitrary search queries in the [same syntax](#) as the inventory.

## Exporting HP iLO data

HP Integrated Lights-Out (iLO) provides remote management, configuration, and monitoring capabilities for HP servers. These capabilities centralize operations for your server environments and streamline tasks like rebooting servers, booting into single user mode, and bypassing authentication.

Being able to identify and find the serial number for HP iLO devices is useful for tracking warranties for support and contract management. If you have runZero Platform, you can export the runZero data as a CSV to feed into warranty tracking tools.

## How to export HP iLO CSV data

In your runZero Console, go to your [inventory](#). From the **Export** menu, choose the **HP iLO CSV format**. This method downloads all HP iLO data from the runZero inventory to a CSV file.

If you want to refine the results in your exported data, you can filter the inventory first. For example, if you only want to export iLOs that have the ProLiant DL360p Gen8 hardware, you can query:

```
alive:t AND ilo.hardware:"=ProLiant DL360p Gen8"
```

And then, from the Export menu, choose the **HP iLO CSV format** from the Export Search Results submenu.

## HP iLO CSV export data

When you export HP iLO data, the CSV file will contain the following fields:

- IP address
- MAC address
- Name
- Serial number
- Product ID
- Model
- Version
- Health status
- IRS
- iLO product name
- Serial number
- Boot block
- HW revision
- FW revision
- PWRM
- Auth local
- Auth Kerberos
- Auth LDAP
- Auth license

- Security message
- Alive
- ID
- Site ID
- Last seen

There is a lot of data in the export that you may not need for your warranty tracker. Fields, such as serial numbers, physical hardware information, health status, and firmware version may be the most useful to import into your tracker.

# Advanced reports

## Network Map

The Network Map is an interactive Layer-2 / Layer-3 topology of every asset in the current scope. Where the [World Map](#) shows assets by physical location, the Network Map shows logical relationships: which subnets contain which assets, which gateways and switches connect them, and which assets pivot between subnets.

The map ships in two modes — a 2D Canvas view and a 3D rotating globe — that share the same data, search syntax, and trace engine.



## Accessing the map

The Network Map can be opened from several places in the console:

- **Inventory grid** — Asset rows show a small map icon when the asset has L2/L3 data. Clicking the icon opens the map filtered to that asset.
- **Inventory search** — The “View on map” action above the inventory grid opens the map with the current inventory search applied as the map filter.
- **Asset details** — The **Manage** menu on an asset’s detail page offers **View on network map** (or **Back to network map** if you arrived from the map). The same menu also offers

**View internet attack path**, which pre-loads a trace from the public internet to the selected asset.

- **Reports** — Open it directly at `/reports/network/map` from the Reports menu, or <https://console.runzero.com/reports/network/map>.
- **Queries** — Saved query rows in the queries grid include a “View on network map” action that prefills the map with the saved query.

When you open the map from another page, runZero records the previous URL as a return target. A **Back** button appears in the top-left and returns you to that exact view, preserving your filters and pagination.

## 2D and 3D modes

- **2D Network Map** (`/reports/network/map`) — Canvas-rendered topology with the full feature set: hop-filter trace, choke-point analysis, multi-tab Network Intel card, search results panel, and rich exports.
- **3D Network Map** (`/reports/network/map/3d`) — A rotating spherical layout that uses the same data, search syntax, and Source / Target trace engine. Useful for executive views and demos.

To switch modes, use the **2D / 3D** button in the toolbar. Filter, search, and trace state are preserved across the switch.

## Subnet layout

The map groups assets into rectangular subnet clusters. Each subnet box contains the assets that have at least one address in that CIDR range, along with any unaddressed (MAC-only) endpoints linked into the subnet by L2 evidence.

## Subnet mask

The toolbar mask selector controls how IPv4 addresses are aggregated into subnets:

- **/24** (default) — each /24 is a separate cluster.
- **/16** — collapses related /24s into the same /16 parent.
- **/15** through **/8** — broader rollups, useful for large flat networks.

A separate selector controls the IPv6 mask, which defaults to /56 and ranges from /32 to /64 in steps of 4.

When runZero scans a subnet that overlaps with a configured site subnet, the configured CIDR is used directly. Otherwise the chosen mask is applied to the asset's address to derive a synthetic subnet boundary.

## Multi-homed assets

A single asset that has IP addresses in more than one subnet appears **once in each subnet it belongs to**. The renderer draws an L3 link between each copy so the map clearly shows the

asset bridges those segments. The same asset ID is used for every copy — clicking any copy opens the same asset detail panel — but each copy has its own position so trace paths through the asset are visually unambiguous.

Multi-homed assets are counted as **Pivots** in the [Network Intel card](#) (`multi_homed:t`).

## Public Internet node

A virtual **Public Internet** node is pinned at the top of the layout. Public IP subnets observed during scanning attach to this node, and the trace engine treats it as a valid source / target for ingress and egress path analysis (see [Tracing paths](#)).

By default the map filters out widely-reused public IPs (for example public DNS resolvers like 1.1.1.1 or 8.8.8.8) that show up on many internal assets and would otherwise produce misleading internet edges. The threshold is configurable in the Trace panel's expanded view.

## Network Intel card

The **Network Intel** card on the left rail is a clickable stats grid. Each cell shows a derived count for the current view; clicking a cell seeds the map's search bar with the corresponding query and re-runs the filter.

| Cell      | What it counts                                                     | Click query                |
|-----------|--------------------------------------------------------------------|----------------------------|
| Assets    | Total assets in view                                               | (display only)             |
| Subnets   | Distinct subnets observed                                          | (display only)             |
| High Risk | Assets with high or critical risk                                  | risk:high OR risk:critical |
| EOL       | Assets running an end-of-life OS                                   | eol:t                      |
| Pivots    | Multi-homed assets bridging two or more subnets                    | multi_homed:t              |
| Public    | Assets with at least one public IP                                 | has_public:t               |
| Misplaced | Assets whose category does not match their subnet's classification | misplaced:t                |
| Outliers  | Assets that are distinct from their neighbors' norm                | outlier_rank:>1            |
| Gateways  | Protocol gateways bridging IP and non-IP networks                  | is:gateway                 |
| Switches  | Managed switches that share neighbor information                   | is:switch                  |

The card also includes a device-type legend; each legend cell is clickable and runs an equivalent `type:` or `category:` query.



## OT Risk tab

When the current view contains OT-scoped assets, an **OT Risk** tab appears alongside **Network**. It is a curated grid of one-click investigations driven by the OT risk register. Each card runs a query against the visible map and shows the result count; click a card to filter the map and the inventory below.

| Card           | What it surfaces                                                                                       |
|----------------|--------------------------------------------------------------------------------------------------------|
| Public OT      | OT-scoped assets reachable from the public internet — <code>ot_scope:t AND has_public:t</code>         |
| ICS on Public  | Native ICS protocols (Modbus, EtherNet/IP, DNP3, BACnet, S7, OPC UA, IEC-104, KNX, ...) on a public IP |
| Path Starters  | Internet-facing management / lateral-movement services (RDP, SMB, SSH, WinRM, VNC, Telnet, VPN, IPMI)  |
| OT Pivots      | OT-scoped assets that are multi-homed across subnets — <code>ot_scope:t AND multi_homed:t</code>       |
| Mgmt Plane     | OT-scoped management-plane services (SSH, RDP, SMB, WinRM, VNC, Telnet, IPMI, BMC, KVM)                |
| Proto Gateways | Protocol gateways bridging IP topology to BACnet, EtherNet/IP, Modbus, KNXnet/IP, C37.118, ...         |
| SIS Candidates | Hardware suggesting Safety Instrumented Systems (Triconex / HIMA / ProSafe)                            |
| OT EOL         | OT-scoped assets on an end-of-life OS or firmware                                                      |
| OT Vulns       | OT-scoped assets with high or critical risk                                                            |
| Misplaced OT   | OT in a non-OT subnet, or non-OT in an OT subnet (segmentation gaps)                                   |
| Operator UIs   | OT-scoped HMIs, Panel PCs, engineering workstations, operator desktops, thin clients                   |
| Weak Mgmt      | OT-scoped Telnet, FTP, VNC, TFTP, SNMPv1/v2c — cleartext or unauthenticated management                 |

OT-scoping is computed from the asset's `category`, the subnet classification, and a small device-type veto list (a Schneider UPS with a stale `category:OT` tag is not OT-scoped).

## Search

The search bar accepts a domain-specific query language with keyword filters, free text, boolean operators (`AND`, `OR`, `NOT`), parentheses, and `*` wildcards. Bare text matches across all fields.

```

192.168.1 # free-text match
name:web-server # keyword
type:server AND os:Linux # boolean
NOT net:10.0.0.0/8 # negation
(name:PC AND type:server) OR os:Windows
multi_homed:t AND has_public:t
type:Server AND geo:30.27,-97.74,25km

```

## Identity, address, and platform

| Keyword      | Aliases                          | Matches                                                  |
|--------------|----------------------------------|----------------------------------------------------------|
| id           |                                  | Node or asset ID                                         |
| address      | addr, addresses, ip, host        | IP address(es)                                           |
| name         | names, hostname, hostnames       | Hostname(s)                                              |
| net          | cidr                             | Subnet / network CIDR                                    |
| type         |                                  | Device type (Server, Desktop, Switch, IP Camera, ...)    |
| os           |                                  | Operating system                                         |
| os_version   | osversion                        | OS version                                               |
| hw           | hardware                         | Hardware model                                           |
| category     | categories                       | Asset category (IT, OT, IoT)                             |
| subnet_class | net_class                        | Subnet classification (OT, IoT, General)                 |
| ot_scope     |                                  | OT-scoped (t / f) — defers to category + subnet + vetoes |
| functions    | function                         | Asset function tags                                      |
| vendor       | mac_vendor,<br>newest_mac_vendor | Newest MAC vendor                                        |
| tag          | tags                             | Asset tags                                               |
| domain       | domains                          | DNS domains                                              |
| site         | site_name, site_id               | Site                                                     |
| explorer     | agent, last_agent_id             | Last Explorer (UUID prefix)                              |
| task         | last_task_id                     | Last task (UUID prefix)                                  |
| detected_by  | det                              | Detection method (arp, icmp, ndp, ...)                   |

## Boolean / flag fields

| Keyword                               | Matches                                                                                 |
|---------------------------------------|-----------------------------------------------------------------------------------------|
| multi_homed                           | Asset has addresses in multiple subnets (t / f)                                         |
| has_public                            | Asset has a public IP                                                                   |
| has_private                           | Asset has a private IP                                                                  |
| has_ipv4                              | Asset has at least one IPv4 address                                                     |
| has_ipv6                              | Asset has at least one IPv6 address                                                     |
| has_link_local                        | Asset has a link-local address                                                          |
| has_mac                               | Asset has MAC address data                                                              |
| has_os, has_hw, has_name,<br>has_type | Asset has the named field populated                                                     |
| eol                                   | Asset is end-of-life (t / f)                                                            |
| misplaced                             | Asset's category does not match its subnet's classification                             |
| unmapped_mac                          | Node is an unmapped-MAC stub from a switch port                                         |
| is                                    | Topology role: switch, gateway, child (sub-asset behind a gateway)                      |
| gateway                               | Legacy alias — t / f for any gateway, or a specific kind. Prefer is:gateway / is:switch |

## Numeric fields

All numeric fields support comparisons (>, <, >=, <=, ranges).

| Keyword             | Matches                              |
|---------------------|--------------------------------------|
| risk_rank           | Numeric risk rank (0-4+)             |
| outlier_rank        | Outlier score rank (0-4+)            |
| criticality         | Criticality rank                     |
| modified_risk_rank  | Modified risk rank (0 = unmodified)  |
| vulnerability_count | Number of known vulnerabilities      |
| software_count      | Number of detected software packages |
| service_count       | Number of detected services          |
| mac_count           | Number of MAC addresses              |
| address_count       | Number of IP addresses               |

|                   |                                                                   |
|-------------------|-------------------------------------------------------------------|
| name_count        | Number of hostnames                                               |
| service_ports_tcp | TCP port numbers (service_ports_tcp:443, service_ports_tcp:>1024) |
| service_ports_udp | UDP port numbers                                                  |
| rtt, lowest_rtt   | Lowest round-trip time in microseconds                            |
| hops, ttl         | Lowest TTL / hop count                                            |
| newest_mac_age    | Newest MAC age in seconds                                         |

## Services and software

| Keyword           | Matches                                                 |
|-------------------|---------------------------------------------------------|
| service_protocols | Service protocols (http, ssh, tls, modbus, bacnet, ...) |
| service_products  | Service product names (Apache*, nginx, OpenSSH*)        |
| protocol          | Convenience alias for service_protocols                 |
| product           | Convenience alias for service_products                  |

## Time

first\_seen (alias firstseen) and last\_seen (alias lastseen) accept relative ranges like <7days, >24hours, <30days, or absolute timestamps like >2026-01-01.

## Risk

| Keyword | Matches                                              |
|---------|------------------------------------------------------|
| risk    | Risk level label (Critical, High, Medium, Low, None) |

## Node and gateway types

| Keyword      | Matches                                                                                                                          |
|--------------|----------------------------------------------------------------------------------------------------------------------------------|
| node_type    | Graph node type: a=asset, n=subnet, u=unaddressed, g=gateway, m=unmapped MAC, r=router hop                                       |
| class        | Object class: asset, subnet, gateway, unaddressed, unmapped-mac, router-hop, plus device-type aliases (switch, firewall, ...)    |
| gateway_kind | Gateway protocol kind: switch, bacnet, modbus, cip, knxnet, s7comm, dnp3, profinet, ethercat, omronfins, melsecq, hartip, c37118 |

|                  |                                                                                                              |
|------------------|--------------------------------------------------------------------------------------------------------------|
| child_kind       | Asset is a sub-asset behind a gateway of the given kind (same vocabulary as gateway_kind)                    |
| downlink_count   | Number of downstream links (switch ports + OT gateway children). Supports comparisons (>5, <=0, >=10)        |
| hops_from_public | Minimum topology hops to a public-facing asset (0 = the asset itself is public-facing). Supports comparisons |
| switch_port      | Switch port observation on an unmapped-MAC stub (e.g. Gi1/0/1)                                               |

## Geographic search

| Keyword     | Aliases                | Matches                                                        |
|-------------|------------------------|----------------------------------------------------------------|
| geo         | geo_within, geo_radius | Distance: geo:<lat>,<lon>[,<radius>] (km, mi, m, default 50km) |
| country     |                        | ISO country code                                               |
| city        |                        | City                                                           |
| state       | subdivision            | State / subdivision                                            |
| continent   |                        | Continent                                                      |
| postal_code | postal                 | Postal code                                                    |
| timezone    |                        | Timezone (America/Los_Angeles)                                 |
| asn         | isp                    | ASN ID or network owner                                        |
| geo_source  |                        | Source of the geo point (isp, cloud-region, attribute-ip, ...) |

See [Geolocation](#) for how runZero produces these values.

## Tracing paths

The Network map's trace engine answers the question "how does traffic get from A to B?" by computing shortest paths through the rendered topology. The **Trace** panel at the top of the map controls every aspect of the trace.

## Sources and targets

A trace needs at least one of:

- One or more **Sources** — the start of the path.
- One or more **Targets** — the end of the path.

You can supply sources and targets in three ways:

1. **Right-click any node** on the map and choose **Add as Source** or **Add as Target**.

2. **Search, then assign** — type a query into the search bar and click **Set Query As Sources** or **Set Query As Targets** in the trace panel. Every node matching the query becomes a source / target.
3. **Open from asset details** with the **View internet attack path** action — this pre-loads **Public Internet** as the source and the asset as the target.

The same selection methods work in 2D and 3D.

## Trace modes

The button in the trace panel changes label based on what is set:

| Sources set | Targets set | Mode label            | Behavior                                                        |
|-------------|-------------|-----------------------|-----------------------------------------------------------------|
| Yes         | Yes         | <b>Trace Path</b>     | Shortest paths between every source and every target.           |
| Yes         | No          | <b>Trace Outbound</b> | All paths reachable outward from the sources.                   |
| No          | Yes         | <b>Trace Inbound</b>  | All paths converging on the targets from the rest of the graph. |

## Trace from the internet

To answer “what can reach this asset from the public internet?”, set **Public Internet** as the source (right-click the public internet node and choose **Add as Source**) and the asset of interest as the target, then click **Trace Path**. The asset details page **View internet attack path** action does this in one click.

To answer “what does this server expose to the internet?”, reverse it — set the asset as the source and **Public Internet** as the target.

## IP version routing

The trace engine respects two IP-version toggles in the panel’s expanded view:

- **Route IPv4** — include IPv4 paths in the trace (default on).
- **Route IPv6** — include IPv6 paths in the trace (default on).

Two additional toggles control IPv6 visibility on the canvas itself, independent of routing:

- **Show IPv6 link-local** — render `fe80::/10` addresses (default on).
- **Show IPv6 global** — render global unicast `2000::/3` addresses (default on).

## Widely-reused public IP filter

By default the trace ignores public subnet nodes that are linked to more than 5 distinct internal assets. This filters out misleading edges introduced by shared resolvers (1.1.1.1, 8.8.8.8) and content-delivery anycast addresses. The threshold is configurable in the trace panel.

## The Trace panel and per-hop filters

The collapsed trace panel shows the current Sources, Targets, and the trace button. Click the ▼ chevron to expand it and reveal the per-hop filter grid and the routing toggles described above.

### Per-hop filters

The hop-filter grid lets you specify an exact attack path, hop by hop. Each row corresponds to one hop from the source and has two inputs:

- **Include** — only nodes matching this query may appear at this hop.
- **Exclude** — nodes matching this query are removed from this hop.

The inputs accept the full search query language described above, and the panel shows the live match count next to each input as you type.

These filters **reduce** any source → target paths to just the hops that satisfy the constraints. They are subtractive: they cannot add a path that doesn't already exist in the graph, but they can carve a much smaller subset out of a noisy trace.

Examples:

- **Force the path to pivot through SSH:** put `service_protocols:ssh` in **Hop 1** → **Include**.
- **Exclude known firewalls and routers:** put `type:Firewall OR type:Router` in **Hop 1** → **Exclude**.
- **Reach OT through a specific HMI:** put `type:HMI AND name:hmi-prod-*` in **Hop 2** → **Include**.
- **Find paths that cross a multi-homed pivot:** put `multi_homed:t` in **Hop 1** → **Include**.
- **Eliminate a specific subnet from consideration:** put `net:10.50.0.0/16` in any hop's **Exclude** input.

Filters apply hop-by-hop counting from the source side. A trace with all four hop filters empty is the unconstrained shortest path; adding any include / exclude carves the result set without re-running the search server-side.

### What each hop row constrains

The hop index refers to the node's position along the traced path, counting from the source:

| Filter       | Applies to                                                       | Typical use                                                                                                                  |
|--------------|------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>Hop 0</b> | The source node(s) themselves — hop 0 is “0 hops from the start” | Restrict which matched sources are actually used, e.g. only internet-facing assets ( <code>has_public:t</code> )             |
| <b>Hop 1</b> | Nodes reachable one step from the source                         | Force or forbid the first pivot, e.g. exclude <code>os:Windows</code> so the path can’t route through Windows hosts at hop 1 |
| <b>Hop 2</b> | Nodes two steps from the source                                  | Same idea for the second pivot, e.g. require <code>port:22</code> so hop 2 must expose SSH                                   |
| <b>Hop 3</b> | Nodes three steps from the source                                | Constrain the deepest explicitly-controlled hop before the path continues to the target                                      |

An **Include** query means “at this hop, only nodes matching this query are allowed on the path”; an **Exclude** query means “nodes matching this query are removed from consideration at this hop.” Both use the same search terms as the map’s global search bar.

## Effect on the Tracing visualization

When you run a trace (outbound, inbound, or shortest path), the path solver treats hop filters as routing constraints:

- The animated trace path (dashed trace lines and flowing path particles) is re-routed so that each hop position honors its include/exclude rules. If the unconstrained shortest path went through a node your Hop 1 exclude removes, the trace will show an alternate — often longer — route instead.
- Per-cluster **hop labels and hop-colored borders** on the map update to reflect the constrained path, so you can visually confirm the path is passing through the node classes you demanded at each depth.
- If no path can satisfy all hop constraints, the trace reports “No path found between source and target” — a useful signal in itself (e.g. “there is *no* route to the OT network that avoids Windows hosts”).

## Effect on the Choke Points panel

The **Choke Points** panel (left rail) is derived entirely from the active trace path: it collects the intermediate pivot assets on that path, scores them (edges, clusters bridged, multi-homed, services, protocol-gateway status, etc.), sorts by choke score, and tags each entry with an **H0/H1/H2/H3 hop badge** (“Hop from source”).

Because hop filters change which path is traced, they directly reshape this list:

- **Include at hop N** → the choke points shown at that hop depth will only be assets matching your query, letting you answer questions like “if an attacker must go through SSH at hop 2, which asset becomes the critical pivot?”
- **Exclude at hop N** → assets matching the exclusion disappear from that hop level, and the choke-point ranking recomputes around whichever alternate pivots the rerouted



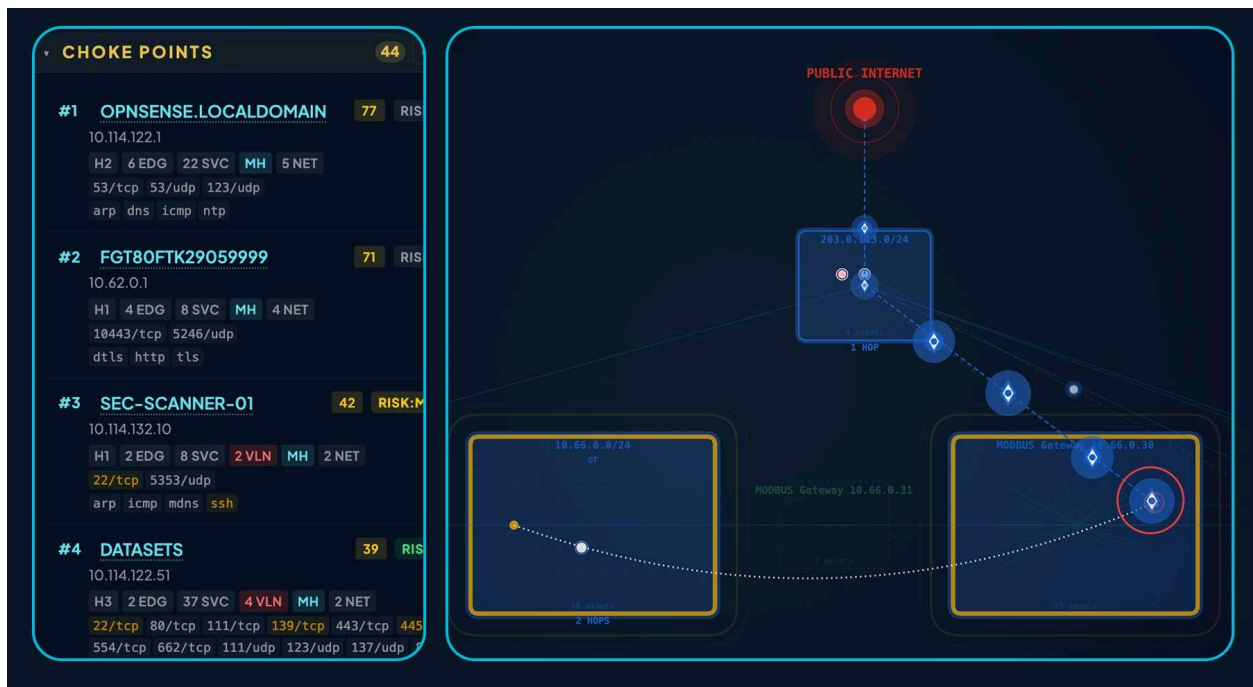
path now uses.

- The choke-score priorities, gateway badges, and the CSV export (which includes the hop column) all reflect the filtered path, so exporting after applying hop filters gives you a prioritized pivot list for that specific constrained attack scenario.

In short: **hop filters are “what-if” routing constraints per path depth** — the trace animation shows you the route that survives your constraints, and the Choke Points panel shows you which assets become the must-pass pivots on that constrained route.

## Choke points

When a trace produces results, a **Choke Points** panel appears on the left rail. It lists the assets that sit on the largest share of source → target paths — the systems that, if removed or hardened, would disrupt the most attack paths.



## How choke points are scored

Each candidate is given a **Pivot Risk Score** that combines connectivity signals with a heavy bias toward devices that act as the only path into a non-IP backplane. Candidates are sorted by this score (descending), with ties broken by hop position and then by asset risk rank. The top of the list is what you should harden first.

The score is built from these signals:

| Signal                                         | Weight                                                   | Why it matters                                                                                                                                                                                                      |
|------------------------------------------------|----------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Protocol-gateway pivot</b>                  | +1000 (dominates the score)                              | A protocol gateway (BACnet, Modbus, EtherNet/IP CIP, KNXnet, S7, DNP3, ...) is the <b>only</b> way to reach the assets on its backplane. Blocking it severs every downstream device, so it always ranks at the top. |
| <b>Gateway pivot is also EOL or vulnerable</b> | +200                                                     | Adds a “replace this first” boost when a protocol-gateway pivot is end-of-life or carries known vulnerabilities.                                                                                                    |
| <b>Cluster count</b> (subnets bridged)         | +15 per additional cluster beyond the first              | Multi-homed assets that bridge two or more subnets concentrate transit traffic — each extra subnet they touch increases their leverage.                                                                             |
| <b>Edge count</b> on the trace                 | +2 per edge, capped at 50 edges                          | Counts how many distinct neighbors the asset talks to on the trace path. Higher fan-in / fan-out means more flows pass through it.                                                                                  |
| <b>Network count</b> (distinct networks)       | +5 per additional network beyond the first, capped at 10 | Rewards bridging across separate networks (different CIDRs or routing domains) rather than just multiple subnets within the same network.                                                                           |
| <b>Multi-homed flag</b>                        | +10 (one-time)                                           | A small constant bump for any multi-homed asset, on top of the cluster / network counts above.                                                                                                                      |
| <b>Service count</b>                           | +1 per service, capped at 20                             | Heavily-exposed assets (many open services) are higher-value pivots than single-purpose hosts.                                                                                                                      |

Candidates that do not meet a minimum bar are excluded from the panel entirely: an asset has to either bridge into a protocol gateway, have more than one neighbor on the trace, or be multi-homed across subnets. Protocol-gateway pivots are always kept regardless of degree because they are by construction the choke point for everything behind them. The list is capped at 100 entries so very large traces stay readable.

## Reading the panel

Each row shows the priority number (1 = highest), the asset’s name and key attributes, and its Pivot Risk Score. Protocol-gateway rows are highlighted and include the gateway protocol kind plus the count of backplane assets that depend on it. Click a row to focus the asset on the canvas and reveal its full details.

The **Show only protocol gateways** toggle filters the list to gateway pivots, and the count badge in the section header still shows how many entries were hidden so you know whether more candidates exist. The full list — filtered or not — can be exported to CSV from the **Export** menu.

## Exporting

The **Export** menu in the toolbar produces both image and graph-data formats from the current map view.

## Images

- **Current view (PNG)** — exactly what is on screen, at the rendered resolution.
- **Full network (PNG)** — the whole topology rendered to a high-resolution PNG, regardless of the current pan and zoom.

The 3D map exports the current rotated view as PNG.

## Graph data

These formats include nodes, edges, subnet clusters, and node positions, suitable for loading into other graph tools:

- **GraphML** — opens in Gephi, yEd, Cytoscape, NetworkX, and any GraphML-aware tool.
- **GEXF** — Gephi-native format with richer attribute typing than GraphML.
- **GML** — Cytoscape, R igraph, and other classic tools.
- **Cytoscape JSON** — direct import into Cytoscape Web and Cytoscape Desktop.
- **Visio (VSDX)** — opens in Microsoft Visio.

## Tabular data

- **Assets (CSV)** — a flat asset table of the current view (or the search-filtered subset when a search is active).
- **Choke Points (CSV)** — one record per choke-point asset for the active trace.

## Sharing a view

The map's URL captures the current filter, pan / zoom, theme, mask, search query, and any active trace. Copy the browser URL — recipients with access to the same organization will see exactly the view you copied.

# World Map

The World Map plots every asset that has a known geographic location on an interactive globe. It is the geographic counterpart to the [Network Map](#): where the Network Map shows logical relationships between assets, the World Map shows where in the world each asset is.

## Prerequisites

The World Map only shows assets that have at least one resolved location. runZero produces locations from MaxMind GeoIP lookups, cloud region centroids, integration attributes, and user-supplied geo\* tags. See [Geolocation](#) for the full list of sources and how to manually pin an asset's location.

Assets with no geolocation data are not shown on the map, and the **View on World Map** action and inventory globe icon are only offered when an asset has at least one resolved location.

## Accessing the map

The World Map can be opened from several places in the console:

- **Inventory** — Asset rows show a small globe icon when the asset has a known location. Clicking the icon opens the World Map filtered to that asset.
- **Inventory search** — The “View on map” action above the inventory grid opens the World Map with the current inventory search applied as the map filter.
- **Asset details** — The **Manage** menu on an asset's detail page offers **View on World Map** (or **Back to World Map** if you arrived from the map). The link is shown only when the asset has geolocation data.
- **Reports** — Open it directly at `/reports/world/map` from the Reports menu, or <https://console.runzero.com/reports/world/map>.
- **Queries** — Saved query rows in the queries grid include a “View on World Map” action that prefills the map with the saved query.

When you open the map from the inventory or from an asset's detail page, runZero records the previous URL as a return target. A **Back** button appears in the top-left of the map and returns you to that exact view, preserving any filters and pagination you had set.

## 2D and 3D modes

The World Map ships in two modes:

- **2D World Map** (`/reports/world/map`) — A flat projection of every asset location, optimised for searching, filtering, and exporting.
- **3D globe** (`/reports/world/map/3d`) — An interactive rotating globe rendering the same point set, useful for executive and dashboard displays.

Both modes use the same data, the same search syntax, and the same filter state. To switch:

- From the 2D map, click the **3D** button in the top-right toolbar.
- From the 3D globe, click the **2D** button in the top-right toolbar.

## Searching the map

The map's search box accepts a domain-specific query language with keyword filters and free text. Type a keyword followed by a colon and a value, or start typing and press **Tab** to autocomplete from the current point set.

Boolean operators **AND**, **OR**, and **NOT** are supported, along with parentheses for grouping. Bare text (no keyword) matches across all displayed fields.

## Supported keywords

| Keyword   | Aliases                      | Matches                               |
|-----------|------------------------------|---------------------------------------|
| country   |                              | ISO alpha-2 country code              |
| city      |                              | City name                             |
| state     | subdivision                  | State, province, or admin subdivision |
| continent |                              | Continent name                        |
| asn       | isp                          | ASN organization name or number       |
| id        |                              | Asset ID                              |
| name      | names, hostname, hostnames   | Asset name or any of its hostnames    |
| addr      | address, addresses, ip, host | Asset IP address                      |
| os        |                              | Operating system                      |
| hw        | hardware                     | Hardware identifier                   |
| type      |                              | Asset type                            |
| category  | categories                   | Asset category (IT / OT / IoT)        |
| risk_rank | risk                         | Numeric risk rank                     |
| last_seen | lastseen                     | Last-seen timestamp                   |
| site      | site_name                    | Site name                             |
| site_id   |                              | Site UUID                             |

Values with spaces or punctuation must be quoted (`city:"San Francisco"`). The search field auto-quotes when you accept an autocomplete suggestion.

## Examples

```
country:US
country:US AND state:CA
city:"San Francisco" OR city:Berlin
asn:"AS15169 GOOGLE"
type:server AND country:DE
category:OT AND continent:EU
NOT country:US
"web-prod-01"
```

## Searching the asset inventory by location

The asset inventory and services inventory accept a `geo` keyword (synonyms: `geo_within`, `geo_radius`) that matches assets whose geolocation is within a circular area around a point. This is the recommended way to scope inventory queries by location.

```
geo:<lat>,<lon>[,<radius>]
```

- `<lat>` and `<lon>` are decimal degrees.
- `<radius>` is optional and defaults to **50 km**. Supply a number followed by `km`, `mi`, or `m` — for example `25km`, `15mi`, `5000m`. Plain numbers are treated as kilometres. The radius is clamped between 1 m and 1,000 km.

## Examples

```
geo:30.2672,-97.7431 # within 50 km of downtown Austin
geo:30.2672,-97.7431,25km # within 25 km
geo:52.5200,13.4050,100mi # within 100 miles of Berlin
geo_within:51.5074,-0.1278,5km # synonym
```

The query matches if **any** of the asset's geolocation sources falls within the circle. See [Geolocation](#) for details on how runZero builds those locations.

## Exporting

The **Export** menu in the top-right toolbar exports the currently filtered point set. The following formats are available:

- **CSV** — a flat table of every visible asset and its location fields.
- **JSON** — full point records for programmatic use.
- **GeoJSON** — standard GeoJSON `FeatureCollection`, suitable for loading into GIS tools.
- **KML** and **KMZ** — for Google Earth and compatible viewers.

## Sharing a view

The **Share link** button copies a URL that captures the current filter, pan, zoom, and any opened asset popup. Recipients with access to the same organization will see exactly the view you copied.

## Organization Overview

The Organization Overview Report captures a point-in-time snapshot of the asset data within your organization and sites. The report organizes data from your asset inventory into relevant sections and summarizes the major findings. The Organization Overview Report is useful for sharing with teams and leaders who may not have access to runZero and need an at-a-glance look into their network.

The report helps you quickly assess high-level metrics across multiple categories for your organization and sites, such as your asset types, operating systems, hardware, protocols, and products. The report also includes a summary of your RFC 1918 coverage, subnet utilization, and switches. For organizations with less than 50,000 assets, you can include additional information from your inventory via asset details and screenshots.

You can schedule the report to run at a specified start time and at a specified frequency. This report can run once, daily, weekly, monthly, or on the Nth weekday of each month. When a report is configured to run recurrently, one or more email addresses can be notified when the report has run.

runZero generates this report in HTML, but you can use your browser's Print to PDF feature to save the file as a PDF for distribution.

## Generating the Organization Overview Report

1. Go to the [Reports page](#).
2. Open the [Organization Overview Report configuration form](#).
3. Enable **Include asset details** if you want additional information about each asset. The asset details will include the asset type, operating system, hardware, outlier score, first seen date, last seen date, site, addresses, names, MAC addresses, protocols, products, and services. This option only applies to organizations with less than 50,000 assets.
4. Enable **Include screenshots** if you want to see captured screenshots in the report. This option only applies to organizations with less than 50,000 assets.
5. Configure **Start time** and **Report frequency** if you want the report to run at a specified time or on a recurring schedule.
6. Provide one or more email addresses in the **Email addresses** field if you want email notifications when the report is run.
7. Generate the report. When the report generation completes, it automatically opens in your browser. Save the report as a PDF to share and distribute as needed.

## Email notifications

You can include a comma-separated list of email addresses when configuring the report to run on a schedule. The specified email addresses will receive notifications when the report has finished each run of the configured schedule. The notification email will contain a link to the report in the runZero Console. Please note that only registered users with access to the relevant organization will be able to access the completed report.



# RFC 1918 Report

Coverage reports help you understand potential blind spots on your network by identifying which IP spaces have been scanned, which ones contain assets, and which ones still are unknown. With this information, you can find things like missing subnets, rogue devices, and misconfigurations.

To access the [coverage reports](#), go to **Reports** on the main menu and scroll down to **RFC 1918 coverage**.

## RFC1918 coverage report

The RFC1918 coverage report helps you better track and identify the subnets that are in use on your internal network, the ones that have been scanned, and the ones that haven't been scanned.

TCP/IP version 4 reserves three ranges of IP addresses for private use. Specified in [RFC1918](#), they are:

| CIDR           | Address range                 | Number of addresses |
|----------------|-------------------------------|---------------------|
| 10.0.0.0/8     | 10.0.0.0 - 10.255.255.255     | 16,777,216          |
| 172.16.0.0/12  | 172.16.0.0 - 172.31.255.255   | 1,048,576           |
| 192.168.0.0/16 | 192.168.0.0 - 192.168.255.255 | 65,536              |

Most companies use these address ranges for their internal IPv4 networks, connecting them to the Internet via Network Address Translation (NAT).

To help you visualize and assess your RFC1918 coverage, the RFC1918 report includes:

- [Coverage maps](#)
- [Subgrids](#)
- [Statistics](#)

## RFC1918 coverage maps

A common network security and administration goal is to scan all of the available private IP addresses, to detect which subnetworks are in use on your internal network. Because of the large number of addresses, this can take a long time, leaving the problem of tracking which addresses have been scanned and which have not.

To solve this problem, runZero's Coverage report shows a graphical map of the RFC1918 private address spaces, showing which pieces have been scanned and to what percentage of completion.

runZero will sometimes detect that a device has additional IP addresses which are not part of the range being scanned. This can indicate that the device is present on an unscanned part of the private IP address space. The coverage reports show this by drawing a red border around the appropriate grid cell.

You can hover the mouse cursor over a cell to see a tooltip showing the CIDR address range the cell represents, how many unscanned hosts are believed to be in that range, and what percentage of the entire range has been scanned.

## **RFC1918 sub-grids**

For the 192.168.0.0/16 map, each cell on the grid represents a /24 (256 addresses). Clicking a cell will take you to the subnet analysis report for that range and list the assets found.

For a map that shows a large address range, such as 10.0.0.0/8, each cell represents an entire /16 range of 65,536 addresses. To help narrow down the search for assets and unscanned hosts, you can click on any cell that represents a /16 range to go a grid map that showing just that range. From a /16 grid, you can use the link at top right to go back to the full range map.

On the /16 sub-grids, each cell is a /24, so clicking one takes you to the subnet analysis for that specific cell's address range, like on the 192.168.0.0/16 map.

## **Statistics**

At the top of the coverage report page, you can see statistics showing how much of the RFC1918 address space you have scanned. Another box breaks the coverage down by the three blocks of reserved addresses.

Clicking the magnifying glass icon in the summary box will create a sample scan task, covering the unscanned address ranges. The refresh buttons create scan tasks to rescan all of the appropriate range.

## External Assets

The External Asset Report captures a point-in-time snapshot of the external asset data within your organization and sites. The report organizes data from your asset inventory into relevant sections and summarizes the major findings. The External Asset Report is useful for sharing with teams and leaders who may not have access to runZero and need an at-a-glance look into their public-facing assets.

The report helps you quickly assess high-level metrics across multiple categories for external assets in your organization and sites, such as asset types, operating systems, hardware, protocols, and products. The report also includes a summary of top Certificate Authorities and GeoIP countries. For organizations with less than 50,000 assets, you can include additional information from your inventory via asset details and screenshots.

You can schedule the report to run at a specified start time and at a specified frequency. This report can run once, daily, weekly, monthly, or on the Nth weekday of each month. When a report is configured to run recurrently, one or more email addresses can be notified when the report has run.

runZero generates this report in HTML, but you can use your browser's Print to PDF feature to save the file as a PDF for distribution.

## Generating the External Asset Report

1. Go to the [Reports page](#).
2. Open the [External Asset Report configuration form](#).
3. Enable **Include asset details** if you want additional information about each asset. The asset details will include the asset type, operating system, hardware, outlier score, first seen date, last seen date, site, addresses, names, MAC addresses, protocols, products, and services. This option only applies to organizations with less than 50,000 assets.
4. Enable **Include screenshots** if you want to see captured screenshots in the report. This option only applies to organizations with less than 50,000 assets.
5. Enable **Include unscanned subnet details** if you want to see results from external subnets that have not been scanned by runZero.
6. Enable **Include TLS certificate details** if you want to see details gathered about TLS certificates.
7. Use the **External IP exclusions** if you are using public IP address space internally and do not want those assets included in the report.
8. Configure **Start time** and **Report frequency** if you want the report to run at a specified time or on a recurring schedule.
9. Provide one or more email addresses in the **Email addresses** field if you want email notifications when the report is run.
10. Generate the report. When the report generation completes, it automatically opens in your browser. Save the report as a PDF to share and distribute as needed.

## Email notifications

You can include a comma-separated list of email addresses when configuring the report to run on a schedule. The specified email addresses will receive notifications when the report has finished each run of the configured schedule. The notification email will contain a link to the report in the runZero Console. Please note that only registered users with access to the relevant organization will be able to access the completed report.

## Site Comparison

The [Site and organization comparison feature](#) lets you generate a side-by-side analysis of two sites, so you can understand:

- [How assets change over time](#) such as their TCP/UDP services, TCP/UDP ports, and service protocols. You can leverage this data to evaluate historical changes for assets for a specific point in time.
- [How exposure changes](#) based on scanning your network from different locations. For example, if you use public IP addresses internally and externally, you may want to scan those addresses from inside and outside your network to understand your potential exposure.

The report provides a summary view of differences. It only captures certain attributes that were added or removed from an asset, such as IP addresses, TCP ports, TCP service counts, UDP ports, UDP service counts, service protocols, and service counts. It does not track every modification to an asset, such as fingerprint or service banner changes.

After you run the report, the data presented in the report will be static. Any changes to your current inventory may result in assets no longer being accessible from the report.

## Generate a site comparison

Generating the site comparison requires selecting a current site and a comparison site. The sites can be in different organizations. You can also select "All sites" to compare all sites in an organization with a different site and organization.

When the report runs it will assemble the two sets of assets specified, compare them using runZero's asset matching algorithms, and generate a set of differences. You can then browse the summarized results in the report.

### To generate a site comparison:

1. Verify that your current organization is one containing an inventory of assets you want to compare.
2. Go to the [site comparison page](#).
3. When the site comparison configuration page appears, the current organization will be set to the one you currently have selected. You can change the current site, if needed. For the comparison, choose the organization and site you want to run the analysis against.
4. Run the report. A task will be created to perform the comparisons, and you will be taken to the task page.
5. When the task is complete, the report will appear in the list of recent analysis reports at the top of the Reports page. You can then [view and search the results](#).

## View how assets change over time

To analyze how assets have changed over time, you can compare the data from an old scan task with your most recent inventory. Setting a point-in-time comparison requires creating a new project that you can import your old scan data into.

Here's how you can set up a point-in-time analysis:

1. Go to the organization or project that contains the task scan data you'd like to use.
2. Go to your [completed tasks](#) and locate the task that contains the data for the point in time you'd like to compare.
3. From the task page, download the task data. It will be in a file with a name starting `scan_` and ending `.json.gz`. This is the file you'll import into your new project. You don't need to uncompress the file, unless you're curious to look at the JSON data.
4. Next, [create a new project](#) for your import. You can create an organization if you intend to perform the analysis regularly.
5. After you create the project, go to the Inventory page and [import your scan task data](#) into it.

Now, you're ready to compare your current inventory with a previous version of it. Go to the [site comparison page](#). You'll need to select the organization and site for your current inventory as the site to compare against.

After the report runs, it shows a table that [highlights the differences](#) between the two sites, which in this case will represent two points in time, going from past to present.

You can also run the comparison by selecting your current organization first, and then choosing the project with the past data as the comparison site. The results will be the same, but with their sense reversed – that is, services which show as added when going from past to present, will show as removed when going from present to past.

## View how exposure differs between networks

You can run the site comparison report to compare how exposure varies based on where you scan your network from. By comparing two inventories from two different perspectives, you can obtain better insights on your attack surface, which can help with active defense or risk reduction. For example, if you use external IPs internally and externally, you may want to scan those addresses from inside your network and outside your network. Then, you can run the site comparison report to compare the results from those two sites.

Here's how you can set up a diff for exposures between networks:

1. Set up a site with an Explorer on one network.
2. Set up another site with an Explorer on a different network. This site can be in the same organization as the first site.
3. Run a scan of the same address range with each Explorer. Verify you have the correct site selected for each scan.
4. After the scans complete, run the [site comparison](#) to generate the diff. The report will show a table that [highlights the differences](#) between the two sites.

## Analyze the results in the site comparison report

The site comparison generates a table to show how assets' addresses, names, services, ports, and protocols differ between sites. Red text, denoted by the minus (-) sign, indicates that attributes were removed going from left to right. Green text, denoted by the plus (+) sign, indicates attributes were added.

Here are some things you should know about the results:

- **Address and Other Address** - The first address column contains the asset's addresses for the organization currently selected when the report was requested. The second address column, *Other Address*, contains the asset's addresses for the organization and site that was compared against the current organization.
- **Name and Other Name** - The first name column contains the asset's names in the organization currently selected when the report was generated. The second address column, *Other Name*, contains the asset's names for the organization and site compared against.
- **TCP and UDP services** - These columns show how the total number of TCP and UDP services differ between sites.
- **TCP and UDP ports** - These columns show the ports that have been added or removed between sites.

You can click on the green info (i) icon to view a more detailed comparison of the asset.

Clicking on one of the asset addresses in the report will bring up the full current asset record, if the asset still exists in the appropriate organization and site.

## Search the site comparison report

You can search the report using the runZero [search query language](#). In the following descriptions, the main set refers to the assets in the organization that was current when you generated the report (i.e., the address and name columns). The comparison set refers to the assets in the organization and site that you chose to compare against (i.e., the other address and other name columns).

| Keyword                   | Meaning                                                                                                                                             |
|---------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------|
| address:                  | Search for assets in the main set with a specified IP address. Use <code>none</code> to find assets that are missing from the main set.             |
| net: or cidr:             | Filter assets by their network CIDR range in the main set.                                                                                          |
| other_address:            | Search for assets in the comparison set with a specified IP address. Use <code>none</code> to find assets that are missing from the comparison set. |
| other_net: or other_cidr: | Filter assets by their network CIDR range in the comparison set.                                                                                    |
| id:                       | Search by asset ID in the main set.                                                                                                                 |

|           |                                                      |
|-----------|------------------------------------------------------|
| other_id: | Search by asset ID in the comparison set.            |
| tcp:      | Search for a TCP port change by number.              |
| udp:      | Search for a UDP port change by number.              |
| protocol: | Search for a TCP or UDP port change by service name. |



## External reporting

External reporting lets you package a dashboard, an inventory view, or a runZero report into a point-in-time **snapshot** and share it with people who do not have a runZero account. Recipients open a link, verify their email address, and view the snapshot in a read-only browser page. Snapshots are immutable, scoped to the organizations you select, and can be regenerated on a recurring schedule.

External reporting is useful when you need to give executives, auditors, customers, or other teams an at-a-glance view of your asset inventory, security posture, or a specific report without provisioning runZero accounts for them.

## What you can share

Two types of external reports are supported:

- **Dashboard** — any dashboard you can view, including [cross-organization shared dashboards](#). The snapshot captures the dashboard widgets for a configurable date window.
- **Inventory** — an asset, software, service, or vulnerability inventory query, including the columns you have selected and the current sort order. The snapshot freezes the matching rows and serves them as a read-only table to the recipient.

## Sharing a report

You can start an external report from anywhere in the runZero Console that displays one of the supported views:

- **From a dashboard:** open the dashboard and click the **Share** button in the header. This option is available on dashboards that can be shared, such as runZero-managed and cross-organization dashboards.
- **From an inventory page:** on the [Assets](#), [Software](#), [Services](#), or [Vulnerabilities](#) page, run the query you want to share and use the Share button to create an external report. The current query, columns, and sort order are saved with the report.

The [External Reports page](#) lists all reports you have already created and lets you edit or delete them.

## Share dialog options

When you create or edit an external report, the share dialog collects the following information:

- **Report name** — identifies the report in the runZero Console and appears in the email sent to recipients.
- **Email recipients** — one or more email addresses that should be notified each time a snapshot is generated. The notification email contains the link recipients use to access the report.

- **Report frequency** — how often the report regenerates. Choose **One time** for a single snapshot, or **Daily**, **Weekly**, or **Monthly** for a recurring schedule.
- **Organizations** (My Organizations view) — the organizations whose data should be included in the snapshot. You can only include organizations you have access to.
- **Automatically include new organizations** (My Organizations view) — when enabled, organizations created after the report is set up are automatically added to future runs of a recurring report, provided you have access to them.

Click **Send report** to save the configuration and queue the first snapshot. Snapshot generation runs in the background; the recipient email is sent as soon as the snapshot is ready.

## How recipients view a report

When a snapshot is generated, each recipient receives an email containing a link to the report. The first time they open the link, recipients are asked to accept the runZero terms of service.

runZero then emails a single-use verification link to that address. Clicking the verification link confirms the recipient's identity, sets a secure session cookie that lasts for 30 days, and redirects them to the report. While the cookie is valid the recipient can return to the same report (or any other report shared with the same email address by the same runZero account) without verifying again.

The verification link itself is short-lived and expires 30 minutes after it is sent. If a recipient does not click it in time, they can request a new one by reopening the report link and entering their email address again.

Recipients with a runZero account that has access to one of the report's organizations can open the report directly using their existing session, without going through the email verification step.

## Managing reports and snapshots

The [External Reports page](#) shows every external report your organization has created. Reports are grouped under three tabs:

- **All** — every external report.
- **One-time** — reports configured to run once.
- **Recurring** — reports configured to run on a daily, weekly, or monthly schedule.

Clicking a report opens its details page, which shows the report's configuration along with a table of every snapshot that has been generated. From this page you can:

- **Edit** the parent report to change its name, recipients, schedule, date window, or organizations. Edits affect future runs only; existing snapshots are not modified.
- **Delete** the parent report. Deleting the parent stops future runs and removes all of its snapshots.
- **Copy** the share link for an individual snapshot.
- **Revoke** a snapshot's access link. Revoking invalidates the URL so it can no longer be used to view the report, but leaves the snapshot record in place for auditing purposes.

- **Delete** an individual snapshot. Deleting a snapshot revokes its link and removes the captured data.

Each generated snapshot link is valid for 90 days from the time the snapshot is created. After that the link stops working automatically; recipients of recurring reports continue to receive a fresh link in each run's email.

## Security and access

External reporting is designed so that report data only ever leaves runZero through links you explicitly create:

- **Cryptographic share links.** Each snapshot link contains a 256-bit random token signed with a per-account HMAC key. Tampered or fabricated links are rejected without a database lookup.
- **Email gate.** Before a recipient can view a snapshot they must prove ownership of an email address by clicking a verification link sent to that address. Identity is then carried in a secure, account-scoped browser cookie.
- **Organization access reconciliation.** Each time a recurring report runs, runZero re-checks the original creator's access to the report's organizations. Organizations they no longer have access to are dropped from the snapshot. If they have lost access to all of the report's organizations, the run is skipped.
- **Audit events.** Every successful access of an external report is recorded as an `external-report-viewed` event for each organization included in the snapshot. The event includes the recipient's email address, IP, and user agent and is visible in the runZero activity log.
- **Rate limiting.** The public verification and access endpoints are rate limited to mitigate abuse.
- **Revocation.** You can revoke any snapshot link at any time from the report's details page. Revoked links cannot be reactivated.

# Switch Topology



Most users should start with the [Network Map](#), which combines switch topology, network bridges, and route pathing into a single interactive view with search, hop-filter tracing, and choke-point analysis.

The runZero switch topology report allows you to view a graph of the switches and routers on your network, and see how they are interconnected. It will also show which assets are connected to each switch.



The switch topology report requires a source of layer 2 topology information. This is typically obtained from SNMP, or by connecting an integration that provides topology information, such as Cisco Meraki.

## Generating the switch topology report

Once you have obtained topology data, you can launch the [switch topology report](#).

The initial view shows a graph of network switches, with the links between them shown as lines. Beneath each switch you can see a count of how many known assets were detected as having that switch as their upstream network connection.

You may also see a count of unmapped assets listed. These are assets inferred to exist by MAC addresses which were found in the topology data, but the MAC address isn't known to belong to any scanned asset. Lists of unmapped MACs are also available in tabular form from the Unmapped MACs report.

At the bottom of the page you will see the total number of assets in the graph, and the total number of unmapped MACs.

The report will attempt to lay out the graph appropriately. You can drag nodes around to make it clearer.

Clicking on a node representing a switch will display a pop-up window with more information about it. This includes a link to view the asset details for the switch, and a link to view a list of the unmapped assets.

Double-clicking a node will expand it to show the individual assets connected to it. Clicking an asset will show a pop-up window with a link to view the asset details.

## Filtering the switch topology report

The switch topology report is limited in the number of assets it can display. To focus on the set of assets you are interested in, you can use the filter box at the top of the report page. It accepts search strings in the standard runZero search language. For example, you could filter to a given subnet using a search string such as `cidr:10.1.6.0/24`.

There is also a drop-down to switch quickly between sites.

Additional buttons in the filter box allow you to collapse or expand all of the switch nodes at once.

The *Export view* button will render a PNG file of the graph as currently displayed, and download it.

## Limitations of the switch topology report

The switch topology report may not always be entirely accurate because of limitations on the data runZero can gather.

In the case where there is SNMP data available, runZero will pull a snapshot of the SNMP data from each device when it is scanned, then use that to build topology. However, in many cases a single infrequent snapshot is not enough to show a complete picture in complicated environments, and links may end up missing. In addition, only recent SNMP data is used — if devices have not been scanned in the last 9 days, their SNMP topology data will not be used.

When there is no SNMP information, runZero will attempt to compute topology based on which switch claims to have seen the MAC, which may not always be the nearest access switch. Our algorithm looks for the port with the least number of shared MACs to find best match, but that depends on the switch cache timeouts and how the switch was scanned, so there may be links shown that don't exist as direct physical connections.

## Cisco Catalyst devices

SNMPv3 on Cisco Catalyst devices will not let you pull the bridge port information that we need unless you specifically enable per-VLAN access.

SNMP v3 access to VLAN ARP/FDB tables requires this access rule:

| Version    | Command                                                                                       |
|------------|-----------------------------------------------------------------------------------------------|
| Newer IOS: | <code>snmp-server group YourGroupName v3 auth context vlan- match prefix</code>               |
| Older IOS: | <code>snmp-server group YourGroupName v3 auth context vlan-1 (repeated for every VLAN)</code> |

Note that even after this is done, runZero will need to send a separate SNMP request for every VLAN. This can significantly slow down scans with SNMP enabled on a network with many Catalyst devices.

# Network Bridge Report



Most users should start with the [Network Map](#). It surfaces the same multi-homed bridge assets — and goes further with subnet clustering, choke-point ranking, hop-filter tracing, and exports to Gephi, Cytoscape, and Visio.

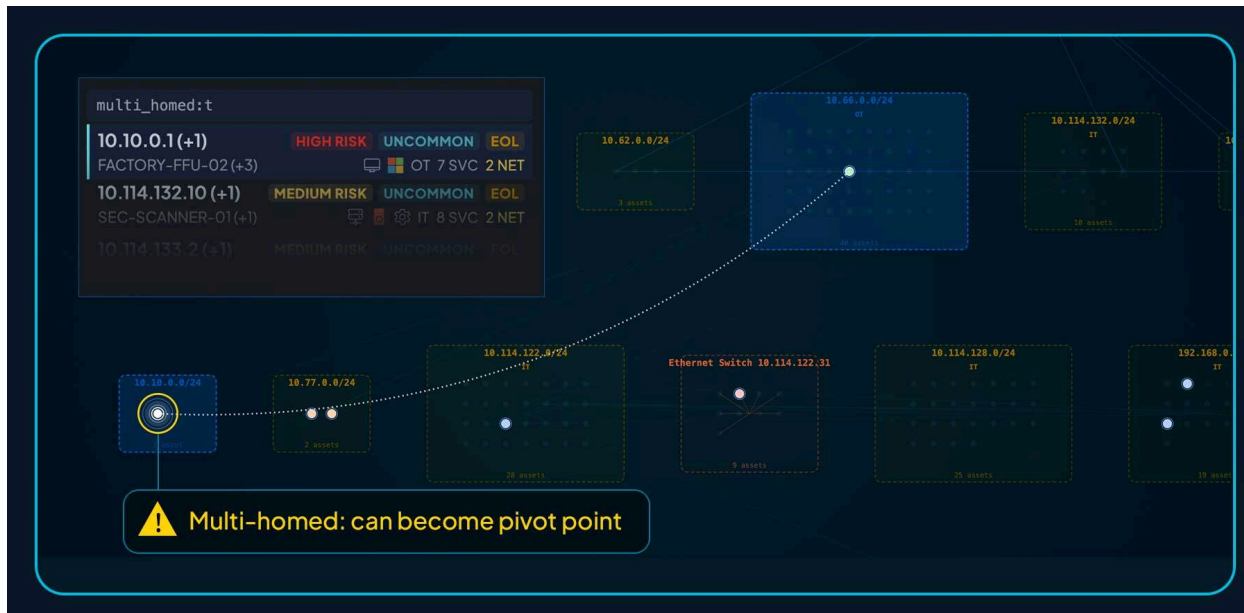
The network bridge report finds assets that bridge two or more network segments. These multi-homed assets are the pivots an attacker uses to move between zones — between guest Wi-Fi and corporate LAN, between IT and OT, or between an internal segment and the public internet.

## Accessing the report

- **Reports** — Open it directly at `/reports/analysis/bridges` from the Reports menu, or <https://console.runzero.com/reports/analysis/bridges>.
- **Network Map** — The same assets appear on the [Network Map](#) under the **Pivots** stat in the Network Intel card (`multi_homed:t`). Each multi-homed asset shows up once in every subnet it has an address in, with edges drawn between the copies so you can see exactly which segments it bridges.

## How bridges are detected

runZero detects network bridges by collecting extra IP addresses returned in responses to common discovery probes — NetBIOS, SNMP, MDNS, UPnP, and others — and by correlating MAC addresses across subnets. A bridge is reported whenever the same asset is observed with addresses in two or more distinct subnets.



The detection is opportunistic. Hardened endpoints (host firewalls, disabled discovery services) may not reveal their other interfaces, so the report can miss bridges. Treat what it shows as a floor, not a ceiling.

## Reading the report

- **External networks** are drawn in red, **internal networks** in green.
- Edges connect a subnet to every multi-homed asset that has an address in it.
- Single-homed assets are omitted to keep the graph readable.

The visualization is intentionally segmentation-focused; it is not a full topology view. For a full Layer-2 / Layer-3 picture, use the [Network Map](#). For switch-port-level connectivity, use the [Switch Topology](#) report.

## Querying multi-homed assets directly

Every bridge in this report is a multi-homed asset. You can query them straight from the inventory:

```
multi_homed:t
multi_homed:t AND has_public:t # bridges with at least one public IP
multi_homed:t AND category:OT # OT pivots – top-of-register OT risk
```

Combine with `subnet_class:OT`, `category:IT`, `risk:high`, or `eol:t` to surface the bridges that matter most for segmentation hardening.

## See also

- [Network Map](#) — full interactive topology with bridge / pivot analysis built in.

- [Understanding network segmentation](#) — background on segmentation gaps and how to remediate them.
- [Switch Topology](#) — switch-port-level connectivity from SNMP and integrations.



# Asset Route Pathing



Most users should start with the [Network Map](#), which lets you set sources and targets, trace inbound or outbound paths, apply per-hop include/exclude filters, and rank choke points — all in one interactive view.

The asset route pathing report generates a visualization of the potential network paths between a source asset and destination asset in an organization. Following the paths, you can see assets connected between the target and source destinations. These assets represent opportunities an attacker could potentially leverage to break into your target asset.

The runZero Explorer performs a traceroute between itself and the source, and then another with the target. runZero then compares the data to infer shared points between the assets. runZero does not get any paths from a direct traceroute. This is runZero's best effort — based on the scan data it has — to identify the assets that it sees as viable points between two assets.

You can share this report with your IT and security teams to highlight assets that could be leveraged as pivot points to your critical assets. Armed with this information, they can identify systems that may need to be hardened. They can assess whether or not the appropriate critical controls are in place to prevent unauthorized access to those assets.

Customers with highly segmented environments can use this report to quickly identify paths from low security assets to critical assets. For example, this report can indicate whether a device in a wireless guest network can reach a system within the PCI cardholder data environment.

## Terminology

Before diving into the asset route pathing report, here are some terms you need to know:

- **Hop** - Any node between the source and destination.
- **Node** - Any asset or hop. Nodes can be an IP, an asset, or unknown.
- **Asset** - Any device that is part of your runZero inventory.
- **Network path** - runZero does not get any paths from a direct traceroute. Instead, runZero uses the data it has to identify and display potential network paths between the target and source.

## Generating the asset route pathing report

1. Launch the [asset route pathing report](#).

2. When the *Asset route pathing* page appears, you will need to select a source asset and a destination asset. Use the search to filter assets by keyword or the table pagination to browse all of your assets.
3. After you have a source and destination asset selected, start the trace.
4. In the generated results, you will see the potential paths between the source and destination asset.

## Analyzing the report

Locate your source asset and target asset. If there are hops between the two assets, you should review them and secure the paths between them. Take a look at the services running on those systems that may provide potential entry points for attacks and harden them.

Nodes in the [asset route pathing report](#) are color-coded to help you identify the source asset and destination asset.

The report uses the following colors:

- Green - The source asset
- Red - The destination asset
- Orange - A multi-homed asset that may act as a pivot point
- Blue - A standard layer-3 routing hop
- Gray - Asset is unknown

A hop labeled `Unknown` indicates an intermediate hop in the layer-3 path that did not respond with ICMP errors for TTL exceeded packets.

## Sharing the report

There are a couple of ways to share the results from the [asset route pathing report](#). You can either export a PNG or dotfile of the report or share a direct link to the report.

- **Export a PNG** - A PNG export will take a snapshot of your report. To share the [asset route pathing report](#), click **Export view**. A PNG will download to your computer.
- **Export a dotfile** - You can export a dotfile and feed it into a Graphviz engine or open source visualization tool. The file allows you to render the image in formats like SVG, PSD, and PNG.
- **Share a link** - When viewing the generated asset route pathing report, copy the URL. You can share the URL directly with other team members who have a runZero account and access to the organization.

## Exporting a dotfile

A dotfile is a text file that can be fed into Graphviz engines or open source visualization tools. With the dot file, you can render the image in other [formats](#), like SVG, PSD, and PNG.

1. Launch the [asset route pathing report](#).
2. When the *Trace path* page appears, you will need to select a source asset and a destination asset. Use the search to filter assets by keyword or the table pagination to browse all of your assets.
3. After you have a source and destination asset selected, generate the report.
4. After the visualization appears, click the **Export report** button. A window appears prompting you to enter a name and download location for the file.
5. Enter a name for the file and choose where you want to save it on your computer.
6. Save the file.

## FAQs

### Why are there hops in the report that aren't in my inventory?

Not every hop is a runZero asset. runZero fills in IP addresses for some asset hops, based on information that is pulled out of the traceroute data. Sometimes, runZero is able to extract asset information from the traceroute data that isn't part of the inventory, which is why you may be seeing them in the report.

### Can runZero determine how two assets are talking to each other?

No, runZero can only identify potential paths—not if they are routable. runZero does not test or validate the paths.

### Why isn't the asset route pathing report available?

The asset route pathing report is only available for runZero Platform customers.

# Cryptographic bill of materials

The **Cryptographic bill of materials** (CBOM) report is available on the [Reports page](#) and gives you an inventory of cryptographic algorithms, protocol versions, certificates, public keys, and reused key material observed in your organization. It helps you identify cryptography that is already weak, find classical public-key cryptography that will require a post-quantum migration, and prioritize remediation.

The cryptographic inventory can also be exported as CycloneDX JSON from the asset export API:

```
https://console.runzero.com/api/v1.0/export/org/assets.cbom.json
```

## Post-quantum status

The report assigns each observed cryptographic component a post-quantum status. The status is not a simple claim that an entire service is either quantum-safe or quantum-unsafe. It considers the algorithm's cryptographic role, protocol version, negotiated key exchange, and known classical weaknesses.

| Status              | Meaning                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Unsafe</b>       | The cryptography is broken or provides little or no protection today, independently of quantum computing. Examples include SSL 2.0, SSL 3.0, RC4, single DES, MD5, NULL, and export-grade cipher suites.                                                                                                                                                                                                                        |
| <b>Deprecated</b>   | The cryptography is superseded and should be removed, but is not considered practically broken in the observed role. Examples include TLS 1.0 and 1.1, SHA-1, 3DES, 64-bit block ciphers such as Blowfish and CAST, RIPEMD, and severely truncated authentication tags.                                                                                                                                                         |
| <b>Transitional</b> | The cryptography is currently classically strong but does not demonstrate post-quantum readiness, or it uses a classical asymmetric algorithm that a sufficiently capable quantum computer could break. Examples include classical key agreement, RSA key transport, classical public-key encryption and signatures, TLS 1.2, TLS 1.3 without a recognized post-quantum key exchange, and SSH with only classical key exchange. |
| <b>Safe</b>         | The observation includes a recognized post-quantum or hybrid algorithm, or a symmetric or hash construction that retains useful security against quantum attacks. AES-128, AES-192, AES-256, SHA-256, and SHA-384 qualify; they are not downgraded solely because of Grover's algorithm.                                                                                                                                        |
| <b>Unknown</b>      | Cryptographic material was observed, but the algorithm, protocol evidence, or cryptographic role could not be recognized reliably. Unknown is a visibility or classification gap, not an assumption that the cryptography is safe.                                                                                                                                                                                              |

## TLS and SSH rules

For TLS, **Safe** requires both TLS 1.3 and a recognized post-quantum or hybrid key exchange. TLS 1.3 without that exchange is **Transitional**. TLS 1.0 or 1.1 is **Deprecated**, and SSL 2.0 or 3.0 is **Unsafe**, even when a service also supports TLS 1.3 and a post-quantum exchange.

For SSH, a recognized post-quantum or hybrid key-exchange algorithm advertised in the server's KEXINIT response is **Safe**. A server that advertises only classical key exchange is **Transitional**. If no actual key-exchange algorithm is observed, the result is **Unknown**.

## Combined results

When a certificate contains both a public-key algorithm and a signature algorithm, the certificate receives the worse status of the two. A cipher suite is similarly assessed using its component algorithms. For an asset, the report uses the worst status observed across its relevant TLS and SSH services.

The status severity order is:

**Unsafe > Deprecated > Transitional > Safe**

**Unknown** remains an explicit uncertainty state rather than being counted as Safe.

## Migration deadlines

The report shows published transition schedules alongside the inventory. For United States federal high-value assets and high-impact systems, Executive Order 14412 sets these dates:

- **December 31, 2030:** post-quantum cryptography for key establishment, as defined in FIPS 203.
- **December 31, 2031:** post-quantum cryptography for digital signatures, as defined in FIPS 186-5.

These dates are published milestones, not runZero estimates. The report keeps FIPS approval separate from post-quantum status: an algorithm can be post-quantum without being FIPS-approved, and a FIPS-approved algorithm can still be vulnerable to quantum attacks.

## Interpreting the results

Prioritize **Unsafe** findings first because they represent present-day cryptographic weaknesses. Address **Deprecated** findings next, then use **Transitional** findings to build and execute your post-quantum migration plan. Investigate **Unknown** findings to determine whether additional scanning, service evidence, or algorithm identification is needed.

# Additional resources

## Leveraging the API

runZero provides three primary APIs as well as integration-specific endpoints:

- The Export API provides read-only access to a specific organizations.
- The Organization API provides read-write access to a specific organizations (Professional and Platform licenses).
- The Account API provides read-write access to all account settings and organizations (Platform license).

To get started, you will need an API key / token or API client credentials.

## API keys and tokens

The console supports five types of runZero API key, with different levels of access to runZero APIs. Each type grants a fixed level of access; for a credential with an explicit set of permissions, use an [API client](#) instead. As an additional security option around API access, runZero allows users to create an IP address allowlist to restrict API access based on IP addresses and network ranges (in CIDR notation).

## Export tokens

An export token only allows access to export data via the [export API endpoints](#) under the `/api/v1.0/export` path. It cannot access any other APIs.

An export token has information about its organization encoded into it. It can only be used to export data for that organization. There is no need to specify the organization when using an export token.

Export tokens can be recognized by their ET prefix.

To generate an export token, go to the Organizations page, click on the desired organization to view its details page. Then, click **Edit organization** and scroll down to the export tokens section and use the button to generate or regenerate the token.

## Download tokens

A download token allows access to download the runZero Explorer. This can be useful if you want to automate downloads for deployment across many machines, or if you need to containerize the Explorer. It cannot be used to access any other API.

Like export tokens, the download token has information about the organization encoded into it. This determines the organization the downloaded Explorer will be associated with.

Download tokens can be recognized by their **DT** prefix.

To obtain the download API token, go to the Organizations page, click on the desired organization to view its details page. Then, click **Edit organization** and scroll down to the download token section.

## Organization API tokens

An organization API token allows read and write access to data within a particular organization, using the organization API. Most of the organization endpoints are under the `/api/v1.0/org` path. Organization tokens can also be used to access the export APIs.

An organization API token has information about the organization encoded into it. It can only be used to access that organization. There is no need to specify the organization when making a call using an organization API token.

Organization API tokens can be recognized by their **OT** prefix.

To generate an organization API token, go to the Organizations page, click on the desired organization to view its details page. Then, click **Edit organization** and scroll down to the organization API tokens section. You can use the buttons to generate named organization API tokens and revoke them.

## Account API tokens

An account API token allows read and write access to the account API endpoints. These provide access to perform account-level operations such as creating and deleting organizations and users. Account API tokens can also be used for organization level access and for the export API.

Account API tokens require a Platform license and are generated from the [Account settings page](#). To use an account API token with the Organization or Export API, specify the additional parameter `_oid=[organization-id]` in the query parameters.

An account API token has no specific organization encoded into it. When using an account API token with an API call that applies to an organization, the organization must be specified. This can be done by appending an `_oid` value to the URL query parameters. The `_oid` should be the unique ID of the organization. This can be found on the organization's information page.

Account API tokens can be recognized by their **CT** prefix (client token).

To generate an account API token, go to the Account settings page under Account in the left navigator. You can create and delete account API tokens as needed.

## API client credentials

The tokens above are bearer strings with fixed access. API client credentials are different: they are a client ID and secret exchanged for a short-lived access token via [OAuth2](#), and each

one carries an explicit set of [permissions](#) and the organizations those permissions apply to.

Because their access is explicit, API clients are the right choice for anything that should not hold full account access. A client can also be tied to the user who created it, so that it narrows automatically when that person's access changes.

Users create their own from [My API clients](#). Superusers manage account-wide ones from the [API clients page](#). Your REST client should use the OAuth 2.0 authorization type and Client Credentials grant type. See the [OpenAPI specification](#) for the access token details.

See [API clients](#) for the full details.

## Authentication

Once you have a token or some API client credentials, you can authenticate against the runZero API.

For export, organization and account tokens, your REST client should use the token with [the Authorization: Bearer standard header](#) to authenticate.

For API client credentials, you must use the generated client ID and client secret to make an OAuth2 call to generate an access token. For example:

```
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" \
 --data-urlencode "grant_type=client_credentials" \
 --data-urlencode "client_id=<CLIENT_ID>" \
 --data-urlencode "client_secret=<CLIENT_SECRET>" \
 https://console.runzero.com/api/v1.0/account/api/token
```

## API rate limiting

API calls are rate limited. You can make as many API calls per day as you have licensed assets in your account. For example, if you have 1,000 licensed assets, you can make 1,000 API calls per day. Each API call returns rate limit information in the HTTP headers of the response:

- X-API-Usage-Total - Total number of calls made to the API
- X-API-Usage-Today - Number of calls made to the API today
- X-API-Usage-Limit - Your daily API call limit, shared across all API keys
- X-API-Usage-Remaining - The number of API calls remaining from your daily limit

In addition, there's a limit of 2,000 requests per 5 minutes per source IP address. If this is exceeded, an HTTP 429 response will be sent with an error of "request limit exceeded", and your client code should delay the next request.

## IP Address Allowlist

As an additional security option around API access, runZero allows users to create an IP address allowlist to restrict API access based on IP addresses and network ranges (in CIDR



notation). Users can access this configuration from the Account Settings section of their console under a setting called "API key IP address allowlist".

This field accepts a comma-separated list of both IPv4 and IPv6 IP addresses and network ranges (in CIDR notation) and applies to all API usage across the console. An empty allowlist will result in the allowlist functionality being disabled and allowing all IP addresses to access the API.

## runZero API usage examples

This guide demonstrates how to use `curl` with the **runZero APIs**, including the **Organization API** for creating scan tasks and the **Export API** for retrieving data.

### Scan API usage

You'll need:

- An **Organization API token** (prefixed with `OT`)
- The **Site ID** where you want scan results to be stored

### Set environment variables

First, set your Organization API token and Site ID as environment variables:

```
export RUNZERO_ORG_TOKEN="OT-YOUR-API-TOKEN-HERE"
export YOUR_SITE_ID="YOUR-SITE-ID-HERE" # Replace with the actual Site ID
```

### API endpoint

The API endpoint to start a scan task within a specific site is:

```
PUT /api/v1.0/org/sites/{site_id}/scan
```

Example base URL:

```
https://console.runzero.com/api/v1.0/org/sites/${YOUR_SITE_ID}/scan
```

### Create a scan task for a specific Explorer

To run a scan on a **specific Explorer**, use the "explorer" property in the JSON payload with the Explorer's UUID.

```
curl -X PUT "https://console.runzero.com/api/v1.0/org/sites/${YOUR_SITE_ID}/scan" \
 -H "Content-Type: application/json" \
 -H "Authorization: Bearer ${RUNZERO_ORG_TOKEN}" \
 -d '{
 "explorer": "<YOUR-EXPLORER-ID>",
 "targets": "192.168.1.0/24",
 "scan-name": "Scan via Specific Explorer",
```

```
"rate": "3000"
}'
```

## Create a scan task using a hosted Explorer

To run an external scan using a **runZero hosted Explorer**, use the "hosted-zone-name" key.

- Set "hosted-zone-name": "auto" to let runZero choose the best hosted zone.
- You can also specify a specific hosted zone name.
- Targets must be **public IP addresses or hostnames**.

```
curl -X PUT "https://console.runzero.com/api/v1.0/org/sites/${YOUR_SITE_ID}/scan" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer ${RUNZERO_ORG_TOKEN}" \
-d '{
 "hosted-zone-name": "auto",
 "targets": "example.com",
 "scan-name": "Scan via Hosted Zone",
 "rate": "1000"
}'
```

## Create a scan task for an Explorer group

To distribute a scan task among a **group of Explorers**, use the "explorer-group-id" key with the Explorer group's UUID. The group will automatically select an available Explorer to run the task.

```
curl -X PUT "https://console.runzero.com/api/v1.0/org/sites/${YOUR_SITE_ID}/scan" \
-H "Content-Type: application/json" \
-H "Authorization: Bearer ${RUNZERO_ORG_TOKEN}" \
-d '{
 "explorer-group-id": "<YOUR-EXPLORER-GROUP-ID>",
 "targets": "10.0.0.0/8",
 "scan-name": "Scan via Explorer group",
 "rate": "1000"
}'
```

## Additional notes

- The payloads above are **minimal working examples**.
- You can include any additional valid parameters, such as:
  - "scan-template"
  - "tcp-ports"
  - "udp-ports"
  - "max-hosts"
  - "max-duration"
  - "probes"
- Always validate your payload against your organization's scan configuration and API schema.

## Export API usage

The **Export API** provides read-only access to your organization's data, such as assets, sites, and scans. It uses an **Export Token** (prefixed with `XT`) and is ideal for integrations or reporting that don't modify data.

## Set environment variables

```
export RUNZERO_EXPORT_TOKEN="XT-YOUR-EXPORT-TOKEN-HERE"
```

## Base endpoint

All export endpoints use the `/api/v1.0/export/org/` path.

Example:

```
https://console.runzero.com/api/v1.0/export/org/assets.json
```

## Export all assets (JSON)

Retrieve all assets in JSON format:

```
curl -X GET "https://console.runzero.com/api/v1.0/export/org/assets.json" \
 -H "Authorization: Bearer ${RUNZERO_EXPORT_TOKEN}" \
 -o assets.json
```

## Export all assets (CSV)

Retrieve the same data in CSV format:

```
curl -X GET "https://console.runzero.com/api/v1.0/export/org/assets.csv" \
 -H "Authorization: Bearer ${RUNZERO_EXPORT_TOKEN}" \
 -o assets.csv
```

## Export sites list

List all sites in your organization:

```
curl -X GET "https://console.runzero.com/api/v1.0/export/org/sites.json" \
 -H "Authorization: Bearer ${RUNZERO_EXPORT_TOKEN}" \
 -o sites.json
```

## Export scans

Retrieve a list of completed scans:

```
curl -X GET "https://console.runzero.com/api/v1.0/export/org/scans.json" \
 -H "Authorization: Bearer ${RUNZERO_EXPORT_TOKEN}" \
 -o scans.json
```

## Notes on export tokens

- **Export tokens are read-only** — they cannot modify organization data.
- They can be safely scoped to limit data access.
- Output may be large for organizations with many assets — consider filtering.
- You can use the `search` query parameter to filter exports, for example:

```
curl -X GET "https://console.runzero.com/api/v1.0/export/org/assets.json?search=alive:true" \
-H "Authorization: Bearer ${RUNZERO_EXPORT_TOKEN}"
```

## Additional documentation

Please see the [Swagger documentation](#) and [runZero OpenAPI specification](#) for details on the individual API calls.

## API clients

An API client is a credential made of a client ID and a client secret, exchanged for a short-lived access token through OAuth2. Unlike the [API keys and tokens](#), which are bearer strings with fixed access, an API client has an explicit grant: a set of [permissions](#) and the organizations they apply to.

The console URLs in this article use `console.runzero.com` as an example. Use the URL for your runZero deployment instead.

There are two kinds, and the difference matters:

|                                       | User-bound                                            | Account-bound                             |
|---------------------------------------|-------------------------------------------------------|-------------------------------------------|
| Created from                          | <a href="#">My API clients</a>                        | <a href="#">Account API clients</a>       |
| Who can create one                    | Any user with access to grant                         | Superusers                                |
| Maximum access                        | The creator's current access                          | Administrator access to the account       |
| When the creator's access changes     | The credential narrows or recovers to match           | Unaffected                                |
| When the creator is locked or removed | The credential stops working                          | Unaffected                                |
| Use it for                            | Automation on behalf of a person, AI clients, scripts | Service integrations, CI, unattended jobs |

The rule of thumb: if the credential is doing something on your behalf, make it user-bound so it dies with your access. If it belongs to a system rather than a person, make it account-bound so it does not break when someone changes jobs.

Both kinds require a Platform license.

## Grants

A grant has two halves, and both are enforced on every request.

**Permissions.** A verb on each area, as described in [How permissions work](#). A grant that does not include an area denies that area.

**Organizations.** Either a list of specific organizations, or every organization. Selecting every organization is an explicit choice. A grant with an empty organization list grants nothing rather than everything.

One grant applies one set of permissions uniformly across the organizations it names. To give a credential different permissions in different organizations, create separate API clients.

## Expiration

Every API client has an expiration, chosen when it is created. The default is 30 days, with presets for 90 days and a year, a maximum of 10 years, and an option for no expiration at all. Prefer a real expiration: a credential nobody has to renew is a credential nobody reviews.

An expired credential stops working rather than being deleted. It no longer exchanges its secret for an access token, and it stays in the list so you can see what expired and when. Access tokens are also capped by the credential's expiration, so a token issued shortly before a credential expires dies with it rather than outliving it.

Expiration is one of the two fields that can be changed after creation, so a credential that has expired, or is about to, can be given a new expiration without reissuing its secret or its grant. This is the way to renew a working integration. It is not a way to widen one, because permissions and organizations stay fixed.

Credentials created by approving an [MCP connection](#) are the exception. They expire 90 days after approval, and while their expiration can be shortened, it can never be moved past that original deadline or removed. Renewing one means reconnecting the application, which creates a new credential. Reviving an expired one counts against the limit of 40 active MCP connections per account.

## Creating a user-bound API client

Go to **My API clients** at [console.runzero.com/settings/api-clients](https://console.runzero.com/settings/api-clients), which lists only the clients you created, and click to register a new one.

1. Enter an **API client name** and choose an **expiration**. See [Expiration](#).
2. Choose the **organizations**. You can pick specific organizations or every organization you can reach. Choosing every organization records the organizations you can reach at that moment; organizations you gain access to later are not added automatically.
3. Optionally choose a **role**, either as a lock or as a starting point. See [Role locks](#).
4. Set the **Access** permissions. The form only offers permissions your current access allows. If you are an administrator in one organization and a viewer in another, selecting just the first organization lets you grant administrator permissions; selecting both clamps the grant to what the two have in common.
5. Review the summary and save. **Copy the client secret before closing the panel.** It is shown once and cannot be retrieved later.

## The creator's access is a continuing ceiling

A user-bound credential is checked against its creator every time it authenticates, not just when it was created:

- If you lose access, the credential loses it too, immediately.
- If you regain access, the credential recovers it, up to the permissions it was issued with.
- The credential can never gain access above its issued grant, even if you are later promoted.

- If your account is locked or removed, the credential stops working.

This is the property that makes user-bound credentials safe to hand to an AI client or a script: revoking the person revokes everything they issued.

## Creating an account-bound API client

Superusers manage account-bound clients from **API clients** under **Account** in the left navigator, at [console.runzero.com/account/api/clients](https://console.runzero.com/account/api/clients).

The form is the same, with two differences. The ceiling is administrator access across every organization in the account rather than your own access, and selecting every organization stays dynamic, so organizations added to the account later are included.

An account-bound credential keeps the access it was issued regardless of what happens to the person who created it. That is the point, and also the risk: it needs to be tracked and rotated deliberately, because nothing else will retire it.

## Role locks

When you choose a role while creating an API client, you pick one of two behaviors:

- **Lock this credential to the role.** The credential's permissions are re-derived from the role every time it authenticates. Editing the role changes what the credential can do. This keeps a fleet of credentials consistent, and lets you revoke a class of access by editing one role. A user-bound locked credential is still capped by its issued grant and by your current access, so a role gaining permissions later does not widen it beyond those.
- **Use the role as a starting point.** The role's permissions are copied into the form so you can adjust them. The result is a fixed grant, and later changes to the role do not apply.

If the locked role is deleted, the credential cannot authenticate until a role with that name exists again. The API clients list shows locked credentials whose role is disabled.

## Managing existing clients

Once a credential exists, only its name and its [expiration](#) can be changed. Its organizations, binding, permissions, and role lock are fixed at creation. To change any of those, create a new client and retire the old one.

The secret can be rotated without changing anything else about the credential, which is the right response to a leaked secret. Rotation is done by the credential itself, authenticated with the secret being replaced, and invalidates the old secret immediately:

```
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" \
 --data-urlencode "client_id=<CLIENT_ID>" \
 --data-urlencode "client_secret=<CURRENT_SECRET>" \
 https://console.runzero.com/api/v1.0/account/api/rotate
```

The response includes the new secret. Rotation needs the current secret, so if it has been lost, the only option is to delete the client and create a new one. Rotation is also refused on an expired credential; give it a new [expiration](#) first, then rotate.

Creation, metadata changes, deletion, and secret rotation are all recorded in the [audit log](#), with the organizations and permissions the credential was granted. Secrets are never recorded.

## Clients created before grants existed

API clients created before explicit grants were introduced show as a legacy grant in the list. They keep the access they have always had, which is unrestricted on the REST API and account-wide on MCP. They are not migrated automatically. Replace them with explicitly scoped clients when you can.

## Using an API client

Exchange the client ID and secret for an access token, then use the token as a bearer token:

```
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" \
 --data-urlencode "grant_type=client_credentials" \
 --data-urlencode "client_id=<CLIENT_ID>" \
 --data-urlencode "client_secret=<CLIENT_SECRET>" \
 https://console.runzero.com/api/v1.0/account/api/token
```

The token is accepted anywhere an account API token is, subject to the grant. A call the grant does not permit is refused, and a call against an organization the grant does not name is refused whether or not the organization was requested explicitly.

API clients can also be presented to the [MCP endpoint](#), where the same grant decides which tools are offered.

The [IP address allowlist](#) and the [rate limits](#) apply to API clients as they do to API keys.

## Credentials created by AI clients

When you connect an AI client to runZero through [MCP](#), approving the connection creates an API client for you. Those credentials appear in the same lists as any other, are user-bound unless a superuser chose otherwise, and can be revoked the same way. They expire 90 days after approval, and that deadline cannot be moved. See [Expiration](#) and [Connecting through OAuth](#).



## Using the CLI

The runZero Command Line Interface (CLI) provides various utility functions. For licensed users, it also allows standalone network scanning.

### Scanner

The `scan` command has the same options as the runZero Explorer, and similar performance characteristics. The output file named `scan.runzero.gz` can be uploaded to the runZero Console through the Inventory *Import* menu. This

The CLI scanner works best with root privileges on Linux/macOS and Administrator privileges on Windows. Although the CLI will function without privileged access, many probe types will be unavailable. The `sudo` command can be used to run the CLI as root on Linux and macOS, while the tool is best run from an elevated command shell on Windows. On the Windows platform, the runZero CLI will look for an existing `npcap` installation and try to install it if the software is not found. This behavior can be disabled with the `--nopcode` flag.

On Windows hosts with Intel Advanced Matrix Extensions (AMX) enabled, the CLI scanner may become unstable or crash. This can affect physical hardware and virtual machines. Disable AMX at the BIOS or host level before running the scanner. For Windows guests on Hyper-V, enabling **Processor Compatibility** can mitigate this issue.

EDR, antivirus, and endpoint protection software on Windows can also interfere with scanner operations, including packet capture and process execution. If scans fail or terminate unexpectedly, add allowlist exceptions for runZero and `npcap`/`winpcap` components, or use monitor-only policy modes while validating scans. In some environments, endpoint protections may need to be temporarily disabled on the host for stable operation.



Some components of the application still reference the name "Rumble" for backwards compatibility. The documentation will be updated as these are changed.

The runZero CLI defaults to a semi-interactive terminal interface that writes multiple output files to a directory. The default directory name is `runzero-[current-date]`. To switch to plain text output, use the `--text` option. To skip artifact generation and only produce the raw JSON output file, use the flags `--text -o disable --output-raw scan.runzero`.

Input can be provided as arguments on the command-line or by specifying an input file using the `--input` (or `-i`) parameter. Input can consist of specific IPv4 addresses or IPv4 CIDRs. Supported formats include:

- `10.0.0.1`
- `10.0.0.0/24`
- `10.0.0.0/255.255.255.0`
- `10.0.0.1-10.0.0.255`
- `example.com`

- `example.com/24`

For hostnames, each IPv4 address in the response will be expanded with the optional mask.

The example below downloads and runs the CLI on a Linux x86\_64 host. This URL will be different for your installation. The current download links for your organization are available from the [CLI](#) page of the runZero Console. If you are using a self-hosted console or a region other than our US-based SaaS, you can find the download link under the Deploy navigation menu.

```
$ wget https://console.runzero.com/download/cli/[unique-link]/runzero-cli-linux-amd64.bin
$ chmod +x runzero-cli-linux-amd64.bin
$ sudo runzero scan 192.168.0.0/24 -o output-dir
```

Please note that the hexadecimal values in the download URL are specific for your account and organization.

## Performance

The default speed of runZero scans is limited to 1,000 packets per second with a single pass. This setting works great for reliable wired networks without stateful firewalls between the scanning system and the destination networks. This rate can be changed via the `--rate` (or `-r`) option, with a reasonable maximum being 10000 for most networks. On slow, unreliable networks, a rate of 300 with `--passes` set to 3 may provide better results.

A second parameter, `--max-host-rate` limits how many packets are sent per second to each individual host. This defaults to 40, which is low, but may be necessary when scanning low-power embedded devices. In cases where a small number of hosts (or a single host) should be scanned quickly, the `--max-host-rate` parameter can be increased to match the `--rate`.

## Examples

The following example demonstrates a scan of 65,535 TCP ports on all hosts in the 192.168.0.0/24 subnet running at 10,000 packets per second:

```
$ sudo runzero scan 192.168.0.0/24 -r 10000 --tcp-ports 1-65535 -o output-dir
```

The following example demonstrates a scan on all hosts in the 192.168.0.0/24 and 10.0.0.0/24 subnets running at 5,000 packets per second:

```
$ sudo runzero scan 192.168.0.0/24 10.0.0.0/24 -r 5000 -o output-dir
```

The following example demonstrates a scan on all hosts in the 192.168.0.0/24 and 10.0.0.0/8 subnets running at a max host rate of 20 packets per host:

```
$ sudo runzero scan 192.168.0.0/24 10.0.0.0/8 --max-host-rate 20 -o output-dir
```

The following example demonstrates a scan on all hosts in the 192.168.0.0/24 subnet and the domain "example.com" running at 7,500 packets per second:

```
$ sudo runzero scan 192.168.0.0/24 example.com -r 7,500 -o output dir
```

The following example demonstrates a scan on all hosts in the 10.0.0.0/8 subnet and a particular ASN4 value at a default speed of 1,000 packets per second.

```
$ sudo runzero scan 10.0.0.0/8 asn4:[ID] -o output dir
```

The following example demonstrates a scan on all hosts in the 192.168.0.0/24 subnet with the max TTL set at 128 and a scan rate of 2,500 packets per second:

```
$ sudo runzero scan 192.168.0.0/24 -r 2,500 --max-ttl 128 -o output-dir
```

The following example demonstrates a scan based on an input file:

```
$ sudo runzero scan -i /path/to/input-file.txt -o output dir
```

Here is an example input file:

```
www.example.com
192.168.0.0/24
```

## Automatic web screenshots

The `--screenshots` option defaults to `true` and tells runZero to obtain a screenshot of all web services identified during the scan. This feature depends on the system running the Explorer having a local installation of the Google Chrome or Chromium browsers. The acquired screenshots will be reported as a base64 string, stored in the `"screenshot.image"` field of the containing service scan result.

To disable automatic web screenshots, set the `--screenshots` option to `false` (`--screenshots=false`).

## Scanner defaults

### Standard ports scanned

|      |      |      |      |      |      |      |      |      |      |      |
|------|------|------|------|------|------|------|------|------|------|------|
| 1    | 7    | 9    | 13   | 17   | 19   | 21   | 22   | 23   | 25   | 37   |
| 42   | 43   | 49   | 53   | 69   | 70   | 79   | 80   | 81   | 82   | 83   |
| 84   | 85   | 88   | 102  | 105  | 109  | 110  | 111  | 113  | 119  | 123  |
| 135  | 137  | 139  | 143  | 161  | 179  | 222  | 264  | 280  | 384  | 389  |
| 402  | 407  | 442  | 443  | 444  | 445  | 465  | 500  | 502  | 512  | 513  |
| 515  | 523  | 524  | 540  | 541  | 548  | 554  | 587  | 617  | 623  | 631  |
| 636  | 664  | 689  | 705  | 717  | 743  | 771  | 783  | 789  | 830  | 873  |
| 888  | 902  | 903  | 910  | 912  | 921  | 949  | 990  | 993  | 995  | 998  |
| 1000 | 1024 | 1030 | 1035 | 1080 | 1083 | 1089 | 1090 | 1091 | 1098 | 1099 |
| 1100 | 1101 | 1102 | 1103 | 1128 | 1129 | 1153 | 1158 | 1199 | 1211 | 1220 |
| 1234 | 1241 | 1260 | 1270 | 1300 | 1311 | 1337 | 1352 | 1433 | 1434 | 1440 |
| 1443 | 1468 | 1494 | 1502 | 1514 | 1521 | 1530 | 1533 | 1581 | 1582 | 1583 |
| 1604 | 1610 | 1611 | 1723 | 1755 | 1801 | 1811 | 1830 | 1883 | 1900 | 1911 |
| 2000 | 2002 | 2021 | 2022 | 2023 | 2024 | 2031 | 2049 | 2068 | 2074 | 2082 |
| 2083 | 2100 | 2103 | 2105 | 2121 | 2181 | 2199 | 2207 | 2222 | 2224 | 2323 |
| 2362 | 2375 | 2376 | 2379 | 2380 | 2381 | 2404 | 2443 | 2455 | 2525 | 2533 |

|       |       |       |       |       |       |       |       |       |       |       |       |
|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| 2598  | 2601  | 2602  | 2604  | 2638  | 2809  | 2947  | 2967  | 3000  | 3001  | 3003  | 3033  |
| 3037  | 3050  | 3052  | 3057  | 3071  | 3080  | 3083  | 3128  | 3142  | 3200  | 3210  | 3217  |
| 3220  | 3260  | 3268  | 3269  | 3273  | 3299  | 3300  | 3306  | 3311  | 3312  | 3351  | 3389  |
| 3460  | 3478  | 3500  | 3502  | 3628  | 3632  | 3690  | 3780  | 3790  | 3817  | 3868  | 3871  |
| 3872  | 3900  | 4000  | 4092  | 4322  | 4343  | 4353  | 4365  | 4366  | 4368  | 4369  | 4406  |
| 4433  | 4443  | 4444  | 4445  | 4567  | 4659  | 4679  | 4730  | 4786  | 4840  | 4843  | 4848  |
| 4900  | 4949  | 4950  | 4987  | 5000  | 5001  | 5003  | 5006  | 5007  | 5022  | 5037  | 5038  |
| 5040  | 5044  | 5050  | 5051  | 5060  | 5061  | 5093  | 5094  | 5168  | 5222  | 5247  | 5250  |
| 5275  | 5347  | 5351  | 5353  | 5355  | 5392  | 5400  | 5405  | 5432  | 5433  | 5466  | 5498  |
| 5520  | 5521  | 5554  | 5555  | 5560  | 5580  | 5601  | 5631  | 5632  | 5666  | 5671  | 5672  |
| 5678  | 5683  | 5800  | 5814  | 5900  | 5901  | 5902  | 5903  | 5904  | 5905  | 5906  | 5907  |
| 5908  | 5909  | 5910  | 5911  | 5920  | 5938  | 5984  | 5985  | 5986  | 5988  | 5989  | 6000  |
| 6001  | 6002  | 6050  | 6060  | 6070  | 6080  | 6082  | 6101  | 6106  | 6112  | 6161  | 6262  |
| 6274  | 6277  | 6333  | 6379  | 6405  | 6432  | 6443  | 6481  | 6502  | 6503  | 6504  | 6514  |
| 6542  | 6556  | 6660  | 6661  | 6667  | 6905  | 6988  | 7000  | 7001  | 7002  | 7021  | 7070  |
| 7071  | 7077  | 7080  | 7100  | 7144  | 7181  | 7210  | 7373  | 7443  | 7444  | 7474  | 7510  |
| 7547  | 7579  | 7580  | 7676  | 7700  | 7770  | 7777  | 7778  | 7780  | 7787  | 7800  | 7801  |
| 7860  | 7879  | 7902  | 7990  | 7999  | 8000  | 8001  | 8002  | 8003  | 8004  | 8006  | 8008  |
| 8009  | 8010  | 8012  | 8014  | 8020  | 8023  | 8028  | 8030  | 8080  | 8081  | 8082  | 8083  |
| 8084  | 8085  | 8086  | 8087  | 8088  | 8089  | 8090  | 8095  | 8098  | 8099  | 8100  | 8123  |
| 8127  | 8160  | 8161  | 8172  | 8180  | 8181  | 8182  | 8188  | 8193  | 8200  | 8205  | 8222  |
| 8265  | 8300  | 8303  | 8333  | 8400  | 8428  | 8443  | 8444  | 8445  | 8446  | 8447  | 8448  |
| 8449  | 8471  | 8488  | 8500  | 8503  | 8530  | 8531  | 8545  | 8649  | 8686  | 8787  | 8800  |
| 8812  | 8833  | 8834  | 8850  | 8871  | 8880  | 8883  | 8888  | 8889  | 8890  | 8899  | 8900  |
| 8901  | 8902  | 8903  | 8983  | 9000  | 9001  | 9002  | 9042  | 9060  | 9080  | 9081  | 9084  |
| 9090  | 9091  | 9092  | 9099  | 9100  | 9111  | 9152  | 9160  | 9200  | 9300  | 9380  | 9390  |
| 9391  | 9401  | 9418  | 9428  | 9440  | 9443  | 9444  | 9471  | 9495  | 9524  | 9527  | 9530  |
| 9593  | 9594  | 9595  | 9600  | 9809  | 9855  | 9997  | 9998  | 9999  | 10000 | 10001 | 10008 |
| 10050 | 10051 | 10080 | 10098 | 10162 | 10202 | 10203 | 10250 | 10255 | 10257 | 10259 | 10443 |
| 10616 | 10628 | 10800 | 11000 | 11099 | 11211 | 11234 | 11333 | 11434 | 11443 | 12174 | 12203 |
| 12221 | 12345 | 12379 | 12397 | 12401 | 13364 | 13400 | 13500 | 13778 | 13838 | 14330 | 15200 |
| 15671 | 15672 | 16102 | 16443 | 16992 | 16993 | 17185 | 17200 | 17472 | 17775 | 17776 | 17777 |
| 17778 | 17781 | 17782 | 17783 | 17784 | 17790 | 17791 | 17798 | 18245 | 18264 | 18789 | 18881 |
| 19000 | 19300 | 19530 | 19810 | 19888 | 20000 | 20010 | 20031 | 20034 | 20101 | 20111 | 20171 |
| 20222 | 20256 | 20293 | 21047 | 22222 | 23472 | 23791 | 23943 | 24442 | 25000 | 25025 | 25565 |
| 25672 | 26000 | 26122 | 27000 | 27017 | 27018 | 27019 | 27080 | 27888 | 28017 | 28222 | 28784 |
| 29418 | 30000 | 31001 | 31099 | 32400 | 32764 | 32843 | 32844 | 32845 | 32913 | 33060 | 34205 |
| 34443 | 34962 | 34963 | 34964 | 37718 | 37777 | 37890 | 37891 | 37892 | 38008 | 38010 | 38080 |
| 38102 | 38292 | 40007 | 40317 | 41025 | 41080 | 41523 | 41524 | 44334 | 44343 | 44818 | 45230 |
| 46823 | 46824 | 47001 | 47002 | 47290 | 48898 | 48899 | 49152 | 50000 | 50013 | 50021 | 50051 |
| 50070 | 50090 | 50121 | 50505 | 51443 | 52302 | 52311 | 53282 | 54321 | 54921 | 54922 | 54923 |
| 55553 | 55580 | 57772 | 61614 | 61616 | 62078 | 62514 | 65002 | 65535 |       |       |       |

## Scan outputs

The runZero CLI generates a directory of output files by default. This directory includes the following items.

- `scan.runzero.gz`: The raw scan data compressed via gzip, this can be imported or reprocessed via `--import`

- `assets.jsonl`: The new optimized format for correlated, fingerprinted assets.
- `nmap.xml`: A Nmap XML compatible data file that can be imported into various security tools.
- `urls.txt`: A list of discovered web services in URL format.
- `protocols.csv`: A list of protocols with their ports and URLs.
- `assets.html`: A rudimentary HTML report with screenshots.
- `screenshots`: A directory of raw screenshot images, headers in JSON format, and HTML bodies.
- Various lists including `addresses.txt`, `addresses_all.txt`, `hostnames.txt`, and `domains.txt`

## Raw Scan Data

The runZero CLI raw data is stored in a file named `scan.runzero.gz` within the output directory. This file contains JSONL-formatted records. An example ARP response record is shown below.

```
{
 "type": "result",
 "host": "192.168.0.1",
 "port": "0",
 "proto": "arp",
 "probe": "arp",
 "name": "192.168.0.1",
 "info": {
 "mac": "f0:9f:c2:11:1a:13",
 "macDateAdded": "2014-12-17",
 "macVendor": "Ubiquiti Networks Inc."
 },
 "ts": 1551584126253853200
}
```

The `info` field is a JSON map of strings to strings. Multiple values are encoded using the `tab` character (0x09), which are otherwise escaped as `\t` (along with `\r` and `\n` for carriage return and line feed bytes and `\x00` for null bytes). runZero scans may return more than one record of the same type for the same host if multiple responses were received.

In addition to the `result` type, there are also records for `status` messages, `stats`, and an initial `config` type that contains the scan parameters.

## runZero Command Line Interface (CLI)

The runZero CLI supports a wide range of commands and options. As well as offline scans, you can run third-party integrations and custom scripts on the command line.

The `--help` output provides basic documentation on the available options.

Most commands below accept the following global flags:

```
--verbose
 Display verbose output.
```

`--very-verbose`  
Display very verbose output.

## Import Censys data files

```
runZero censys [avro files] [targets] [flags]
```

### Flags:

`-i, --input-targets string`  
Read search targets from the specified input file

`--output-raw string`  
Write results to the specified output file

## Import Censys data from a local database

```
runZero censys-db [path-to-database] [targets] [flags]
```

### Flags:

`-i, --input-targets string`  
Read search targets from the specified input file

`--output-raw string`  
Write results to the specified output file

## Convert Censys Avro files into a local database

```
runZero censys-db-convert [path-to-avro-directory] [path-to-db-directory] [flags]
```

### Flags:

`--shard-max-records int`  
Specify the maximum record count per shard (default 500000)

`--test-mode`  
Replace keys in the source data with random values

`--test-multiplier int`  
Multiply the source data by a given factor (default 1)

## Serves a Censys database from a web server

```
runZero censys-db-server [path-to-database] [flags]
```

**Flags:**

--port int  
The TCP port for the web server to listen on (default 55555)

**Generate the autocompletion script for the specified shell**

```
runZero completion [command]
```

**Available commands:**

bash: Generate the autocompletion script for bash

fish: Generate the autocompletion script for fish

powershell: Generate the autocompletion script for powershell

zsh: Generate the autocompletion script for zsh

**Help about any command**

```
runZero help [command] [flags]
```

**Display license information**

```
runZero license [flags]
```

**Start a runZero active scan, passive discovery, or integration task**

```
runZero scan <options> [targets] [flags]
```

**Flags:**

--api-key string  
Specify the runZero API key

--api-no-verify  
Disable TLS verification for API communication

--api-url string  
Specify the runZero API server hostname (default  
"<https://console.runzero.com/api/v1.0>")

--arp-fast  
Enables fast mode by ARP scanning at the scan rate vs host rate

```

--atg-ports string
 The destination ports for ATG probes (default "10001")

--aws-instances-access-key string
 The access key for the AWS account

--aws-instances-assume-role-name string
 The role to assume for all accounts in the organization for cross-account access

--aws-instances-delete-stale
 Automatically delete stale AWS assets

--aws-instances-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--aws-instances-include-stopped
 Include assets that are not currently running

--aws-instances-regions string
 The comma-separated list of regions for the AWS account

--aws-instances-secret-access-key string
 The secret access key for the AWS account

--aws-instances-service-options string
 The comma-separated list of services to sync data from
 (defaults,ec2,elb,elbv2,rds,lambda) (default "defaults")

--aws-instances-site-per-account
 Automatically create a new site per account

--aws-instances-site-per-vpc
 Automatically create a new site per VPC

--aws-instances-token string
 The session token for the AWS account

--azure-client-id string
 The application ID (client ID) for the Azure account

--azure-client-secret string
 The client secret for the Azure account

--azure-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--azure-multi-subscription
 Access all subscriptions in the directory (tenant) for the Azure account

--azure-password string
 The password for the Azure account

--azure-service-options string
 The comma-separated list of services to sync data from
 (defaults,vm,vmss,azsql,cosmos,lb,functionapp) (default "defaults")

--azure-site-per-subscription
 Automatically create a new site per subscription

```



--azure-subscription-id string  
The subscription ID for the Azure account

--azure-tenant-id string  
The directory ID (tenant ID) for the Azure account

--azure-username string  
The username for the Azure account

--azuread-client-id string  
The application ID (client ID) for the Azure account

--azuread-client-secret string  
The client secret for the Azure account

--azuread-exclude-unknown  
Exclude assets that cannot be merged into an existing asset

--azuread-filter string  
An optional filter. Only import devices that match this filter.

--azuread-include-inactive  
Include assets that are marked as inactive in the AzureAD account

--azuread-password string  
The password for the AzureAD account

--azuread-service-options string  
The comma-separated list of services to sync data from (defaults,dev,user,group)  
(default "defaults")

--azuread-tenant-id string  
The directory ID (tenant ID) for the Azure account

--azuread-username string  
The username for the AzureAD account

--bacnet-ports string  
The destination ports for BACnet probes (default "46808,47808,48808")

-b, --baseline string  
Use the specified file as an asset baseline for tracking

--bedrock-ports string  
The destination ports for Bedrock probes (default "19132")

--bjnp-printer-ports string  
The UDP ports to send Canon printer discovery requests (default "8611")

--bjnp-scanner-ports string  
The UDP ports to send Canon scanner discovery requests (default "8612")

--censys-api-url string  
The API endpoint to use for Censys Search (default "<https://search.censys.io>")

--censys-client-id string  
The Client ID to use for Censys Search authentication

--censys-client-secret string  
The Client Secret to use for Censys Search authentication

```

--censys-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--censys-mode string
 The search mode (assets or query). The assets option queries the scan targets (default
 "assets")

--censys-query string
 The search string to use in query mode

--coap-port uint
 The destination port for CoAP probes (default 5683)

-c, --config string
 Specify the config file name to load. It must be in a JSON format

--cpu string
 Write a cpu profile after the scan completes

--crestron-port uint
 The destination port for Crestron probes (default 41794)

--crowdstrike-api-url string
 The URL used for the CrowdStrike account's API access

--crowdstrike-client-id string
 The client ID for the CrowdStrike account

--crowdstrike-client-secret string
 The client secret for the CrowdStrike account

--crowdstrike-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--crowdstrike-filter string
 An optional Falcon Query Language (FQL) filter for imported assets

--crowdstrike-fingerprint-only
 Import vulnerabilites for fingerprinting purposes only

--crowdstrike-risks string
 Minimum risk of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")

--crowdstrike-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

--custom-integration-entry-function-name string
 Function to call that will return ImportAssets (default "main")

--custom-integration-id string
 UUID of custom integration from console

--custom-integration-script-args string
 Arguments for the script

--custom-integration-script-kwargs string
 Keyword arguments for the script

```

```

--custom-integration-script-source string
 Source code of the custom integration script

--dahua-dhip-ports string
 The destination ports for Dahua DHIP discovery probes (default "37810")

--defender365-client-id string
 The application ID (client ID) for the Azure account

--defender365-client-secret string
 The client secret for the Azure account

--defender365-exclude-notonboarded
 Exclude assets that have not been fully onboarded

--defender365-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--defender365-filter string
 Exclude assets using a Graph API $filter string

--defender365-include-inactive
 Include assets that have stopped reporting to the Microsoft 365 Defender service

--defender365-tenant-id string
 The directory ID (tenant ID) for the Azure account

--disabled-probes string
 Specifically exclude these probes, comma-delimited

--dnp3-address-probe-timeout int
 Time limit (in seconds) for DNP address discovery. (default 30)

--dnp3-banner-address-discovery string
 One of 'require', 'prefer', or 'ignore'. (default "ignore")

--dnp3-destination-address-discovery-range string
 A numeric range of addresses to attempt to discover. (default "0-32")

--dnp3-explorer-address int
 Source DNP3 address for the explorer. (default -1)

--dns-disable-google-myaddr
 Disables resolution of upstream DNS via Google myaddr service

--dns-disable-meraki-detection
 Disables detection of Meraki DNS interception

--dns-port uint
 The destination port for DNS probes (default 53)

--dns-resolve-name string
 The target hostname for DNS queries ('off' to disable) (default "www.google.com")

--dns-trace-domain string
 The subdomain to use for trace requests ('off' to disable) (default
 "helper.rumble.network")

--dtls-ports string
 The destination ports for DTLS probes (default "443,3391,4433,5246,5349,5684")

```

```

--echo-report-errors
 Report errors from intermediate in-scope hosts

--ethernetip-cip-enumeration-method string
 (BETA) Set this to the preferred CIP enumeration method. (default "none")

--ethernetip-udp-ports string
 The destination ports for EtherNet/IP UDP probes (default "44818")

--exclude string
 Specify scan exclusions

--excludefile string
 Read exclusions from an input file

--filter-base64
 Filter base64-encoded fields

-f, --fingerprints string
 Use the specified directory as an alternate fingerprint database

--fingerprints-debug
 Enable debug output for the fingerprint processor

--fins-port uint
 The destination port for FINS probes (default 9600)

--gcp-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--gcp-key-path string
 Path to GCP service account key file

--gcp-service-options string
 The comma-separated list of services to sync data from (defaults,vm,lb,cloudsql)
 (default "defaults")

--gcp-site-per-project
 Automatically create a new site per project

--genudp-payload-base64 string
 The generic udp payload as base64

--genudp-payload-hex string
 The generic udp payload as hexadecimal

--genudp-payload-text string
 The generic udp payload as plain text

--genudp-ports string
 The destination ports for the generic udp probe

--googleworkspace-client-email string
 The email address of the service account

--googleworkspace-client-id string
 The ID of the service account

--googleworkspace-customer-id string
 An optional customer ID for multi-tenant environments (default "my_customer")

```

```

--googleworkspace-delegate string
 The email address of an admin account with directory access
--googleworkspace-exclude-unknown
 Exclude assets that cannot be merged into an existing asset
--googleworkspace-private-key string
 The PEM encoded private key
--googleworkspace-private-key-id string
 The ID of the private key
--googleworkspace-project-id string
 The project ID of the service account
--googleworkspace-service-options string
 The comma-separated list of services to sync data from
 (defaults,chromeos,mobile,endpoint,user,group) (default "defaults")
--goroutines string
 Write a goroutine dump after the scan completes
--heap string
 Write a heap profile after the scan completes
--hiddiscoveryd-port uint
 The destination port for HID discoveryd probes (default 4070)
--host-ping
 Only scan hosts that respond to a ping scan using the host-ping settings
--host-ping-max-attempts int
 Set the maximum number of attempts for each probe (default 2)
--host-ping-max-ttl int
 Set the default TTL on host-ping probe packets (default 255)
--host-ping-passes int
 Set the number of passes for the host-ping phase (default 1)
--host-ping-probes string
 Launch a subset of the probes for the host-ping, comma-delimited (default
 "arp,echo,syn,connect,netbios,snmp,ntp,sunrpc,ike,openvpn,mdns")
--host-ping-tcp-ports string
 The list of TCP ports to host-ping using the syn and connect probes (default
 "22,80,135,179,443,3389,5040,7547,62078")
--host-ping-tos int
 Set the default ToS on host-ping probe packets
--igel-discovery-ports string
 The destination ports for IGEL discovery probes (default "30005")
--ike-port uint
 The destination port for IKE probes (default 500)
-I, --import stringArray
 Import existing scan data from the specified input files ('scan.rumble' format)

```

```

--import-pcap stringArray
 Import pcap packet capture from the specified input files ('.pcap' or '.pcapng' format)
-i, --input-targets string
 Read scan targets from the specified input file
--insightvm-api-url string
 The URL used for the InsightVM account's API access
--insightvm-exclude-unknown
 Exclude assets that cannot be merged into an existing asset
--insightvm-fingerprint-only
 Import vulnerabilities for fingerprinting purposes only
--insightvm-insecure
 Set this to true to authenticate to untrusted endpoints (self-signed or no IP SAN) (default
 true)
--insightvm-password string
 The password for the InsightVM account
--insightvm-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")
--insightvm-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")
--insightvm-thumbprints string
 A set of IP=SHA256:B64HASH pairs to trust for authentication
--insightvm-username string
 The username for the InsightVM account
--insightvmcloud-host string
 The host for the InsightVM Cloud API
--insightvmcloud-api-key string
 The API key for the InsightVM Cloud account
--insightvmcloud-exclude-unknown
 Exclude assets that cannot be merged into an existing asset
--insightvmcloud-exclude-tags
 Do not collect asset tags
--insightvmcloud-exclude-vulns-import
 Do not collect asset vulnerabilities
--insightvmcloud-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")
--insightvmcloud-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

```

```

--intune-client-id string
 The application ID (client ID) for the Azure account
--intune-client-secret string
 The client secret for the Azure account
--intune-exclude-unknown
 Exclude assets that cannot be merged into an existing asset
--intune-filter string
 An optional filter. Only import devices that match this filter.
--intune-password string
 The password for the Intune account
--intune-tenant-id string
 The directory ID (tenant ID) for the Azure account
--intune-username string
 The username for the Intune account
--ipmi-port uint
 The destination port for IPMI probes (default 623)
--ipp-browse-port uint
 The destination port for IPP-browse probes (default 631)
--iscsi-discover
 Enable iSCSI target discovery probe (default true)
--kerberos-port uint
 The destination port for kerberos probes (default 88)
--knxnet-ports string
 The destination ports for knxnet probes (default "3671")
--l2t-port uint
 The destination port for L2T probes (default 2228)
--l2tp-ports string
 The destination ports for L2TP probes (default "1701")
--lantronix-port uint
 The destination port for Lantronix probes (default 30718)
--layer2-add-targets
 Set this false to skip scanning discovered targets (default true)
--layer2-force
 Set this to true to force discovery even without local targets
--layer2-max-retries uint
 The desired number of retries (default 3)
--layer2-tcp-ports string
 The TCP ports to ping for local device discovery (default
 "22,80,135,179,443,3389,5040,7547,62078")
--layer2-udp-trace-port uint
 The UDP port number to use for UDP trace requests (default 9)

```

```

--ldap-base-dn string
 The base DN used for LDAP searches

--ldap-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--ldap-insecure
 Set this to true to authenticate to untrusted endpoints (self-signed or no IP SAN)

--ldap-legacy-tls
 Set this to true to authenticate over legacy TLS versions (< 1.2)

--ldap-password string
 The password for the LDAP account

--ldap-service-options string
 The comma-separated list of services to sync data from (defaults,computer,user,group)
 (default "defaults")

--ldap-thumbprints string
 A set of IP=SHA256:B64HASH pairs to trust for authentication

--ldap-url string
 The URL used for the LDAP server

--ldap-username string
 The username for the LDAP account

--max-attempts int
 Set the maximum number of attempts for each probe (default 3)

-G, --max-group-size int
 Set the maximum number of targets to process in each group (default 4096)

-R, --max-host-rate int
 Set the maximum packet rate per target (including ARP broadcast) (default 40)

--max-scan-duration int
 Set the maximum scan duration in seconds before aborting

--max-sockets int
 Set the maximum number of concurrent sockets (default 2048)

--max-ttl int
 Set the default TTL on probe packets (default 255)

--mdns-port uint
 The destination port for MDNS probes (default 5353)

--mecm-database-connection-string string
 The connection string for your MECM Microsoft SQL Server database

--mecm-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--memcache-port uint
 The destination port for memcached probes (default 11211)

--meraki-api-key string
 The access key for the Meraki.io account

```



```

--meraki-api-url string
 The URL used for the Meraki.io account's API access (default
 "https://api.meraki.com/api/v1")

--meraki-exclude-no-vlan-clients

--meraki-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--meraki-excluded-ssids string

--meraki-excluded-vlans string

--meraki-networks string
 An optional list of network names or IDs. Only import devices in the specified networks.

--meraki-organizations string
 An optional list of organization names or IDs. Only import devices in the specified
 organizations.

--miradore-api-key string
 The API key for the Miradore account

--miradore-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--miradore-hostname string
 The Miradore web console hostname (url)

--modbus-identification-level string
 Identification level, one of 'basic', 'regular', or 'extended'. (default "regular")

--mssql-port uint
 The destination port for MSSQL Browser probes (default 1434)

--nameservers string
 One or more nameservers to use for DNS resolution

--natpmp-port uint
 The destination port for NATPMP probes (default 5351)

--nessus-access-key string
 The access key for the Nessus Professional account

--nessus-api-url string
 The URL used for the Nessus Professional account's API access

--nessus-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--nessus-fingerprint-only
 Import vulnerabilities for fingerprinting purposes only

--nessus-insecure
 Set this to true to authenticate to untrusted endpoints (self-signed or no IP SAN) (default
 true)

```

```

--nessus-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")

--nessus-secret-key string
 The secret key for the Nessus Professional account

--nessus-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

--nessus-thumbprints string
 A set of IP=SHA256:B64HASH pairs to trust for authentication

--netbios-port uint
 The destination port for NetBIOS Name Service probes (default 137)

--netbox-api-key string
 The access key for the NetBox instance

--netbox-api-url string
 The URL to the NetBox instance

--netbox-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--netbox-include-no-ip
 Include NetBox assets that have no associated IP addresses

--netbox-include-site-names
 Comma-separated list of NetBox site names to include assets from

--netbox-include-cidrs
 Comma-separated list of CIDRs to include assets from

--netbox-field-mappings
 Map NetBox custom fields to runZero attributes. See the NetBox documentation for
 more information

--nowait
 Exit the user interface immediately upon completion

--ntp-port uint
 The destination port for NTP probes (default 123)

--openvpn-ports string
 The destination ports for OpenVPN probes (default "1194")

--oracledb-fingerprint
 Enable Oracle DB version fingerprinting using a TNS connect sequence (default true)

-o, --output string
 Output directory for scan results and analysis ('disable' to skip)

--output-raw string
 Set the raw output file for scan data

--overwrite
 Overwrite and replace the output directory if it already exists

```

```

--passes int
 Set the number of passes for each probe (default 1)

--pca-port uint
 The destination port for PCAnywhere probes (default 5632)

--pcworx-ports string
 The destination ports for PCWORX probes (default "1962")

--probes string
 Launch a subset of the probes, comma-delimited (default "defaults")

--psdisco-ports string
 The destination ports for playstation discovery probes (default "987,9302")

--qualys-api-url string
 The URL used for the Qualys account's API access

--qualys-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--qualys-fingerprint-only
 Import vulnerabilites for fingerprinting purposes only

--qualys-include-unscanned
 Include assets that have not been assessed for vulnerabilities

--qualys-password string
 The password for the Qualys account

--qualys-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")

--qualys-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

--qualys-tags string
 An optional list of tags. Only import devices that match any of the specified tags

--qualys-network-ids string
 An optional list of network IDs. Only import devices that match any of the specified
 network IDs

--qualys-username string
 The username for the Qualys account

-r, --rate int
 Set the maximum packet rate for the overall scan (default 1000)

--rdns-max-concurrent int
 The maximum number of concurrent DNS lookups (default 64)

--rdns-timeout uint
 The DNS PTR lookup timeout in seconds (default 3)

--rpcbind-port uint
 The destination port for RPCBind probes (default 111)

```

```

--rpcbind-port-nfs uint
 The destination port for NFS probes (default 2049)

--s7comm-request-extended-information
 If true, request extended device information.

--sadb-ports string
 The destination ports for Hikvision SADP discovery probes (default "37020")

--sample-duration string
 Specify the duration in seconds to sample network traffic (or '0' for non-stop) (default
 "300")

--sample-excludes string
 Specify host exclusions

--sample-interfaces string
 Specify a comma-separated list of network interfaces (or 'all' for everything)

--sample-targets string
 Specify the discovery scope (default "10.0.0.0/8 172.16.0.0/12 192.168.0.0/16
 169.254.0.0/16")

-S, --screenshots
 Capture screenshots from scan target web services (default true)

--sentinelone-api-url string
 The URL used for the SentinelOne account's API access

--sentinelone-api-key string
 The API key for the SentinelOne accounts API access

--sentinelone-exclude-software-import
 Do not import software records from the SentinelOne account

--sentinelone-exclude-vulnerability-import
 Do not import vulnerability records from the SentinelOne account

--sentinelone-severities string
 Severity levels of imported vulnerabilities (False Positive, LOW, MEDIUM, HIGH,
 CRITICAL) (default "False Positive,LOW,MEDIUM,HIGH,CRITICAL")

--sentinelone-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--servicetag-port uint
 The destination port for Solaris Service Tag probes (default 6481)

--shodan-api-key string
 The key used for the Shodan account's API access

--shodan-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--shodan-mode string
 The search mode (assets or query). The assets option queries the scan targets (default
 "assets")

--shodan-query string
 The search string to use in query mode

```

```

--sip-port uint
 The destination port for SIP probes (default 5060)

--snmp-comms string
 The comma-separated list of SNMP v1/v2c communities (default "public,private")

--snmp-disable-bulk
 If true, do not use bulk walking operations

--snmp-max-repetitions uint
 The maximum number of repetitions in a bulk walk operation (default 16)

--snmp-max-retries int
 The maximum number of retries for an SNMP operation (default 1)

--snmp-poll-interval uint
 The minimum number of seconds between polling each host after initial discovery
 (default 300)

--snmp-port uint
 The destination port for SNMP probes (default 161)

--snmp-timeout uint
 The maximum number of seconds for each individual SNMP operation (default 5)

--snmp-v3-auth-passphrase string
 The authentication passphrase

--snmp-v3-auth-protocol string
 The authentication protocol (none, md5, sha, sha224, sha256, sha384, sha512) (default
 "none")

--snmp-v3-context string
 The optional SNMP v3 context to supply

--snmp-v3-privacy-passphrase string
 The privacy passphrase

--snmp-v3-privacy-protocol string
 The privacy protocol (none, des, aes, aes192, aes256, aes192c, aes256c) (default
 "none")

--snmp-v3-username string
 The username to use for SNMP v3 authentication

--snmp-walk-timeout uint
 The maximum number of seconds for each SNMP walk operation (default 60)

--ssdp-port uint
 The destination port for UPnP/SSDP probes (default 1900)

--ssh-fingerprint
 Enable fingerprinting using partial authentication (default true)

--ssh-fingerprint-username string
 The username to use for partial authentication SSH fingerprinting (default "STATUS")

--steam-ports string
 The destination ports for Steam discovery probes (default "27036")

```

```

--subnet-ping
 Only scan subnets that have at least one active response using the subnet-ping settings
--subnet-ping-max-attempts int
 Set the maximum number of attempts for each probe (default 1)
--subnet-ping-max-ttl int
 Set the default TTL on subnet-ping probe packets (default 255)
--subnet-ping-mode string
 Set the subnet-ping discovery profile: auto (default "auto")
--subnet-ping-net-size int
 Set the subnet size to use for the subnet ping (default 256)
--subnet-ping-passes int
 Set the number of passes for the subnet-ping phase (default 1)
--subnet-ping-probes string
 Launch a subset of the probes for the subnet-ping, comma-delimited (default
 "arp,echo,syn,connect,netbios,snmp,ntp,sunrpc,ike,openvpn,mdns")
--subnet-ping-sample-rate int
 Set the sample rate of addresses within each subnet as a percentage (default 4)
--subnet-ping-tcp-ports string
 The list of TCP ports to subnet-ping using the syn and connect probes (default
 "22,80,135,179,443,3389,5040,7547,62078")
--subnet-ping-tos int
 Set the default ToS on subnet-ping probe packets
--syn-disable-bogus-filter
 Disable bogus service detection and filtering
--syn-forwarding-check
 Perform an IP forwarding check as part of the scan (default true)
--syn-forwarding-check-target string
 An external IPv4 address for the forwarding check (default:runzero) (default
 "13.248.161.247")
--syn-max-retries uint
 The maximum number of retries trace and SYN requests (default 2)
--syn-report-resets
 Set this to true to report RST responses (default true)
--syn-reset-sessions
 Reset middle-box/firewall sessions automatically (default true)
--syn-reset-sessions-delay uint
 Minimum delay in milliseconds between a SYN and a session reset
--syn-reset-sessions-limit uint
 Maximum number of in-flight sessions before forcing session resets (default 50)
--syn-traceroute
 Perform a multi-protocol traceroute as part of the scan (default true)

```

```

--syn-udp-trace-port uint
 The UDP port number to use for UDP trace requests (default 9)

--tanium-api-token string
 The API token for the Tanium account

--tanium-api-url string
 The URL used for the Tanium account's API access

--tanium-computer-groups string
 Filter endpoints to members of the specified computer groups

--tanium-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--tanium-severities string
 Severity levels of imported vulnerabilities (Low, Medium, High, Critical) (default
 "Low,Medium,High,Critical")

--tcp-excludes string
 The list of TCP ports to always exclude

-p, --tcp-ports string
 The list of TCP ports scan using the syn and connect probes (see below for default)

--tcp-skip-protocol
 Set this to skip protocol detection on TCP ports

--tenable-access-key string
 The access key for the Tenable.io account

--tenable-api-url string
 The URL used for the Tenable.io account's API access

--tenable-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--tenable-fingerprint-only
 Import vulnerabilites for fingerprinting purposes only

--tenable-include-unscanned
 Include assets that have not been assessed for vulnerabilities

--tenable-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")

--tenable-secret-key string
 The secret key for the Tenable.io account

--tenable-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

--tenable-sources string
 An optional list of sources. Only import devices that match the specified sources

--tenable-tags string
 An optional list of tags. Only import devices that match the specified tags

```

```

--tenablesecuritycenter-access-key string
 The access key for the Tenable Security Center account

--tenablesecuritycenter-api-url string
 The URL used for the Tenable Security Center account's API access

--tenablesecuritycenter-batch-size string
 The number of records to request at a time. (between 2000 and 10000) (default "2000")

--tenablesecuritycenter-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--tenablesecuritycenter-fingerprint-only
 Import vulnerabilities for fingerprinting purposes only

--tenablesecuritycenter-insecure string
 Set this to true to authenticate to untrusted endpoints (self-signed or no IP SAN)

--tenablesecuritycenter-query-id string
 The ID of an existing vulnerability query in the Tenable Security Center account

--tenablesecuritycenter-query-mode string
 Set to 'filters' to provide 'severities' and 'risks' values to import. Set to 'query-id' to
 provide a value for 'query-id'. (default "filters")

--tenablesecuritycenter-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default
 "None,Low,Medium,High,Critical")

--tenablesecuritycenter-secret-key string
 The secret key for the Tenable Security Center account

--tenablesecuritycenter-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default
 "Info,Low,Medium,High,Critical")

--tenablesecuritycenter-sync-since string
 Specify an initial date to sync data from.

--tenablesecuritycenter-thumbprints string
 A set of IP=SHA256:B64HASH pairs to trust for authentication

--text
 Force text-only mode (no console ui)

--tftp-ports string
 The destination ports for TFTP probes (default "69")

--tos int
 Set the default ToS on probe packets

--ubnt-port uint
 The destination port for Ubiquiti probes (default 10001)

--upload
 Automatically upload results to the runZero Console

-u, --upload-site string
 Specify the Site ID or Name to upload the raw scan results to if --upload is specified
 (default "Primary")

```



```
--vmware-insecure
 Set this to true to authenticate to untrusted endpoints (self-signed or no IP SAN) (default true)

--vmware-password string
 The password to use for VMware SDK authentication (read-only)

--vmware-thumbprints string
 A set of IP=SHA256:B64HASH pairs to trust for authentication

--vmware-username string
 The username to use for VMware SDK authentication (read-only)

--webmin-ports string
 The destination ports for webmin probes (default "10000")

--wiz-api-url string
 The URL used for the Wiz account's API access

--wiz-auth-url string
 The URL used for the Wiz account's authentication

--wiz-client-id string
 The client ID for the Wiz account

--wiz-client-secret string
 The client secret for the Wiz account

--wiz-exclude-unknown
 Exclude assets that cannot be merged into an existing asset

--wiz-fingerprint-only
 Import vulnerabilities for fingerprinting purposes only

--wiz-include-unscanned
 Include assets that have not been assessed for vulnerabilities (default true)

--wiz-risks string
 Risk levels of imported vulnerabilities (None, Low, Medium, High, Critical) (default "None,Low,Medium,High,Critical")

--wiz-severities string
 Severity levels of imported vulnerabilities (Info, Low, Medium, High, Critical) (default "Info,Low,Medium,High,Critical")

--wlan-list-poll-interval uint
 The minimum number of seconds between polls of the access point list (default 300)

--wsd-port uint
 The destination port for WSD probes (default 3702)

--xdmcp-ports string
 The destination ports for XDMCP probes (default "177")
```

## Run a custom starlark script

```
runZero script --filename filename [--args a] [--args b] [--kwargs a=b] [--kwargs b=c] [flags]
```

```
runZero script [command]
```

### **Available commands:**

repl: Run a custom starlark script in a REPL

### **Flags:**

```
--args stringArray
 args to pass into script
-f, --filename string
 file of script to load and run
--kwargs stringToString
 kwargs to pass into script (default [])
```

## **Upgrade to the latest version of the runZero Scanner**

```
runZero upgrade [flags]
```

### **Flags:**

```
--force
 Apply the update without checking the version
```

## **Perform an internal signature verification**

```
runZero verify [flags]
```

## **Print the version number of runZero**

```
runZero version [flags]
```

# Model Context Protocol

runZero supports the Model Context Protocol (MCP) at the `/mcp` endpoint and utilizes streaming-HTTP for communication. This integration allows clients to directly access runZero's comprehensive security inventory data, enabling advanced, AI-driven queries and insights into network assets and vulnerabilities.

The `console.runzero.com` URLs in this article are examples. Use the endpoint for your runZero deployment instead.

The MCP server offers the same tools as the console's [AI threads](#), so a question you can ask in the console can also be asked from an external client. See [Available tools](#).

Some models are not designed with "tool use" or "function calling" in mind and will struggle. They might lack the internal mechanisms or sufficient training to reliably interpret tool schemas, decide when to call a tool, or correctly parse its output. Trying to force MCP onto such models would likely lead to frequent errors, hallucinations, or simply an inability to use the tools effectively.

The amount of data that is available in runZero will impact the quality of responses. Larger models tend to be able to process larger data sets better but prompting for exactly what is needed along with filtering criteria will help get the information being asked for.

## MCP Terms

### AI Client

The *interface and communication layer* between a human user (or another system) and an AI model. Its primary job is to enable interaction and present AI outputs. It's often *reactive* to user input. It facilitates the use of AI.

### AI Agent

An advanced version of that chat application that, upon receiving "Summarize this document," might first decide it needs to use a "file reading tool" to access the document. The LLM can then summarize the retrieved document contents, and then *proactively* suggest sharing the summary via email without being explicitly told to. The agent orchestrated multiple steps and made decisions.

## Model Context Protocol

From the [MCP specification](#):

**Model Context Protocol** (MCP) is an open protocol that enables seamless integration between LLM applications and external data sources and tools. Whether you're building an AI-powered IDE, enhancing a chat interface, or

creating custom AI workflows, MCP provides a standardized way to connect LLMs with the context they need.

## Authentication

There are two ways to connect. Which one you use depends on what your AI client supports.

**OAuth.** The client sends you to the runZero console, you approve what it may reach, and runZero issues it a scoped credential of its own. Nothing is copied and pasted, the credential is limited to what you approved, and it expires. This is the better option, and most current MCP clients support it. See [Connecting through OAuth](#).

**A credential you supply.** You create an [API key](#) or an [API client](#) yourself and configure it in the client, usually as a request header. Use this when the client cannot do OAuth, or when the connection belongs to a system rather than a person. See [Presenting a credential directly](#).

## What a credential can reach

The credential decides the access. The MCP server offers a client only the tools its credential permits, so a read-only credential is not offered [start\\_scan](#) and cannot call it.

| Credential                             | Prefix | Organizations                      | Access                                             |
|----------------------------------------|--------|------------------------------------|----------------------------------------------------|
| <a href="#">OAuth connection</a>       | -      | What you approved                  | What you approved, never more than your own access |
| <a href="#">API client</a>             | -      | The client's <a href="#">grant</a> | The client's grant                                 |
| <a href="#">Account API token</a>      | CT     | Every organization in the account  | Read-write                                         |
| <a href="#">Organization API token</a> | OT     | The token's organization           | Read-write                                         |
| <a href="#">Export token</a>           | ET     | The token's organization           | Read-only                                          |

[Download tokens](#) (DT) are refused, because they do not map onto an inventory scope.

## Connecting through OAuth

Point the client at <https://console.runzero.com/mcp> and let it authenticate. The client discovers runZero as its authorization server, registers itself, and opens a browser. You sign in to the console if you are not already signed in, and land on an approval page.

The approval page tells you which application is asking, where the authorization code will be sent, and asks two questions:

- **What this application may reach.** Read one organization, read and write one organization, or, for superusers, read and write every organization in the account. Only organizations where you can manage API clients are offered.

- **A name for the credential**, so you can tell your connections apart later.

Superusers get a third choice, **Credential binding**, between a user-bound credential tied to their own access and an account-bound credential that is not. Everyone else gets a user-bound credential.

Approving creates an [API client](#) on your account and hands the application an authorization code, which it exchanges for an access token. Denying creates nothing.

## What the connection can and cannot do

The credential an approval creates is deliberately narrow:

- It **cannot** manage API credentials, scan credentials, console users, or account settings, whatever you approved and whatever your own access is.
- It is limited to what you approved, and, when user-bound, to your current access. If your access narrows, the connection narrows with it. If your account is locked or removed, the connection stops working.
- It **expires 90 days** after approval. That deadline cannot be moved, so renewing the connection means reconnecting the application, which creates a new credential. See [Expiration](#).
- Its access token is only valid for the MCP endpoint. It is refused by the REST API.

Approvals are capped at 40 active connections per account.

## Managing and revoking connections

A connection created this way is an API client like any other. Find it under [My API clients](#), or, for account-bound ones, on the [Account API clients](#) page, and delete it to revoke access immediately. Approvals and the credentials they create are recorded in the [audit log](#).

## Who can approve a connection

To approve a connection you need permission to manage API credentials (`tokens:manage`) in at least one organization, which the Administrator role grants. Without it, the approval page tells you so and nothing is created. Approving for the whole account requires a superuser with a Platform license.

## Presenting a credential directly

For clients that cannot do OAuth, or for unattended connections, create a credential yourself and configure it in the client. Three shapes are accepted.

**API key as a bearer token.** The simplest option, for any client that can send a custom header:

```
Authorization: Bearer CT*****
```

**API key as an OAuth secret.** For a client that can only be configured with OAuth client credentials. Use the client ID `token` and put the API key in the client secret. Send it as HTTP basic auth against the MCP endpoint, or exchange it at the [token endpoint](#) first, which validates the key and returns it as the access token:

```
curl -X POST -H "Content-Type: application/x-www-form-urlencoded" \
 --data-urlencode "grant_type=client_credentials" \
 --data-urlencode "client_id=token" \
 --data-urlencode "client_secret=CT*****" \
 https://console.runzero.com/api/v1.0/account/api/token
```

**Registered API client.** Use the client ID and secret of an [API client](#) you created. Send them as HTTP basic auth against the MCP endpoint, or exchange them at the token endpoint for an access token the way the [REST API](#) does. The client's grant decides what the connection can reach, so this is the way to give an unattended integration a narrow, long-lived MCP connection.

## OAuth endpoints

A request with no usable credential is answered with a 401 whose `WWW-Authenticate` header points at the protected resource metadata, so a client can find its way without being configured by hand.

| Endpoint                                             | Purpose                                                                                                                                                                   |
|------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>/.well-known/oauth-protected-resource</code>   | Names the console as the authorization server for <code>/mcp</code> ( <a href="#">RFC 9728</a> ). Also served at <code>/.well-known/oauth-protected-resource/mcp</code> . |
| <code>/.well-known/oauth-authorization-server</code> | Advertises the endpoints, grant, and PKCE support ( <a href="#">RFC 8414</a> ).                                                                                           |
| <code>/oauth/register</code>                         | Dynamic client registration ( <a href="#">RFC 7591</a> ), for clients with no client ID metadata document.                                                                |
| <code>/oauth/authorize</code>                        | The approval page.                                                                                                                                                        |
| <code>/api/v1.0/account/api/token</code>             | The token endpoint.                                                                                                                                                       |

The authorization code grant is offered, with PKCE using S256, and both dynamic client registration and client ID metadata documents are supported, so a client can register itself without anyone provisioning it first. Registration issues a public client ID and stores nothing; redirect URIs must use HTTPS, or HTTP on a loopback address. Authorization codes are single use and no refresh token is issued.

The client credentials exchange described in [Presenting a credential directly](#) remains available for API keys and API clients, but it is not advertised as an OAuth grant.

## Organization scope

Every inventory tool takes an optional `organization_ids` argument. Omit it and the call covers the whole scope of the credential: the organizations named in an API client's grant or approved for an OAuth connection, every organization in the account for an account API token, and the single organization encoded into an organization key or export token.

An organization-scoped credential is narrowed to its own organization whether or not `organization_ids` was passed. Naming an organization outside the credential's scope returns an error rather than quietly substituting a different one.

## Clients

While the core concept of MCP is standardized communication between AI agents and external tools, how different clients implement this integration can vary significantly based on their design and intended use.

## Connecting to remote MCP servers

Point the client at <https://console.runzero.com/mcp> and select Streamable HTTP as the transport.

If the client supports OAuth, that is all the configuration it needs. It will send you to the console to approve the connection, as described in [Connecting through OAuth](#). If it does not, configure a credential yourself as described in [Presenting a credential directly](#).

## Claude

Claude Desktop and Claude.ai connect to remote servers as a custom connector rather than through `claude_desktop_config.json`.

On a Free, Pro, or Max plan, go to **Settings** → **Connectors** and choose **Add custom connector**. On a Team or Enterprise plan, an owner adds it from **Organization settings** → **Connectors** with **Add** → **Custom**, and members then connect to it individually.

Enter <https://console.runzero.com/mcp> as the server URL and leave **Authentication** on **Always required**, which Claude pre-fills because the MCP endpoint answers an anonymous request with a 401. Leave the **OAuth client** setting on its default; Claude registers itself with runZero automatically.

The first time the connector is used, Claude opens the console in a browser. Sign in if you are not already signed in, choose what the connector may reach, name it, and approve. Claude finishes the connection and the connector is ready to use.

Because each member of a Team or Enterprise organization connects the shared connector separately, each one approves their own credential and reaches only what their own runZero access permits. Nobody sees another member's inventory through the connector, and

revoking one person's runZero access revokes their connection without affecting anyone else's.

The credential expires after 90 days. Reconnecting the connector in Claude creates a new one.

Anthropic documents the dialog field by field in [Third party connectors with remote MCP](#).

## Claude Code

Add the server with the http transport and no credential:

```
claude mcp add --transport http runzero https://console.runzero.com/mcp
```

The first call to the server opens a browser for approval. Confirm the result with `claude mcp list`, which reports whether each configured server connected, and use `/mcp` inside Claude Code to check or re-run authentication.

To use a credential you created yourself instead, pass it as a header, which turns off the OAuth flow for that server:

```
claude mcp add --transport http runzero https://console.runzero.com/mcp --header "Authorization: Bearer CT*****"
```

## Gemini CLI

The [Gemini CLI](#) also has MCP support, primarily configured through the `settings.json` file.

User-scoped `settings.json` files should be found at `~/.gemini/settings.json` (where `~` is your home directory).

The system-wide `settings.json` file can be found or created at:

- Linux: `/etc/gemini-cli/settings.json`
- macOS: `/Library/Application Support/GeminiCli/settings.json`
- Windows: `C:\ProgramData\gemini-cli\settings.json`

For more configuration information, please consult the [project's configuration documentation](#).

Replace the client token value below with a token created as described in [Presenting a credential directly](#), and leave other configuration details (theme, selected authentication type, etc) unchanged from values provided by Gemini.

```
{
 "mcpServers": {
 "runzero": {
 "httpUrl": "https://console.runzero.com/mcp",
 "headers": {
 "Authorization": "Bearer CT*****"
 }
 }
 }
}
```



```
}
}
```

## Other Options

The configurations above should be able to be adapted to other tools. Most of these clients support OAuth, in which case no credential needs to be configured at all. Here are some potential options, including links to their respective documentation. Follow each client's documentation for remote or streaming-HTTP servers rather than its `stdio` instructions.

- [VSCode](#)
- [OpenAI Codex](#)
- [Cursor](#)
- [Windsurf](#)
- [Zed Editor](#)

## Available tools

MCP and [AI threads](#) run the same tools against the same inventory. Everything below is shared with AI threads except `start_scan`, which only MCP exposes. MCP offers the whole set at once; inside the console, each turn is given only the tools its task needs.

Every tool takes an optional `organization_ids` argument; see [Organization scope](#).

## Inventory search

Live queries against the inventory, written in [runZero search syntax](#).

- **query\_assets** – Run an asset query and return the exact match count plus an optional risk-ranked sample. Views: `count`, `compact`, `lifecycle`, `security`, `detail`.
- **query\_asset\_counts** – Count up to eight labeled asset queries in one batch, with no rows. Use this for comparisons and category breakdowns instead of repeated `query_assets` calls.
- **query\_services** – Search the service inventory, including TLS fields.
- **query\_certificates** – Search the certificate inventory, sorted by earliest validity end.
- **query\_tasks** – Search scan and integration tasks. The `series` view returns up to 2,000 runs with timings and result counts for charting.
- **query\_inventory** – Search any other inventory entity: `software`, `software-group`, `vulnerability` (with CVE, CVSS, and remediation text), `vulnerability-group`, `wireless`, `finding`, `event`, `user`, `group`, `task`, `issue`, `goal`, `credential`, `query`, and `site`.
- **get\_attributes** – Return selected host-level attributes for 1-20 assets in one batch.
- **list\_organizations** – List the organizations the credential can reach, with summary asset counts.

## Writing queries

- **get\_field\_definitions** – The searchable fields for an entity or integration source, each with a description and example values. Call it with no argument for the index of areas and the integration sources present in the inventory.
- **get\_query\_syntax** – The query-writing documentation for one entity: operators, worked examples, and that entity's fields. Every entity area `get_field_definitions` lists is supported.
- **translate\_query** – Turn a plain-language phrase into runZero syntax. entity: "asset" with question: "find all cisco devices with ssh exposed" returns (os:cisco or hw:cisco or newest\_mac\_vendor:cisco) and protocol:ssh. The query is compiled but not executed, so it stays fast on large inventories; pass the result to `query_assets` or `query_inventory` to run it.

`translate_query` calls the account's configured model, so it needs AI to be enabled and is subject to the same [entitlement and budget](#) as the console. When AI is unavailable the tool is not offered.

## Detection coverage and rapid response

- **search\_vuln\_coverage** – Search the vulnerability match rules runZero ships, by CVE, vendor, or product. This is platform detection coverage, not your findings.
- **search\_scan\_checks** – Search the active scan checks runZero ships, by CVE, check ID, product, or tag.
- **list\_rapid\_response** – The account's [Rapid Response](#) feed, each entry with the inventory query that finds affected assets.

## Organization analytics

These read an organization-wide snapshot rather than running a query.

- **get\_overview** – Device type, OS, and attack surface distributions, security-control signal counts, and the total asset count.
- **get\_subnets** – Per-subnet rollups (/24 for IPv4, /48 for IPv6) with dominant device types and gateway hints.
- **list\_assets\_in\_subnet** – The assets whose primary address falls within a CIDR.
- **search\_assets** – Compact asset rows matching a case-insensitive substring across address, name, OS, type, and service products.
- **get\_outliers** – Assets with notable outlier scores, annotated with the dimensions that make them unusual.
- **get\_exposed\_assets** – Externally exposed and high-risk internal assets, tagged so the two are not conflated.
- **get\_datastores** – Assets exposing databases, file shares, NFS, SMB, and object storage.
- **get\_security\_controls** – The histogram of detected EDR, MDM, encryption, firewall, and TLS posture signals.
- **get\_critical\_systems** – High-impact assets tagged by role: network core, OT control, identity, hypervisor, datastore, management plane.

- **get\_rogue\_assets** – Hosts seen by discovery with no identity, EDR, MDM, or configuration-management source.
- **get\_eol\_assets** – Expired and near-end-of-life assets with their OS version, EOL date, site, and risk.
- **get\_eol\_crosstab** – The end-of-life heat-map cells by site or business unit and OS family.
- **get\_explorers** – Connected Explorers with their reachable networks, site, and system details.
- **get\_recent\_tasks** – Recent scans and integration syncs, useful for judging data freshness.
- **get\_latest\_metrics** – The latest precomputed metric snapshots.
- **get\_goals** – The security goals the organization is tracking.

## Scanning

- **start\_scan** – Start a scan in an organization, optionally against a specific site or Explorer.

This is the only tool that changes state and the only one MCP does not share with AI threads. Read-only credentials neither see it nor can call it.

## Retired tools

The tool set converged on the AI thread tools, so a number of narrower tools were removed. Their replacements:

| Retired tool                                                         | Use instead                                                               |
|----------------------------------------------------------------------|---------------------------------------------------------------------------|
| query_software                                                       | query_inventory with entity software OR software-group                    |
| query_vulnerabilities                                                | query_inventory with entity vulnerability OR vulnerability-group          |
| query_findings, critical_findings, external_assets_critical_findings | query_inventory with entity finding                                       |
| query_wireless                                                       | query_inventory with entity wireless                                      |
| query_users                                                          | query_inventory with entity user                                          |
| query_directory_group                                                | query_inventory with entity group                                         |
| list_sites                                                           | query_inventory with entity site                                          |
| asset_remediation                                                    | query_inventory with entity vulnerability, which returns remediation text |

For bulk, paginated exports of full records, use the [export API](#) rather than MCP.

## Prompts and resources

Alongside the tools, the server offers:

- The **critical\_findings** prompt, which takes an `organization` name or UUID and asks for that organization's critical findings.
- The **docs://querysyntax** resource, which returns the [search query syntax](#) reference as Markdown.

## Exploring MCP tools, resources, and prompts

[MCP Inspector](#) can be run locally and allows for the API key to be added directly in the UI. Once connected, the server entry can be copied and used for connection to other agents, although it may not be the exact configuration the client needs.

```
$ npx @modelcontextprotocol/inspector@latest
```

1. Enter Authorization as the API client key.
2. Connect to the server to view the tools and resources available.

# Glossary

As you read through the documentation, you will see commonly used terms. These are the definitions for those terms, so you can familiarize yourself with them ahead of time.

## Terms

### Alerts

[Alerts](#) are triggered when a certain events occurs based on [rules](#) defined in the Rules Engine.

### Analysis reports

[Analysis reports](#) are reports which run as tasks, rather than being generated on-the-fly. These reports are static, so any changes to your [inventory](#) may result in assets no longer being accessible from the report.

### Asset

An [asset](#) is a unique network entity from the perspective of the system running the [Explorer](#).

### Automatic queries

[Automatic queries](#) are certain queries you always want to run after a scan. After the query runs, you will be able to view its results in the [queries table](#).

### Dashboard

The [dashboard](#) provides trend data and insights that will help you assess how your inventory is changing over time.

### Discovery scan

A [discovery scan](#) finds, identifies, and builds an [inventory](#) of all the connected devices and assets on your internal network.

### Explorer

The [Explorer](#) is a lightweight scan engine that enables network and asset discovery.

### Insights

[Insights](#) are queries that run automatically after each scan. They will populate on your [dashboard](#).

### Inventory

The [inventory](#) displays all assets within the Organization and can be sorted, filtered,

### Organization

An [organization](#) represents a distinct entity; this can be your business, a specific department within your business, or one of your customers.

### Outliers

[Outliers](#) show how often different values occur in specific attributes of assets and services.

### Queries

[Queries](#) are filters that can be applied to your Inventory to find assets of interest.

## Rules

A [rule](#) defines the action that is taken based on a set of conditions. You can create rules to proactively alert your team when there are changes to things like Explorers, assets, scans, organizations, and sites.

## Scheduled scans

[Scheduled scans](#) allow you to set a date and frequency for your scan task.

## Self-hosted

The [self-hosted](#) version runZero allows you to run the entire platform on-premises or within your own cloud environment.

## Site

A [site](#) represents a distinct network segment, usually defined by addressing or accessibility.

## Frequently asked questions

Here you can browse the solutions to some common runZero issues and the answers to some frequently asked questions (FAQs).

For more solutions and FAQs, check out the knowledgebase on [the runZero support portal](#).

## Issues and FAQs

- [Why are there so many identical assets in my inventory?](#)
- [How do I run runZero without crashing my router?](#)
- [How do I scan VMware virtual machines without crashing the host?](#)
- [Why do Windows Explorers or CLI scans crash on Intel AMX-capable systems?](#)
- [Can EDR or antivirus software interfere with Windows Explorer or CLI scans?](#)
- [Why didn't the runZero Explorer capture screenshots?](#)
- [What protocols does runZero support?](#)
- [What ports does runZero scan?](#)
- [Can I safely scan my IoT or OT environments?](#)
- [Which browsers are supported when accessing the runZero Console?](#)

Still can't find your answer? [Let us know](#).

## Identical assets in inventory

### Why are there so many identical assets in my inventory?

Some enterprise routers and firewalls, like Cisco ASA devices, are designed to reply to all unexpected attempts on a particular port with a TCP reset (RST). On top of that, some routers listen to SIP traffic on all addresses and automatically respond to it. runZero tries to automatically detect and avoid most of the SIP helper implementations, but can't always do so without possibly losing real results.

runZero will generally detect when a router or firewall is replying to every connection attempt and avoid creating assets based on those responses. However, if you have a network appliance that runZero doesn't detect is spoofing response, there may be a substantial number of identical assets that will appear in your inventory.

Here are a couple of workarounds if you can't prevent your device from replying to all connections:

- Exclude the ports the device responds to from the scan configuration.
- Exclude all or part of the router's IP address range from the scan.

If you need help deleting unwanted records, please [contact our support team](#).

## Scanning routers

### How do I run runZero without crashing my router?

If your router is crashing while being scanned, the likely issue is that your router is stateful and it is keeping track of every connection going through it. Since our scanning process involves thousands of attempted connections, your router likely ran out of available stateful sessions. This usually occurs when a router is using Network Address Translation (NAT) or is acting as a stateful security firewall.

If this happens, here's what you can do:

- Avoid scanning across routed networks (wired and WiFi, multiple VLANs, etc) by deploying additional Explorers.
- Reduce the Max group size in your scan configuration. This limits the number of targets runZero can scan at once, which correlates to the number of connections the router sees. Default is 4096.
- Reduce the scan speed. This will give failed connections more time to expire before new ones are attempted.
- If a router can run in bridge mode, and you don't need its NAT features, bridge mode will likely be more reliable. For example, if you have an ISP-provided router connected to a WiFi mesh system, you will likely want to run the mesh system in bridge mode and let the ISP router handle all routing including NAT; you should then be able to scan across your WiFi network without crashing the base stations.

## Scanning VMWare virtual machines

### How do I scan VMware virtual machines without crashing the host?

runZero can be used to scan VMware virtual machines. However, there are some precautions you should take.

VMnet interfaces normally use Network Address Translation (NAT) to route traffic between the host system and the virtual machines. The VMware software effectively operates as a stateful router. As explained above, this can cause problems when runZero tries to open thousands of connections.

For scanning VMware systems, the best option is to deploy a runZero Explorer inside VMware, on a virtual machine connected to the VMnet you want to scan. That Explorer should be able to scan all VMs on the same VMnet without VMware needing to track all of the connections.



## Scanning on Windows with Intel AMX

### Why do Windows Explorers or CLI scans crash on Intel AMX-capable systems?

Some Windows hosts expose Intel Advanced Matrix Extensions (AMX) CPU features that can interfere with runZero scan operations and cause Explorer or CLI scanner instability and crashes. This can affect both physical hardware and virtual machines.

To resolve or mitigate this issue:

1. Disable AMX support at the BIOS level (physical systems) or at the host/hypervisor level before installing or running runZero.
2. If Windows is running as a guest on Hyper-V, enable **Processor Compatibility** for that VM to help mitigate AMX-related crashes.

If the issue persists after applying one of the above mitigations, reinstall the Explorer or rerun the CLI scanner and collect diagnostics for runZero support.

## Scanning on Windows with EDR or antivirus

### Can EDR or antivirus software interfere with Windows Explorer or CLI scans?

Yes. Some EDR, antivirus, HIDS, and endpoint protection products on Windows can interfere with runZero Explorer and CLI scanner operations. Common impacts include blocked packet capture, failed raw socket operations, unstable scan behavior, and process termination.

To reduce or resolve these issues:

1. Add allowlist exceptions for runZero binaries, services, and directories used by the Explorer or CLI scanner.
2. Ensure packet capture components (including npcap/winpcap where applicable) are not blocked by endpoint security policies.
3. If your endpoint tooling supports policy modes, use a monitoring-only mode while validating scan operations.
4. In some environments, endpoint protection controls may need to be temporarily disabled on the host to confirm whether they are the root cause.
5. Re-run the scan or reinstall/restart the Explorer after policy updates.

If issues continue, collect Explorer diagnostics or CLI output and contact runZero support.

## Explorer not capturing screenshots

### Why didn't the runZero Explorer capture screenshots?

The runZero Explorer needs a working install of Google Chrome to obtain screenshots. To check for Google Chrome, the Explorer looks in the following locations on each OS.

#### Windows

The runZero Explorer looks for Chrome on Windows in:

`c:\Program Files (x86)\Google\Chrome\Application\chrome.exe`

The Explorer also checks the following environment variables:

- `ProgramFiles(x86)`
- `ProgramFiles`
- `ProgramW6432`

Each may list another directory, in which case the Explorer looks in `\Google\Chrome\Application\chrome.exe` under each of those directories as well.

To find what the environment variables are set to, open a Windows command prompt and entering the command `set`.

For a default Windows 10 install, the default value of `ProgramFiles` and `ProgramW6432` is `c:\Program Files`, which means the Explorer also checks the following location:

`c:\Program Files\Google\Chrome\Application\chrome.exe`

This is the default location for Chrome on a 64 bit Windows 10 system.

#### MacOS

On macOS, the Explorer checks for Google Chrome in the following locations:

- `/Applications/Google Chrome.app/Contents/MacOS/Google Chrome`
- `/Applications/Google Chrome Canary.app/Contents/MacOS/Google Chrome Canary`
- `/Applications/Chromium.app/Contents/MacOS/Chromium`

#### Linux

On Linux systems, the Explorer checks for Google Chrome at the following locations:

- `/usr/bin/google-chrome`
- `/usr/bin/google-chrome-beta`
- `/usr/bin/google-chrome-unstable`
- `/usr/local/bin/chrome`
- `/usr/bin/chrome`

- `/opt/google/chrome/google-chrome`
- `/usr/bin/chromium`
- `/usr/bin/chromium-browser`

## Chrome is installed, but screenshots still don't work

If Google Chrome is installed in one of the standard locations, but isn't being found, it could be a permissions issue. It is also possible for Chrome to fail to run for other reasons, such as a corrupt Chrome profile. The next step to take is to check the [Explorer log file](#) which will have more detail about the scan operation.

In addition, you won't see a screenshot for http(s) services if any of the following are true:

- It's located on a IPv6 link-local IP address (Chrome cannot access these)
- It uses a TLS version lower than 1.2 (Chrome no longer supports accessing these)
- The browser timed out while attempting to take a screenshot (e.g. unresponsive host)
- The host returned an error code.

## Explorer security model

runZero Explorers are responsible for running network scans, sampling traffic for passive discovery, and implementing integrations within their deployed environment. This document describes the security model of runZero Explorers.

## Development

runZero Explorers are built using the Go programming language. This language provides cross-platform compatibility and memory safety by default. The runZero engineering team updates the Explorer code and its dependencies on a continuous basis to ensure that any dependency-level vulnerabilities are resolved quickly. All engineers are required to use MFA to access our Git repository. All code changes go through manual review before being merged to our development branch, deployed to a test environment, reviewed again before being merged to our staging branch, and deployed to a staging environment. During the build process, Explorer binaries are signed with a private Ed25519 key and the signature is stamped into the end of the binary. For Windows binaries, an Authenticode signature is also applied, using an extended-validation certificate stored in a cloud-based HSM. Once all automated tests pass and any changes to behavior are confirmed, the staging environment binaries are tagged for release, and moved to the production deployment location in AWS S3.

## Deployment

runZero Explorers are initially provisioned by hand through direct downloads from the runZero Console (whether cloud or self-hosted). During this download process the link token and organization ID is stamped into the binary while still preserving the code signatures. The binary will run on the deployed system, connect back to its stamped Console URL, validate TLS, and register. The registration process starts with the Explorer generating a unique host ID using the operating system's random source. The ID is stored with limited permissions on

the Explorer's file system (non-Windows) or registry (Windows) and will be used to confirm that Explorer's identity for future connections.

During the first connection from a new Explorer, the Console records the Explorer-generated Host ID, and establishes a database record for that instance. Other Explorers, even those running on the same system, are not able to claim to be this unique Explorer record without also possessing and verifying the Host ID. New Explorers, by default, do not receive any data from the Console and are not sent any tasks (including those with credentials) unless manually selected by the user. As a result, it's relatively safe to have untrusted Explorers registered, as long as they aren't used for tasks that include sensitive data (typically integrations with credentials).

Note that once an Explorer has the Organization ID, the link token (Download Token) is not needed for the Explorer to connect and register. runZero plans to enforce Explorer use of the current Download Token in the near future, but this is not yet a requirement, as there are few if any security risks exposed by unauthorized Explorer registrations. Once the current Download Token becomes a requirement for registration, older Explorers will not be able to reconnect after the Download Token is rotated. Active Explorers should always receive the latest Download Token as part of the upgrade request.

## Commands

The runZero Console provides the ability to send new tasks and specific requests to Explorers. Tasks are limited to active scans, passive traffic sampling, or integrations. Integrations can use custom code, written in the Starlark language, and while they have access to the network, they do not have access to the Explorer host machine or file system. Outside of tasks, Explorers accept requests to uninstall, update, and post diagnostics (limited to stack traces and Explorer attributes like operating system and hardware specifications). Explorers do not implement any commands that allow network tunneling or host access to the Explorer's machine.

Credentials are only sent to an Explorer if they are associated with an integration task and if the integration task scope matches the CIDR or other configuration for those credentials. Credentials are never sent to runZero-hosted External Explorers regardless of the task configuration. Credential parameters are sent across in the same TLS connection as the request, but are also encrypted to a unique per-process AGE encryption key. Even if the runZero communication is logged through a TLS proxy and pinned CA, credentials cannot be recovered without also extracting the private, in-process-only, and temporary AGE key.

## Updates

The runZero Console verifies that Explorers are running the latest version before scheduling a task (scan, passive, or integration). If the Explorer is out of date, it will be updated to the latest version first, and then the task will be scheduled. If the update takes longer than an internal threshold, or fails due to things like a read-only filesystem on the Explorer side, the task will be scheduled anyways after this threshold is reached.

The update process receives a URL to the new binary. The Explorer connects to this URL, verifies TLS parameters, and downloads the binary to a temporary directory. The Explorer

then verifies that the binary has a valid code signature embedded and rejects the update if this verification fails. If the code signature is valid, the Explorer launches the new binary in update mode, which typically replaces the running service with the new executable.

It is possible and supported for the update to change the Organization or even Console for the Explorer service. This feature is used to move Explorers between installations and organizations. In either case, the TLS connection must be valid, and the binary must have a valid code signature. This verification process also rejects version downgrades by comparing the signed version metadata in the binary against the running version.

## Uninstallation

runZero Explorers are automatically uninstalled when their associated runZero Organization is deleted. Explorers can also be uninstalled from the runZero Console or by using the “uninstall” CLI option of the Explorer binary. Explorers try to clean up as many artifacts as possible during the uninstallation process. Any artifacts remaining post-uninstall can be manually removed as needed (common on Windows systems where the OS locks the program files and directory while they run).

## Defense in Depth

runZero strives to protect customer data by layering multiple defenses into the runZero architecture:

- An attacker that is able to push code to our internal repository would still need to pass review before it can be merged into our protected staging branch.
- An attacker that is able to compromise our binary distribution point (AWS S3) would still need a valid signing key before any customer updates could be affected. Downgrade attacks are prevented by Explorer-time version comparisons of authenticated and signed metadata.
- An attacker that obtains the download link for a customer can register new Explorers, but is not able to obtain any data from the Console unless those Explorers are specifically chosen for a task.
- An attacker that is able to send commands to an Explorer (by compromising a Console) is limited to a few relatively-safe commands and is not able to force downgrades or arbitrary code execution on the Explorer host.
- An attacker that is able to MiTM the TLS communication with the Console is not able to decrypt credentials used in integration tasks.
- An attacker that is able to schedule a custom integration task with a malicious Starlark script can interact with the network from the perspective of the Explorer, but is not able to interface with the Explorer host.

In all of these situations, runZero also operates extensive logging and alerting, so that any malicious action can be investigated, and the impact of any issue readily assessed.

## Browsers supported by the runZero Console

### Which browsers are supported when accessing the runZero Console?

We maintain compatibility with the following browser versions or newer:

- *Chrome 123* (released March 18, 2024)
- *Edge 123* (released March 22, 2024)
- *Safari 17.5* (released May 12, 2024)
- *Firefox 120* (released November 20, 2023)
- *Opera 109* (released March 26, 2024)

Internet Explorer is *not* supported.

Mobile browser support is experimental and is not guaranteed to work.

## Common sign-in issues

### User account has been locked from too many failed sign-in attempts.

**Cause:** The account you are signing into is configured for password authentication and has been locked due to repeated incorrect sign-in attempts.

**Remediation:** Contact your runZero administrator and have them follow these steps:

1. In the runZero console, visit the Team page
2. Select the locked account
3. Use the "Unlock user accounts" option in the "Reset" dropdown menu.

If contacting an administrator is not an option, please reach out to [support@runzero.com](mailto:support@runzero.com).

### Certificate information for single sign-on has expired and must be updated in order to sign in

**Cause:** The SAML certificate configured for the target runZero tenant has expired.

**Remediation:**

1. Acquire a new valid certificate from your identity provider (for example, Okta, Google Workspace, etc)
2. Sign into runZero using a superuser account that is configured for link-based or password-based authentication

3. Paste the new certificate contents into the "Certificate" field of the Identity provider settings page

- The "Certificate" field can be found by navigating to the Team page in the product, then clicking the "SSO settings" button in the page header.

**If you are the administrator for your account and are unable to sign in at all**, please contact [support@runzero.com](mailto:support@runzero.com) and provide the required certificate for further assistance.

## Invalid SAML response

**Cause:** The data that your identity provider sent to the runZero console was in an invalid format.

**Remediation:** Review the SAML configuration in your identity provider (for example, Okta, Google Workspace, etc)

## The information provided by your identity provider was incomplete or not valid.

**Cause:** The data that your identity provider sent to the runZero console did not include one or more required assertion elements. Required elements are Subject and NameID.

**Remediation:** Review the configuration of your identity provider to ensure it's configured to send a valid Subject and NameID to the runZero console.

## The email address {email address} is already in-use by an existing user account.

**Cause:** The email address provided by the SSO identity provider already exists in a different tenant in the runZero console.

**Remediation:** Change the email address of the existing runZero account or delete the existing runZero account. There are a few possible ways to fix this:

- Sign into the runZero console with the email address mentioned in the error message and change the associated email address to something different.
- Contact the administrators of the tenant where your email address is registered and request that they delete your account from the Team page.
- Contact runZero support via email at [support@runzero.com](mailto:support@runzero.com) and request that they delete your existing account.

Once the email address is no longer in use, you will be able to sign in with SSO.

## The identity provider has provided an invalid NameID

**Cause:** Your SSO identity provider (for example, Okta, Google Workspace, etc) has provided the runZero console with a NameID attribute that is not formatted like an email address. The runZero console uses this attribute as your email address in the product.

**Remediation:** Review the configuration of your identity provider and ensure it is configured to provide an email address in the NameID attribute, or contact your administrators to do so.

## runZero data formats

runZero consumes and produces a handful of data formats. This page provides examples of these formats and describes the fields and use cases for each.

### Formats

- [Scan data \(sample\)](#)
- [Asset data \(sample\)](#)
- [Change reports \(sample\)](#)

### Scan data

The raw output produced by the runZero Explorer and the runZero CLI is the **scan data**. This is newline-delimited JSON – JSONL – that represents the unprocessed output of the scan engine. This format is returned when downloading the task data for an Explorer-run scan and correlates to the `scan.runzero.gz` file created by the CLI. The runZero Inventory view is built by processing scan data in chronological order to create the current state at a given point in time.

Scan data can be imported into an existing site through the Inventory Import menu of the web console and through the `--import` parameter of the CLI. Each line of the file is a JSON object that specifies a type and a 64-bit Unix timestamp.

The example below is the raw scan data for a single Apple Mac Mini:

```
{ "type": "config", "ts": 1597259738842951567, "probes": ["arp", "bacnet", "dns", "dtls", "echo", "ike", "ipmi", "mdns", "memcache", "mssql", "natpmp", "netbios", "ntp", "openvpn", "pca", "rdns", "rpcbind", "sip", "snmp", "ssdp", "syn", "tftp", "ubnt", "wlan-list", "wsd"], "addresses": ["192.168.0.1", "192.168.30.1", "192.168.40.1"], "networks": ["192.168.0.1/24", "192.168.30.1/24", "192.168.40.1/24"], "params": { "arp-fast": "false", "bacnet-port": "47808", "clock-offset": "0", "dns-port": "53", "dns-resolve-name": "www.google.com", "dns-trace-domain": "helper.rumble.network", "dtls-ports": "443,3391,4433,5246,5349,5684", "excludes": "", "ike-port": "500", "ipmi-port": "623", "max-group-size": "4096", "max-host-rate": "40", "max-sockets": "512", "mdns-port": "5353", "memcache-port": "11211", "mssql-port": "1434", "nameservers": "", "natpmp-port": "5351", "netbios-port": "137", "nopcap": "false", "ntp-port": "123", "openvpn-ports": "1194", "passes": "1", "pca-port": "5632", "probe-s": "arp,bacnet,dns,dtls,echo,ike,ipmi,mdns,memcache,mssql,natpmp,netbios,ntp,openvpn,pca,rdns,rpcbind,sip,snmp,ssdp,syn,connect,tftp,ubnt,wlan-list,wsd", "rate": "1000", "rdns-max-concurrent": "64", "rpcbind-port": "111", "rpcbind-port-nfs": "2049", "screenshots": "true", "sip-port": "5060", "skip-broadcast": "true", "s
```



```

uth-passphrase":"","snmp-v3-auth-protocol":"none","snmp-v3-context":"","snmp-v3-privacy-passphrase":"","snmp-v3-privacy-protocol":"none","snmp-v3-username":"","ssdp-port":"1900","syn-max-retries":"2","syn-udp-trace-port":"65535","tcp-ports":"1300,5554,8020,20034,47001,41080,2601,2604,2638,5060,7181,10202,4679,2181,34205,13,2323,5601,18881,50070,139,1129,2199,2375,4444,902,1440,2103,32913,1311,9524,8028,8883,13364,37718,512,3200,5683,10203,81,1091,5222,8081,13838,37777,1,5672,8095,65535,21,540,548,1102,27080,28017,34443,40007,6060,6542,8300,27888,4786,9443,2049,3050,5984,46823,12221,1352,6405,26122,7210,41025,1103,1530,1883,8834,443,9100,45230,1234,3128,5432,12397,111,993,3780,5250,6112,524,5247,20031,1211,1755,5985,6070,8880,1241,3690,6002,1035,4000,8080,9081,2362,23,587,921,8903,31001,143,2598,3273,6101,8812,10628,25,113,513,1720,2533,6905,32764,38080,5040,20010,6001,6660,8471,82,2222,5093,6262,6379,8545,384,5168,20222,7579,998,3057,3217,6106,9391,9,2380,5520,9060,19300,30718,49,84,161,5900,10001,8009,19,617,2100,5580,38292,85,6667,10443,42,2121,5986,23791,515,1199,10008,16993,631,2083,8443,9527,13500,27017,30000,41523,554,5061,4659,8333,9855,5355,7001,1000,1220,5521,11234,20000,6988,3351,7547,7,1900,7778,9160,31099,1030,10616,7902,8090,12174,1533,135,5631,623,5038,9300,19888,14330,109,1433,15672,1581,3790,5632,9999,80,2381,4840,7800,61616,1101,1128,1494,3311,9092,11000,110,995,1098,5800,523,8087,10098,28784,407,7777,9090,19810,34963,50000,502,1100,8161,8180,9152,11099,2379,8023,88,1582,20101,16102,16992,1583,5814,5938,20111,11211,636,27000,1158,5400,5920,7443,9530,20171,8800,9099,7474,8222,10000,2082,8902,50013,689,771,7080,8098,8686,22,1024,12345,105,9080,9111,47002,4433,44818,4848,6080,7071,8303,62078,705,873,6000,7077,8503,9495,34964,5666,17200,5433,7801,11333,12401,12203,25025,264,2525,3628,9809,26000,50090,9000,2967,137,4730,5051,8899,10050,52302,8400,53,389,402,4443,7700,62514,1090,5353,6082,6661,40317,8089,17185,912,5405,28222,465,4445,6503,8014,57772,23472,1080,7021,8088,22222,5000,9084,18264,8888,6050,7144,41524,69,1811,44334,102,6502,1521,2809,9471,888,5351,5498,123,34962,8205,9042,2947,3389,5555,10080,10162,9200,500,1089,7510,15200,2207,3500,8008,9418,3460,6504,7770,25000,55553,179,783,8012,9595,46824,7580,5560,49152,83,903,1604,3632,4322,4567,445,8030,9390,3817,8901,10051,9002,27019,910,1099,3000,3299,222,7787,37,2000,3306,48899,7879,79,1723,3037,3312,8000,23943","tftp-ports":"69","ubnt-port":"10001","verbose":"true","wlan-list-poll-interval":"300","wsd-port":"3702"},"scan_targets":{"networks":["192.168.0.5/32"],"enable_dns":true,"enable_ip6":false,"inputs":["192.168.0.5"],"dns_timeout":2000000000,"concurrency":24,"version":"1.10.0 (build 20200804052508) [eae4e551f9f0ce5ab3bf0a1410b2ed5098db097e]"}
{"type":"status","ts":1597259758851278069,"level":"info","source":"connect","msg":"waiting on TCP probes to complete"}
{"type":"stats","ts":1597259769858733899,"stats":{"elapsed":31,"progress":94,"rateLimitTime":24246445895,"recv":546,"recvBytes":37404,"recvError":10,"recvRate":17,"resultCount":22,"secondsLeft":1,"sent":503,"sentBytes":36833,"sentError":0,"sentRate":16,"startTime":1597259738843056937}}
{"type":"result","ts":1597259738844372618,"host":"192.168.0.5","port":0,"proto":"icmp","probe":"echo","info":{"icmp.addr":"192.168.0.5","icmp.rtt":"541214","ip.flags":"DF","ip.id":0,"ip.tos":0,"ip.ttl":"64"}}
{"type":"result","ts":1597259739077542717,"host":"192.168.0.5","port":137,"name":"MACMINI-EE7C7B","proto":"udp","probe":"netbios","info":{"netbios.domain":"WORKGROUP","netbios.mac":"f0:18:98:ee:7c:7b","netbios.macDateAdded":"2017-12-23","netbios.macVendor":"Apple, Inc."}}
{"type":"result","ts":1597259739219939211,"host":"192.168.0.5","port":137,"proto":"udp","probe":"netbios","info":{"netbios.addr":"192.168.0.5"}}
{"type":"result","ts":1597259739471688048,"host":"192.168.0.5","port":0,"proto":"arp","probe":"arp","info":{"arp.mac":"f0:18:98:ee:7c:7b","arp.macDateAdded":"2017-12-23","arp.macVendor":"Apple, Inc.","source":"arp"}}
{"type":"result","ts":1597259739826114224,"host":"192.168.0.5","port":5353,"name":"Developers-Mac-mini","proto":"udp","probe":"mdns","info":{"mdns.replies":"5.0.168.192.in-addr.arpa.=PTR,Developers-Mac-mini.local."}}
{"type":"result","ts":1597259740197709484,"host":"192.168.0.5","port":445,"proto":"tcp","probe":"connect","info":{"ntlmssp.dnsComputer":"Developers-Mac-mini.local","ntlmssp.dnsDomain":"local","ntlmssp.negotiationFlags":"0x62898235","ntlmssp.netbiosComputer":"DEVELOPERS-MAC-MINI","ntlmssp.netbiosDomain":"MACMINI-EE7C7B","ntlmssp.ntlmRevision":"15","ntlmssp.targetName":"MACMINI-EE7C7B","ntlmssp.timestamp":"0x01d670dd06500880","ntlmssp.version":"6.1.7600","protocol":"smb1\\tsmb2\\tsmb3","smb.capabilities":"0x00000066","smb.dialect":"0x0302","smb.guid":"ff12583f-5ba1-53b8-8ff3-a48a394056f7","smb.nativeL-

```

```
M": "@(#)PROGRAM:smbd PROJECT:smbx-499.60.1", "smb.nativeOS": "Darwin", "smb.sessionID": "0x9551b7cb000000001", "smb.signing": "required", "source": "mdns"}}
```

The data contains four types of object:

- **Scan config:** The {"type": "config"} object contains the full set of parameters for the scan as well as the version of the scan engine, and on Windows, the version of nmap installed. This record is used to determine the scan targets, which is used by the analysis engine to determine whether a given IP address was in scope.
- **Scan status:** The {"type": "status"} object contains diagnostic output from the scan engine. This can highlight issues that occurred while the scan was running.
- **Scan stats:** The {"type": "stats"} object represents point in time statistics for the scan. This will include the number of packets sent, received, and the progress estimate.
- **Scan result:** The {"type": "result"} object is target response for a specific probe. This can include TCP SYN+ACK replies, ICMP replies, or the result of application-layer probes, such as SNMP query responses, or HTTP screenshots. Scan Results are analyzed and correlated to create the Asset Data format.

The scan stats sub-fields are defined below:

| Field         | Description                                                      |
|---------------|------------------------------------------------------------------|
| cpu           | CPU Core Percent * 100. 100% of one core would be 100000.        |
| elapsed       | The number of seconds since the scan started.                    |
| fdcount       | The number of open file descriptors.                             |
| memory        | The current memory usage in bytes.                               |
| progress      | The estimated progress as a percentage (90 = 90%).               |
| rateLimitTime | The number of Unix nanoseconds spent idling in the rate limiter. |
| recv          | The number of packets received from the network.                 |
| recvBytes     | The number of bytes received from the network.                   |
| recvError     | The number of errors receiving from the network.                 |
| recvRate      | The average packet receive rate for the scan.                    |
| resultCount   | The total number of findings from the scan.                      |
| routines      | The number of internal goroutines in the scan engine.            |
| secondsLeft   | The estimated seconds left to complete the scan.                 |
| sent          | The number of packets sent the network.                          |
| sentBytes     | The number of bytes sent the network.                            |
| sentError     | The number of errors sending to the network.                     |

|           |                                                             |
|-----------|-------------------------------------------------------------|
| sentRate  | The average packet send rate for the scan.                  |
| startTime | The Unix timestamp in nanoseconds of when the scan started. |

The scan result object type contains the following fields in addition to `type` and `ts`:

| Field | Description                                                                                                    |
|-------|----------------------------------------------------------------------------------------------------------------|
| host  | The IP address associated with the response.                                                                   |
| name  | An optional hostname returned as part of this response.                                                        |
| port  | The TCP or UDP port or zero for other protocols.                                                               |
| proto | The transport protocol, one of <code>arp</code> , <code>icmp</code> , <code>tcp</code> , or <code>udp</code> . |
| probe | The specific internal probe name that returned this response.                                                  |
| info  | The result details object where all keys and values are strings.                                               |

The `info` object contains probe-specific response data. The key names are typically in the format of `probe.subfield`, with a few exceptions, and the values are always strings, even for numeric and array content. Multiple values for a key are represented as a tab-delimited array. Empty values are never reported for `info` keys. A given scan may return multiple result objects for a single probe, sometimes with duplicate values. These responses are correlated, deduplicated, and merged during the next phase of processing.

## Asset data

The correlated and fingerprinted assets shown in the web console Inventory view and in the `assets.jsonl` file produced by the runZero CLI are the **asset data**. This data represents the state of each unique asset at a point in time and is built up by processing one or more sets of scan data.

runZero supports a few variants of the asset data, including line-delimited JSON (JSONL), standard JSON documents, and a simplified CSV export. The JSONL format is the easiest to work with as it supports incremental processing without having to load the entire response into memory.

The example below is the correlated asset data for a scan of a single Apple Mac Mini:

```
{
 "id": "b73f8e09-78a6-4d2b-979d-e63908f28251",
 "created_at": 1597259778,
 "updated_at": 1597259778,
 "organization_id": "b7fb13a7-701d-4ca5-b0e6-6f28f06cc866",
 "site_id": "52d60c51-8dee-4f09-94e5-2dee30050a25",
 "alive": true,
 "last_seen": 1597259750,
 "first_seen": 1597259738,
 "detected_by": "arp",
 "type": "Desktop",
 "os": "Apple macOS",
 "os_version": "10.15",
 "hw": "Apple Mac Mini (Late 2018)",
 "addresses": ["192.168.0.5"],
 "addresses_extra": ["fe80::1c9d:c567:8db1:d79b"],
 "macs": ["f0:18:98:ee:7c:7b"],
 "mac_vendors": ["Apple, Inc."],
 "names": ["MACMINI-EE7C7B", "DEVELOPERS-MAC-MINI", "DEVELOPERS-MAC-MINI.LOCAL"],
 "tags": {},
 "domains": [],
 "services": {
 "192.168.0.5/0/arp": {
 "arp.mac": "f0:18:98:ee:7c:7b",
 "arp.macDateAdded": "2017-12-23",
 "arp.macVendor": "Apple, Inc.",
 "source": "arp",
 "ts": "1597259739"
 },
 "192.168.0.5/0/icmp": {
 "icmp.addr": "192.168.0.5",
 "icmp.rtt": "541214",
 "ip.flags": "DF",
 "ip.id": "0",
 "ip.tos": "0",
 "ip.ttl": "64",
 "ts": "1597259738"
 },
 "192.168.0.5/137/udp": {
 "netbios.addr": "192.168.0.5",
 "netbios.domain": "WORKGROUP",
 "netbios.mac": "f0:18:98:ee:7c:7b",
 "netbios.macDateAdded": "2017-12-23",
 "netbios.macVendor": "Apple, Inc.",
 "protocol": "netbio-"
 }
 }
}
```

```

s", "ts": "1597259739"}, "192.168.0.5/22/tcp/": {"banner": "SSH-2.0-OpenSSH_7.9", "ip.flags": "DF", "ip.id": "0", "ip.tos": "0", "ip.ttl": "64", "protocol": "ssh", "service.cpe23": "cpe:/a:openbsd:openssh:7.9", "service.family": "OpenSSH", "service.vendor": "OpenBSD", "service.version": "7.9", "source": "mdns", "ssh.hostKey.data": "AAAAB3NzaC1yc2EAAAADAQABAAQCGjYTFcSp2Fs/R8dbOLYiQ6PPrULZYanYH3SCYYr5QgC1SIF3AURGYTMnUDAS+tTI/Pquwowkgiq3rtfsQMAsCrabbPahwiOLTupsuLNp3evXYYSf8ZQFyBN8iz5cys06u+yczqWG7Fu8mgsS8zwCwN7yRrbFwd8+Hp6GgFUU4Z6jUQoZu7iajpbSX1TA90YKXQIZ0m8qc4mPLT/uHw9nxNmExWA1V/2ZeoS59NGSV8zFMKb52SOXKhkvHAIUVh5NJDaudxK4uP4eG6dxr8btYtVKIOYKlsLdSBSfHvScvVVLb7DKJBiMXG+qspt33Zd73o4S9Ich20aSbVt7h/NZ3", "ssh.hostKey.md5": "75:9b:a2:e6:10:da:72:8a:11:91:3f:a1:43:14:7f:2e", "ssh.hostKey.sha256": "SHA256:xVJfddKBJ9E5jstVCj0zY8763Rnxy2ppqzaLZX0+cHc", "ssh.hostKey.type": "ssh-rsa", "syn.rtt": "542019", "tcp.options": "MSS:05b4", "tcp.ts": "2009546838", "tcp.urg": "0", "tcp.win": "65535", "ts": "1597259740"}, "192.168.0.5/3031/tcp/": {"source": "mdns", "ts": "1597259740"}, "192.168.0.5/3283/tcp/": {"source": "mdns", "ts": "1597259740"}, "192.168.0.5/445/tcp/": {"ip.flags": "DF", "ip.id": "0", "ip.tos": "0", "ip.ttl": "64", "ntlmssp.dnsComputer": "Developers-Mac-mini.local", "ntlmssp.dnsDomain": "local", "ntlmssp.negotiationFlags": "0x62898235", "ntlmssp.netbiosComputer": "DEVELOPERS-MAC-MINI", "ntlmssp.netbiosDomain": "MACMINI-EE7C7B", "ntlmssp.ntlmRevision": "15", "ntlmssp.targetName": "MACMINI-EE7C7B", "ntlmssp.timestamp": "0x01d670dd06500880", "ntlmssp.version": "6.1.7600", "protocol": "smb1\\tsmb2\\tsmb3", "smb.capabilities": "0x00000066", "smb.dialect": "0x0302", "smb.guid": "ff12583f-5ba1-53b8-8ff3-a48a394056f7", "smb.nativeLM": "@(#)PROGRAM:smbd PROJECT:smbx-499.60.1", "smb.nativeOS": "Darwin", "smb.sessionID": "0x9551b7cb00000001", "smb.signing": "required", "source": "mdns", "syn.rtt": "624414", "tcp.options": "MSS:05b4", "tcp.ts": "2009544186", "tcp.urg": "0", "tcp.win": "65535", "ts": "1597259740"}, "192.168.0.5/5353/udp/": {"hw.device": "Desktop", "hw.family": "Mac mini", "hw.product": "Mac mini (Late 2018)", "hw.vendor": "Apple", "mdns.addr": "fe80::1c9d:c567:8db1:d79b\\t192.168.0.5", "mdns.device.model": "Macmini8,1", "mdns.device.osxvers": "19", "mdns.ports": "eppc/tcp=3031\\tnet-assistant/udp=3283\\trfb/tcp=5900\\tsftp-ssh/tcp=22\\tsmb/tcp=445\\tssh/tcp=22", "mdns.replies": "5.0.168.192.in-addr.arpa.=PTR,Developers-Mac-mini.local.\\tDeveloper\\226\\128\\153s\\ Mac\\ mini._device-info._tcp.local.=TXT,model=Macmini8,1 osxvers=19\\tDevelopers-Mac-mini.local.=A,192.168.0.5\\tDevelopers-Mac-mini.local.=AAAA,fe80::1c9d:c567:8db1:d79b\\t_eppc._tcp.local.=PTR,Developer\\226\\128\\153s\\ Mac\\ mini._eppc._tcp.local.\\t_net-assistant._udp.local.=PTR,Developer\\226\\128\\153s\\ Mac\\ mini._net-assistant._udp.local.\\t_rfb._tcp.local.=PTR,Developer\\226\\128\\153s\\ Mac\\ mini._rfb._tcp.local.\\t_sftp-ssh._tcp.local.=PTR,Developer\\226\\128\\153s\\ Mac\\ mini._sftp-ssh._tcp.local.\\t_smb._tcp.local.=PTR,Developer\\226\\128\\153s\\ Mac\\ mini._ssh._tcp.local.", "mdns.services": "ssh/tcp\\tsftp-ssh/tcp\\teppc/tcp\\trfb/tcp\\tsmb/tcp\\tnet-assistant/udp", "os.cpe23": "cpe:/o:apple:mac_os_x:10.15", "os.family": "Mac OS X", "os.product": "Mac OS X", "os.vendor": "Apple", "os.version": "10.15", "protocol": "mdns", "ts": "1597259740"}, "192.168.0.5/5900/tcp/": {"ip.flags": "DF", "ip.id": "0", "ip.tos": "0", "ip.ttl": "64", "protocol": "vnc", "source": "mdns", "syn.rtt": "625419", "tcp.options": "MSS:05b4", "tcp.ts": "2009549134", "tcp.urg": "0", "tcp.win": "65535", "ts": "1597259749", "vnc.version": "RFB 003.889"}, "credentials": {}, "rtts": {"icmp/echo": [541214]}, "attributes": {"_macs.ipmap": "f0:18:98:ee:7c:7b=192.168.0.5", "ip.ttl.hops": "0", "ip.ttl.host": "192.168.0.5", "ip.ttl.port": "22", "ip.ttl.source": "64", "ip.ttl.source.icmp": "64", "ip.ttl.win": "65535", "match.db": "mdns-device-info-txt", "match.score": "90", "ntlmssp.dnsComputer": "Developers-Mac-mini.local", "ntlmssp.dnsDomain": "local", "ntlmssp.version": "6.1.7600", "os.cpe23": "cpe:/o:apple:mac_os_x:10.15", "os.family": "Mac OS X", "os.product": "Mac OS X", "os.vendor": "Apple", "os.version": "10.15", "smb.guid": "ff12583f-5ba1-53b8-8ff3-a48a394056f7", "smb.nativeLM": "@(#)PROGRAM:smbd PROJECT:smbx-499.60.1", "smb.nativeOS": "Darwin"}, "service_count": 9, "service_count_tcp": 5, "service_count_udp": 2, "service_count_arp": 1, "service_count_icmp": 1, "lowest_ttl": 0, "lowest_rtt": 541214, "last_agent_id": "ca811190-329c-4da3-8cbe-3fd2ddff2663", "last_task_id": "de5a4176-3614-4b71-8939-95b9108124aa", "newest_mac": "f0:18:98:ee:7c:7b", "newest_mac_vendor": "Apple, Inc.", "newest_mac_age": 1513987200000000000, "comments": null, "service_ports_tcp": ["22", "445", "3031", "3283", "5900"], "service_ports_udp": ["137", "5353"], "service_protocols": ["mdns", "netbios", "smb1", "smb2", "smb3", "ssh", "vnc"], "service_products": ["openbsd openssh"], "org_name": "Test Lab", "site_name": "MAC", "agent_name": "TENTACULAR"}

```

Asset Data uses a number of data types for top-level fields, including string arrays, objects, strings, and integers. runZero tracks multiple IP addresses and MACs per asset and these are represented as arrays. For asset-level attributes and services, these are stored as objects with additional structure. Assets are uniquely identified by the `id` field (a V4 UUID) and nearly every

other field can be changed between scans, as assets move around the network, change IPs, and open and close services.

Every asset belongs to an organization and a site within that organization.

The core asset data fields are defined below.

| Field           | Description                                                                           |
|-----------------|---------------------------------------------------------------------------------------|
| id              | The unique ID of this asset defined as a v4 UUID.                                     |
| created_at      | The asset created time represented as a 64-bit Unix timestamp in seconds.             |
| updated_at      | The asset last update time represented as a 64-bit Unix timestamp in seconds.         |
| organization_id | The organization identifier defined as a v4 UUID.                                     |
| site_id         | The site identifier defined as a v4 UUID.                                             |
| alive           | A boolean indicating whether this asset was found during the last scan of the site.   |
| last_seen       | The time the asset last responded represented as a 64-bit Unix timestamp in seconds.  |
| first_seen      | The time the asset first responded represented as a 64-bit Unix timestamp in seconds. |
| detected_by     | The protocol used to first detect that this asset was alive during the last scan.     |
| type            | A classification that represents a guess of the asset's purpose.                      |
| os_vendor       | The operating system vendor name as determined by the fingerprinting engine.          |
| os_product      | The operating system product name as determined by the fingerprinting engine.         |
| os_version      | The operating system version as determined by the fingerprinting engine.              |
| os              | The operating system name as determined by the fingerprinting engine.                 |
| hw_vendor       | The hardware vendor name as determined by the fingerprinting engine.                  |
| hw_product      | The hardware product name as determined by the fingerprinting engine.                 |
| hw_version      | The hardware version as determined by the fingerprinting engine.                      |
| hw              | The hardware definition as determined by the fingerprinting engine.                   |
| addresses       | An array of IP (v4/v6) addresses for the asset that were within the scan scope.       |
| addresses_extra | An array of IP (v4/v6) addresses for the asset that were outside the scan scope.      |

|                     |                                                                                                 |
|---------------------|-------------------------------------------------------------------------------------------------|
| macs                | An array of MAC addresses associated with this asset.                                           |
| mac_vendors         | An array of MAC address vendors associated with this asset.                                     |
| names               | An array of unique hostnames associated with this asset (uppercase).                            |
| domains             | An array of unique domain names associated with this asset (uppercase).                         |
| tags                | A text representation of the user-specified tags associated with this asset.                    |
| attributes          | An object containing a map of key-value string attributes for this asset.                       |
| services            | An object containing each associated service with the key representing the service description. |
| credentials         | An object containing a map of any associated credentials (SNMP v2/v3).                          |
| rtts                | An object containing a map of round-trip measurement times in milliseconds.                     |
| service_count       | A count of TCP, UDP, ARP, and ICMP services.                                                    |
| service_count_tcp   | A count of TCP services.                                                                        |
| service_count_udp   | A count of UDP services.                                                                        |
| service_count_arp   | A count of ARP services (0 or 1).                                                               |
| service_count_icmp  | A count of ICMP services.                                                                       |
| software_count      | A count of software results.                                                                    |
| vulnerability_count | A count of vulnerability results.                                                               |
| lowest_ttl          | The lowest observed source TTL for this asset.                                                  |
| lowest_rtt          | The lowest observed source RTT for this asset.                                                  |
| last_agent_id       | The v4 UUID of the Explorer responsible for the last scan of this asset.                        |
| last_task_id        | The v4 UUID of the task responsible associated with the last scan of this asset.                |
| last_task_id        | The v4 UUID of the task responsible associated with the last scan of this asset.                |
| newest_mac          | The "newest" MAC address by registration date.                                                  |
| newest_mac_vendor   | The "newest" MAC address vendor by registration date.                                           |
| newest_mac_age      | The "newest" MAC address registration date as a Unix timestamp in nanoseconds.                  |
| comments            | User-specified comments associated with this asset.                                             |

|                   |                                                                                                   |
|-------------------|---------------------------------------------------------------------------------------------------|
| service_ports_tcp | An array of strings representing the unique TCP ports found on this asset.                        |
| service_ports_udp | An array of strings representing the unique UDP ports found on this asset.                        |
| service_protocols | An array of strings representing the unique protocols found on this asset.                        |
| service_products  | An array of strings representing the unique products found on this asset.                         |
| scanned           | A TRUE or FALSE value indicating whether the asset has been scanned by runZero.                   |
| source_ids        | The ID of the data source, mapped to <a href="#">this table</a> .                                 |
| eol_os            | The operating system End-of-Life time represented as a 64-bit Unix timestamp in seconds.          |
| eol_os_ext        | The operating system extended End-of-Life time represented as a 64-bit Unix timestamp in seconds. |
| outlier_score     | The 0-5 score range indicating how unusual an asset is compared to the rest of the inventory.     |
| outlier_raw       | The heuristic score indicating how unusual an asset is compared to the rest of the inventory.     |
| sources           | The name of the data source, mapped to <a href="#">this table</a> .                               |
| org_name          | The name of the organization associated with this asset.                                          |
| site_name         | The name of the site associated with this asset.                                                  |
| agent_name        | The name of the Explorer associated with this asset.                                              |
| agent_external_ip | The external IP address of the Explorer associated with this asset.                               |
| hosted_zone_name  | The name of the hosted zone associated with this asset.                                           |
| subnets           | The registered subnets associated with the site this asset is in.                                 |

The `services` field contains string keys that contain the unique service identifier with values stored as strings. Multiple values may be stored as tab-delimited strings in the service values. A typical service key looks like `192.168.0.5/22/tcp/`. The components of the service key name consist of address, port, transport, and virtual host (which can be blank).

## Change reports

The runZero platform calculates a **change report** after processing each scan. This is a JSON document available for download from the Task Details page with the following structure.

```
{
 "assets":{
```

```

"new":{ "<asset-UUID>": { "Asset Data Fields":"" } },
 "online":{ "<asset-UUID>": { "Asset Data Fields":"" } },
 "offline":{ "<asset-UUID>": { "Asset Data Fields":"" } },
 "changed":{ "<asset-UUID>": { "Asset Data Fields":"" } },
 "summary":{
 "changed":#,
 "new":#,
 "total":#,
 "unchanged":#
 } },
"directory_users":{
 "new":{ "<user-UUID>": { "User Data Fields":"" } },
 "changed":{ "<user-UUID>": { "User Data Fields":"" } },
 "summary":{
 "changed":#,
 "new":#,
 "total":#,
 "unchanged":#
 } },
"directory_groups":{
 "new":{ "<group-UUID>": { "Group Data Fields":"" } },
 "changed":{ "<group-UUID>": { "Group Data Fields":"" } },
 "summary":{
 "changed":#,
 "new":#,
 "total":#,
 "unchanged":#
 } },
 "truncated": <true/false>
}

```

The `new`, `online`, `offline`, and `changed` objects each contain keys consisting of the modified asset IDs with the values represented in the **asset data** format. The `summary` field indicates overall change statistics for this task. The `truncated` field is set to `true` if the change report is incomplete due to reaching the maximum change threshold (1000 asset changes today).

## runZero data dictionary

runZero discovers and catalogs the following fields from known protocols and services:

| Protocol | Attribute name        | Data type | Single-value |
|----------|-----------------------|-----------|--------------|
| acop     | acop.cellID           | string    | true         |
| acop     | acop.channelID        | string    | true         |
| acop     | acop.clientID         | string    | true         |
| acop     | acop.controllerName   | string    | true         |
| acop     | acop.controllerSerial | string    | true         |
| acop     | acop.controllerTime   | string    | true         |



|                |                             |         |      |
|----------------|-----------------------------|---------|------|
| acop           | acop.controllerVersion      | string  | true |
| acop           | acop.error                  | string  | true |
| acop           | acop.opVersion              | string  | true |
| acop           | acop.rbuType                | string  | true |
| acop           | acop.revision               | string  | true |
| acop           | acop.stationID              | string  | true |
| acop           | acop.stationName            | string  | true |
| acop           | acop.supplierCode           | string  | true |
| acop           | acop.supportLinking         | numeric | true |
| acop           | acop.supportSeqNum          | numeric | true |
| acop           | acop.systemSubtype          | string  | true |
| acop           | acop.systemType             | string  | true |
| acop           | acop.toolData               | string  | true |
| acop           | acop.toolVersion            | string  | true |
| activemq       | activemq.jvm.vendor         | string  | true |
| activemq       | activemq.jvm.version        | string  | true |
| activemq       | activemq.os                 | string  | true |
| activemq       | activemq.protocolVersion    | string  | true |
| activemq       | activemq.wireformat.data    | string  | true |
| activemq       | activemq.wireformat.version | string  | true |
| adb            | adb.access                  | string  | true |
| ads            | ads.deviceName              | string  | true |
| ads            | ads.routeCount              | numeric | true |
| ads            | ads.routes                  | string  | true |
| ads            | ads.version                 | string  | true |
| ads            | ads.versionBuild            | string  | true |
| ads            | ads.versionMajor            | string  | true |
| ads            | ads.versionMinor            | string  | true |
| advantech-adam | advantech.firmware          | string  | true |
| advantech-adam | advantech.ipv4              | string  | true |
| advantech-adam | advantech.mac               | string  | true |

|                |                                  |         |       |
|----------------|----------------------------------|---------|-------|
| advantech-adam | advantech.magic                  | string  | true  |
| advantech-adam | advantech.model                  | string  | true  |
| airplay        | airplay.build                    | numeric | true  |
| airplay        | airplay.canRecordScreenStream    | boolean | true  |
| airplay        | airplay.deviceID                 | string  | true  |
| airplay        | airplay.features                 | numeric | true  |
| airplay        | airplay.featuresEx               | string  | true  |
| airplay        | airplay.firmwareBuildDate        | string  | true  |
| airplay        | airplay.firmwareRevision         | numeric | true  |
| airplay        | airplay.hardwareRevision         | numeric | true  |
| airplay        | airplay.hasUDPMirroringSupport   | boolean | true  |
| airplay        | airplay.initialVolume            | numeric | true  |
| airplay        | airplay.keepAliveLowPower        | boolean | true  |
| airplay        | airplay.keepAliveSendStatsAsBody | boolean | true  |
| airplay        | airplay.macAddress               | string  | true  |
| airplay        | airplay.manufacturer             | string  | true  |
| airplay        | airplay.model                    | string  | true  |
| airplay        | airplay.name                     | string  | true  |
| airplay        | airplay.osBuildVersion           | string  | true  |
| airplay        | airplay.osinfo                   | string  | true  |
| airplay        | airplay.pi                       | uuid    | true  |
| airplay        | airplay.pk                       | string  | true  |
| airplay        | airplay.playbackCapabilities     | string  | false |
| airplay        | airplay.protocolVersion          | numeric | true  |
| airplay        | airplay.ptpinfo                  | string  | true  |
| airplay        | airplay.receiverHDRCapability    | string  | true  |
| airplay        | airplay.screenDemoMode           | boolean | true  |
| airplay        | airplay.sdk                      | string  | true  |
| airplay        | airplay.senderAddress            | string  | true  |
| airplay        | airplay.sourceVersion            | numeric | true  |
| airplay        | airplay.statusFlags              | string  | true  |

|         |                           |         |       |
|---------|---------------------------|---------|-------|
| airplay | airplay.supportedFormats  | string  | true  |
| airplay | airplay.volumeControlType | numeric | true  |
| airplay | airplay.vv                | numeric | true  |
| airplay | apache.serverInfo         | string  | true  |
| amqp    | amqp.capabilities         | string  | true  |
| amqp    | amqp.clusterName          | string  | true  |
| amqp    | amqp.copyright            | string  | true  |
| amqp    | amqp.frameType            | string  | true  |
| amqp    | amqp.information          | string  | true  |
| amqp    | amqp.locales              | string  | true  |
| amqp    | amqp.mechanisms           | string  | true  |
| amqp    | amqp.platform             | string  | true  |
| amqp    | amqp.product              | string  | true  |
| amqp    | amqp.protocolID           | string  | true  |
| amqp    | amqp.protocolVersion      | string  | true  |
| amqp    | amqp.saslRequired         | string  | true  |
| amqp    | amqp.serverVersion        | string  | true  |
| amqp    | amqp.tlsRequired          | string  | true  |
| amqp    | amqp.version              | string  | true  |
| atg     | atg.cpe                   | string  | true  |
| atg     | atg.functionCode          | string  | true  |
| atg     | atg.modile                | string  | true  |
| atg     | atg.module                | string  | true  |
| atg     | atg.products              | numeric | true  |
| atg     | atg.software              | string  | true  |
| atg     | atg.stationName           | string  | true  |
| atg     | atg.tankCount             | numeric | true  |
| atg     | atg.version               | string  | true  |
| bacnet  | bacnet.address            | string  | true  |
| bacnet  | bacnet.bbmd               | string  | false |
| bacnet  | bacnet.bbmdTable          | string  | true  |

|        |                                |         |       |
|--------|--------------------------------|---------|-------|
| bacnet | bacnet.databaseRevision        | string  | true  |
| bacnet | bacnet.decodeWarning           | string  | false |
| bacnet | bacnet.deviceCount             | numeric | true  |
| bacnet | bacnet.deviceType              | string  | true  |
| bacnet | bacnet.fdt                     | string  | false |
| bacnet | bacnet.foreignDeviceRegistered | string  | true  |
| bacnet | bacnet.foreignDeviceTable      | string  | true  |
| bacnet | bacnet.gateway                 | string  | true  |
| bacnet | bacnet.iHave                   | string  | false |
| bacnet | bacnet.instance                | string  | true  |
| bacnet | bacnet.instanceID              | string  | true  |
| bacnet | bacnet.maxAPDU                 | string  | true  |
| bacnet | bacnet.maxApdu                 | string  | true  |
| bacnet | bacnet.maxApduLengthAccepted   | string  | true  |
| bacnet | bacnet.modelName               | string  | true  |
| bacnet | bacnet.network                 | string  | true  |
| bacnet | bacnet.objectCount             | numeric | true  |
| bacnet | bacnet.pid                     | string  | false |
| bacnet | bacnet.port                    | numeric | true  |
| bacnet | bacnet.profileName             | string  | true  |
| bacnet | bacnet.protocolRevision        | string  | true  |
| bacnet | bacnet.readFailure             | string  | false |
| bacnet | bacnet.remote                  | string  | true  |
| bacnet | bacnet.remoteDevice            | string  | false |
| bacnet | bacnet.remoteDeviceCount       | numeric | true  |
| bacnet | bacnet.route                   | string  | true  |
| bacnet | bacnet.routerNetwork           | string  | false |
| bacnet | bacnet.routerNetworks          | string  | true  |
| bacnet | bacnet.routingEntries          | string  | true  |
| bacnet | bacnet.routingEntry            | string  | false |
| bacnet | bacnet.segmentation            | string  | true  |

|                 |                              |         |      |
|-----------------|------------------------------|---------|------|
| bacnet          | bacnet.segmentationSupported | numeric | true |
| bacnet          | bacnet.serialNumber          | string  | true |
| bacnet          | bacnet.systemStatus          | string  | true |
| bacnet          | bacnet.target                | string  | true |
| bacnet          | bacnet.vendorID              | string  | true |
| bacnet          | bacnet.vendorIDLookup        | string  | true |
| bacnet          | bacnet.vendorLookup          | string  | true |
| bedrock         | bedrock.contents             | string  | true |
| bedrock         | bedrock.guid                 | string  | true |
| bedrock         | bedrock.uptime               | string  | true |
| bgp             | bgp.addressFamilies          | string  | true |
| bgp             | bgp.asn                      | string  | true |
| bgp             | bgp.asn32                    | string  | true |
| bgp             | bgp.bgpID                    | string  | true |
| bgp             | bgp.capabilities             | string  | true |
| bgp             | bgp.capabilityCount          | numeric | true |
| bgp             | bgp.error                    | string  | true |
| bgp             | bgp.holdTime                 | string  | true |
| bgp             | bgp.identifier               | string  | true |
| bgp             | bgp.messageType              | string  | true |
| bgp             | bgp.notificationCode         | string  | true |
| bgp             | bgp.notificationCodeID       | string  | true |
| bgp             | bgp.notificationSubcode      | string  | true |
| bgp             | bgp.notificationSubcodeName  | string  | true |
| bgp             | bgp.subCode                  | string  | true |
| bgp             | bgp.version                  | string  | true |
| bitdefender-app | bitdefender.deviceID         | string  | true |
| bitdefender-app | bitdefender.hostname         | string  | true |
| bitdefender-app | bitdefender.model            | string  | true |
| bitdefender-app | bitdefender.os               | string  | true |
| bitdefender-app | bitdefender.type             | string  | true |

|                 |                             |        |      |
|-----------------|-----------------------------|--------|------|
| bitdefender-app | service.product             | string | true |
| bitdefender-app | service.vendor              | string | true |
| bitdefender-app | service.version             | string | true |
| bjnp            | bjnp.ipv4                   | string | true |
| bjnp            | bjnp.ipv6                   | string | true |
| bjnp            | bjnp.macAddress             | string | true |
| bjnp            | bjnp.type                   | string | true |
| brother         | brother.firmware.version    | string | true |
| brother         | brother.hostname            | string | true |
| brother         | brother.ip                  | string | true |
| brother         | brother.mainFirmwareVersion | string | true |
| brother         | brother.modelName           | string | true |
| brother         | brother.scanner             | string | true |
| brother         | brother.serialNbr           | string | true |
| brother         | brother.sub1FirmwareVersion | string | true |
| brother         | brother.sub3FirmwareVersion | string | true |
| cacti           | cacti.version               | string | true |
| cassandra       | cassandra.cluster           | string | true |
| cassandra       | cassandra.compression       | string | true |
| cassandra       | cassandra.confidence        | string | true |
| cassandra       | cassandra.cqlVersion        | string | true |
| cassandra       | cassandra.protocolVersions  | string | true |
| cassandra       | cassandra.version           | string | true |
| cdp             | cdp.addresses               | string | true |
| cdp             | cdp.capabilities            | string | true |
| cdp             | cdp.checksum                | string | true |
| cdp             | cdp.deviceID                | string | true |
| cdp             | cdp.isHost                  | string | true |
| cdp             | cdp.isIGMPCapable           | string | true |
| cdp             | cdp.isL2Switch              | string | true |
| cdp             | cdp.isL3Router              | string | true |

|            |                              |         |      |
|------------|------------------------------|---------|------|
| cdp        | cdp.isRepeater               | string  | true |
| cdp        | cdp.isSourceRouteBridge      | string  | true |
| cdp        | cdp.isTransparentBridge      | string  | true |
| cdp        | cdp.mgmtAddresses            | string  | true |
| cdp        | cdp.nativeVLAN               | string  | true |
| cdp        | cdp.osVersion                | string  | true |
| cdp        | cdp.platform                 | string  | true |
| cdp        | cdp.port                     | numeric | true |
| cdp        | cdp.portID                   | numeric | true |
| cdp        | cdp.powerConsumption         | string  | true |
| cdp        | cdp.sysName                  | string  | true |
| cdp        | cdp.target                   | string  | true |
| cdp        | cdp.ttl                      | string  | true |
| cdp        | cdp.version                  | string  | true |
| cdp        | cdp.voipVLAN                 | string  | true |
| chargen    | chargen.osGuess              | string  | true |
| checkmk    | banner                       | string  | true |
| checkmk    | checkmk.agentOS              | string  | true |
| checkmk    | checkmk.arch                 | string  | true |
| checkmk    | checkmk.buildDate            | string  | true |
| checkmk    | checkmk.hostname             | string  | true |
| checkmk    | checkmk.osName               | string  | true |
| checkmk    | checkmk.osType               | string  | true |
| checkmk    | checkmk.osVersion            | string  | true |
| checkmk    | checkmk.version              | string  | true |
| chromecast | chromecast.buildVersion      | numeric | true |
| chromecast | chromecast.capabilities      | string  | true |
| chromecast | chromecast.castBuildRevision | string  | true |
| chromecast | chromecast.connected         | bool    | true |
| chromecast | chromecast.ethernetConnected | bool    | true |
| chromecast | chromecast.generation        | numeric | true |

|            |                                                |         |       |
|------------|------------------------------------------------|---------|-------|
| chromecast | chromecast.hasUpdate                           | bool    | true  |
| chromecast | chromecast.hotspotBssid                        | string  | true  |
| chromecast | chromecast.ipAddress                           | string  | true  |
| chromecast | chromecast.locale                              | string  | true  |
| chromecast | chromecast.location.countryCode                | string  | true  |
| chromecast | chromecast.location.latitude                   | numeric | true  |
| chromecast | chromecast.location.longitude                  | numeric | true  |
| chromecast | chromecast.macAddress                          | string  | true  |
| chromecast | chromecast.name                                | string  | true  |
| chromecast | chromecast.optIn.crash                         | bool    | true  |
| chromecast | chromecast.optIn.opencast                      | bool    | true  |
| chromecast | chromecast.optIn.stats                         | bool    | true  |
| chromecast | chromecast.publicKey                           | string  | true  |
| chromecast | chromecast.releaseTrack                        | string  | true  |
| chromecast | chromecast.setupState                          | numeric | true  |
| chromecast | chromecast.setupStats.historicallySucceeded    | bool    | true  |
| chromecast | chromecast.setupStats.numCheckConnectivity     | numeric | true  |
| chromecast | chromecast.setupStats.numConnectWifi           | numeric | true  |
| chromecast | chromecast.setupStats.numConnectedWifiNotSaved | numeric | true  |
| chromecast | chromecast.setupStats.numInitialEurekaInfo     | numeric | true  |
| chromecast | chromecast.setupStats.numObtainIpnumeric       | true    | false |
| chromecast | chromecast.ssdPudn                             | string  | true  |
| chromecast | chromecast.ssid                                | string  | true  |
| chromecast | chromecast.timeFormat                          | numeric | true  |
| chromecast | chromecast.timezone                            | string  | true  |
| chromecast | chromecast.tosAccepted                         | bool    | true  |
| chromecast | chromecast.umaClientId                         | string  | true  |
| chromecast | chromecast.uptime                              | numeric | true  |
| chromecast | chromecast.version                             | numeric | true  |
| chromecast | chromecast.wpaConfigured                       | bool    | true  |
| chromecast | chromecast.wpaId                               | numeric | true  |



|            |                                 |         |       |
|------------|---------------------------------|---------|-------|
| chromecast | chromecast.wpaState             | numeric | true  |
| cip        | cip.address                     | string  | true  |
| cip        | cip.backplaneSlot               | string  | true  |
| cip        | cip.deviceType                  | string  | true  |
| cip        | cip.deviceTypeID                | string  | true  |
| cip        | cip.discoveredPorts             | numeric | true  |
| cip        | cip.ethernetLinkCount           | numeric | true  |
| cip        | cip.gateway                     | string  | true  |
| cip        | cip.gateway.ethernetLinkCount   | numeric | true  |
| cip        | cip.gateway.ips                 | string  | true  |
| cip        | cip.gateway.macs                | string  | true  |
| cip        | cip.gateway.tcpipInterfaceCount | numeric | true  |
| cip        | cip.id                          | string  | true  |
| cip        | cip.identityPort                | numeric | true  |
| cip        | cip.ips                         | string  | true  |
| cip        | cip.macPairs                    | string  | true  |
| cip        | cip.macs                        | string  | true  |
| cip        | cip.moduleCount                 | numeric | true  |
| cip        | cip.moduleStatus                | string  | true  |
| cip        | cip.modules                     | string  | false |
| cip        | cip.port                        | numeric | true  |
| cip        | cip.product                     | string  | true  |
| cip        | cip.productCode                 | string  | true  |
| cip        | cip.productCodeName             | string  | true  |
| cip        | cip.productID                   | string  | true  |
| cip        | cip.productName                 | string  | true  |
| cip        | cip.revision                    | string  | true  |
| cip        | cip.safetyEnumerationSuppressed | string  | true  |
| cip        | cip.safetySupervisor            | string  | true  |
| cip        | cip.serialNumber                | string  | true  |
| cip        | cip.serialNumberHex             | string  | true  |

|             |                                  |         |      |
|-------------|----------------------------------|---------|------|
| cip         | cip.slotStatus                   | string  | true |
| cip         | cip.state                        | string  | true |
| cip         | cip.stateCode                    | string  | true |
| cip         | cip.status                       | string  | true |
| cip         | cip.statusCode                   | string  | true |
| cip         | cip.supportsCommunicationService | numeric | true |
| cip         | cip.supportsTCPEncapsulation     | numeric | true |
| cip         | cip.supportsUDPEncapsulation     | numeric | true |
| cip         | cip.target                       | string  | true |
| cip         | cip.tcpipInterfaceCount          | numeric | true |
| cip         | cip.vendor                       | string  | true |
| cip         | cip.vendorID                     | string  | true |
| cisco-phone | ciscocp.applloadid               | string  | true |
| cisco-phone | ciscocp.bootloadid               | string  | true |
| cisco-phone | ciscocp.hostname                 | string  | true |
| cisco-phone | ciscocp.mac                      | string  | true |
| cisco-phone | ciscocp.messagewaiting           | string  | true |
| cisco-phone | ciscocp.model                    | string  | true |
| cisco-phone | ciscocp.phonedn                  | string  | true |
| cisco-phone | ciscocp.revision                 | string  | true |
| cisco-phone | ciscocp.serial                   | string  | true |
| cisco-phone | ciscocp.series                   | string  | true |
| cisco-phone | ciscocp.udi                      | string  | true |
| cisco-phone | ciscocp.version                  | string  | true |
| ciscosmi    | ciscosmi.reply                   | string  | true |
| coap        | coap.contentFormat               | string  | true |
| coap        | coap.contentUnknown              | string  | true |
| coap        | coap.encodedPayload              | string  | true |
| coap        | coap.jsonData.cid                | string  | true |
| coap        | coap.jsonData.name               | string  | true |
| coap        | coap.jsonData.type               | string  | true |

|          |                     |        |       |
|----------|---------------------|--------|-------|
| coap     | coap.messageID      | string | true  |
| coap     | coap.options        | string | true  |
| coap     | coap.payload        | string | true  |
| coap     | coap.resources      | string | true  |
| coap     | coap.responseCode   | string | true  |
| coap     | coap.token          | string | true  |
| coap     | coap.type           | string | true  |
| coap     | coap.version        | string | true  |
| cockpit  | cockpit.os          | string | true  |
| cockpit  | cockpit.variant     | string | true  |
| cockpit  | cockpit.variantID   | string | true  |
| cockpit  | host.name           | string | false |
| codesys  | codesys.banner      | string | true  |
| codesys  | codesys.cpe23       | string | true  |
| codesys  | codesys.format      | string | true  |
| codesys  | codesys.osName      | string | true  |
| codesys  | codesys.osType      | string | true  |
| codesys  | codesys.productType | string | true  |
| codesys  | codesys.version     | string | true  |
| codesys2 | codesys2.byteOrder  | string | true  |
| codesys2 | codesys2.osType     | string | true  |
| codesys2 | codesys2.osVersion  | string | true  |
| codesys2 | codesys2.vendor     | string | true  |
| control  | dnsServer           | string | true  |
| control  | gateway             | string | true  |
| control  | hostname            | string | true  |
| control  | ipAddress           | string | true  |
| control  | macAddress          | string | true  |
| control  | manufacturer        | string | true  |
| control  | modelName           | string | true  |
| control  | modelNumber         | string | true  |

|            |                        |         |      |
|------------|------------------------|---------|------|
| control    | netmask                | string  | true |
| control    | osVersion              | string  | true |
| control    | serialNumber           | string  | true |
| confluence | confluence.baseURL     | string  | true |
| confluence | confluence.build       | string  | true |
| confluence | confluence.version     | string  | true |
| consul     | consul.config.version  | string  | true |
| couchdb    | couchdb.version        | string  | true |
| crestron   | crestron.banner        | string  | true |
| crestron   | crestron.buildDate     | string  | true |
| crestron   | crestron.hostname      | string  | true |
| crestron   | crestron.id            | string  | true |
| crestron   | crestron.mac           | string  | true |
| crestron   | crestron.model         | string  | true |
| crestron   | crestron.version       | string  | true |
| crimsonv3  | crimsonv3.manufacturer | string  | true |
| crimsonv3  | crimsonv3.model        | string  | true |
| crowd      | crowd.version          | string  | true |
| crowd      | service.product        | string  | true |
| crowd      | service.vendor         | string  | true |
| crowd      | service.version        | string  | true |
| daytime    | daytime.osGuess        | string  | true |
| daytime    | daytime.timestamp      | string  | true |
| db2        | db2.database           | string  | true |
| db2        | db2.extnam             | string  | true |
| db2        | db2.hostname           | string  | true |
| db2        | db2.instance           | string  | true |
| db2        | db2.nodeName           | string  | true |
| db2        | db2.port               | numeric | true |
| db2        | db2.raw                | string  | true |

|        |                          |         |      |
|--------|--------------------------|---------|------|
| db2    | db2.serverType           | string  | true |
| db2    | db2.serviceName          | string  | true |
| db2    | db2.srvnam               | string  | true |
| db2    | db2.srvrslsv             | string  | true |
| db2    | db2.version              | string  | true |
| dcerpc | dcerpc.accepted          | string  | true |
| dcerpc | dcerpc.acceptedContexts  | numeric | true |
| dcerpc | dcerpc.assocGroup        | string  | true |
| dcerpc | dcerpc.authLength        | string  | true |
| dcerpc | dcerpc.byteOrder         | string  | true |
| dcerpc | dcerpc.callID            | string  | true |
| dcerpc | dcerpc.dataRep           | string  | true |
| dcerpc | dcerpc.fragLength        | string  | true |
| dcerpc | dcerpc.interface         | string  | true |
| dcerpc | dcerpc.interfaceUUID     | string  | true |
| dcerpc | dcerpc.interfaceVersion  | string  | true |
| dcerpc | dcerpc.maxRecvFrag       | string  | true |
| dcerpc | dcerpc.maxXmitFrag       | string  | true |
| dcerpc | dcerpc.oxid.addresses    | string  | true |
| dcerpc | dcerpc.oxid.bindingCount | numeric | true |
| dcerpc | dcerpc.oxid.bindings     | string  | true |
| dcerpc | dcerpc.oxid.comVersion   | string  | true |
| dcerpc | dcerpc.oxid.error        | string  | true |
| dcerpc | dcerpc.oxid.machineName  | string  | true |
| dcerpc | dcerpc.oxid.security     | string  | true |
| dcerpc | dcerpc.pduType           | string  | true |
| dcerpc | dcerpc.port              | numeric | true |
| dcerpc | dcerpc.rejectReason      | string  | true |
| dcerpc | dcerpc.rejectReasonCode  | string  | true |
| dcerpc | dcerpc.rejectedContexts  | numeric | true |
| dcerpc | dcerpc.secondaryAddr     | string  | true |

|        |                                   |         |       |
|--------|-----------------------------------|---------|-------|
| dcerpc | dcerpc.serviceCount               | numeric | true  |
| dcerpc | dcerpc.services                   | string  | true  |
| dcerpc | dcerpc.target                     | string  | true  |
| dcerpc | dcerpc.version                    | string  | true  |
| dcerpc | epm.addrLen                       | numeric | true  |
| dcerpc | epm.address                       | string  | true  |
| dcerpc | epm.assocGroup                    | string  | true  |
| dcerpc | epm.authRequired                  | string  | true  |
| dcerpc | epm.dataRep                       | string  | true  |
| dcerpc | epm.maxRecvFrag                   | numeric | true  |
| dcerpc | epm.maxSendFrag                   | numeric | true  |
| dcerpc | epm.oxid.addresses                | string  | true  |
| dcerpc | epm.oxid.security                 | string  | true  |
| dcerpc | epm.oxidVersion                   | string  | true  |
| dcerpc | epm.version                       | string  | true  |
| dhcp   | dhcp.authentication               | string  | true  |
| dhcp   | dhcp.bootFile                     | string  | true  |
| dhcp   | dhcp.broadcast                    | string  | true  |
| dhcp   | dhcp.broadcastAddress             | string  | true  |
| dhcp   | dhcp.class                        | string  | false |
| dhcp   | dhcp.clientFqdn                   | string  | true  |
| dhcp   | dhcp.clientIP                     | string  | true  |
| dhcp   | dhcp.clientMAC                    | string  | true  |
| dhcp   | dhcp.clientSystemArchitecture     | string  | false |
| dhcp   | dhcp.clientSystemArchitectureCode | string  | true  |
| dhcp   | dhcp.clientid.enterprise          | string  | false |
| dhcp   | dhcp.domainName                   | string  | true  |
| dhcp   | dhcp.domainNameServer             | string  | true  |
| dhcp   | dhcp.flags                        | string  | true  |
| dhcp   | dhcp.hardwareAddressLength        | string  | true  |
| dhcp   | dhcp.hardwareType                 | string  | true  |

|          |                             |         |      |
|----------|-----------------------------|---------|------|
| dhcp     | dhcp.hops                   | string  | true |
| dhcp     | dhcp.hostname               | string  | true |
| dhcp     | dhcp.isRequest              | string  | true |
| dhcp     | dhcp.leaseTime              | string  | true |
| dhcp     | dhcp.leaseTimeFormatted     | string  | true |
| dhcp     | dhcp.maximumMessageSize     | string  | true |
| dhcp     | dhcp.message                | string  | true |
| dhcp     | dhcp.messageType            | string  | true |
| dhcp     | dhcp.messageTypeCode        | string  | true |
| dhcp     | dhcp.nextServerIP           | string  | true |
| dhcp     | dhcp.ntpServers             | string  | true |
| dhcp     | dhcp.offeredIP              | string  | true |
| dhcp     | dhcp.op                     | string  | true |
| dhcp     | dhcp.port                   | numeric | true |
| dhcp     | dhcp.rebindingTime          | string  | true |
| dhcp     | dhcp.rebindingTimeFormatted | string  | true |
| dhcp     | dhcp.relayAgentIP           | string  | true |
| dhcp     | dhcp.relayAgentInformation  | string  | true |
| dhcp     | dhcp.renewalTime            | string  | true |
| dhcp     | dhcp.renewalTimeFormatted   | string  | true |
| dhcp     | dhcp.router                 | string  | true |
| dhcp     | dhcp.seconds                | string  | true |
| dhcp     | dhcp.serverIdentifier       | string  | true |
| dhcp     | dhcp.serverName             | string  | true |
| dhcp     | dhcp.subnetMask             | string  | true |
| dhcp     | dhcp.target                 | string  | true |
| dhcp     | dhcp.tftpServerAddress      | string  | true |
| dhcp     | dhcp.tftpServerName         | string  | true |
| dhcp     | dhcp.transactionID          | string  | true |
| dhcp     | dhcp.vendorClass            | string  | true |
| diameter | diameter.cpe                | string  | true |

|               |                           |        |      |
|---------------|---------------------------|--------|------|
| diameter      | diameter.example.com      | string | true |
| diameter      | diameter.firmwareRevision | string | true |
| diameter      | diameter.originHost       | string | true |
| diameter      | diameter.originRealm      | string | true |
| diameter      | diameter.productName      | string | true |
| diameter      | diameter.resultCode       | string | true |
| diameter      | diameter.vendor           | string | true |
| diameter      | diameter.vendorID         | string | true |
| diametersctp  | diameter.cpe              | string | true |
| diametersctp  | diameter.firmwareRevision | string | true |
| diametersctp  | diameter.originHost       | string | true |
| diametersctp  | diameter.originRealm      | string | true |
| diametersctp  | diameter.productName      | string | true |
| diametersctp  | diameter.vendor           | string | true |
| diametersctp  | diameter.vendorID         | string | true |
| digi-realport | realport.banner           | string | true |
| digi-realport | realport.baud             | int    | true |
| digi-realport | realport.name             | string | true |
| digi-realport | realport.portCount        | int    | true |
| digi-realport | realport.serialPort       | int    | true |
| digi-realport | realport.version          | string | true |
| dnp3          | dnp3.destination          | string | true |
| dnp3          | dnp3.deviceID             | string | true |
| dnp3          | dnp3.deviceName           | string | true |
| dnp3          | dnp3.hardwareVersion      | string | true |
| dnp3          | dnp3.location             | string | true |
| dnp3          | dnp3.manufacturer         | string | true |
| dnp3          | dnp3.model                | string | true |
| dnp3          | dnp3.owner                | string | true |
| dnp3          | dnp3.serialNumber         | string | true |
| dnp3          | dnp3.softwareVersion      | string | true |



|      |                   |         |       |
|------|-------------------|---------|-------|
| dnp3 | dnp3.source       | string  | true  |
| dns  | dns.addrS         | string  | true  |
| dns  | dns.ancount       | numeric | true  |
| dns  | dns.answerAddrS   | string  | true  |
| dns  | dns.answerNames   | string  | true  |
| dns  | dns.answerTXTs    | string  | true  |
| dns  | dns.answerTypes   | string  | true  |
| dns  | dns.authors.bind  | string  | true  |
| dns  | dns.edns0         | string  | true  |
| dns  | dns.edns0.cookie  | string  | true  |
| dns  | dns.edns0.nsid    | string  | true  |
| dns  | dns.edns0.subnet  | string  | true  |
| dns  | dns.edns0.udpSize | string  | true  |
| dns  | dns.edns0.unknown | string  | true  |
| dns  | dns.edns0.version | string  | true  |
| dns  | dns.flags         | string  | true  |
| dns  | dns.hostname.bind | string  | true  |
| dns  | dns.hostnameBind  | string  | true  |
| dns  | dns.id            | string  | true  |
| dns  | dns.id.server     | string  | true  |
| dns  | dns.idServer      | string  | true  |
| dns  | dns.isRequest     | string  | true  |
| dns  | dns.meraki        | string  | true  |
| dns  | dns.opcode        | string  | true  |
| dns  | dns.ptr           | string  | false |
| dns  | dns.qdcount       | numeric | true  |
| dns  | dns.rcode         | string  | true  |
| dns  | dns.recursion     | string  | true  |
| dns  | dns.resolvers     | string  | true  |
| dns  | dns.resolves      | string  | true  |
| dns  | dns.rttS          | numeric | true  |

|        |                              |         |       |
|--------|------------------------------|---------|-------|
| dns    | dns.subnets                  | numeric | true  |
| dns    | dns.telco.epc                | string  | false |
| dns    | dns.tracer                   | string  | false |
| dns    | dns.tracer.edns0             | string  | false |
| dns    | dns.transport                | string  | true  |
| dns    | dns.version                  | string  | true  |
| dns    | dns.version.bind             | string  | true  |
| dns    | dns.version.server           | string  | true  |
| dns    | dns.versionBind              | string  | true  |
| docker | docker.architecture          | string  | true  |
| docker | docker.dockerRootDir         | string  | true  |
| docker | docker.kernelVersion         | string  | true  |
| docker | docker.memTotal              | string  | true  |
| docker | docker.name                  | string  | true  |
| docker | docker.ncpu                  | string  | true  |
| docker | docker.operatingSystem       | string  | true  |
| docker | docker.ostype                | string  | true  |
| docker | docker.systemTime            | string  | true  |
| docker | docker.version               | string  | true  |
| dtls   | dtls.alert                   | string  | true  |
| dtls   | dtls.contentType             | string  | true  |
| dtls   | dtls.cookieLength            | string  | true  |
| dtls   | dtls.encapsulation           | string  | true  |
| dtls   | dtls.handshakeFragmentLength | string  | true  |
| dtls   | dtls.handshakeFragmentOffset | string  | true  |
| dtls   | dtls.handshakeLength         | string  | true  |
| dtls   | dtls.handshakeMessageSeq     | string  | true  |
| dtls   | dtls.handshakeType           | string  | true  |
| dtls   | dtls.recordEpoch             | string  | true  |
| dtls   | dtls.recordLength            | string  | true  |
| dtls   | dtls.recordSequence          | string  | true  |

|               |                              |        |       |
|---------------|------------------------------|--------|-------|
| dtls          | dtls.serverVersion           | string | true  |
| dtls          | dtls.unknown                 | string | true  |
| dtls          | dtls.verifyCookie            | string | true  |
| dtls          | dtls.version                 | string | true  |
| echo          | echo.data                    | string | true  |
| echo          | echo.match                   | string | true  |
| echo          | echo.probeHex                | string | true  |
| echo          | echo.probeLength             | string | true  |
| echo          | echo.responseHex             | string | true  |
| echo          | echo.responseLength          | string | true  |
| elasticsearch | elasticsearch.version.number | string | true  |
| epo           | epo.guid                     | string | true  |
| epo           | epo.hostname                 | string | true  |
| epo           | epo.server                   | string | true  |
| epo           | epo.version                  | string | true  |
| epo           | mcafeeAgent.hostname         | string | true  |
| epo           | mcafeeAgent.version          | string | true  |
| erlangdp      | epmd.details                 | string | false |
| erlangdp      | epmd.names                   | string | false |
| erlangdp      | erldp.challenge              | string | true  |
| erlangdp      | erldp.flags                  | string | true  |
| erlangdp      | erldp.hostname               | string | true  |
| erlangdp      | erldp.node                   | string | true  |
| erlangdp      | erldp.status                 | string | true  |
| erlangdp      | erldp.version                | string | true  |
| etcd2         | etcd2.access                 | string | true  |
| etcd2         | etcd2.keys                   | string | true  |
| ethercat      | ethercat.ado                 | string | true  |
| ethercat      | ethercat.adp                 | string | true  |
| ethercat      | ethercat.command             | string | true  |
| ethercat      | ethercat.commandType         | string | true  |

|          |                          |         |      |
|----------|--------------------------|---------|------|
| ethercat | ethercat.datagramCount   | numeric | true |
| ethercat | ethercat.detected        | boolean | true |
| ethercat | ethercat.index           | string  | true |
| ethercat | ethercat.workingCounter  | numeric | true |
| fgfm     | fgfm.deviceName          | string  | true |
| fgfm     | fgfm.firmwareVersion     | string  | true |
| fgfm     | fgfm.mgmtIP              | string  | true |
| fgfm     | fgfm.platform            | string  | true |
| fgfm     | fgfm.serialNumber        | string  | true |
| firebird | firebird.opcode          | string  | true |
| firebird | firebird.protocolVersion | string  | true |
| firebird | firebird.version         | string  | true |
| fox      | fox.appName              | string  | true |
| fox      | fox.appVersion           | string  | true |
| fox      | fox.authAgent            | string  | true |
| fox      | fox.brandId              | string  | true |
| fox      | fox.cpe                  | string  | true |
| fox      | fox.hostAddress          | string  | true |
| fox      | fox.hostId               | string  | true |
| fox      | fox.hostname             | string  | true |
| fox      | fox.language             | string  | true |
| fox      | fox.osName               | string  | true |
| fox      | fox.osVersion            | string  | true |
| fox      | fox.stationName          | string  | true |
| fox      | fox.sysInfo              | string  | true |
| fox      | fox.timeZone             | string  | true |
| fox      | fox.vendor               | string  | true |
| fox      | fox.version              | string  | true |
| fox      | fox.vmName               | string  | true |
| fox      | fox.vmUuid               | string  | true |
| fox      | fox.vmVersion            | string  | true |

|            |                               |         |      |
|------------|-------------------------------|---------|------|
| gesrtp     | gesrtp.cpe                    | string  | true |
| gesrtp     | gesrtp.deviceIndicator        | string  | true |
| gesrtp     | gesrtp.plcName                | string  | true |
| git        | git.branches                  | string  | true |
| git        | git.capabilities              | string  | true |
| git        | git.error                     | string  | true |
| git        | git.headRef                   | string  | true |
| git        | git.protocolVersion           | string  | true |
| git        | git.refs                      | string  | true |
| git        | git.serverSoftware            | string  | true |
| git        | git.tags                      | string  | true |
| git-http   | gitlab.manifest.hash          | string  | true |
| googlewifi | googleWifi.software.version   | string  | true |
| googlewifi | googleWifi.system.countryCode | string  | true |
| googlewifi | googleWifi.system.hardwareID  | string  | true |
| googlewifi | googleWifi.system.modelID     | string  | true |
| googlewifi | googleWifi.wan.gateway        | string  | true |
| googlewifi | googleWifi.wan.localIP        | string  | true |
| gtpc       | gtpc.restartCounter           | numeric | true |
| gtpc       | gtpc.teid                     | string  | true |
| gtpc       | gtpc.teidPresent              | string  | true |
| gtpc       | gtpc.version                  | string  | true |
| gtpprime   | gtpprime.detected             | boolean | true |
| gtpprime   | gtpprime.pt                   | string  | true |
| gtpprime   | gtpprime.restartCounter       | numeric | true |
| gtpprime   | gtpprime.teid                 | string  | true |
| gtpprime   | gtpprime.version              | string  | true |
| gtpu       | gtpu.detected                 | boolean | true |
| gtpu       | gtpu.pt                       | string  | true |
| gtpu       | gtpu.restartCounter           | numeric | true |
| gtpu       | gtpu.sequenceNumber           | string  | true |

|               |                                   |        |       |
|---------------|-----------------------------------|--------|-------|
| gtpu          | gtpu.teid                         | string | true  |
| gtpu          | gtpu.version                      | string | true  |
| hartip        | hartip.inactivityTimeout          | string | true  |
| hartip        | hartip.masterType                 | string | true  |
| hartip        | hartip.messageType                | string | true  |
| hartip        | hartip.status                     | string | true  |
| hartip        | hartip.statusDescription          | string | true  |
| hartip        | hartip.transactionID              | string | true  |
| hartip        | hartip.version                    | string | true  |
| hiddiscoveryd | hiddiscoveryd.address             | string | true  |
| hiddiscoveryd | hiddiscoveryd.hostname            | string | true  |
| hiddiscoveryd | hiddiscoveryd.mac                 | string | true  |
| hiddiscoveryd | hiddiscoveryd.model               | string | true  |
| hiddiscoveryd | hiddiscoveryd.unpatchedVertXploit | string | true  |
| hiddiscoveryd | hiddiscoveryd.version             | string | true  |
| hiddiscoveryd | hiddiscoveryd.versionDate         | string | true  |
| hikvision     | hikvision.codebase.lastModified   | string | true  |
| hikvision     | hikvision.codebase.platform       | string | true  |
| hikvision     | hikvision.codebase.plugins        | string | false |
| hikvision     | hikvision.deviceCode              | string | true  |
| hikvision     | hikvision.deviceType              | string | true  |
| hikvision     | hikvision.firmwareVersion         | string | true  |
| hikvision     | hikvision.firmwareVersionEmbedded | string | true  |
| hikvision     | hikvision.loginPageLastModified   | string | true  |
| hikvision     | hikvision.modelFamily             | string | true  |
| hikvision     | hikvision.pluginVersion           | string | true  |
| hikvision     | hikvision.realm                   | string | true  |
| hikvision     | hikvision.webVersion              | string | true  |
| http          | html.copyright                    | string | true  |
| http          | html.favicon                      | string | true  |
| http          | html.generator                    | string | true  |

|      |                                           |         |      |
|------|-------------------------------------------|---------|------|
| http | html.title                                | string  | true |
| http | http.body                                 | numeric | true |
| http | http.body.mmh3                            | string  | true |
| http | http.code                                 | numeric | true |
| http | http.contentLength                        | string  | true |
| http | http.contentType                          | string  | true |
| http | http.head.acceptRanges                    | string  | true |
| http | http.head.cacheControl                    | string  | true |
| http | http.head.contentLength                   | numeric | true |
| http | http.head.contentSecurityPolicy           | string  | true |
| http | http.head.contentType                     | string  | true |
| http | http.head.cookieNames                     | string  | true |
| http | http.head.date                            | string  | true |
| http | http.head.etag                            | string  | true |
| http | http.head.expires                         | string  | true |
| http | http.head.faviconPath                     | string  | true |
| http | http.head.lastModified                    | string  | true |
| http | http.head.location                        | string  | true |
| http | http.head.microsoftsharepointteamservices | string  | true |
| http | http.head.server                          | string  | true |
| http | http.head.setCookie                       | string  | true |
| http | http.head.spiislatency                    | string  | true |
| http | http.head.vary                            | string  | true |
| http | http.head.wwwAuthenticate                 | string  | true |
| http | http.head.xConfluenceRequestTime          | string  | true |
| http | http.head.xInfluxdbBuild                  | string  | true |
| http | http.head.xInfluxdbVersion                | string  | true |
| http | http.head.xJenkins                        | string  | true |
| http | http.head.xNtnxEnv                        | string  | true |
| http | http.head.xOracleDmsEcid                  | string  | true |
| http | http.head.xOracleDmsRid                   | string  | true |

|      |                                  |         |      |
|------|----------------------------------|---------|------|
| http | http.head.xPoweredBy             | string  | true |
| http | http.head.xSharepointhealthscore | string  | true |
| http | http.head.xconfluencerequesttime | string  | true |
| http | http.head.xframeoptions          | string  | true |
| http | http.location                    | string  | true |
| http | http.message                     | string  | true |
| http | http.method                      | string  | true |
| http | http.ntlm.challenge              | string  | true |
| http | http.ntlm.path                   | string  | true |
| http | http.owa.version                 | string  | true |
| http | http.owa.version.full            | string  | true |
| http | http.path                        | string  | true |
| http | http.port                        | numeric | true |
| http | http.server                      | string  | true |
| http | http.setCookie                   | string  | true |
| http | http.sonicwall.hardware          | string  | true |
| http | http.sonicwall.softwareVersion   | string  | true |
| http | http.status                      | string  | true |
| http | http.statusCode                  | string  | true |
| http | http.target                      | string  | true |
| http | http.technologies                | string  | true |
| http | http.title                       | string  | true |
| http | http.uri                         | string  | true |
| http | http.url                         | string  | true |
| http | http.version                     | string  | true |
| http | http.vmware.thumbprint           | string  | true |
| http | http.vmware.username             | string  | true |
| http | landesk.configPath               | string  | true |
| http | landesk.providerVersion          | string  | true |
| http | landesk.serverVersion            | string  | true |
| iax2 | iax2.destCallNumber              | string  | true |



|       |                       |         |       |
|-------|-----------------------|---------|-------|
| iax2  | iax2.detected         | boolean | true  |
| iax2  | iax2.isseqno          | string  | true  |
| iax2  | iax2.oseqno           | string  | true  |
| iax2  | iax2.response         | string  | true  |
| iax2  | iax2.sourceCallNumber | string  | true  |
| iax2  | iax2.subclass         | string  | true  |
| iax2  | iax2.subclassName     | string  | true  |
| iax2  | iax2.ts               | numeric | true  |
| icmp  | icmp.addr             | string  | true  |
| icmp  | icmp.rtt              | string  | true  |
| icmp  | icmp.typeCode         | string  | true  |
| icmp  | icmp.typeCodeN        | string  | true  |
| icmp  | icmp6.addr            | string  | true  |
| icmp  | icmp6.rtt             | string  | true  |
| icmp  | ip.tos                | string  | false |
| icmp  | ip.ttl                | string  | false |
| ident | ident.error           | string  | true  |
| ident | ident.opSys           | string  | true  |
| ident | ident.osGuess         | string  | true  |
| ident | ident.username        | string  | true  |
| idrac | idrac.adEnabled       | boolean | true  |
| idrac | idrac.fwVersion       | string  | true  |
| idrac | idrac.gui.titleBar    | string  | true  |
| idrac | idrac.hostname        | string  | true  |
| idrac | idrac.isOEMBranded    | boolean | true  |
| idrac | idrac.license         | string  | true  |
| idrac | idrac.name            | string  | true  |
| idrac | idrac.oem.hostname    | string  | true  |
| idrac | idrac.serverGen       | string  | true  |
| idrac | idrac.ssoEnabled      | boolean | true  |
| idrac | idrac.status          | string  | true  |

|                |                         |         |      |
|----------------|-------------------------|---------|------|
| idrac          | idrac.sysDesc           | string  | true |
| idrac          | idrac.systemLockdown    | string  | true |
| idrac          | idrac.systemModelName   | string  | true |
| idrac          | idrac.tfaEnabled        | boolean | true |
| idrac          | idrac.version           | string  | true |
| idrac          | service.product         | string  | true |
| idrac          | service.vendor          | string  | true |
| idrac          | service.version         | string  | true |
| iec104         | iec104.asduCOT          | string  | true |
| iec104         | iec104.asduCommonAddr   | string  | true |
| iec104         | iec104.asduCommonAddrs  | string  | true |
| iec104         | iec104.asduInfoObjAddr  | string  | true |
| iec104         | iec104.asduNegative     | string  | true |
| iec104         | iec104.asduTest         | string  | true |
| iec104         | iec104.asduTypes        | string  | true |
| iec104         | iec104.command          | string  | true |
| iec104         | iec104.commands         | string  | true |
| iec104         | iec104.frameCount       | numeric | true |
| iec104         | iec104.frameType        | string  | true |
| iec104         | iec104.frameTypes       | string  | true |
| iec104         | iec104.initCause        | string  | true |
| iec104         | iec104.initLocalFlag    | string  | true |
| iec104         | iec104.startDTConfirmed | string  | true |
| iec104         | iec104.testFRConfirmed  | string  | true |
| igel-discovery | igel.firmwareVersion    | string  | true |
| igel-discovery | igel.hostname           | string  | true |
| igel-discovery | igel.ipAddress          | string  | true |
| igel-discovery | igel.macAddress         | string  | true |
| igel-discovery | igel.osType             | string  | true |
| igel-discovery | igel.productID          | string  | true |
| igel-discovery | igel.productName        | string  | true |

|     |                       |         |      |
|-----|-----------------------|---------|------|
| iis | http.owa.version      | string  | true |
| iis | http.owa.version.full | string  | true |
| iis | rdg.authScheme        | string  | true |
| iis | rdg.transport         | string  | true |
| ike | ike.authMethod        | string  | true |
| ike | ike.dhGroup           | string  | true |
| ike | ike.encryptionAlgo    | string  | true |
| ike | ike.esn               | string  | true |
| ike | ike.exchangeType      | string  | true |
| ike | ike.flags             | string  | true |
| ike | ike.hashAlgo          | string  | true |
| ike | ike.initiatorSPI      | string  | true |
| ike | ike.integrityAlgo     | string  | true |
| ike | ike.keyLength         | string  | true |
| ike | ike.lifeDuration      | string  | true |
| ike | ike.lifeType          | string  | true |
| ike | ike.messageID         | string  | true |
| ike | ike.messageLength     | string  | true |
| ike | ike.nextPayload       | string  | true |
| ike | ike.nonESPMarker      | string  | true |
| ike | ike.notifyTypes       | string  | true |
| ike | ike.payload           | string  | true |
| ike | ike.payloadCount      | numeric | true |
| ike | ike.prfAlgo           | string  | true |
| ike | ike.reply             | string  | true |
| ike | ike.responderSPI      | string  | true |
| ike | ike.saProtocol        | string  | true |
| ike | ike.sha1              | string  | true |
| ike | ike.vendorID          | string  | true |
| ike | ike.vendorIDCount     | numeric | true |
| ike | ike.version           | string  | true |

|          |                           |         |       |
|----------|---------------------------|---------|-------|
| influxdb | influxdb.build            | string  | true  |
| influxdb | influxdb.databases        | string  | false |
| influxdb | influxdb.version          | string  | true  |
| ipmi     | ipmi.additionalSupport    | numeric | true  |
| ipmi     | ipmi.additionalSupportRaw | numeric | true  |
| ipmi     | ipmi.authAnon             | string  | true  |
| ipmi     | ipmi.authNull             | string  | true  |
| ipmi     | ipmi.authTypes            | string  | true  |
| ipmi     | ipmi.auxFirmwareRev       | string  | true  |
| ipmi     | ipmi.channel              | string  | true  |
| ipmi     | ipmi.cipherZero           | string  | true  |
| ipmi     | ipmi.ciphers              | string  | true  |
| ipmi     | ipmi.ciphersPageError     | string  | true  |
| ipmi     | ipmi.ciphersRaw           | string  | true  |
| ipmi     | ipmi.ciphersText          | string  | true  |
| ipmi     | ipmi.command              | string  | true  |
| ipmi     | ipmi.completionCode       | string  | true  |
| ipmi     | ipmi.completionCodeText   | string  | true  |
| ipmi     | ipmi.connVersions         | string  | true  |
| ipmi     | ipmi.connVersionsRaw      | string  | true  |
| ipmi     | ipmi.deviceID             | string  | true  |
| ipmi     | ipmi.deviceRevision       | string  | true  |
| ipmi     | ipmi.deviceRevisionRaw    | string  | true  |
| ipmi     | ipmi.fakeUsername         | string  | true  |
| ipmi     | ipmi.firmware             | string  | true  |
| ipmi     | ipmi.firmwareRevision     | string  | true  |
| ipmi     | ipmi.guid.mac             | string  | true  |
| ipmi     | ipmi.guid.mac.vendor      | string  | true  |
| ipmi     | ipmi.guid.partID          | string  | true  |
| ipmi     | ipmi.guid.serial          | string  | true  |
| ipmi     | ipmi.ipmiVersion          | string  | true  |

|      |                                   |         |      |
|------|-----------------------------------|---------|------|
| ipmi | ipmi.manufacturer                 | string  | true |
| ipmi | ipmi.manufacturerID               | string  | true |
| ipmi | ipmi.oemData                      | string  | true |
| ipmi | ipmi.oemID                        | string  | true |
| ipmi | ipmi.oemName                      | string  | true |
| ipmi | ipmi.passAuth                     | string  | true |
| ipmi | ipmi.passAuthRaw                  | string  | true |
| ipmi | ipmi.productID                    | string  | true |
| ipmi | ipmi.rakp.authCiphers             | string  | true |
| ipmi | ipmi.rakp.count                   | numeric | true |
| ipmi | ipmi.rakp.cracked                 | string  | true |
| ipmi | ipmi.rakp.crackedCount            | numeric | true |
| ipmi | ipmi.rakp.guids                   | string  | true |
| ipmi | ipmi.rakp.hashes                  | string  | true |
| ipmi | ipmi.rakp.passwords               | string  | true |
| ipmi | ipmi.rakp.sessionIDs              | string  | true |
| ipmi | ipmi.rakp.usernames               | string  | true |
| ipmi | ipmi.response.commandMismatch     | string  | true |
| ipmi | ipmi.response.data                | string  | true |
| ipmi | ipmi.response.dataLen             | string  | true |
| ipmi | ipmi.response.parseError          | string  | true |
| ipmi | ipmi.response.raw                 | string  | true |
| ipmi | ipmi.response.text                | string  | true |
| ipmi | ipmi.rmcpPlus                     | string  | true |
| ipmi | ipmi.session.ID                   | string  | true |
| ipmi | ipmi.session.activateAuthType     | string  | true |
| ipmi | ipmi.session.activateAuthTypeName | string  | true |
| ipmi | ipmi.session.activateID           | string  | true |
| ipmi | ipmi.session.activateInboundSeq   | string  | true |
| ipmi | ipmi.session.activateMaxPriv      | string  | true |
| ipmi | ipmi.session.activateMaxPrivName  | string  | true |

|           |                                    |        |      |
|-----------|------------------------------------|--------|------|
| ipmi      | ipmi.session.activateStatus        | string | true |
| ipmi      | ipmi.session.challenge             | string | true |
| ipmi      | ipmi.session.challengeAuthType     | string | true |
| ipmi      | ipmi.session.challengeAuthTypeName | string | true |
| ipmi      | ipmi.session.challengeID           | string | true |
| ipmi      | ipmi.session.challengeStatus       | string | true |
| ipmi      | ipmi.session.consoleID             | string | true |
| ipmi      | ipmi.session.status                | string | true |
| ipmi      | ipmi.session.statusCode            | string | true |
| ipmi      | ipmi.session.username              | string | true |
| ipmi      | ipmi.srcAddr                       | string | true |
| ipmi      | ipmi.srcLun                        | string | true |
| ipmi      | ipmi.system.guid                   | string | true |
| ipmi      | ipmi.tgtAddr                       | string | true |
| ipmi      | ipmi.tgtLun                        | string | true |
| ipmi      | ipmi.userAuth                      | string | true |
| ipmi      | ipmi.userAuthRaw                   | string | true |
| ipmi      | ipmi.v15.authMethods               | string | true |
| ipmi      | ipmi.v15.authNonePrivilege         | string | true |
| ipmi      | ipmi.v15.authNoneProof             | string | true |
| ipmi      | ipmi.v15.authNoneSession           | string | true |
| ipmi      | ipmi.v15.authNoneSessionID         | string | true |
| ipmi      | ipmi.v2.0                          | string | true |
| ipmi      | ipmi.weakCiphers                   | string | true |
| ippbrowse | ippbrowse.info                     | string | true |
| ippbrowse | ippbrowse.location                 | string | true |
| ippbrowse | ippbrowse.makeModel                | string | true |
| ippbrowse | ippbrowse.state                    | string | true |
| ippbrowse | ippbrowse.type                     | string | true |
| ippbrowse | ippbrowse.uri                      | string | true |
| irc       | irc.banner                         | string | true |

|       |                                   |         |      |
|-------|-----------------------------------|---------|------|
| irc   | irc.channelCount                  | numeric | true |
| irc   | irc.channelModes                  | string  | true |
| irc   | irc.core.sim                      | string  | true |
| irc   | irc.cpe23                         | string  | true |
| irc   | irc.createdDate                   | string  | true |
| irc   | irc.error                         | string  | true |
| irc   | irc.example.net                   | string  | true |
| irc   | irc.networkName                   | string  | true |
| irc   | irc.product                       | string  | true |
| irc   | irc.serverName                    | string  | true |
| irc   | irc.serverSoftware                | string  | true |
| irc   | irc.userCount                     | numeric | true |
| irc   | irc.userModes                     | string  | true |
| irc   | irc.vendor                        | string  | true |
| irc   | irc.version                       | string  | true |
| iscsi | iscsi.acknowledge                 | string  | true |
| iscsi | iscsi.authMethod                  | string  | true |
| iscsi | iscsi.bidiReadResidualCount       | numeric | true |
| iscsi | iscsi.bufferOffset                | string  | true |
| iscsi | iscsi.continue                    | string  | true |
| iscsi | iscsi.currentStage                | string  | true |
| iscsi | iscsi.dataDigest                  | string  | true |
| iscsi | iscsi.dataSN                      | string  | true |
| iscsi | iscsi.expCmdSN                    | string  | true |
| iscsi | iscsi.expDataSN                   | string  | true |
| iscsi | iscsi.final                       | string  | true |
| iscsi | iscsi.flags.readResidualOverflow  | string  | true |
| iscsi | iscsi.flags.readResidualUnderflow | string  | true |
| iscsi | iscsi.flags.residualOverflow      | string  | true |
| iscsi | iscsi.flags.residualUnderflow     | string  | true |
| iscsi | iscsi.flags.status                | string  | true |

|       |                                         |         |      |
|-------|-----------------------------------------|---------|------|
| iSCSI | iscsi.headerDigest                      | string  | true |
| iSCSI | iscsi.immediateDelivery                 | string  | true |
| iSCSI | iscsi.initiatorTaskTag                  | string  | true |
| iSCSI | iscsi.isid                              | string  | true |
| iSCSI | iscsi.keyvalue.MaxRecvDataSegmentLength | string  | true |
| iSCSI | iscsi.keyvalue.authMethod               | string  | true |
| iSCSI | iscsi.keyvalue.dataDigest               | string  | true |
| iSCSI | iscsi.keyvalue.defaultTime2Retain       | string  | true |
| iSCSI | iscsi.keyvalue.defaultTime2Wait         | string  | true |
| iSCSI | iscsi.keyvalue.errorRecoveryLevel       | string  | true |
| iSCSI | iscsi.keyvalue.headerDigest             | string  | true |
| iSCSI | iscsi.keyvalue.ifmarker                 | string  | true |
| iSCSI | iscsi.keyvalue.maxConnections           | numeric | true |
| iSCSI | iscsi.keyvalue.ofmarker                 | string  | true |
| iSCSI | iscsi.keyvalue.targetAddress            | string  | true |
| iSCSI | iscsi.keyvalue.targetName               | string  | true |
| iSCSI | iscsi.keyvalue.xCom.cisco.pingTimeout   | string  | true |
| iSCSI | iscsi.keyvalue.xCom.cisco.protocol      | string  | true |
| iSCSI | iscsi.keyvalue.xCom.cisco.sendAsyncText | string  | true |
| iSCSI | iscsi.logout.response                   | string  | true |
| iSCSI | iscsi.lun                               | string  | true |
| iSCSI | iscsi.lunAddressMode                    | string  | true |
| iSCSI | iscsi.maxCmdSN                          | string  | true |
| iSCSI | iscsi.nextStage                         | string  | true |
| iSCSI | iscsi.opcode                            | string  | true |
| iSCSI | iscsi.residualCount                     | numeric | true |
| iSCSI | iscsi.response                          | string  | true |
| iSCSI | iscsi.statSN                            | string  | true |
| iSCSI | iscsi.status                            | string  | true |
| iSCSI | iscsi.statusClass                       | string  | true |
| iSCSI | iscsi.statusDetail                      | string  | true |



|           |                            |        |      |
|-----------|----------------------------|--------|------|
| iscsi     | iscsi.targetAddress        | string | true |
| iscsi     | iscsi.targetAddresses      | string | true |
| iscsi     | iscsi.targetCount          | string | true |
| iscsi     | iscsi.targetName           | string | true |
| iscsi     | iscsi.targetNames          | string | true |
| iscsi     | iscsi.time2Retain          | string | true |
| iscsi     | iscsi.time2Wait            | string | true |
| iscsi     | iscsi.transferTaskTag      | string | true |
| iscsi     | iscsi.transit              | string | true |
| iscsi     | iscsi.tsih                 | string | true |
| iscsi     | iscsi.unknownData          | string | true |
| iscsi     | iscsi.versionActive        | string | true |
| iscsi     | iscsi.versionMax           | string | true |
| iscsi     | iscsi.versionMin           | string | true |
| iua       | iua.errorCode              | string | true |
| iua       | iua.infoString             | string | true |
| iua       | iua.messageClass           | string | true |
| iua       | iua.messageType            | string | true |
| jdwp      | jdwp.banner                | string | true |
| jdwp      | jdwp.description           | string | true |
| jdwp      | jdwp.major                 | string | true |
| jdwp      | jdwp.minor                 | string | true |
| jdwp      | jdwp.packetID              | string | true |
| jdwp      | jdwp.version               | string | true |
| jdwp      | jdwp.vmName                | string | true |
| jdwp      | jdwp.vmVersion             | string | true |
| jetdirect | jetdirect.banner           | string | true |
| jetdirect | jetdirect.cpe23            | string | true |
| jetdirect | jetdirect.filesystemAccess | string | true |
| jetdirect | jetdirect.firmware         | string | true |
| jetdirect | jetdirect.id               | string | true |

|           |                               |         |       |
|-----------|-------------------------------|---------|-------|
| jetdirect | jetdirect.model               | string  | true  |
| jetdirect | jetdirect.status              | string  | true  |
| jetdirect | jetdirect.vendor              | string  | true  |
| jetdirect | pjl.id                        | string  | true  |
| jira      | jira.baseUrl                  | string  | true  |
| jira      | jira.build                    | string  | true  |
| jira      | jira.version                  | string  | true  |
| jms       | jms.admin                     | string  | true  |
| jms       | jms.cluster                   | string  | true  |
| jms       | jms.jms                       | string  | true  |
| jms       | jms.jmsdirect                 | string  | true  |
| jms       | jms.jmxrmi                    | string  | true  |
| jms       | jms.jmxrmi.env.url            | string  | true  |
| jms       | jms.jmxrmi.env.version        | string  | true  |
| jms       | jms.mqdirect2                 | string  | true  |
| jms       | jms.portmapper                | numeric | true  |
| jms       | jms.portmapper.env.imqhome    | numeric | true  |
| jms       | jms.portmapper.env.imqvarhome | numeric | true  |
| jms       | jms.portmapper.env.sessionid  | numeric | true  |
| jms       | jms.services                  | string  | false |
| jms       | jms.tcp.ports                 | numeric | false |
| jms       | jms.version                   | string  | true  |
| kafka     | kafka.correlationID           | string  | true  |
| kafka     | kafka.version                 | string  | true  |
| kasa      | kasa.deviceAlias              | string  | true  |
| kasa      | kasa.deviceId                 | string  | true  |
| kasa      | kasa.deviceType               | string  | true  |
| kasa      | kasa.hardwareAddress          | string  | true  |
| kasa      | kasa.hardwareId               | string  | true  |
| kasa      | kasa.hardwareVersion          | string  | true  |
| kasa      | kasa.model                    | string  | true  |

|          |                             |         |       |
|----------|-----------------------------|---------|-------|
| kasa     | kasa.obdSource              | string  | true  |
| kasa     | kasa.oemId                  | string  | true  |
| kasa     | kasa.softwareVersion        | string  | true  |
| kerberos | kerberos.error              | string  | true  |
| kerberos | kerberos.errorCode          | string  | true  |
| kerberos | kerberos.errorCodeName      | string  | true  |
| kerberos | kerberos.messageType        | string  | true  |
| kerberos | kerberos.messageTypeCode    | string  | true  |
| kerberos | kerberos.microseconds       | string  | true  |
| kerberos | kerberos.port               | numeric | true  |
| kerberos | kerberos.realm              | string  | true  |
| kerberos | kerberos.serverTS           | numeric | true  |
| kerberos | kerberos.servicePrincipal   | string  | true  |
| kerberos | kerberos.target             | string  | true  |
| kerberos | kerberos.ticket.realm       | string  | true  |
| kerberos | kerberos.version            | string  | true  |
| knxnet   | knxnet.address              | string  | true  |
| knxnet   | knxnet.bus.address          | string  | true  |
| knxnet   | knxnet.bus.applicationID    | string  | true  |
| knxnet   | knxnet.bus.descriptor       | string  | true  |
| knxnet   | knxnet.bus.firmwareRevision | string  | true  |
| knxnet   | knxnet.bus.hardwareType     | string  | true  |
| knxnet   | knxnet.bus.manufacturerID   | string  | true  |
| knxnet   | knxnet.bus.name             | string  | true  |
| knxnet   | knxnet.bus.orderInfo        | string  | true  |
| knxnet   | knxnet.bus.programVersion   | string  | true  |
| knxnet   | knxnet.bus.serialNumber     | string  | true  |
| knxnet   | knxnet.busDevice            | string  | false |
| knxnet   | knxnet.busDeviceCount       | numeric | true  |
| knxnet   | knxnet.channel              | string  | true  |
| knxnet   | knxnet.decodeErrors         | string  | true  |

|        |                           |         |       |
|--------|---------------------------|---------|-------|
| knxnet | knxnet.decodeWarning      | string  | false |
| knxnet | knxnet.descriptor         | string  | true  |
| knxnet | knxnet.firmwareRevision   | string  | true  |
| knxnet | knxnet.hardwareType       | string  | true  |
| knxnet | knxnet.hpaiDataEP         | string  | false |
| knxnet | knxnet.mac                | string  | true  |
| knxnet | knxnet.manufacturerData   | string  | true  |
| knxnet | knxnet.manufacturerID     | string  | true  |
| knxnet | knxnet.medium             | string  | true  |
| knxnet | knxnet.multicast          | string  | true  |
| knxnet | knxnet.multicastAddress   | string  | true  |
| knxnet | knxnet.name               | string  | true  |
| knxnet | knxnet.orderInfo          | string  | true  |
| knxnet | knxnet.remoteDevice       | string  | false |
| knxnet | knxnet.remoteDeviceCount  | numeric | true  |
| knxnet | knxnet.serial             | string  | true  |
| knxnet | knxnet.serialNumber       | string  | true  |
| knxnet | knxnet.service.Core       | string  | true  |
| knxnet | knxnet.service.Tunnelling | string  | true  |
| knxnet | knxnet.serviceFamilies    | string  | true  |
| knxnet | knxnet.status             | string  | true  |
| knxnet | knxnet.tunnelData         | string  | true  |
| knxnet | knxnet.tunnelEP           | string  | false |
| knxnet | knxnet.type               | string  | true  |
| l2t    | l2t.assignedTunnelID      | string  | true  |
| l2t    | l2t.bearerCaps            | string  | true  |
| l2t    | l2t.firmwareRevision      | string  | true  |
| l2t    | l2t.framingCaps           | string  | true  |
| l2t    | l2t.hostName              | string  | true  |
| l2t    | l2t.messageType           | string  | true  |
| l2t    | l2t.protocolVersion       | string  | true  |

|           |                                 |        |      |
|-----------|---------------------------------|--------|------|
| l2t       | l2t.vendorName                  | string | true |
| l2tp      | l2tp.assignedTunnelID           | string | true |
| l2tp      | l2tp.bearerCaps                 | string | true |
| l2tp      | l2tp.controlCode                | string | true |
| l2tp      | l2tp.errorCode                  | string | true |
| l2tp      | l2tp.errorMessage               | string | true |
| l2tp      | l2tp.firmwareRevision           | string | true |
| l2tp      | l2tp.framingCaps                | string | true |
| l2tp      | l2tp.hostName                   | string | true |
| l2tp      | l2tp.nr                         | string | true |
| l2tp      | l2tp.ns                         | string | true |
| l2tp      | l2tp.protoVersion               | string | true |
| l2tp      | l2tp.recvWindowSize             | string | true |
| l2tp      | l2tp.resultCode                 | string | true |
| l2tp      | l2tp.sessionID                  | string | true |
| l2tp      | l2tp.tunnelID                   | string | true |
| l2tp      | l2tp.unknownType                | string | true |
| l2tp      | l2tp.vendorName                 | string | true |
| landesk   | landesk.configPath              | string | true |
| landesk   | landesk.providerVersion         | string | true |
| landesk   | landesk.serverVersion           | string | true |
| langflow  | service.product                 | string | true |
| langflow  | service.version                 | string | true |
| lantronix | lantronix.fwinfo                | string | true |
| lantronix | lantronix.mac                   | string | true |
| lantronix | lantronix.macVendor             | string | true |
| lantronix | lantronix.serial                | string | true |
| lantronix | lantronix.type                  | string | true |
| lantronix | lantronix.version               | string | true |
| ldap      | ldap.auth                       | string | true |
| ldap      | ldap.configurationNamingContext | string | true |

|      |                                    |         |       |
|------|------------------------------------|---------|-------|
| ldap | ldap.currentTime                   | string  | true  |
| ldap | ldap.defaultNamingContext          | string  | true  |
| ldap | ldap.diagMessage                   | string  | true  |
| ldap | ldap.dnsHostName                   | string  | true  |
| ldap | ldap.domainControllerFunctionality | string  | true  |
| ldap | ldap.domainFunctionality           | string  | true  |
| ldap | ldap.dsServiceName                 | string  | true  |
| ldap | ldap.error                         | string  | true  |
| ldap | ldap.example.com                   | string  | true  |
| ldap | ldap.forestFunctionality           | string  | true  |
| ldap | ldap.isGlobalCatalogReady          | string  | true  |
| ldap | ldap.isSynchronized                | string  | true  |
| ldap | ldap.matchedDN                     | string  | true  |
| ldap | ldap.namingContexts                | numeric | true  |
| ldap | ldap.notes                         | string  | false |
| ldap | ldap.resultCode                    | string  | true  |
| ldap | ldap.resultCodeName                | string  | true  |
| ldap | ldap.rootDomainNamingContext       | string  | true  |
| ldap | ldap.schemaNamingContext           | string  | true  |
| ldap | ldap.searchresult                  | string  | true  |
| ldap | ldap.serverName                    | string  | true  |
| ldap | ldap.serverType                    | string  | true  |
| ldap | ldap.startTLS                      | string  | true  |
| ldap | ldap.subschemaSubentry             | string  | true  |
| ldap | ldap.supportedControl              | string  | true  |
| ldap | ldap.supportedControls             | numeric | true  |
| ldap | ldap.supportedExtension            | string  | true  |
| ldap | ldap.supportedExtensions           | numeric | true  |
| ldap | ldap.supportedLDAPVersion          | string  | true  |
| ldap | ldap.supportedSASLMechanisms       | numeric | true  |
| ldap | ldap.supportsStartTLS              | numeric | true  |

|           |                                   |        |      |
|-----------|-----------------------------------|--------|------|
| ldap      | ldap.vendorName                   | string | true |
| ldap      | ldap.vendorVersion                | string | true |
| liebert   | liebert.hostname                  | string | true |
| liebert   | liebert.mac                       | string | true |
| liebert   | liebert.model                     | string | true |
| liebert   | liebert.state                     | string | true |
| liebert   | liebert.sysname                   | string | true |
| liebert   | liebert.version                   | string | true |
| llmnr     | llmnr.ipv4                        | string | true |
| llmnr     | llmnr.ipv6                        | string | true |
| llmnr     | llmnr.name                        | string | true |
| lockdownd | lockdownd.cpuArchitecture         | string | true |
| lockdownd | lockdownd.deviceName              | string | true |
| lockdownd | lockdownd.hardwareModel           | string | true |
| lockdownd | lockdownd.productName             | string | true |
| lockdownd | lockdownd.productType             | string | true |
| lockdownd | lockdownd.productVersion          | string | true |
| lockdownd | lockdownd.supportedDeviceFamilies | string | true |
| lpd       | lpd.banner                        | string | true |
| lwm2m     | lwm2m.bootstrapServer             | string | true |
| lwm2m     | lwm2m.payload                     | string | true |
| lwm2m     | lwm2m.registrationDirectory       | string | true |
| lwm2m     | lwm2m.serverImpl                  | string | true |
| m2pa      | m2pa.errorCode                    | string | true |
| m2pa      | m2pa.infoString                   | string | true |
| m2pa      | m2pa.linkState                    | string | true |
| m2pa      | m2pa.linkStateName                | string | true |
| m2pa      | m2pa.messageClass                 | string | true |
| m2pa      | m2pa.messageType                  | string | true |
| m2ua      | m2ua.errorCode                    | string | true |

|             |                            |         |      |
|-------------|----------------------------|---------|------|
| m2ua        | m2ua.infoString            | string  | true |
| m2ua        | m2ua.messageClass          | string  | true |
| m2ua        | m2ua.messageType           | string  | true |
| m3ua        | m3ua.errorCode             | string  | true |
| m3ua        | m3ua.infoString            | string  | true |
| m3ua        | m3ua.messageClass          | string  | true |
| m3ua        | m3ua.messageType           | string  | true |
| managesieve | managesieve.banner         | string  | true |
| managesieve | managesieve.extensions     | string  | true |
| managesieve | managesieve.implementation | string  | true |
| managesieve | managesieve.language       | string  | true |
| managesieve | managesieve.sasl           | string  | true |
| managesieve | managesieve.startTLS       | string  | true |
| managesieve | managesieve.version        | string  | true |
| megaco      | megaco.banner              | string  | true |
| megaco      | megaco.error               | string  | true |
| megaco      | megaco.errorCode           | string  | true |
| megaco      | megaco.format              | string  | true |
| megaco      | megaco.mid                 | string  | true |
| megaco      | megaco.port                | numeric | true |
| megaco      | megaco.profile             | string  | true |
| megaco      | megaco.replyID             | string  | true |
| megaco      | megaco.serviceChange       | string  | true |
| megaco      | megaco.target              | string  | true |
| megaco      | megaco.transactionID       | string  | true |
| megaco      | megaco.version             | string  | true |
| melsecq     | melsecq.cpe                | string  | true |
| melsecq     | melsecq.cpuModel           | string  | true |
| memcache    | memcache.binaryProtocol    | string  | true |
| memcache    | memcache.bytes             | string  | true |
| memcache    | memcache.currConnections   | numeric | true |



|                |                            |         |      |
|----------------|----------------------------|---------|------|
| memcache       | memcache.currItems         | string  | true |
| memcache       | memcache.error             | string  | true |
| memcache       | memcache.libevent          | string  | true |
| memcache       | memcache.limitMaxbytes     | string  | true |
| memcache       | memcache.maxConnections    | numeric | true |
| memcache       | memcache.pid               | string  | true |
| memcache       | memcache.pointerSize       | string  | true |
| memcache       | memcache.threads           | string  | true |
| memcache       | memcache.totalConnections  | string  | true |
| memcache       | memcache.totalItems        | string  | true |
| memcache       | memcache.uptime            | numeric | true |
| memcache       | memcache.version           | string  | true |
| mgcp           | mgcp.endpoint              | string  | true |
| mgcp           | mgcp.endpoints             | numeric | true |
| mgcp           | mgcp.localOptions          | string  | true |
| mgcp           | mgcp.mode                  | string  | true |
| mgcp           | mgcp.packages              | string  | true |
| mgcp           | mgcp.responseCode          | string  | true |
| mgcp           | mgcp.signals               | string  | true |
| mikrotikwinbox | mikrotikwinbox.banner      | string  | true |
| mikrotikwinbox | mikrotikwinbox.subProtocol | string  | true |
| minecraft      | minecraft.currentplayers   | numeric | true |
| minecraft      | minecraft.maxplayers       | numeric | true |
| minecraft      | minecraft.motd             | string  | true |
| minecraft      | minecraft.version          | string  | true |
| modbus         | modbus.applicationName     | string  | true |
| modbus         | modbus.deviceCount         | numeric | true |
| modbus         | modbus.exception           | string  | true |
| modbus         | modbus.function            | string  | true |
| modbus         | modbus.gateway             | string  | true |
| modbus         | modbus.gatewayPort         | numeric | true |

|         |                                      |         |      |
|---------|--------------------------------------|---------|------|
| modbus  | modbus.identifier                    | string  | true |
| modbus  | modbus.modelName                     | string  | true |
| modbus  | modbus.port                          | numeric | true |
| modbus  | modbus.productCode                   | string  | true |
| modbus  | modbus.productName                   | string  | true |
| modbus  | modbus.registers                     | string  | true |
| modbus  | modbus.remote                        | string  | true |
| modbus  | modbus.revision                      | string  | true |
| modbus  | modbus.subDeviceCount                | numeric | true |
| modbus  | modbus.target                        | string  | true |
| modbus  | modbus.unitID                        | string  | true |
| modbus  | modbus.vendor                        | string  | true |
| modbus  | modbus.vendorURL                     | string  | true |
| mongodb | mongodb allocator                    | string  | true |
| mongodb | mongodb.auth                         | string  | true |
| mongodb | mongodb.bits                         | numeric | true |
| mongodb | mongodb.buildEnvironment.cc          | string  | true |
| mongodb | mongodb.buildEnvironment.ccflags     | string  | true |
| mongodb | mongodb.buildEnvironment.cxx         | string  | true |
| mongodb | mongodb.buildEnvironment.cxxflags    | string  | true |
| mongodb | mongodb.buildEnvironment.distarch    | string  | true |
| mongodb | mongodb.buildEnvironment.distmod     | string  | true |
| mongodb | mongodb.buildEnvironment.linkflags   | string  | true |
| mongodb | mongodb.buildEnvironment.target_arch | string  | true |
| mongodb | mongodb.buildEnvironment.target_os   | string  | true |
| mongodb | mongodb.cpuArch                      | string  | true |
| mongodb | mongodb.debug                        | bool    | true |
| mongodb | mongodb.error                        | string  | true |
| mongodb | mongodb.gitVersion                   | string  | true |
| mongodb | mongodb.hostname                     | string  | true |
| mongodb | mongodb.isMaster                     | string  | true |

|         |                                   |         |       |
|---------|-----------------------------------|---------|-------|
| mongodb | mongodb.javascriptEngine          | string  | true  |
| mongodb | mongodb.jsEngine                  | string  | true  |
| mongodb | mongodb.legacyWireVersion         | string  | true  |
| mongodb | mongodb.maxBsonObjectSize         | numeric | true  |
| mongodb | mongodb.maxWireVersion            | string  | true  |
| mongodb | mongodb.memSizeMB                 | string  | true  |
| mongodb | mongodb.minWireVersion            | string  | true  |
| mongodb | mongodb.modules                   | string  | false |
| mongodb | mongodb.mongos                    | string  | true  |
| mongodb | mongodb.numCores                  | string  | true  |
| mongodb | mongodb.ok                        | string  | true  |
| mongodb | mongodb.opcode                    | string  | true  |
| mongodb | mongodb.openssl.compiled          | string  | true  |
| mongodb | mongodb.openssl.running           | string  | true  |
| mongodb | mongodb.os                        | string  | true  |
| mongodb | mongodb.osVersion                 | string  | true  |
| mongodb | mongodb.readOnly                  | string  | true  |
| mongodb | mongodb.setParameter.md5          | string  | true  |
| mongodb | mongodb.storageEngines            | string  | false |
| mongodb | mongodb.sysInfo                   | string  | true  |
| mongodb | mongodb.version                   | string  | true  |
| mongodb | mongodb.versionArray              | string  | false |
| moxa    | moxa.ip                           | string  | true  |
| moxa    | moxa.mac                          | string  | true  |
| moxa    | moxa.model                        | string  | true  |
| moxa    | moxa.productLine                  | string  | true  |
| moxa    | moxa.vendor                       | string  | true  |
| mqtt    | mqtt.broker.authRequired          | boolean | true  |
| mqtt    | mqtt.broker.supportedProtocols    | numeric | true  |
| mqtt    | mqtt.broker.sys.buildTime         | string  | true  |
| mqtt    | mqtt.broker.sys.clients.connected | string  | true  |

|       |                                         |         |       |
|-------|-----------------------------------------|---------|-------|
| mqtt  | mqtt.broker.sys.clients.maximum         | string  | true  |
| mqtt  | mqtt.broker.sys.clients.total           | string  | true  |
| mqtt  | mqtt.broker.sys.messages.retained.count | numeric | true  |
| mqtt  | mqtt.broker.sys.messages.stored         | string  | true  |
| mqtt  | mqtt.broker.sys.subscriptions.count     | numeric | true  |
| mqtt  | mqtt.broker.sys.time                    | string  | true  |
| mqtt  | mqtt.broker.sys.uptime                  | numeric | true  |
| mqtt  | mqtt.broker.sys.version                 | string  | true  |
| mqtt  | mqtt.connAckCode                        | string  | true  |
| mqtt  | mqtt.connAckMessage                     | string  | true  |
| mqtt  | mqtt.isV5                               | string  | true  |
| mqtt  | mqtt.port                               | numeric | true  |
| mqtt  | mqtt.protocolVersion                    | string  | true  |
| mqtt  | mqtt.sessionPresent                     | string  | true  |
| mqtt  | mqtt.target                             | string  | true  |
| mssql | mssql.browser.response                  | string  | false |
| mssql | mssql.encryption                        | string  | true  |
| mssql | mssql.encryptionCode                    | string  | true  |
| mssql | mssql.instanceName                      | string  | true  |
| mssql | mssql.instanceNames                     | string  | true  |
| mssql | mssql.isClustered                       | string  | true  |
| mssql | mssql.mars                              | string  | true  |
| mssql | mssql.nps                               | string  | true  |
| mssql | mssql.ntlm.dnsComputer                  | string  | true  |
| mssql | mssql.ntlm.dnsDomain                    | string  | true  |
| mssql | mssql.ntlm.dnsTree                      | string  | true  |
| mssql | mssql.ntlm.negotiateFlags               | string  | true  |
| mssql | mssql.ntlm.netbiosComputer              | string  | true  |
| mssql | mssql.ntlm.netbiosDomain                | string  | true  |
| mssql | mssql.ntlm.osBuild                      | string  | true  |
| mssql | mssql.ntlm.osVersion                    | string  | true  |

|       |                            |         |       |
|-------|----------------------------|---------|-------|
| mssql | mssql.ntlm.targetName      | string  | true  |
| mssql | mssql.ntlm.timestamp       | numeric | true  |
| mssql | mssql.packetType           | string  | true  |
| mssql | mssql.packetTypeCode       | string  | true  |
| mssql | mssql.port                 | numeric | true  |
| mssql | mssql.product              | string  | true  |
| mssql | mssql.requiresEncryption   | string  | true  |
| mssql | mssql.requiresTDS8Strict   | string  | true  |
| mssql | mssql.serverName           | string  | true  |
| mssql | mssql.serverNames          | string  | true  |
| mssql | mssql.sessionEncryption    | string  | true  |
| mssql | mssql.target               | string  | true  |
| mssql | mssql.tcp                  | string  | true  |
| mssql | mssql.tcp.ports            | numeric | false |
| mssql | mssql.version              | string  | true  |
| mssql | mssql.versionBuild         | string  | true  |
| mssql | mssql.versionMajor         | string  | true  |
| mssql | mssql.versionMinor         | string  | true  |
| mssql | mssql.versionName          | string  | true  |
| mssql | mssql.versionSubBuild      | string  | true  |
| mssql | mssql.versions             | string  | true  |
| munin | munin.capabilities         | string  | false |
| munin | munin.hostname             | string  | true  |
| munin | munin.metrics              | string  | false |
| munin | munin.nodes                | string  | false |
| munin | munin.rawBanner            | string  | true  |
| munin | munin.tlsRequired          | boolean | true  |
| munin | munin.version              | string  | true  |
| mysql | mysql.authPluginDataLength | string  | true  |
| mysql | mysql.authPluginName       | string  | true  |
| mysql | mysql.capabilities         | string  | true  |

|        |                                  |         |      |
|--------|----------------------------------|---------|------|
| mysql  | mysql.capabilityFlags            | string  | true |
| mysql  | mysql.characterSet               | string  | true |
| mysql  | mysql.characterSetName           | string  | true |
| mysql  | mysql.connectionID               | string  | true |
| mysql  | mysql.error                      | string  | true |
| mysql  | mysql.errorCode                  | string  | true |
| mysql  | mysql.errorMessage               | string  | true |
| mysql  | mysql.errorSQLState              | string  | true |
| mysql  | mysql.family                     | string  | true |
| mysql  | mysql.normalizedVersion          | string  | true |
| mysql  | mysql.packetType                 | string  | true |
| mysql  | mysql.product                    | string  | true |
| mysql  | mysql.protocolVersion            | string  | true |
| mysql  | mysql.ssl                        | string  | true |
| mysql  | mysql.sslSupported               | numeric | true |
| mysql  | mysql.status                     | string  | true |
| mysql  | mysql.statusFlags                | string  | true |
| mysql  | mysql.version                    | string  | true |
| mysqlx | mysqlx.authentication.mechanisms | string  | true |
| mysqlx | mysqlx.client.interactive        | string  | true |
| mysqlx | mysqlx.client.pwdExpireOk        | string  | true |
| mysqlx | mysqlx.compression.algorithm     | string  | true |
| mysqlx | mysqlx.doc.formats               | numeric | true |
| mysqlx | mysqlx.error                     | string  | true |
| mysqlx | mysqlx.errorCode                 | string  | true |
| mysqlx | mysqlx.errorSeverity             | string  | true |
| mysqlx | mysqlx.messageType               | string  | true |
| mysqlx | mysqlx.nodeType                  | string  | true |
| mysqlx | mysqlx.tls                       | string  | true |
| natpmp | natpmp.externalIP                | string  | true |
| natpmp | natpmp.lastChange                | string  | true |

|        |                     |         |      |
|--------|---------------------|---------|------|
| natpmp | natpmp.responseCode | string  | true |
| nats   | nats.authRequired   | boolean | true |
| nats   | nats.banner         | string  | true |
| nats   | nats.clientID       | string  | true |
| nats   | nats.clientIP       | string  | true |
| nats   | nats.cluster        | string  | true |
| nats   | nats.connectURLs    | string  | true |
| nats   | nats.domain         | string  | true |
| nats   | nats.gitCommit      | string  | true |
| nats   | nats.goVersion      | string  | true |
| nats   | nats.headers        | string  | true |
| nats   | nats.host           | string  | true |
| nats   | nats.info           | string  | true |
| nats   | nats.jetStream      | string  | true |
| nats   | nats.ldm            | string  | true |
| nats   | nats.maxPayload     | string  | true |
| nats   | nats.nonce          | string  | true |
| nats   | nats.ok             | string  | true |
| nats   | nats.pong           | string  | true |
| nats   | nats.port           | numeric | true |
| nats   | nats.proto          | string  | true |
| nats   | nats.serverID       | string  | true |
| nats   | nats.serverName     | string  | true |
| nats   | nats.tlsAvailable   | string  | true |
| nats   | nats.tlsRequired    | boolean | true |
| nats   | nats.tlsVerify      | string  | true |
| nats   | nats.version        | string  | true |
| ndmp   | ndmp.reason         | string  | true |
| ndmp   | ndmp.status         | string  | true |
| ndmp   | ndmp.statusName     | string  | true |
| ndmp   | ndmp.timestampTS    | numeric | true |

|         |                                 |         |       |
|---------|---------------------------------|---------|-------|
| ndmp    | ndmp.version                    | string  | true  |
| neo4j   | neo4j.boltVersion               | string  | true  |
| neo4j   | neo4j.neo4jVersion              | string  | true  |
| neo4j   | neo4j.server                    | string  | true  |
| neo4j   | neo4j.serverVersion             | string  | true  |
| netbios | netbios.addrs                   | string  | true  |
| netbios | netbios.domain                  | string  | true  |
| netbios | netbios.domainController        | string  | true  |
| netbios | netbios.mac                     | string  | true  |
| netbios | netbios.macDateAdded            | string  | true  |
| netbios | netbios.macVendor               | string  | true  |
| netbios | netbios.name                    | string  | true  |
| netbios | netbios.names                   | string  | true  |
| netbios | netbios.nodeType                | string  | true  |
| netbios | netbios.primarydomainController | string  | true  |
| netbios | netbios.responseType            | string  | true  |
| netbios | netbios.username                | string  | true  |
| nfs     | nfs.allowed                     | string  | false |
| nfs     | nfs.detected                    | boolean | true  |
| nfs     | nfs.exports                     | string  | true  |
| nfs     | nfs.maxVersion                  | string  | true  |
| nfs     | nfs.minVersion                  | string  | true  |
| nfs     | nfs.unknownReply                | string  | true  |
| nfs     | nfs.verifierFlavor              | string  | true  |
| nfs     | nfs.versions                    | string  | true  |
| nfs     | nfs.xid                         | string  | true  |
| nrpe    | nrpe.commandArgsEnabled         | boolean | true  |
| nrpe    | nrpe.output                     | string  | true  |
| nrpe    | nrpe.resultCode                 | string  | true  |
| nrpe    | nrpe.version                    | string  | true  |
| ntp     | ntp.controlAssociationID        | string  | true  |



|     |                            |         |      |
|-----|----------------------------|---------|------|
| ntp | ntp.controlOpcode          | string  | true |
| ntp | ntp.controlSequence        | string  | true |
| ntp | ntp.controlStatus          | string  | true |
| ntp | ntp.controlVersion         | string  | true |
| ntp | ntp.interval               | string  | true |
| ntp | ntp.leapIndicator          | string  | true |
| ntp | ntp.leapIndicatorCode      | string  | true |
| ntp | ntp.mode                   | string  | true |
| ntp | ntp.modeCode               | string  | true |
| ntp | ntp.monlistItemSize        | string  | true |
| ntp | ntp.monlistNumItems        | string  | true |
| ntp | ntp.monlistSupported       | numeric | true |
| ntp | ntp.originTimestamp        | numeric | true |
| ntp | ntp.poll                   | string  | true |
| ntp | ntp.pollSeconds            | string  | true |
| ntp | ntp.precision              | string  | true |
| ntp | ntp.precisionSeconds       | string  | true |
| ntp | ntp.processor              | string  | true |
| ntp | ntp.readVar                | string  | true |
| ntp | ntp.receiveTimestamp       | numeric | true |
| ntp | ntp.referenceID            | string  | true |
| ntp | ntp.referenceIDDescription | string  | true |
| ntp | ntp.referenceTimestamp     | numeric | true |
| ntp | ntp.rootDelay              | string  | true |
| ntp | ntp.rootDispersion         | string  | true |
| ntp | ntp.serverTime             | string  | true |
| ntp | ntp.skew                   | string  | true |
| ntp | ntp.skewMS                 | string  | true |
| ntp | ntp.stratum                | string  | true |
| ntp | ntp.stratumDescription     | string  | true |
| ntp | ntp.swVersion              | string  | true |

|           |                                     |         |       |
|-----------|-------------------------------------|---------|-------|
| ntp       | ntp.system                          | string  | true  |
| ntp       | ntp.timestamp                       | string  | true  |
| ntp       | ntp.transmitTimestamp               | numeric | true  |
| ntp       | ntp.version                         | string  | true  |
| omronfins | omronfins.banner                    | string  | true  |
| omronfins | omronfins.model                     | string  | true  |
| omronfins | omronfins.transport                 | numeric | true  |
| omronfins | omronfins.version                   | string  | true  |
| opcua     | opcua.applications                  | string  | false |
| opcua     | opcua.channelCertSHA256             | string  | true  |
| opcua     | opcua.clientThumbprint              | string  | true  |
| opcua     | opcua.endpointCerts                 | numeric | true  |
| opcua     | opcua.endpointCertsSuppressed       | string  | true  |
| opcua     | opcua.endpointCount                 | numeric | true  |
| opcua     | opcua.endpointUniqueCerts           | numeric | true  |
| opcua     | opcua.endpoints                     | numeric | true  |
| opcua     | opcua.error                         | string  | true  |
| opcua     | opcua.errorReason                   | string  | true  |
| opcua     | opcua.errorText                     | string  | true  |
| opcua     | opcua.maxChunkCount                 | numeric | true  |
| opcua     | opcua.maxMessageSize                | string  | true  |
| opcua     | opcua.messageType                   | string  | true  |
| opcua     | opcua.port                          | numeric | true  |
| opcua     | opcua.protocol                      | string  | true  |
| opcua     | opcua.receiveBufferSize             | string  | true  |
| opcua     | opcua.revisedLifetime               | string  | true  |
| opcua     | opcua.secureChannelID               | string  | true  |
| opcua     | opcua.securityPolicy                | string  | true  |
| opcua     | opcua.sendBufferSize                | string  | true  |
| opcua     | opcua.supportedMessageSecurityModes | numeric | true  |
| opcua     | opcua.supportedSecurityPolicies     | numeric | true  |

|          |                               |         |      |
|----------|-------------------------------|---------|------|
| opcua    | opcua.supportedUserTokenTypes | numeric | true |
| opcua    | opcua.target                  | string  | true |
| opcua    | opcua.tokenID                 | string  | true |
| opcua    | opcua.unknownNodeID           | string  | true |
| openvpn  | openvpn.detected              | boolean | true |
| openvpn  | openvpn.keyID                 | string  | true |
| openvpn  | openvpn.messagePacketID       | string  | true |
| openvpn  | openvpn.opcode                | string  | true |
| openvpn  | openvpn.packetID              | string  | true |
| openvpn  | openvpn.packetIDArrayLength   | string  | true |
| openvpn  | openvpn.remoteSessionID       | string  | true |
| openvpn  | openvpn.reply                 | string  | true |
| openvpn  | openvpn.service               | string  | true |
| openvpn  | openvpn.sessionID             | string  | true |
| oracle   | oracle.connectFlags0          | string  | true |
| oracle   | oracle.connectFlags1          | string  | true |
| oracle   | oracle.didResend              | string  | true |
| oracle   | oracle.globalServiceOptions   | string  | true |
| oracle   | oracle.nsn.Authentication     | string  | true |
| oracle   | oracle.nsn.Encryption         | string  | true |
| oracle   | oracle.nsnVersion             | string  | true |
| oracle   | oracle.packetType             | string  | true |
| oracle   | oracle.packetTypeCode         | string  | true |
| oracle   | oracle.redirectData           | string  | true |
| oracle   | oracle.redirectDescriptor     | string  | true |
| oracle   | oracle.refuseData             | string  | true |
| oracle   | oracle.refuseDescriptor       | string  | true |
| oracle   | oracle.refuseReason           | string  | true |
| oracle   | oracle.tnsVersion             | string  | true |
| oracle   | oracle.version                | string  | true |
| oracledb | oracledb.errCode              | string  | true |

|           |                                 |        |       |
|-----------|---------------------------------|--------|-------|
| oracledb  | oracledb.packetType             | string | true  |
| oracledb  | oracledb.tns.error              | string | true  |
| oracledb  | oracledb.tns.version            | string | true  |
| oracledb  | oracledb.tns.vsn                | string | true  |
| oracledb  | oracledb.version                | string | true  |
| oracledb  | oracledb.vsnnum                 | string | true  |
| orion     | orion.components                | string | true  |
| orion     | orion.version                   | string | true  |
| panxmlapi | host.mac                        | string | false |
| panxmlapi | host.name                       | string | false |
| panxmlapi | pan.api.advancedRouting         | string | true  |
| panxmlapi | pan.api.appVersion              | string | true  |
| panxmlapi | pan.api.arpcache.ports          | string | true  |
| panxmlapi | pan.api.avDatabaseVersion       | string | true  |
| panxmlapi | pan.api.deviceCertificateStatus | string | true  |
| panxmlapi | pan.api.globalProtectVersion    | string | true  |
| panxmlapi | pan.api.logdbVersion            | string | true  |
| panxmlapi | pan.api.model                   | string | true  |
| panxmlapi | pan.api.multiVsys               | string | true  |
| panxmlapi | pan.api.operationalMode         | string | true  |
| panxmlapi | pan.api.osVersion               | string | true  |
| panxmlapi | pan.api.plugins                 | string | true  |
| panxmlapi | pan.api.serialNumber            | string | true  |
| panxmlapi | pan.api.threatDatabaseVersion   | string | true  |
| panxmlapi | pan.api.urlDatabaseVersion      | string | true  |
| panxmlapi | pan.api.vmMode                  | string | true  |
| panxmlapi | pan.api.vpnDisableMode          | string | true  |
| panxmlapi | pan.api.wildfireDatabaseVersion | string | true  |
| panxmlapi | pan.api.wildfireRt              | string | true  |
| panxmlapi | snmp.arpcache                   | string | false |
| pca       | pca.caps                        | string | true  |

|        |                        |         |      |
|--------|------------------------|---------|------|
| pca    | pca.isRequest          | string  | true |
| pca    | pca.name               | string  | true |
| pca    | pca.status             | string  | true |
| pcworx | pcworx.firmwareDate    | string  | true |
| pcworx | pcworx.firmwareTime    | string  | true |
| pcworx | pcworx.firmwareVersion | string  | true |
| pcworx | pcworx.modelNumber     | string  | true |
| pcworx | pcworx.plcType         | string  | true |
| pega   | pega.version           | string  | true |
| pfcp   | pfcp.nodeID            | string  | true |
| pfcp   | pfcp.nodeIDType        | string  | true |
| pfcp   | pfcp.recoveryTimestamp | numeric | true |
| pfcp   | pfcp.seidPresent       | string  | true |
| pfcp   | pfcp.sequenceNumber    | string  | true |
| pfcp   | pfcp.version           | string  | true |
| pop3   | pop3.apop              | string  | true |
| pop3   | pop3.apopBanner        | string  | true |
| pop3   | pop3.authMethods       | string  | true |
| pop3   | pop3.banner            | string  | true |
| pop3   | pop3.capa.message      | string  | true |
| pop3   | pop3.capa.status       | string  | true |
| pop3   | pop3.capabilities      | string  | true |
| pop3   | pop3.capabilityCount   | numeric | true |
| pop3   | pop3.expire            | string  | true |
| pop3   | pop3.implementation    | string  | true |
| pop3   | pop3.loginDelay        | string  | true |
| pop3   | pop3.message           | string  | true |
| pop3   | pop3.pipelining        | string  | true |
| pop3   | pop3.responseCodes     | string  | true |
| pop3   | pop3.startTLS          | string  | true |
| pop3   | pop3.status            | string  | true |

|            |                                      |         |      |
|------------|--------------------------------------|---------|------|
| pop3       | pop3.top                             | string  | true |
| pop3       | pop3.uidl                            | string  | true |
| pop3       | pop3.userAuth                        | string  | true |
| postgresql | postgresql.auth.details              | string  | true |
| postgresql | postgresql.auth.method               | string  | true |
| postgresql | postgresql.backendPID                | string  | true |
| postgresql | postgresql.clientEncoding            | string  | true |
| postgresql | postgresql.dateStyle                 | string  | true |
| postgresql | postgresql.error.code                | numeric | true |
| postgresql | postgresql.error.file                | string  | true |
| postgresql | postgresql.error.line                | string  | true |
| postgresql | postgresql.error.message             | string  | true |
| postgresql | postgresql.error.routine             | string  | true |
| postgresql | postgresql.error.severity            | string  | true |
| postgresql | postgresql.error.text                | string  | true |
| postgresql | postgresql.error.unknown             | string  | true |
| postgresql | postgresql.integerDatetimes          | string  | true |
| postgresql | postgresql.isSuperuser               | string  | true |
| postgresql | postgresql.notice.message            | string  | true |
| postgresql | postgresql.notice.severity           | string  | true |
| postgresql | postgresql.packetType                | string  | true |
| postgresql | postgresql.routine                   | string  | true |
| postgresql | postgresql.serverEncoding            | string  | true |
| postgresql | postgresql.serverVersion             | string  | true |
| postgresql | postgresql.sessionAuthorization      | string  | true |
| postgresql | postgresql.sslSupported              | string  | true |
| postgresql | postgresql.standardConformingStrings | string  | true |
| postgresql | postgresql.timeZone                  | string  | true |
| postgresql | postgresql.txStatus                  | string  | true |
| pptp       | pptp.bearerCapabilities              | string  | true |
| pptp       | pptp.errorCode                       | string  | true |

|          |                                 |         |      |
|----------|---------------------------------|---------|------|
| pptp     | pptp.errorText                  | string  | true |
| pptp     | pptp.firmwareRevision           | string  | true |
| pptp     | pptp.framingCapabilities        | string  | true |
| pptp     | pptp.fwRevision                 | string  | true |
| pptp     | pptp.hostname                   | string  | true |
| pptp     | pptp.maxChannels                | string  | true |
| pptp     | pptp.port                       | numeric | true |
| pptp     | pptp.protocolVersion            | string  | true |
| pptp     | pptp.protocolVersionRaw         | string  | true |
| pptp     | pptp.resultCode                 | string  | true |
| pptp     | pptp.resultText                 | string  | true |
| pptp     | pptp.target                     | string  | true |
| pptp     | pptp.vendor                     | string  | true |
| proconos | proconos.bootProject            | string  | true |
| proconos | proconos.ladderLogicRuntime     | string  | true |
| proconos | proconos.plcType                | string  | true |
| proconos | proconos.projectName            | string  | true |
| proconos | proconos.projectSourceCode      | string  | true |
| proconos | prosoft.moduleName              | string  | true |
| proconos | prosoft.operatingSystemRevision | string  | true |
| proconos | prosoft.softwareRevision        | string  | true |
| profinet | profinet.annotation             | string  | true |
| profinet | profinet.cpe                    | string  | true |
| profinet | profinet.deviceName             | string  | true |
| profinet | profinet.deviceType             | string  | true |
| profinet | profinet.vendor                 | string  | true |
| prosoft  | firmwareDate                    | string  | true |
| prosoft  | moduleName                      | string  | true |
| prosoft  | moduleRevision                  | string  | true |
| prosoft  | moduleSerial                    | string  | true |
| prosoft  | moduleStatus                    | string  | true |

|         |                              |         |      |
|---------|------------------------------|---------|------|
| psdisco | psdisco.appName              | string  | true |
| psdisco | psdisco.appTitleID           | string  | true |
| psdisco | psdisco.code                 | string  | true |
| psdisco | psdisco.id                   | string  | true |
| psdisco | psdisco.name                 | string  | true |
| psdisco | psdisco.protoVersion         | string  | true |
| psdisco | psdisco.requestPort          | string  | true |
| psdisco | psdisco.status               | string  | true |
| psdisco | psdisco.sysVersion           | string  | true |
| psdisco | psdisco.type                 | string  | true |
| qotd    | qotd.quote                   | string  | true |
| qualys  | qualys.correlationID         | string  | true |
| radius  | radius.attributes            | string  | true |
| radius  | radius.calledStationId       | string  | true |
| radius  | radius.code                  | numeric | true |
| radius  | radius.codeID                | string  | true |
| radius  | radius.identifier            | string  | true |
| radius  | radius.nasIdentifier         | string  | true |
| radius  | radius.replyMessage          | string  | true |
| rdp     | rdp.auth.nla                 | string  | true |
| rdp     | rdp.auth.sspeua              | string  | true |
| rdp     | rdp.auth.supportsNegotiation | numeric | true |
| rdp     | rdp.auth.tls                 | string  | true |
| rdp     | rdp.failureCode              | string  | true |
| rdp     | rdp.failureCodeName          | string  | true |
| rdp     | rdp.features                 | string  | true |
| rdp     | rdp.flags                    | string  | true |
| rdp     | rdp.negotiation              | string  | true |
| rdp     | rdp.ntlm.dnsComputer         | string  | true |
| rdp     | rdp.ntlm.dnsDomain           | string  | true |
| rdp     | rdp.ntlm.dnsTree             | string  | true |



|         |                           |         |      |
|---------|---------------------------|---------|------|
| rdp     | rdp.ntlm.netbiosComputer  | string  | true |
| rdp     | rdp.ntlm.netbiosDomain    | string  | true |
| rdp     | rdp.ntlm.osBuild          | string  | true |
| rdp     | rdp.ntlm.osMajor          | string  | true |
| rdp     | rdp.ntlm.osMinor          | string  | true |
| rdp     | rdp.ntlm.osVersion        | string  | true |
| rdp     | rdp.ntlm.serverChallenge  | string  | true |
| rdp     | rdp.ntlm.targetName       | string  | true |
| rdp     | rdp.securityLevel         | string  | true |
| rdp     | rdp.selectedProtocol      | string  | true |
| rdp     | rdp.selectedProtocolName  | string  | true |
| rdp     | rdp.tls.fingerprintSHA256 | string  | true |
| rdp     | rdp.tls.issuer            | string  | true |
| rdp     | rdp.tls.notAfter          | string  | true |
| rdp     | rdp.tls.notBefore         | string  | true |
| rdp     | rdp.tls.subject           | string  | true |
| redfish | redfish.name              | string  | true |
| redfish | redfish.product           | string  | true |
| redfish | redfish.uuid              | string  | true |
| redfish | redfish.vendor            | string  | true |
| redfish | redfish.version           | string  | true |
| redis   | redis.archBits            | numeric | true |
| redis   | redis.authRequired        | boolean | true |
| redis   | redis.buildID             | string  | true |
| redis   | redis.clusterEnabled      | boolean | true |
| redis   | redis.connectedClients    | numeric | true |
| redis   | redis.error               | string  | true |
| redis   | redis.gccVersion          | string  | true |
| redis   | redis.gitSHA1             | string  | true |
| redis   | redis.maxMemory           | string  | true |
| redis   | redis.memAllocator        | string  | true |

|           |                        |         |       |
|-----------|------------------------|---------|-------|
| redis     | redis.mode             | string  | true  |
| redis     | redis.os               | string  | true  |
| redis     | redis.ping             | string  | true  |
| redis     | redis.port             | numeric | true  |
| redis     | redis.protectedMode    | string  | true  |
| redis     | redis.redisVersion     | string  | true  |
| redis     | redis.role             | string  | true  |
| redis     | redis.runID            | string  | true  |
| redis     | redis.totalCommands    | string  | true  |
| redis     | redis.totalConnections | string  | true  |
| redis     | redis.uptime           | numeric | true  |
| redis     | redis.usedMemory       | string  | true  |
| redis     | redis.valkeyVersion    | string  | true  |
| redis     | redis.version          | string  | true  |
| riak      | riak.nodename          | string  | true  |
| riak      | riak.version           | string  | true  |
| riak-http | riak-http.version      | string  | true  |
| rip       | rip.auth               | string  | true  |
| rip       | rip.nextHop            | string  | false |
| rip       | rip.route              | string  | false |
| rip       | rip.routeCidr          | string  | false |
| rip       | rip.routeCount         | string  | true  |
| rip       | rip.routingDomain      | string  | true  |
| rip       | rip.version            | string  | true  |
| roomalert | roomalert.ipAddress    | string  | false |
| roomalert | roomalert.macAddress   | string  | false |
| roomalert | roomalert.model        | string  | false |
| roomalert | roomalert.osVersion    | string  | false |
| roomalert | roomalert.serialNumber | string  | true  |
| rpcbind   | rpcbind.acceptState    | string  | true  |
| rpcbind   | rpcbind.acceptStatus   | string  | true  |

|         |                        |         |      |
|---------|------------------------|---------|------|
| rpcbind | rpcbind.addrs          | string  | true |
| rpcbind | rpcbind.authStatus     | string  | true |
| rpcbind | rpcbind.error          | string  | true |
| rpcbind | rpcbind.mappingCount   | numeric | true |
| rpcbind | rpcbind.mountd         | string  | true |
| rpcbind | rpcbind.nfs            | string  | true |
| rpcbind | rpcbind.nlockmgr       | string  | true |
| rpcbind | rpcbind.programEntries | string  | true |
| rpcbind | rpcbind.programs       | string  | true |
| rpcbind | rpcbind.rejectStatus   | string  | true |
| rpcbind | rpcbind.replyStatus    | string  | true |
| rpcbind | rpcbind.transport      | numeric | true |
| rpcbind | rpcbind.versionRange   | string  | true |
| rsync   | rsync.banner           | string  | true |
| rsync   | rsync.modules          | string  | true |
| rsync   | rsync.service          | string  | true |
| rsync   | rsync.version          | string  | true |
| rtmp    | rtmp.version           | string  | true |
| rtsp    | rtsp.banner            | string  | true |
| rtsp    | rtsp.contentType       | string  | true |
| rtsp    | rtsp.cseq              | string  | true |
| rtsp    | rtsp.head.*            | string  | true |
| rtsp    | rtsp.methodCount       | numeric | true |
| rtsp    | rtsp.methods           | string  | true |
| rtsp    | rtsp.port              | numeric | true |
| rtsp    | rtsp.public            | string  | true |
| rtsp    | rtsp.reasonPhrase      | string  | true |
| rtsp    | rtsp.request.cseq      | string  | true |
| rtsp    | rtsp.request.method    | string  | true |
| rtsp    | rtsp.request.uri       | string  | true |
| rtsp    | rtsp.server            | string  | true |

|        |                                 |         |      |
|--------|---------------------------------|---------|------|
| rtsp   | rtsp.session                    | string  | true |
| rtsp   | rtsp.statusClass                | string  | true |
| rtsp   | rtsp.statusCode                 | string  | true |
| rtsp   | rtsp.statusName                 | string  | true |
| rtsp   | rtsp.target                     | string  | true |
| rtsp   | rtsp.transport                  | numeric | true |
| rtsp   | rtsp.version                    | string  | true |
| rtsp   | rtsp.wwwAuthenticate            | string  | true |
| s7comm | s7comm.componentCount           | numeric | true |
| s7comm | s7comm.components               | string  | true |
| s7comm | s7comm.copyright                | string  | true |
| s7comm | s7comm.cpuType                  | string  | true |
| s7comm | s7comm.firmwareExtension        | string  | true |
| s7comm | s7comm.firmwareExtensionVersion | string  | true |
| s7comm | s7comm.firmwareName             | string  | true |
| s7comm | s7comm.firmwareVersion          | string  | true |
| s7comm | s7comm.hardwareName             | string  | true |
| s7comm | s7comm.hardwareVersion          | string  | true |
| s7comm | s7comm.locationDesignation      | string  | true |
| s7comm | s7comm.memoryCardSerial         | string  | true |
| s7comm | s7comm.moduleName               | string  | true |
| s7comm | s7comm.moduleType               | string  | true |
| s7comm | s7comm.moduleTypeName           | string  | true |
| s7comm | s7comm.moduleVersion            | string  | true |
| s7comm | s7comm.plantDesignation         | string  | true |
| s7comm | s7comm.protocolVersion          | string  | true |
| s7comm | s7comm.serialNumber             | string  | true |
| s7comm | s7comm.systemName               | string  | true |
| sadp   | sadp.activated                  | string  | true |
| sadp   | sadp.analogChannelNum           | string  | true |
| sadp   | sadp.bootTime                   | string  | true |

|            |                                 |        |      |
|------------|---------------------------------|--------|------|
| sadp       | sadp.cmdPort                    | string | true |
| sadp       | sadp.deviceDesc                 | string | true |
| sadp       | sadp.deviceSerial               | string | true |
| sadp       | sadp.deviceType                 | string | true |
| sadp       | sadp.dhcp                       | string | true |
| sadp       | sadp.digitalChannelNum          | string | true |
| sadp       | sadp.dspVersion                 | string | true |
| sadp       | sadp.httpPort                   | string | true |
| sadp       | sadp.ipv4.address               | string | true |
| sadp       | sadp.ipv4.gateway               | string | true |
| sadp       | sadp.ipv4.subnet                | string | true |
| sadp       | sadp.ipv6.address               | string | true |
| sadp       | sadp.ipv6.gateway               | string | true |
| sadp       | sadp.ipv6.maskLen               | string | true |
| sadp       | sadp.mac                        | string | true |
| sadp       | sadp.oemInfo                    | string | true |
| sadp       | sadp.passwordResetAbility       | string | true |
| sadp       | sadp.softwareVersion            | string | true |
| sadp       | sadp.uuid                       | string | true |
| sccp       | sccp.deviceType                 | string | true |
| sccp       | sccp.protocolVersion            | string | true |
| sccp       | sccp.rejected                   | string | true |
| securemote | securemote.hostname             | string | true |
| securemote | securemote.server               | string | true |
| servicetag | serviceTag.agentURN             | string | true |
| servicetag | serviceTag.agentVersion         | string | true |
| servicetag | serviceTag.registryVersion      | string | true |
| servicetag | serviceTag.servicePort          | string | true |
| servicetag | serviceTag.sysinfo.cpuInfo.name | string | true |
| servicetag | serviceTag.sysinfo.cpuMfg       | string | true |
| servicetag | serviceTag.sysinfo.hostID       | string | true |

|            |                                 |         |      |
|------------|---------------------------------|---------|------|
| servicetag | serviceTag.sysinfo.hostname     | string  | true |
| servicetag | serviceTag.sysinfo.mfg          | string  | true |
| servicetag | serviceTag.sysinfo.platform     | string  | true |
| servicetag | serviceTag.sysinfo.release      | string  | true |
| servicetag | serviceTag.sysinfo.serialNumber | string  | true |
| servicetag | serviceTag.sysinfo.system       | string  | true |
| servicetag | servicetag.cpuMfg               | string  | true |
| servicetag | servicetag.cpuName              | string  | true |
| servicetag | servicetag.hostid               | string  | true |
| servicetag | servicetag.hostname             | string  | true |
| servicetag | servicetag.manufacturer         | string  | true |
| servicetag | servicetag.platform             | string  | true |
| servicetag | servicetag.port                 | numeric | true |
| servicetag | servicetag.registryVersion      | string  | true |
| servicetag | servicetag.release              | string  | true |
| servicetag | servicetag.serial               | string  | true |
| servicetag | servicetag.system               | string  | true |
| servicetag | servicetag.urn                  | string  | true |
| servicetag | servicetag.version              | string  | true |
| sgsap      | sgsap.messageType               | string  | true |
| sgsap      | sgsap.sgsCause                  | string  | true |
| sip        | sip.accept                      | string  | true |
| sip        | sip.allow                       | string  | true |
| sip        | sip.contact                     | string  | true |
| sip        | sip.head.*                      | string  | true |
| sip        | sip.local                       | string  | true |
| sip        | sip.port                        | numeric | true |
| sip        | sip.reasonPhrase                | string  | true |
| sip        | sip.server                      | string  | true |
| sip        | sip.statusCode                  | string  | true |
| sip        | sip.supported                   | numeric | true |

|            |                         |         |      |
|------------|-------------------------|---------|------|
| sip        | sip.target              | string  | true |
| sip        | sip.userAgent           | string  | true |
| sip        | sip.via                 | string  | true |
| sip        | sip.wwwAuthenticate     | string  | true |
| sixnet-udr | redlion.authRequired    | string  | true |
| sixnet-udr | redlion.crc             | string  | true |
| sixnet-udr | redlion.exposure        | string  | true |
| sixnet-udr | redlion.model           | string  | true |
| sixnet-udr | redlion.sync            | string  | true |
| slp        | slp.attributes          | string  | true |
| slp        | slp.da.attributes       | string  | true |
| slp        | slp.da.bootTimestamp    | numeric | true |
| slp        | slp.da.scopes           | string  | true |
| slp        | slp.da.spi              | string  | true |
| slp        | slp.da.url              | string  | true |
| slp        | slp.error               | string  | true |
| slp        | slp.errorCode           | string  | true |
| slp        | slp.flags               | string  | true |
| slp        | slp.function            | string  | true |
| slp        | slp.functionID          | string  | true |
| slp        | slp.language            | string  | true |
| slp        | slp.length              | numeric | true |
| slp        | slp.nextExtensionOffset | string  | true |
| slp        | slp.port                | numeric | true |
| slp        | slp.sa.attributes       | string  | true |
| slp        | slp.sa.scopes           | string  | true |
| slp        | slp.sa.url              | string  | true |
| slp        | slp.serviceAgentURL     | string  | true |
| slp        | slp.serviceTypeCount    | numeric | true |
| slp        | slp.serviceTypes        | string  | true |
| slp        | slp.serviceTypesRaw     | string  | true |

|     |                     |         |       |
|-----|---------------------|---------|-------|
| slp | slp.target          | string  | true  |
| slp | slp.urlCount        | numeric | true  |
| slp | slp.urls            | string  | true  |
| slp | slp.version         | string  | true  |
| slp | slp.xid             | string  | true  |
| smb | smb.accessControl   | string  | true  |
| smb | smb.bootTime        | string  | true  |
| smb | smb.bootTimeTS      | string  | true  |
| smb | smb.capabilities    | string  | true  |
| smb | smb.cipherAlg       | string  | false |
| smb | smb.cipherAlgCnt    | numeric | true  |
| smb | smb.compAlg         | string  | false |
| smb | smb.compAlgCnt      | numeric | true  |
| smb | smb.compFlags       | string  | true  |
| smb | smb.dialect         | string  | true  |
| smb | smb.dialectIndex    | string  | true  |
| smb | smb.dialectName     | string  | true  |
| smb | smb.dialectRevision | string  | true  |
| smb | smb.dnsComputer     | string  | true  |
| smb | smb.dnsDomain       | string  | true  |
| smb | smb.dnsTree         | string  | true  |
| smb | smb.guid            | string  | true  |
| smb | smb.hashAlg         | string  | false |
| smb | smb.hashAlgCnt      | numeric | true  |
| smb | smb.hashSaltLen     | string  | true  |
| smb | smb.maxReadSize     | string  | true  |
| smb | smb.maxTransactSize | string  | true  |
| smb | smb.maxWriteSize    | string  | true  |
| smb | smb.nativeLM        | string  | true  |
| smb | smb.nativeOS        | string  | true  |
| smb | smb.negCtxCnt       | numeric | true  |



|      |                                |         |       |
|------|--------------------------------|---------|-------|
| smb  | smb.netName                    | string  | true  |
| smb  | smb.netbiosComputer            | string  | true  |
| smb  | smb.netbiosDomain              | string  | true  |
| smb  | smb.ntStatus                   | string  | true  |
| smb  | smb.ntlm.serverChallenge       | string  | true  |
| smb  | smb.ntlmssp                    | string  | true  |
| smb  | smb.ntlmsspNegFlags            | string  | true  |
| smb  | smb.ntlmsspVersion             | string  | true  |
| smb  | smb.posixExtensions            | string  | true  |
| smb  | smb.rdmaTransformCnt           | numeric | true  |
| smb  | smb.rdmaTransforms             | string  | false |
| smb  | smb.securityMode               | string  | true  |
| smb  | smb.serverGUID                 | string  | true  |
| smb  | smb.sessionID                  | string  | true  |
| smb  | smb.signing                    | string  | true  |
| smb  | smb.signingAlg                 | string  | false |
| smb  | smb.signingAlgCnt              | numeric | true  |
| smb  | smb.signingEnabled             | boolean | true  |
| smb  | smb.signingRequired            | boolean | true  |
| smb  | smb.supportsEncryptedPasswords | numeric | true  |
| smb  | smb.timestamp                  | numeric | true  |
| smb  | smb.timestampTS                | numeric | true  |
| smb  | smb.transportSecurity          | numeric | true  |
| smb  | smb.version                    | string  | true  |
| smb3 | ntlmssp.dnsComputer            | string  | true  |
| smb3 | ntlmssp.dnsDomain              | string  | true  |
| smb3 | ntlmssp.negFlags               | string  | true  |
| smb3 | ntlmssp.netbiosComputer        | string  | true  |
| smb3 | ntlmssp.netbiosDomain          | string  | true  |
| smb3 | ntlmssp.ntlmRevision           | numeric | true  |
| smb3 | ntlmssp.targetName             | string  | true  |

|      |                                |         |       |
|------|--------------------------------|---------|-------|
| smb3 | ntlmssp.version                | string  | true  |
| smb3 | ntlmssp.versionInvalid         | string  | true  |
| smb3 | smb.accessControl              | string  | true  |
| smb3 | smb.capabilities               | string  | true  |
| smb3 | smb.cipherAlg                  | string  | false |
| smb3 | smb.cipherAlgCnt               | string  | true  |
| smb3 | smb.compAlg                    | string  | false |
| smb3 | smb.compAlgCnt                 | string  | true  |
| smb3 | smb.compFlags                  | numeric | true  |
| smb3 | smb.dialect                    | string  | true  |
| smb3 | smb.guid                       | string  | true  |
| smb3 | smb.hashAlg                    | string  | false |
| smb3 | smb.hashAlgCnt                 | string  | true  |
| smb3 | smb.hashSaltLen                | string  | true  |
| smb3 | smb.nativeLM                   | string  | true  |
| smb3 | smb.nativeOS                   | string  | true  |
| smb3 | smb.netName                    | string  | true  |
| smb3 | smb.netbiosComputer            | string  | true  |
| smb3 | smb.netbiosDomain              | string  | true  |
| smb3 | smb.posixExtensions            | string  | true  |
| smb3 | smb.rdmaTransformCnt           | string  | true  |
| smb3 | smb.rdmaTransforms             | string  | false |
| smb3 | smb.sessionID                  | string  | true  |
| smb3 | smb.signing                    | string  | true  |
| smb3 | smb.signingAlg                 | string  | false |
| smb3 | smb.signingAlgCnt              | string  | true  |
| smb3 | smb.supportsEncryptedPasswords | string  | true  |
| smb3 | smb.transportSecurity          | string  | true  |
| snmp | snmp._secrets                  | string  | true  |
| snmp | snmp.a10.softwareVersion       | string  | true  |
| snmp | snmp.arpCount                  | numeric | true  |

|      |                            |         |       |
|------|----------------------------|---------|-------|
| snmp | snmp.arpcache              | string  | true  |
| snmp | snmp.arpcache.oids         | string  | false |
| snmp | snmp.arpcache.ports        | numeric | false |
| snmp | snmp.cdpCapabilities       | string  | true  |
| snmp | snmp.cdpDeviceIDs          | string  | true  |
| snmp | snmp.cdpIPs                | string  | true  |
| snmp | snmp.cdpNeighborCount      | numeric | true  |
| snmp | snmp.cdpPlatforms          | string  | true  |
| snmp | snmp.cdpPortIDs            | string  | true  |
| snmp | snmp.cisco.hardware        | string  | true  |
| snmp | snmp.cisco.hardwareRev     | string  | true  |
| snmp | snmp.cisco.softwareVersion | string  | true  |
| snmp | snmp.community             | string  | true  |
| snmp | snmp.credentials           | string  | true  |
| snmp | snmp.defaultCommunities    | string  | true  |
| snmp | snmp.engineBoots           | numeric | true  |
| snmp | snmp.engineID.enterpriseID | string  | true  |
| snmp | snmp.engineID.raw          | string  | true  |
| snmp | snmp.engineID.vendor       | string  | true  |
| snmp | snmp.engineTime            | string  | true  |
| snmp | snmp.errorIndex            | string  | true  |
| snmp | snmp.errorStatus           | string  | true  |
| snmp | snmp.errorStatusName       | string  | true  |
| snmp | snmp.extendOsReleaseFull   | string  | true  |
| snmp | snmp.f5.hardware           | string  | true  |
| snmp | snmp.f5.softwareBuild      | string  | true  |
| snmp | snmp.f5.softwareVersion    | string  | true  |
| snmp | snmp.failedAuth            | string  | true  |
| snmp | snmp.fortinet.fgSysVersion | string  | true  |
| snmp | snmp.fortinet.fnSysSerial  | string  | true  |
| snmp | snmp.fortinet.fnSysVersion | string  | true  |

|      |                              |         |       |
|------|------------------------------|---------|-------|
| snmp | snmp.hostname                | string  | true  |
| snmp | snmp.ifCount                 | numeric | true  |
| snmp | snmp.inauthentic             | string  | true  |
| snmp | snmp.installedSWErr          | string  | true  |
| snmp | snmp.installedSWHWHint       | string  | true  |
| snmp | snmp.installedSWOSHint       | string  | true  |
| snmp | snmp.installedSWProtoVer     | string  | true  |
| snmp | snmp.installedSoftware       | string  | true  |
| snmp | snmp.installedSoftwareCount  | numeric | true  |
| snmp | snmp.interfaceAddrs          | string  | true  |
| snmp | snmp.interfaceAddrsMap       | string  | true  |
| snmp | snmp.interfaceAliases        | string  | true  |
| snmp | snmp.interfaceAliasesMap     | string  | true  |
| snmp | snmp.interfaceCount          | numeric | true  |
| snmp | snmp.interfaceMacs           | string  | true  |
| snmp | snmp.interfaceMacsMap        | string  | true  |
| snmp | snmp.interfaceNames          | string  | true  |
| snmp | snmp.interfaceNamesMap       | string  | true  |
| snmp | snmp.interfaces              | string  | true  |
| snmp | snmp.ivantiv.esapVersion     | string  | true  |
| snmp | snmp.ivantiv.productName     | string  | true  |
| snmp | snmp.ivantiv.productVersion  | string  | true  |
| snmp | snmp.juniper.serialNumbers   | string  | true  |
| snmp | snmp.juniper.softwareVersion | string  | true  |
| snmp | snmp.juniperModel            | string  | true  |
| snmp | snmp.lldp.neighbor           | string  | false |
| snmp | snmp.lldpChassisIDTypes      | string  | true  |
| snmp | snmp.lldpChassisIDs          | string  | true  |
| snmp | snmp.lldpNeighborCount       | numeric | true  |
| snmp | snmp.lldpPortDescs           | numeric | true  |
| snmp | snmp.lldpPortIDs             | numeric | true  |

|      |                                |         |      |
|------|--------------------------------|---------|------|
| snmp | snmp.lldpPortNums              | numeric | true |
| snmp | snmp.lldpSysDescrs             | string  | true |
| snmp | snmp.lldpSysNames              | string  | true |
| snmp | snmp.macs.ports                | numeric | true |
| snmp | snmp.macs.vlans                | string  | true |
| snmp | snmp.modelNames                | string  | true |
| snmp | snmp.msgMaxSize                | string  | true |
| snmp | snmp.pan.appDatabaseVersion    | string  | true |
| snmp | snmp.pan.avDatabaseVersion     | string  | true |
| snmp | snmp.pan.hardwareVersion       | string  | true |
| snmp | snmp.pan.model                 | string  | true |
| snmp | snmp.pan.serialNumber          | string  | true |
| snmp | snmp.pan.softwareVersion       | string  | true |
| snmp | snmp.pan.threatDatabaseVersion | string  | true |
| snmp | snmp.pan.urlDatabaseVersion    | string  | true |
| snmp | snmp.pan.vpnClientVersion      | string  | true |
| snmp | snmp.port                      | numeric | true |
| snmp | snmp.routeCount                | numeric | true |
| snmp | snmp.routes                    | string  | true |
| snmp | snmp.secretCommunities         | string  | true |
| snmp | snmp.serialNumbers             | string  | true |
| snmp | snmp.sonicwall.hardware        | string  | true |
| snmp | snmp.sonicwall.serialNumber    | string  | true |
| snmp | snmp.sonicwall.softwareVersion | string  | true |
| snmp | snmp.sophos.ha.peer            | string  | true |
| snmp | snmp.sophos.ha.status          | string  | true |
| snmp | snmp.sophos.hardware           | string  | true |
| snmp | snmp.sophos.softwareVersion    | string  | true |
| snmp | snmp.sysContact                | string  | true |
| snmp | snmp.sysDesc                   | string  | true |
| snmp | snmp.sysDescr                  | string  | true |

|        |                                   |         |      |
|--------|-----------------------------------|---------|------|
| snmp   | snmp.sysLocation                  | string  | true |
| snmp   | snmp.sysName                      | string  | true |
| snmp   | snmp.sysObjectID                  | string  | true |
| snmp   | snmp.sysServices                  | string  | true |
| snmp   | snmp.sysUpTimeFormatted           | string  | true |
| snmp   | snmp.sysUptime                    | string  | true |
| snmp   | snmp.sysUptimeFormatted           | string  | true |
| snmp   | snmp.target                       | string  | true |
| snmp   | snmp.username                     | string  | true |
| snmp   | snmp.usmStatsDecryptionErrors     | string  | true |
| snmp   | snmp.usmStatsNotInTimeWindows     | string  | true |
| snmp   | snmp.usmStatsUnknownEngineIDs     | string  | true |
| snmp   | snmp.usmStatsUnknownUserNames     | string  | true |
| snmp   | snmp.usmStatsUnsupportedSecLevels | numeric | true |
| snmp   | snmp.usmStatsWrongDigests         | numeric | true |
| snmp   | snmp.varbindCount                 | numeric | true |
| snmp   | snmp.version                      | string  | true |
| snmp   | snmp.vlanCount                    | numeric | true |
| snmp   | snmp.vlans                        | string  | true |
| socks4 | socks4.anonymousAccess            | string  | true |
| socks4 | socks4.status                     | string  | true |
| socks5 | socks5.authMethod                 | string  | true |
| socks5 | socks5.authRequired               | boolean | true |
| socks5 | socks5.noAcceptable               | string  | true |
| socks5 | socks5.open                       | string  | true |
| socks5 | socks5.socks4                     | string  | true |
| socks5 | socks5.version                    | string  | true |
| solr   | service.product                   | string  | true |
| solr   | service.version                   | string  | true |
| solr   | solr.jvm.name                     | string  | true |
| solr   | solr.jvm.version                  | string  | true |

|                 |                     |         |       |
|-----------------|---------------------|---------|-------|
| solr            | solr.lucene.version | string  | true  |
| solr            | solr.system.arch    | string  | true  |
| solr            | solr.system.name    | string  | true  |
| solr            | solr.system.version | string  | true  |
| solr            | solr.version        | string  | true  |
| splunk          | splunk.build        | string  | true  |
| splunk          | splunk.version      | string  | true  |
| spotify-connect | brandDisplayName    | string  | true  |
| spotify-connect | clientID            | string  | true  |
| spotify-connect | deviceID            | string  | true  |
| spotify-connect | deviceType          | string  | true  |
| spotify-connect | modelDisplayName    | string  | true  |
| spotify-connect | remoteName          | string  | true  |
| spotify-connect | scope               | string  | true  |
| spotify-connect | status              | string  | true  |
| spotify-connect | statusString        | string  | true  |
| ssh             | ssh.authMethods     | string  | false |
| ssh             | ssh.banner          | string  | true  |
| ssh             | ssh.comments        | numeric | true  |
| ssh             | ssh.error           | string  | true  |
| ssh             | ssh.exitStatus      | string  | true  |
| ssh             | ssh.extensionNames  | string  | true  |
| ssh             | ssh.extensions      | string  | false |
| ssh             | ssh.firstKexFollows | string  | true  |
| ssh             | ssh.hostKey         | string  | true  |
| ssh             | ssh.hostKey.bkhash  | string  | true  |
| ssh             | ssh.hostKey.data    | string  | true  |
| ssh             | ssh.hostKey.md5     | string  | true  |
| ssh             | ssh.hostKey.sha256  | string  | true  |
| ssh             | ssh.hostKeyAlgs     | string  | true  |
| ssh             | ssh.hostKeyCount    | numeric | true  |

|       |                          |        |       |
|-------|--------------------------|--------|-------|
| ssh   | ssh.hostKeyTypes         | string | false |
| ssh   | ssh.kbdInstructions      | string | true  |
| ssh   | ssh.kbdName              | string | true  |
| ssh   | ssh.kbdQuestion          | string | true  |
| ssh   | ssh.kexAlgs              | string | true  |
| ssh   | ssh.kexCookie            | string | true  |
| ssh   | ssh.kexPadding           | string | true  |
| ssh   | ssh.preAuthBanner        | string | true  |
| ssh   | ssh.protoVersion         | string | true  |
| ssh   | ssh.service              | string | true  |
| ssh   | ssh.sessionMethod        | string | true  |
| ssh   | ssh.sessionOutput        | string | true  |
| ssh   | ssh.softwareVersion      | string | true  |
| ssh   | ssh.stage                | string | true  |
| ssh   | ssh.toClientCiphers      | string | true  |
| ssh   | ssh.toClientCompression  | string | true  |
| ssh   | ssh.toClientComps        | string | true  |
| ssh   | ssh.toClientLangs        | string | true  |
| ssh   | ssh.toClientMACs         | string | true  |
| ssh   | ssh.toServerCiphers      | string | true  |
| ssh   | ssh.toServerCompression  | string | true  |
| ssh   | ssh.toServerComps        | string | true  |
| ssh   | ssh.toServerLangs        | string | true  |
| ssh   | ssh.toServerMACs         | string | true  |
| ssh   | ssh.version              | string | true  |
| steam | steam.authKeyIDs         | string | true  |
| steam | steam.broadcastingActive | string | true  |
| steam | steam.clientID           | string | true  |
| steam | steam.connectPort        | string | true  |
| steam | steam.contentCachePort   | string | true  |
| steam | steam.deviceID           | string | true  |



|       |                            |         |      |
|-------|----------------------------|---------|------|
| steam | steam.downloadLANPeerGroup | string  | true |
| steam | steam.eUniverse            | string  | true |
| steam | steam.enabledServices      | string  | true |
| steam | steam.gamesRunning         | string  | true |
| steam | steam.hostname             | string  | true |
| steam | steam.instanceID           | string  | true |
| steam | steam.ipAddresses          | string  | true |
| steam | steam.is64Bit              | string  | true |
| steam | steam.isSteamDeck          | string  | true |
| steam | steam.macAddresses         | string  | true |
| steam | steam.minVersion           | string  | true |
| steam | steam.osType               | string  | true |
| steam | steam.publicIPAddress      | string  | true |
| steam | steam.remotePlayActive     | string  | true |
| steam | steam.screenLocked         | string  | true |
| steam | steam.steamIDs             | string  | true |
| steam | steam.steamVersion         | string  | true |
| steam | steam.supportedServices    | string  | true |
| steam | steam.timestamp            | string  | true |
| steam | steam.version              | string  | true |
| steam | steam.vrActive             | string  | true |
| stun  | stun.attributeCount        | numeric | true |
| stun  | stun.attributes            | string  | true |
| stun  | stun.errorCode             | string  | true |
| stun  | stun.errorReason           | string  | true |
| stun  | stun.fingerprint           | string  | true |
| stun  | stun.mappedAddress         | string  | true |
| stun  | stun.mappedIP              | string  | true |
| stun  | stun.mappedPort            | numeric | true |
| stun  | stun.messageClass          | string  | true |
| stun  | stun.messageLength         | string  | true |

|        |                       |        |      |
|--------|-----------------------|--------|------|
| stun   | stun.messageMethod    | string | true |
| stun   | stun.otherAddress     | string | true |
| stun   | stun.responseOrigin   | string | true |
| stun   | stun.software         | string | true |
| stun   | stun.transactionID    | string | true |
| stun   | stun.type             | string | true |
| stun   | stun.xorMappedAddress | string | true |
| sua    | sua.errorCode         | string | true |
| sua    | sua.infoString        | string | true |
| sua    | sua.messageClass      | string | true |
| sua    | sua.messageType       | string | true |
| sunrpc | rpcbind.mappingCount  | string | true |
| sunrpc | rpcbind.mountd        | string | true |
| sunrpc | rpcbind.nfs           | string | true |
| sunrpc | rpcbind.programs      | string | true |
| svn    | svn.authMechanisms    | string | true |
| svn    | svn.banner            | string | true |
| svn    | svn.capabilities      | string | true |
| svn    | svn.maxVersion        | string | true |
| svn    | svn.minVersion        | string | true |
| svn    | svn.repoURL           | string | true |
| sybase | sybase.target         | string | true |
| sybase | sybase.version        | string | true |
| syslog | syslog.appName        | string | true |
| syslog | syslog.facility       | string | true |
| syslog | syslog.facilityName   | string | true |
| syslog | syslog.hostname       | string | true |
| syslog | syslog.priority       | string | true |
| syslog | syslog.rfc            | string | true |
| syslog | syslog.severity       | string | true |
| syslog | syslog.severityName   | string | true |

|            |                           |                      |      |
|------------|---------------------------|----------------------|------|
| syslog     | syslog.version            | string               | true |
| teamviewer | teamviewer.command        | string               | true |
| teamviewer | teamviewer.magic          | string               | true |
| teamviewer | teamviewer.response       | string (hex-encoded) | true |
| tftp       | tftp.banner               | string               | true |
| tftp       | tftp.error                | string               | true |
| tftp       | tftp.example.test         | string               | true |
| tftp       | tftp.message              | string               | true |
| tftp       | tftp.mode                 | string               | true |
| tftp       | tftp.opcode               | string               | true |
| tftp       | tftp.rrq.filename         | string               | true |
| tftp       | tftp.wrq.filename         | string               | true |
| time       | time.skewMS               | string               | true |
| time       | time.timestamp            | string               | true |
| time       | time.timestampTS          | numeric              | true |
| time       | time.value                | numeric              | true |
| tls        | tls.alert                 | string               | true |
| tls        | tls.alertDescription      | string               | true |
| tls        | tls.alertLevel            | string               | true |
| tls        | tls.alpn                  | string               | true |
| tls        | tls.authorityKeyID        | string (hex-encoded) | true |
| tls        | tls.caUnknown             | string               | true |
| tls        | tls.certificate           | string               | true |
| tls        | tls.certificateCount      | numeric              | true |
| tls        | tls.certificateParseError | string               | true |
| tls        | tls.certificates          | string               | true |
| tls        | tls.chainFingerprints     | numeric              | true |
| tls        | tls.cipher                | string               | true |
| tls        | tls.cipherName            | string               | true |
| tls        | tls.cipherSuiteNames      | string               | true |
| tls        | tls.cipherSuites          | string               | true |

|     |                              |         |       |
|-----|------------------------------|---------|-------|
| tls | tls.cn                       | string  | true  |
| tls | tls.compression              | string  | true  |
| tls | tls.crl                      | string  | false |
| tls | tls.emails                   | string  | false |
| tls | tls.errors                   | string  | true  |
| tls | tls.expired                  | string  | true  |
| tls | tls.extensionNames           | string  | true  |
| tls | tls.extensions               | string  | true  |
| tls | tls.fp.bkhash                | string  | true  |
| tls | tls.fp.caSha1                | string  | false |
| tls | tls.fp.sha1                  | string  | true  |
| tls | tls.fp.sha1Hex               | string  | true  |
| tls | tls.fp.sha256                | string  | true  |
| tls | tls.helloVersion             | string  | true  |
| tls | tls.hostTime                 | string  | true  |
| tls | tls.hostTimeTS               | numeric | true  |
| tls | tls.issuer                   | string  | true  |
| tls | tls.issuingURL               | string  | false |
| tls | tls.names                    | string  | false |
| tls | tls.negotiationError         | string  | true  |
| tls | tls.notAfter                 | string  | true  |
| tls | tls.notAfterTS               | numeric | true  |
| tls | tls.notBefore                | string  | true  |
| tls | tls.notBeforeTS              | numeric | true  |
| tls | tls.ocsp                     | string  | false |
| tls | tls.ocspResponse             | string  | true  |
| tls | tls.port                     | numeric | true  |
| tls | tls.pqHybridKeyExchangeCodes | string  | false |
| tls | tls.pqHybridKeyExchangeNames | string  | false |
| tls | tls.pqc                      | string  | false |
| tls | tls.publicKeyAlgorithm       | string  | true  |

|      |                               |                      |       |
|------|-------------------------------|----------------------|-------|
| tls  | tls.random                    | string               | true  |
| tls  | tls.requiresClientCertificate | string               | true  |
| tls  | tls.rzfp                      | string               | true  |
| tls  | tls.rzfp0                     | string               | true  |
| tls  | tls.sans                      | string               | true  |
| tls  | tls.selfSigned                | string               | true  |
| tls  | tls.serial                    | string               | true  |
| tls  | tls.sessionID                 | string               | true  |
| tls  | tls.signatureAlgorithm        | string               | true  |
| tls  | tls.stack                     | string               | true  |
| tls  | tls.subject                   | string               | true  |
| tls  | tls.subjectKeyID              | string (hex-encoded) | true  |
| tls  | tls.supportedVersionCodes     | string               | true  |
| tls  | tls.supportedVersionNames     | string               | true  |
| tls  | tls.supportedVersions         | string               | false |
| tls  | tls.target                    | string               | true  |
| tls  | tls.uri                       | string               | false |
| tls  | tls.version                   | string               | true  |
| tls  | tls.versionName               | string               | true  |
| turn | turn.attributeCount           | numeric              | true  |
| turn | turn.attributes               | string               | true  |
| turn | turn.errorCode                | string               | true  |
| turn | turn.errorReason              | string               | true  |
| turn | turn.fingerprint              | string               | true  |
| turn | turn.lifetime                 | string               | true  |
| turn | turn.mappedAddress            | string               | true  |
| turn | turn.mappedIP                 | string               | true  |
| turn | turn.mappedPort               | numeric              | true  |
| turn | turn.messageClass             | string               | true  |
| turn | turn.messageMethod            | string               | true  |
| turn | turn.nonce                    | string               | true  |

|            |                          |         |       |
|------------|--------------------------|---------|-------|
| turn       | turn.realm               | string  | true  |
| turn       | turn.relayedIP           | string  | true  |
| turn       | turn.relayedPort         | numeric | true  |
| turn       | turn.requestedTransport  | numeric | true  |
| turn       | turn.software            | string  | true  |
| turn       | turn.type                | string  | true  |
| turn       | turn.xorMappedAddress    | string  | true  |
| turn       | turn.xorRelayedAddress   | string  | true  |
| ubnt       | ubnt.addr5               | string  | false |
| ubnt       | ubnt.configStatus        | string  | true  |
| ubnt       | ubnt.directConnectDomain | string  | true  |
| ubnt       | ubnt.essid               | string  | true  |
| ubnt       | ubnt.firmware            | string  | true  |
| ubnt       | ubnt.hostName            | string  | true  |
| ubnt       | ubnt.interfaceMap        | string  | false |
| ubnt       | ubnt.macs                | string  | false |
| ubnt       | ubnt.modelFull           | string  | true  |
| ubnt       | ubnt.modelShort          | string  | true  |
| ubnt       | ubnt.protoVersion        | string  | true  |
| ubnt       | ubnt.sourceMAC           | string  | false |
| ubnt       | ubnt.unifiVersion        | string  | true  |
| ubnt       | ubnt.uptime              | string  | true  |
| ubnt       | ubnt.webMgmtPort         | string  | true  |
| ubnt       | ubnt.webMgmtTLS          | string  | true  |
| ubnt       | ubnt.wmode               | string  | false |
| unitronics | unitronics.cpe           | string  | true  |
| unitronics | unitronics.hwVersion     | string  | true  |
| unitronics | unitronics.model         | string  | true  |
| unitronics | unitronics.osVersion     | string  | true  |
| unitronics | unitronics.unitID        | string  | true  |
| upnp       | upnp.controlURL          | string  | true  |

|        |                        |        |      |
|--------|------------------------|--------|------|
| upnp   | upnp.deviceType        | string | true |
| upnp   | upnp.eventSubURL       | string | true |
| upnp   | upnp.friendlyName      | string | true |
| upnp   | upnp.manufacturer      | string | true |
| upnp   | upnp.manufacturerURL   | string | true |
| upnp   | upnp.modelDescription  | string | true |
| upnp   | upnp.modelName         | string | true |
| upnp   | upnp.modelNumber       | string | true |
| upnp   | upnp.modelURL          | string | true |
| upnp   | upnp.presentationURL   | string | true |
| upnp   | upnp.scpdURL           | string | true |
| upnp   | upnp.serialNumber      | string | true |
| upnp   | upnp.udn               | string | true |
| upnp   | upnp.upc               | string | true |
| upnp   | upnp.url               | string | true |
| upnp   | upnp.urlBase           | string | true |
| upnp   | upnp.wifiMac           | string | true |
| upnp   | upnp.wiredMac          | string | true |
| uscan  | uscan.makeAndModel     | string | true |
| uscan  | uscan.manufacturer     | string | true |
| uscan  | uscan.serialNumber     | string | true |
| uscan  | uscan.version          | string | true |
| vault  | vault.version          | string | true |
| vmware | http.vmware.thumbprint | string | true |
| vmware | service.product        | string | true |
| vmware | service.vendor         | string | true |
| vmware | service.version        | string | true |
| vnc    | vnc.banner             | string | true |
| vnc    | vnc.desktopName        | string | true |
| vnc    | vnc.error              | string | true |
| vnc    | vnc.framebufferHeight  | string | true |

|        |                       |         |       |
|--------|-----------------------|---------|-------|
| vnc    | vnc.framebufferWidth  | string  | true  |
| vnc    | vnc.majorVersion      | string  | true  |
| vnc    | vnc.minorVersion      | string  | true  |
| vnc    | vnc.noAuth            | string  | true  |
| vnc    | vnc.securityTypeCount | numeric | true  |
| vnc    | vnc.securityTypes     | string  | true  |
| vnc    | vnc.version           | string  | true  |
| wbsm   | wbsm.active           | string  | true  |
| webmin | webmin.port           | string  | true  |
| webmin | webmin.scheme         | string  | true  |
| webmin | webmin.server         | string  | true  |
| wiznet | wiznet.fields         | string  | true  |
| wsd    | wsd.addrs             | string  | false |
| wsd    | wsd.types             | string  | true  |
| wsman  | wsman.body            | string  | true  |
| x11    | x11.accessGranted     | string  | true  |
| x11    | x11.anonymousAccess   | string  | true  |
| x11    | x11.displayNumber     | string  | true  |
| x11    | x11.failureReason     | string  | true  |
| x11    | x11.status            | string  | true  |
| x11    | x11.vendor            | string  | true  |
| x11    | x11.version           | string  | true  |
| x2ap   | x2ap.criticality      | string  | true  |
| x2ap   | x2ap.messageType      | string  | true  |
| x2ap   | x2ap.procedureCode    | string  | true  |
| xdmcp  | xdmcp.authName        | string  | true  |
| xdmcp  | xdmcp.hostname        | string  | true  |
| xdmcp  | xdmcp.opcode          | string  | true  |
| xdmcp  | xdmcp.status          | string  | true  |
| xdmcp  | xdmcp.version         | string  | true  |



|              |                              |         |       |
|--------------|------------------------------|---------|-------|
| zabbix       | zabbix.agentData             | string  | true  |
| zabbix       | zabbix.agentVersion          | string  | true  |
| zabbix       | zabbix.cpe23                 | string  | true  |
| zabbix       | zabbix.remoteCommandsEnabled | boolean | true  |
| zabbix       | zabbix.version               | string  | true  |
| zabbix-agent | zabbix.agentVersion          | string  | true  |
| zebra        | host.name                    | string  | true  |
| zebra        | zebra.deviceName             | string  | false |
| zebra        | zebra.firmware               | string  | false |
| zebra        | zebra.ip                     | string  | false |
| zebra        | zebra.model                  | string  | false |
| zebra        | zebra.response               | string  | false |
| zebra        | zebra.serial                 | string  | false |
| zookeeper    | zk.access                    | string  | true  |
| zookeeper    | zookeeper.access             | string  | true  |
| zookeeper    | zookeeper.banner             | string  | true  |
| zookeeper    | zookeeper.connections        | numeric | true  |
| zookeeper    | zookeeper.imok               | string  | true  |
| zookeeper    | zookeeper.mode               | string  | true  |
| zookeeper    | zookeeper.nodeCount          | numeric | true  |
| zookeeper    | zookeeper.port               | numeric | true  |
| zookeeper    | zookeeper.restricted         | string  | true  |
| zookeeper    | zookeeper.restrictionReason  | string  | true  |
| zookeeper    | zookeeper.ruok               | string  | true  |
| zookeeper    | zookeeper.srvr               | string  | true  |
| zookeeper    | zookeeper.target             | string  | true  |
| zookeeper    | zookeeper.version            | string  | true  |
| zookeeper    | zookeeper.versionRaw         | string  | true  |
| zyxel        | zyxel.builddate              | string  | true  |
| zyxel        | zyxel.dhcpstate              | string  | true  |
| zyxel        | zyxel.firmware               | string  | true  |

|       |                  |        |      |
|-------|------------------|--------|------|
| zyxel | zyxel.firstlogin | string | true |
| zyxel | zyxel.gateway    | string | true |
| zyxel | zyxel.hostname   | string | true |
| zyxel | zyxel.ip         | string | true |
| zyxel | zyxel.mac        | string | true |
| zyxel | zyxel.maxport    | string | true |
| zyxel | zyxel.model      | string | true |
| zyxel | zyxel.subnetmask | string | true |
| zyxel | zyxel.uptime     | string | true |

## Release notes

No release notes available.

# Index

## 2

2FA: [100](#)

## A

Account API: [101](#) [642](#)  
 Action: [490](#)  
 Activating your account: [3](#)  
 Active scan: [134](#)  
 Active scanning: [130](#)  
 Adaptive Cards: [493](#)  
 Add multiple users: [113](#)  
 Address ranges: [135](#)  
 Administrators: [96](#)  
 AI threads and artifacts: [4](#)  
 Alert messages: [492](#)  
 Alerts: [8](#)  
 Alerts: [489](#) [489](#) [595](#)  
 Alive: [453](#)  
 Amazon Web Services: [222](#)  
 Analysis reports: [429](#)  
 Annotators: [96](#)  
 API client: [643](#) [649](#)  
 API client credentials: [642](#)  
 API key: [642](#)  
 Area: [103](#)  
 ARP: [453](#)  
 Asset category: [451](#)  
 Asset data: [703](#)  
 Asset functions: [452](#)  
 Asset geolocation: [461](#)  
 Asset risk report: [458](#)  
 Asset route pathing: [480](#)  
 Asset route pathing report: [637](#)  
 Assets: [442](#) [450](#)  
 Assets back online: [425](#)  
 Assets changed: [425](#)  
 Assets ignored: [425](#)  
 Assets marked offline: [425](#)  
 Assets unchanged: [425](#)  
 Atlassian Insight: [372](#)  
 Authenticated SNMP v3: [155](#)  
 Authentication passphrase: [155](#)  
 Authentication protocol: [155](#)  
 AuthNoPriv: [154](#)

|                       |         |
|-----------------------|---------|
| AuthPriv:             | 154     |
| Automated installers: | 31      |
| Automatic queries:    | 597     |
| Autonomous discovery: | 140     |
| AWS:                  | 222 232 |
| Azure AD:             | 123 307 |

## B

|                      |     |
|----------------------|-----|
| Billing:             | 96  |
| Billing information: | 128 |
| Binary signature:    | 38  |
| Block Kit:           | 493 |
| Bridge report:       | 479 |
| Browser versions:    | 698 |
| Bulk asset update:   | 447 |
| Bulk import users:   | 113 |

## C

|                                  |     |
|----------------------------------|-----|
| CA:                              | 460 |
| Caret:                           | 511 |
| CDE:                             | 479 |
| CEE:                             | 71  |
| Censys Search:                   | 234 |
| Certificate authority:           | 460 |
| Certificates:                    | 9   |
| Change report:                   | 707 |
| Channel:                         | 491 |
| Channels:                        | 489 |
| CIDR allow list:                 | 155 |
| Cisco ASA:                       | 691 |
| Cisco Catalyst:                  | 633 |
| Cisco Meraki Dashboard:          | 236 |
| Client token:                    | 643 |
| CMDB:                            | 384 |
| Colors:                          | 638 |
| Common Name:                     | 459 |
| Community strings:               | 154 |
| Comparison:                      | 625 |
| Compliance:                      | 455 |
| Connect:                         | 448 |
| Connector task:                  | 218 |
| Container:                       | 20  |
| Coverage reports:                | 621 |
| Credentials:                     | 153 |
| Criticality:                     | 457 |
| CrowdStrike:                     | 238 |
| Cryptographic bill of materials: | 640 |
| CSV:                             | 598 |
| Curly brackets:                  | 495 |

|                                     |                     |
|-------------------------------------|---------------------|
| Custom integration:                 | <a href="#">244</a> |
| Custom integration:                 | <a href="#">245</a> |
| Custom integration script versions: | <a href="#">4</a>   |
| Custom role:                        | <a href="#">103</a> |

## D

|                                          |                     |
|------------------------------------------|---------------------|
| Dashboard:                               | <a href="#">8</a>   |
| Dashboards:                              | <a href="#">430</a> |
| Data deletion after account termination: | <a href="#">4</a>   |
| Data formats:                            | <a href="#">700</a> |
| Data retention:                          | <a href="#">91</a>  |
| Debug SNMP v3:                           | <a href="#">157</a> |
| Defender:                                | <a href="#">291</a> |
| Department:                              | <a href="#">51</a>  |
| Deploy:                                  | <a href="#">20</a>  |
| Deployment plan:                         | <a href="#">41</a>  |
| Detected by:                             | <a href="#">453</a> |
| Device type:                             | <a href="#">451</a> |
| DHCP:                                    | <a href="#">442</a> |
| Diagnostics collection:                  | <a href="#">88</a>  |
| Differences:                             | <a href="#">625</a> |
| Directory services:                      | <a href="#">449</a> |
| Distinguished name:                      | <a href="#">459</a> |
| DN:                                      | <a href="#">459</a> |
| DNS:                                     | <a href="#">450</a> |
| Docker:                                  | <a href="#">20</a>  |
| Domain membership:                       | <a href="#">429</a> |
| Download:                                | <a href="#">20</a>  |
| Download token:                          | <a href="#">642</a> |
| Dragos:                                  | <a href="#">283</a> |
| Duplicate assets:                        | <a href="#">53</a>  |

## E

|                      |                                                           |
|----------------------|-----------------------------------------------------------|
| EC2:                 | <a href="#">232</a>                                       |
| ElasticSearch:       | <a href="#">71</a>                                        |
| Email:               | <a href="#">489</a>                                       |
| Email runZero Users: | <a href="#">489</a>                                       |
| Embedded devices:    | <a href="#">21</a>                                        |
| End-of-Life:         | <a href="#">455</a>                                       |
| Entra:               | <a href="#">123</a>                                       |
| Entra ID:            | <a href="#">307</a>                                       |
| ESSID:               | <a href="#">570</a>                                       |
| ESXi:                | <a href="#">367</a>                                       |
| Event records:       | <a href="#">4</a>                                         |
| Events:              | <a href="#">489</a>                                       |
| Example queries:     | <a href="#">522</a>                                       |
| Explorer:            | <a href="#">20</a> <a href="#">33</a> <a href="#">134</a> |
| Export API:          | <a href="#">642</a>                                       |
| Export asset data:   | <a href="#">598</a>                                       |

|                             |     |
|-----------------------------|-----|
| Export token:               | 642 |
| External Asset Report:      | 623 |
| External network discovery: | 20  |
| External reporting:         | 629 |
| External users:             | 115 |

## F

|                      |     |
|----------------------|-----|
| Filter:              | 520 |
| Finding code:        | 578 |
| Findings:            | 455 |
| Findings:            | 578 |
| Fingerprint results: | 442 |
| Firewall:            | 691 |
| First scans:         | 146 |
| Flat networks:       | 53  |

## G

|                         |         |
|-------------------------|---------|
| GCP:                    | 285     |
| GeoNames:               | 463 465 |
| Global credential:      | 153     |
| Glossary:               | 689     |
| Goal tracking:          | 474     |
| Goals:                  | 474     |
| Google Chrome:          | 694     |
| Google Cloud Platform:  | 285     |
| Google Workspace:       | 288     |
| Grace period:           | 6       |
| Grant:                  | 649     |
| Graphviz:               | 638     |
| Group directory:        | 449     |
| Group mapping search:   | 590     |
| Group search:           | 589     |
| Groups:                 | 9       |
| Groups changed:         | 426     |
| Groups unchanged:       | 426     |
| Groups updated by task: | 426     |

## H

|                           |             |
|---------------------------|-------------|
| Hardware:                 | 453         |
| High-availability:        | 84          |
| Historical changes:       | 625         |
| Hop:                      | 637         |
| Host ping:                | 139         |
| Hosted External Explorer: | 135         |
| Hostnames:                | 135         |
| Hostnames:                | 442 450 450 |
| HP iLO:                   | 599         |

**I**

|                     |         |
|---------------------|---------|
| IAM:                | 232     |
| ICMP:               | 453     |
| ICMP ping:          | 140     |
| Insights:           | 8       |
| Insights:           | 432     |
| InsightVM:          | 338 338 |
| InsightVM Cloud:    | 338     |
| Installation:       | 3       |
| IntegrationHub ETL: | 384     |
| Internet Exposure:  | 455     |
| Inventory:          | 442     |
| Inventory views:    | 8       |
| Invoices:           | 129     |
| IoT:                | 212     |
| IP addresses:       | 442 450 |
| IPv4:               | 621     |
| IRE:                | 384     |
| Issues:             | 422     |
| ITOM:               | 385     |

**J**

|                          |     |
|--------------------------|-----|
| Jira:                    | 377 |
| Jira Service Management: | 372 |
| Journal:                 | 71  |
| Journalctl:              | 71  |
| JSON:                    | 598 |
| JSONL:                   | 700 |

**K**

|                |     |
|----------------|-----|
| Knowledgebase: | 691 |
|----------------|-----|

**L**

|            |     |
|------------|-----|
| LDAP:      | 297 |
| Licensing: | 128 |
| Logging:   | 71  |

**M**

|                         |         |
|-------------------------|---------|
| MAC addresses:          | 442 450 |
| Magnifying glass:       | 622     |
| Mattermost:             | 489     |
| Max group size:         | 137     |
| Max group size:         | 692     |
| MFA:                    | 5       |
| Microsoft 365:          | 291     |
| Microsoft 365 Defender: | 291     |



|                                           |     |
|-------------------------------------------|-----|
| Microsoft Azure:                          | 300 |
| Microsoft Endpoint Configuration Manager: | 305 |
| Microsoft Identity Platform:              | 123 |
| Microsoft Intune:                         | 314 |
| Microsoft Teams:                          | 493 |
| MID Server:                               | 384 |
| Miradore:                                 | 321 |
| Missing subnets:                          | 621 |
| MSI package:                              | 31  |
| Multi-factor authentication:              | 100 |
| Multi-homed:                              | 634 |
| Mustache:                                 | 493 |

## N

|                          |         |
|--------------------------|---------|
| NameID:                  | 117     |
| NAT:                     | 692     |
| Nessus:                  | 363     |
| Nessus Professional:     | 360     |
| NetBox:                  | 324     |
| Network bridge report:   | 634     |
| Network bridges:         | 428     |
| Network examples:        | 46      |
| Network Map:             | 149 601 |
| Network paths:           | 637     |
| Network segmentation:    | 479     |
| Newly discovered assets: | 425     |
| Newly discovered groups: | 425     |
| Newly discovered users:  | 425     |
| Nexpose:                 | 338 343 |
| No access:               | 97      |
| NoAuthNoPriv:            | 154     |
| Npcap:                   | 37      |

## O

|                               |         |
|-------------------------------|---------|
| Offline:                      | 450     |
| Offline asset expiration:     | 4       |
| Offline installation:         | 82      |
| Okta:                         | 125     |
| On-premises:                  | 56      |
| Open Access:                  | 455     |
| Operating system:             | 442 451 |
| Organization:                 | 705     |
| Organization API:             | 642     |
| Organization API token:       | 643     |
| Organization Overview Report: | 620     |
| Organization search:          | 592     |
| Organizations:                | 134     |
| OT:                           | 212     |
| Outlier score:                | 454     |

|            |     |
|------------|-----|
| Outliers:  | 428 |
| Ownership: | 468 |

## P

|                       |     |
|-----------------------|-----|
| Packet rate:          | 136 |
| Panther:              | 382 |
| Passive:              | 130 |
| Passive sampling:     | 163 |
| Password rules:       | 19  |
| PCI DSS:              | 479 |
| Planning:             | 3   |
| Ports:                | 137 |
| PostgreSQL:           | 82  |
| Prebuilt searches:    | 597 |
| Prescan modes:        | 139 |
| Primary address:      | 450 |
| Prisma Cloud:         | 335 |
| Privacy passphrase:   | 155 |
| Privacy protocol:     | 155 |
| Private IP addresses: | 621 |
| Probes:               | 137 |
| Profile images:       | 19  |
| Project:              | 52  |
| Project assets:       | 128 |
| Promiscuous mode:     | 163 |
| Protocol gateway:     | 481 |
| Protocols:            | 708 |
| Public key algorithm: | 459 |

## Q

|                 |     |
|-----------------|-----|
| Qualys VMDR:    | 330 |
| Queries:        | 519 |
| Queries:        | 597 |
| Query language: | 597 |
| Query library:  | 597 |

## R

|                   |         |
|-------------------|---------|
| Rapid Response:   | 455     |
| Rapid7 InsightVM: | 338     |
| Rapid7 Nexpose:   | 343     |
| Raspberry Pi:     | 32      |
| Recent assets:    | 128     |
| Recurring scan:   | 472     |
| Red border:       | 622     |
| Reinstall:        | 35      |
| Remote logging:   | 71      |
| Report:           | 625     |
| Reports:          | 8       |
| REST:             | 644 644 |

|                  |             |
|------------------|-------------|
| RFC 1918:        | 146 148 428 |
| RFC1918:         | 53 621      |
| Risk:            | 457         |
| Risk Management: | 455         |
| Risk rank:       | 579         |
| Role:            | 103         |
| Roles:           | 95          |
| Router:          | 691         |
| RPMs:            | 82          |
| RTT:             | 453         |
| Rules:           | 489 489     |
| Rules:           | 492         |
| Rules submenu:   | 492         |

## S

|                         |          |
|-------------------------|----------|
| SAML:                   | 117 121  |
| Sample Queries:         | 426      |
| Sample rate:            | 140      |
| Scan completed:         | 501      |
| Scan data:              | 700      |
| Scan data expiration:   | 4        |
| Scan probe:             | 218      |
| Scan result:            | 703      |
| Scan scope:             | 450      |
| Scan speed:             | 6        |
| Scan speed:             | 692      |
| Scan stats:             | 702      |
| Scan template:          | 150      |
| Scanner:                | 160      |
| Scans:                  | 471      |
| Scheduled scans:        | 6        |
| Scope:                  | 135      |
| Screenshot:             | 34       |
| Screenshots:            | 23 8 655 |
| Search:                 | 442      |
| Search assets:          | 528      |
| Search certificates:    | 565      |
| Search credentials:     | 593      |
| Search explorers:       | 585      |
| Search groups:          | 575      |
| Search queries:         | 594      |
| Search query rule:      | 500      |
| Search reports:         | 584      |
| Search scan templates:  | 577      |
| Search services:        | 546      |
| Search software:        | 553      |
| Search syntax:          | 520      |
| Search users:           | 573      |
| Search vulnerabilities: | 556      |
| Search WiFi networks:   | 570      |

|                                         |               |
|-----------------------------------------|---------------|
| Searching tasks:                        | 580           |
| Secondary addresses:                    | 450           |
| SecurityGate.io:                        | 383           |
| Segmentation:                           | 479           |
| Self-hosted:                            | 56 160        |
| Self-signed:                            | 460           |
| Sendgrid:                               | 489           |
| SentinelOne:                            | 345           |
| Service Graph connector:                | 384           |
| ServiceNow:                             | 384           |
| Services:                               | 442 8 708     |
| Shodan:                                 | 348           |
| Side-by-side:                           | 625           |
| Sign up:                                | 15 19         |
| Signature algorithm:                    | 459           |
| SIP:                                    | 691           |
| Site:                                   | 442 450 705   |
| Site search:                            | 591           |
| Sites:                                  | 134           |
| Slack:                                  | 489 493 518   |
| SNMP:                                   | 154           |
| SNMP credentials:                       | 155           |
| SNMP v2 Communities:                    | 155           |
| Software:                               | 9             |
| Software groups:                        | 553           |
| SPAN:                                   | 163 163       |
| Splunk:                                 | 387 387       |
| SSID:                                   | 570           |
| SSO:                                    | 5 100 117 125 |
| SSO group mapping:                      | 121           |
| Stale asset expiration:                 | 4             |
| Stale integration attribute expiration: | 4             |
| Stale vulnerability expiration:         | 4             |
| Starlark:                               | 244           |
| Stateful firewalls:                     | 134           |
| Statistics:                             | 622           |
| Sub-asset:                              | 481           |
| Subject:                                | 459           |
| Subject Alternative Name:               | 459           |
| Subnet:                                 | 603           |
| Subnet analysis:                        | 622           |
| Subnet sampling:                        | 139           |
| Subnet size:                            | 140           |
| Subnet utilization:                     | 149           |
| Subnet utilization:                     | 427           |
| Subnets:                                | 442 621       |
| Sumo Logic:                             | 388 388 394   |
| Superuser:                              | 96            |
| Suppress:                               | 477           |
| Suppressing:                            | 477           |

|                      |         |
|----------------------|---------|
| Suppression:         | 477     |
| Switch topology:     | 149 632 |
| Switch topology:     | 427 428 |
| Syslog:              | 71      |
| System events:       | 595     |
| System requirements: | 3       |

## T

|                                   |         |
|-----------------------------------|---------|
| Tailscale:                        | 351     |
| Tanium:                           | 352     |
| TAP:                              | 163     |
| Task details:                     | 425     |
| Tasks:                            | 471     |
| Tasks overview page:              | 471     |
| TCP ping:                         | 140     |
| TCP services:                     | 453     |
| Team member:                      | 98      |
| Templates:                        | 492     |
| Tenable Nessus:                   | 355 363 |
| Tenable Nessus Professional:      | 359     |
| Tenable Security Center:          | 355     |
| Tenable Vulnerability Management: | 355     |
| Tenable.io:                       | 355 356 |
| Tenable.sc:                       | 355 364 |
| Thinkst Canary:                   | 410     |
| Time To Live:                     | 137     |
| Tines:                            | 401     |
| TLS certificates:                 | 459     |
| Token:                            | 642     |
| Topology:                         | 427     |
| Total assets seen by task:        | 425     |
| Traceroute:                       | 637     |
| Traffic sampling:                 | 130     |
| Two-factor authentication:        | 100     |
| Type of Service:                  | 138 138 |

## U

|                  |     |
|------------------|-----|
| UAC:             | 21  |
| UDP ping:        | 140 |
| UDP services:    | 453 |
| Unmapped:        | 427 |
| Unmapped assets: | 632 |
| Unmapped MACs:   | 150 |
| Upgrade:         | 35  |
| User directory:  | 449 |
| User groups:     | 110 |
| User search:     | 587 |
| Users:           | 9   |
| Users changed:   | 425 |

|                        |     |
|------------------------|-----|
| Users unchanged:       | 425 |
| Users updated by task: | 425 |
| UUID:                  | 704 |

**V**

|                          |            |
|--------------------------|------------|
| VCenter:                 | 367        |
| Verb:                    | 103        |
| Verifier:                | 38         |
| Viewers:                 | 97         |
| Virtual machine:         | 367        |
| Virtual machines:        | 692        |
| Virtualized system:      | 20         |
| VMware:                  | 20 367 692 |
| VPN gateways:            | 134        |
| Vulnerabilities:         | 9 455      |
| Vulnerability groups:    | 554        |
| Vulnerability instances: | 556        |

**W**

|                    |     |
|--------------------|-----|
| Web screenshots:   | 3   |
| Webhook:           | 489 |
| Webhooks:          | 493 |
| Widget library:    | 432 |
| WiFi:              | 449 |
| Wireless networks: | 9   |
| Wiz:               | 368 |
| World Map:         | 616 |

**X**

|        |     |
|--------|-----|
| X.509: | 459 |
|--------|-----|