

A [in]Segurança dos Sistemas Governamentais Brasileiros: Um Estudo de Caso em Sistemas Web e Redes Abertas

Marcus Botacin¹, André Grégio¹

¹ Universidade Federal do Paraná (UFPR) - {mfbotacin, gregio}@inf.ufpr.br

Abstract. *Whereas the world relies on computer systems for providing public services, there is a lack of academic work that systematically assess the security of government systems. To partially fill this gap, we conducted a security evaluation of publicly available systems from public institutions. We revisited OWASP top-10 and identified multiple vulnerabilities in deployed services by scanning public government networks. Overall, the unprotected services found have inadequate security level, which must be properly discussed and addressed.*

Disclaimer. Nenhum dos sistemas avaliados foi atacado, tampouco houve intenção maliciosa na realização dos testes. O objetivo dos testes realizados foi o de identificar se os referidos sistemas estavam vulneráveis via acesso Web, com base no documento do OWASP top 10. Tais testes foram verificados previamente a fim de garantir que não haveria violação da integridade, disponibilidade e confidencialidade dos dados dos sistemas avaliados. Todas as vulnerabilidades encontradas foram reportadas aos contatos responsáveis por estes sistemas quando da condução destes (finalizados em maio/2019), bem como posteriormente para o CTIR.gov. Informações sobre os *hostnames* e domínios dos sistemas avaliados foram sanitizadas para preservação de suas identidades.

1. Introdução

Um dos principais agentes da vida cotidiana a se beneficiar das plataformas computacionais modernas, como a *web* e a Internet, são os órgãos governamentais, que agora oferecem serviços *online* aos cidadãos através das plataformas de governo eletrônico (e-gov) [Campos and Marques 2007]. Embora eficientes e práticas, a adoção de plataformas de e-gov também traz novos desafios, como a manutenção de níveis de segurança, um requisito fundamental para sistemas que lidam com dados sensíveis, como as informações dos cidadãos. A manutenção dos níveis de segurança requer um nível alto de maturidade das práticas de desenvolvimento e de proteção de sistemas, que devem ser constantemente avaliadas.

Apesar dos riscos envolvidos na operação de sistemas do tipo e-gov já terem sido apontados pela literatura acadêmica internacional [Prandini and Ramilli 2011], ainda são raras as iniciativas voltadas para a avaliação da segurança dos serviços de e-gov no Brasil, uma clara lacuna de desenvolvimento de impacto não desprezível. Em realidade, pode-se observar na prática o impacto de vazamentos de dados, como os que afetaram os milhares de usuários cadastrados no Procon [Folha 2019]. Dados do CTIR.gov mostram números não menos alarmantes sobre a quantidade de ataques recebidos por órgãos governamentais [CTIR.gov 2019].

Desta forma, visando dar início a estudos que possam suprir esta lacuna de conhecimento, este trabalho se propõe a conduzir uma avaliação preliminar da segurança da infraestrutura dos sistemas computacionais que suportam os serviços providos por órgãos governamentais e traçar um panorama das práticas sendo atualmente adotadas. Mais especificamente, focamos na avaliação da segurança dos serviços *web* e na infraestrutura de rede acessíveis via Internet, dois pontos extremamente críticos e reconhecidos pela comunidade como propensos a apresentar vulnerabilidades. Para a avaliação da segurança *web*, realizamos uma busca *web* por serviços indexados que apresentassem vulnerabilidades conhecidas categorizadas pelo projeto OWASP

top10 [OWASP 2017], uma ferramenta padrão para a realização deste tipo de avaliação. Para a avaliação da infraestrutura de rede, realizamos um escaneamento das redes cujos endereços IPs são associados a órgãos governamentais.

Nossos resultados indicam que o nível de proteção provido pelos serviços e redes governamentais está abaixo do adequado. Pudemos identificar a ocorrência das principais vulnerabilidades OWASP em serviços web colocados em produção pelos órgãos governamentais. Fomos capazes até mesmo de recuperar o arquivo de senhas de um dos servidores através de uma vulnerabilidade do tipo *directory traversal*. Mais ainda, o escaneamento das redes indicou que muitos servidores ainda se limitam a aceitar tráfego não-cifrado, tanto para serviços web, quanto para serviços de e-mail e nomes de domínio. Esperamos que os resultados aqui apresentados possam servir de indicadores e guias para a melhoria dos níveis de segurança dos serviços providos pelos órgãos governamentais. É importante destacar que estas falhas não ocorrem apenas em sistemas computacionais governamentais, mas acreditamos que a investigação desses seja essencial devido ao alto impacto de eventuais brechas.

Em resumo, este trabalho apresenta as seguintes contribuições: (i) Uma revisão das principais vulnerabilidades OWASP ilustradas através de exemplos reais encontrados em sites e serviços governamentais; (ii) Uma análise das principais vulnerabilidades de rede encontradas através de *scans* em larga escala das redes públicas governamentais; e (iii) Uma discussão crítica do estado atual da segurança da infraestrutura computacional governamental e possíveis ações de melhoria.

2. Trabalhos Relacionados: Contextualização

Sistemas de governo eletrônico (e-gov) tem apresentado uma grande evolução ao longo do tempo [Campos and Marques 2007], permitindo que governo disponibilizem aos seus cidadãos os mais diversos tipos de serviço. A possibilidade de agendar eventos ou mesmo realizar tarefas sem sair de casa torna este tipo de plataforma muito popular, independente da sua área de abrangência. Atualmente, plataformas de e-gov estão disponíveis até mesmo em regiões rurais africanas [Abid et al. 2013]. Como esperado, a tendência de adesão e migração para plataformas de e-gov também chegou ao Brasil, de modo que o governo federal apresenta seu próprio portal de serviços eletrônicos disponíveis para os cidadãos [da Economia 2019].

Ao mesmo tempo em que traz benefícios, plataformas de e-gov também trazem novos riscos, sobretudo de violações segurança. Eventos como vazamentos de dados podem violar a privacidade e até mesmo colocar em risco milhares ou milhões de cidadãos. Apesar disto, pouco tem se falado da necessidade de se proteger estes tipos de sistemas. Tipicamente, a segurança de sistemas governamentais tem se voltado para questões de espionagem [Verble 2014]. Contudo, neste novo cenário, governos devem também se preocupar contra ataques oriundos de agentes internos além dos usuais agentes externos, e esperamos que possamos salientar esta necessidade através deste trabalho.

Poucos trabalhos encontrados na literatura acadêmica versam sobre a segurança em sistemas e-gov (e.g., veja um bom exemplo em [Prandini and Ramilli 2011]). Dentre estes, nenhum menciona o cenário Brasileiro, sendo esta uma lacuna de desenvolvimento que pretendemos suprir através deste trabalho. No contexto brasileiro, a segurança de alguns serviços públicos que afetam uma grande massa de usuários começou a ser sistematicamente avaliada recentemente. Como resultado significativo, muitas vulnerabilidades e más decisões de projeto foram encontradas em aplicações bancárias [Botacin et al. 2019, da Cruz and Aranha 2016]. Neste trabalho, estendemos estas avaliações para os serviços públicos disponibilizados via Internet.

3. Metodologia: Como agem os atacantes?

A avaliação foi dividida em duas etapas: (i) inicialmente, procedemos ao entendimento das principais vulnerabilidades, suas origens e implicações; (ii) posteriormente, realizamos uma busca, em maior escala, para identificar a ocorrência destas na prática.

Entendendo Vulnerabilidades. Para o entendimento e caracterização das vulnerabilidades, adotamos o *framework* OWASP top10 [OWASP 2017], que identifica as vulnerabilidades mais comuns no desenvolvimento de sistemas web. O *framework* OWASP tem se mostrado efetivo na identificação e mitigação de vulnerabilidades em casos reais, sendo adotado tanto academicamente quanto pela indústria [Thai and Hieu 2019].

Buscando Vulnerabilidades. Para a identificação das vulnerabilidades na prática, buscamos simular o comportamento de um atacante, que buscaria, primariamente, por vulnerabilidades conhecidas. Para tanto, primeiramente realizamos uma busca por *strings* associadas a vulnerabilidades; por exemplo, buscamos por `login.php` para a identificação de formulários de *login* acessíveis sem nenhuma proteção. Este tipo de busca é conhecido como *google dorks* e tem sido frequentemente utilizado por atacantes para a exploração em cenários reais [Catakoglu et al. 2017]. Pode-se encontrar na Internet coleções de *dorks* para a identificação de diferentes vulnerabilidades [ExploitDB 2019]. Para a identificação das vulnerabilidades de rede, realizamos um escaneamento das redes cujos IPs são mapeados para órgãos governamentais através da ferramenta *shodan* [Shodan 2019]. Esta é uma plataforma de indexação e busca de dispositivos e serviços expostos na Internet, utilizada tanto por acadêmicos para a realização de escaneamentos direcionados [Wang et al. 2017], quanto por atacantes para a identificação de seus alvos.

Reportando Vulnerabilidades. Devemos deixar claro que todos os testes conduzidos tiveram fins puramente acadêmicos e que nenhuma vulnerabilidade foi ativamente explorada de modo a comprometer os sistemas envolvidos. Mais ainda, contatamos todos os administradores dos sistemas analisados de modo a reportar as vulnerabilidades identificadas. O contato foi realizado com antecedência em relação a escrita deste artigo, de modo a garantir tempo hábil para que os administradores corrigissem as falhas apontadas. Lamentamos que nem todos os envolvidos tenham nos contatados em busca de maiores informações ou para confirmação das correções.

4. Vulnerabilidades OWASP: Ocorrências na Prática

Nesta seção, discutimos cada uma das principais vulnerabilidades identificadas pelo projeto OWASP top-10 e exemplificamos a ocorrência destas em serviços reais de órgãos governamentais brasileiros expostos à Internet. Destaca-se que múltiplas instâncias dessas vulnerabilidades foram encontradas (por exemplo, o Google apresenta mais de 100 mil e 600 mil resultados para páginas contendo os *scripts* `admin.php` e `login.php` em domínios `.gov.br`), sendo as apresentadas abaixo selecionadas para serem aqui descritas devido ao caráter didático destas.

Ataques de Injeção são vetores populares de infecção de sites e serviços web. Eles derivam da ausência ou falha na implementação de mecanismos de validação de campos de entrada de dados, sejam de forma explícita ou implícita, de modo a permitir que um atacante insira um comando ou trecho de código sob seu controle no fluxo de execução original do serviço atacado. No passado, os ataques de injeção mais populares eram as injeções SQL [Zhang and Zhang 2018], na quais consultas à base de dados eram abusadas para se obter informações privilegiadas (violação de confidencialidade) ou mesmo para executar comandos de administração, o que poderia resultar até mesmo na remoção da base de dados (violação de integridade). Ao longo do tempo, ataques de injeção também se tornaram populares em outras tecnologias web, como em *scripts*.

Ataques de injeção do tipo XSS e CSRF são os tipos mais populares de ataques de injeção de código *script* atualmente, dado a popularidade destas tecnologias em aplicações web modernas. Em particular, a injeção de código Javascript é prevalente dentre os tipos de injeção. Em ataques XSS, o atacante insere, por exemplo, um *script* em um formulário de texto persistente de modo que, quando o site for apresentado para os demais usuários, o *script* inserido entre em execução [Gupta and Gupta 2015]. A ocorrência deste tipo de ataque foi identificada na prática, por exemplo, no site de uma prefeitura. O Código 1 ilustra que, quando da tentativa de acesso ao site

original, o *script* malicioso inserido na página redireciona o usuário para uma página sob controle do atacante (botsqq). Note que o *script* presente na página controlada pelo atacante (x.php) recebe o endereço da página original como parâmetro, permitindo ao atacante contabilizar o número de “vítimas” e redirecioná-las de acordo com a origem, promovendo ataques direcionados.

```
1 Request URL: http://www.xxxxxxxxxxxx.yyy.gov.br/8vLC8QVTzS
2 Status Code: 302 Found (MOVED) location:
3 http://botsqq.com/x.php?www.xxxxxxxxxxxx.yyy.gov.br/8vLC8QVTzS
```

Código 1. Ataque de injeção de script em página web. O script inserido redireciona o usuário da página original para uma página sob controle do atacante.

Além do controle de origem no servidor de destino, descobrimos que o *script* do atacante também realiza o controle de origem na página atacada, de modo a ocultar a infecção. Notamos que o *script* não é ativado quando o site é acessado diretamente, mas apenas quando o visitante é oriundo de outra página. Em termos técnicos, a ativação do *script* só ocorre quando o cabeçalho *referer* do protocolo HTTP está definido, como mostrado no Código 2.

```
1 curl -L 'http://www.xxxxxxxxxxxx
   .yyy.gov.br/8vLC8QVTzS' -H 'Referer:_https://www.google.com.br/'
```

Código 2. Ativação do script malicioso. O script só realiza o direcionamento quando o cabeçalho referer do protocolo HTTP esta presente.

O cabeçalho *referer* é usualmente setado para usuários provenientes de mecanismos de busca, de modo que infecção pode ser observada em buscadores como o Google, como mostrado na Figura 1. Neste caso, o atacante redireciona os usuários para uma página de conteúdo adulto.

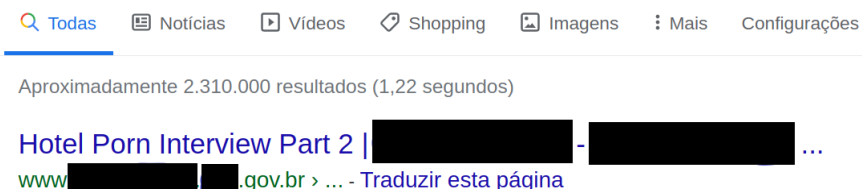


Figura 1. Infecção indexada no Google. Usuários são redirecionados para uma página de conteúdo adulto.

Falhas de mecanismos de autenticação e de controle de acesso são vulnerabilidades graves que permitem que um usuário acesse recursos não autorizados, o que frequentemente implica na escalada de privilégios. Estas vulnerabilidades são frequentemente exploradas através de ataques de injeção de código ou de outras falhas de verificação de entradas. Um tipo de ataque que possibilita violação de princípios de controle de acesso e autenticação é o *path traversal*, no qual falhas na verificação das URLs permitem acesso a arquivos internos do servidor. Na prática, este tipo de ataque foi encontrado no site da prefeitura de uma cidade do Sul do Brasil. O *script* `download.php` não trata adequadamente o parâmetro de entrada `file` e o servidor *web* não está confinado para rodar em um diretório especial, sendo executado, portanto, através da raiz (/) do sistema. Deste modo, podemos atravessar a árvore de diretórios até acessar os arquivos de senhas `shadow` e `passwd` (`https://www.xxxxxxxxxx.yyy.gov.br/data/download.php?file=../[...]/../etc/passwd`). Deste modo, um atacante poderia ter acesso a todos os usuários da máquina e decifrar suas senhas em um ataque de força bruta, ganhando completo acesso ao sistema. Não exibiremos o arquivo `shadow`, mas o arquivo `passwd` é ilustrado no Código 3.

```
1 SSH:/var/empty/sshd:/sbin/nologin
2 dovecot:<range>:Dovecot IMAP
3 root:x:<range>:root:/root:/bin/bash
4 cpaneleximfilter:x:<range>::/var
    /cpanel/userhomes/cpaneleximfilter:/usr/local/cpanel/bin/noshell
5 postfix:x:<range>::/var/spool/postfix:/sbin/nologin
6 postgres:x:<range>:PostgreSQL Server:/var/lib/pgsql:/bin/bash
7 <USERNAME>:x:<range>::/home/<USERNAME>:/bin/bash
```

Código 3. Arquivo de senhas passwd. Um atacante poderia se utilizar destas informações para ganhar acesso completo ao sistema.

É importante ressaltar que a correção do problema passa pela adoção de práticas de confinamento (*jail*) do servidor web e da correta filtragem de parâmetros via *script*. Muitas implementações, contudo, optam por apenas mascarar a vulnerabilidade através de rotinas de ofuscação. O Código 4 ilustra o caso de site de câmara de vereadores de uma cidade no Sudeste do Brasil, no qual o *script* de *download* aceita parâmetros em `base64`, tornando menos evidente o acesso à um diretório. Contudo, a decodificação do mesmo ilustra que o *script* aceita endereços de arquivo como parâmetro.

```
1 base64 -d http://www.zzzzzzzzzzzzzzzzzzzzz.yy.gov.br/_download.php?file=
2 aHR0cDovL2NhbWFYWYNhc2NhbGhvcmljbjY5tZy5nb3YyYnIvdXBsb2Fkcy9kb2N1bWVud
3 GFjYW8vQXRhLzIwMTcvQXRhLTAxNSw5ZGY=
4 http://zzzzzzzzzzzzzzzzzzzz
   .yy.gov.br/uploads/documentacao/Ata/2017/Ata-015.pdf
```

Código 4. Caminho do arquivo codificado em base64. Estratégia de ofuscação não elimina a vulnerabilidade.

Exposição de dados sensíveis é um resultado frequente da exploração de vulnerabilidades de implementação ou da ausência de mecanismos de autenticação e de controle de acesso. Na prática, podemos observar este caso em servidores de dados do tipo FTP expostos à Internet, como no caso de outra prefeitura de cidade do Sul do país (<http://ww2.vvvvvvvv.zz.gov.br:8181/>), como mostrado na Figura 2.



Figura 2. FTP de Prefeitura de cidade no Sul do país. Dados estão acessíveis sem qualquer mecanismo de autenticação ou controle de acesso.

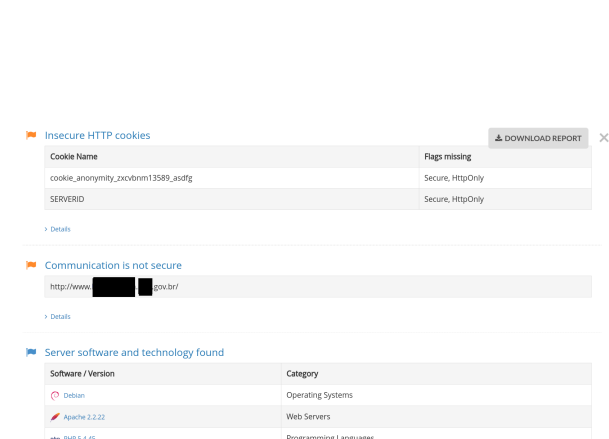


Figura 3. Scanner de Vulnerabilidades Web. Um escaneamento típico revelaria as múltiplas vulnerabilidades conhecidas presentes no serviço.


No caso de órgãos governamentais, é difícil avaliar se os dados foram intencionalmente disponibilizados ou não, pois governos seguem políticas de dados abertos [da União 2017] e de transparência [Transparência 2017]. Desta forma, muitos órgãos disponibilizam seus próprios dados em servidores FTP (e.g., <http://xxxxxxxxx.xxxxxxx.xx.gov.br/ftp/>, www.yyyyyyy.yy.gov.br/arquivos/ftp, ftp.zzzzzzzzzzzzzzzzzzz.zzz.gov.br), o que é uma boa prática quando estes servidores são bem configurados. Contudo, a mistura entre arquivos de sistema, dados de séries históricas e de licitações correntes mostram a ausência de políticas claras para a classificação de informações.

Falhas de configuração de mecanismos de segurança são um agravante frequente dos incidentes de exploração de vulnerabilidade *web*, uma vez que a correta configuração destes poderia ter impedido ou mitigado os efeitos das explorações. Dentre as falhas mais comuns estão a ausência de cabeçalhos que estabeleçam proteções, o uso de configurações padrão não otimizadas, a expiração e invalidação de certificados SSL, entre outros. Na prática, todas essas práticas foram encontradas no site de uma prefeitura de Estado do Sudeste. Uma varredura de vulnerabilidades típica [Pentest-Tools 2019] seria capaz de identificá-las, como mostrado na Figura 3.

De um modo geral, as falhas de configuração em mecanismos de segurança ocorrem por que estas costumam ser o último fator a ser considerado nos processos de desenvolvimento [Pitchford 2018]. Contudo, em muitas vezes, até mesmos as etapas de desenvolvimento apresentam falhas significativas. Não é raro encontrar arquivos de desenvolvimento deixados no servidor após o lançamento da aplicação. Fomos capazes de encontrar até mesmos esquemas de tabelas de banco de dados (<http://www.aaaaaaa.aa.gov.br/wp-content/uploads/database.sql>).

Componentes Vulneráveis. A atualização de componentes de *software* é um requisito básico para a manutenção da segurança de qualquer sistema, pois além de novos recursos e funcionalidades, atualizações implementam a correção de falhas para impedir a exploração de vulnerabilidades. Vulnerabilidades em *software* se popularizam rapidamente via Internet, de modo que atacantes estão sempre aptos a explorar sistemas que apresentem vulnerabilidades conhecidas por não terem sido atualizados. Na prática, contudo, ainda podemos encontrar sistemas defasados expostos na Internet, como mostrado na Figura 4. O site de uma agência de fiscalização de um Governo de Estado do Sudeste, além de vulnerável, apresenta todas as versões dos serviços em execução por não ter removido o arquivo de informações do php (<https://bbbbbbbbbbbbbbbbb.bbbbbbb.bb.gov.br/info.php>). Isto permite que um atacante identifique, com precisão, quais vulnerabilidades estão presentes naquele servidor.

Requisições e Redirecionamentos Não Validados. Serviços e *websites* modernos são construídos de forma modular, de modo que é comum se encontrar nestes diversos pontos de redirecionamento, o que permite a comunicação entre componentes e a obtenção de dados de diferentes origens. O maior desafio deste tipo de construção é garantir, através de processos de validação, a origem legítima dos dados. Caso garantias de validade das requisições não sejam oferecidas, redirecionamentos podem levar a qualquer outro endereço que não o originalmente previsto e dados de diferentes fontes podem ser incorporados ao serviço original. Estas possibilidades são especialmente interessantes à agentes maliciosos, pois permitem redirecionar o usuário para páginas sob controle do atacante e/ou injetar código nos sites vulneráveis. Este tipo de vulnerabilidade é exemplificada, na prática, pelo redirecionamento não tratado no site do portal da transparência de Estado do Centro-Oeste. Como se pode observar na requisição <http://www.cccccccccccccc.cc.gov.br/externo.php?pagina=site.com.br>, o código php responsável pelo tratamento do redirecionamento não valida o parâmetro passado, permitindo que qualquer site especificado na URL seja incorporado ao corpo do site principal, tal como ilustrado na Figura 5.



PHP Version 5.1.4

System	Windows NT SRVWE02 6.1 build 7601
Build Date	May 4 2006 10:30:29
Configure Command	cscript /nologo configure.js "--enable-snapshot-build" "--with-gd=shared"
Server API	CGI/FastCGI
Virtual Directory Support	enabled
Configuration File (php.ini) Path	C:\PHP\php.ini
PHP API	20041225
PHP Extension	20050922
Zend Extension	220051025
Debug Build	no
Thread Safety	enabled
Zend Memory Manager	enabled
IPv6 Support	enabled
Distributed/URL Streams	should be for compression

Figura 4. Agência de fiscalização governamental. Sistemas defasados e vulneráveis são mantidos *online* expostos a ataques.

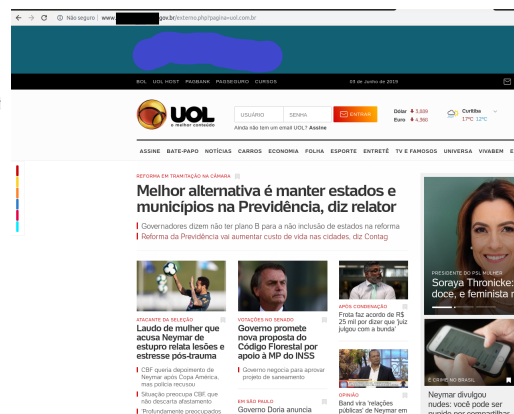


Figura 5. Portal de Transparência de Estado do Centro-Oeste. Parâmetro de redireção na URL não tratado permite embutir qualquer outro site no corpo do site original.

Log e Monitoração Insuficientes A implementação de mecanismos de monitoração é essencial para o estabelecimento de processos seguros, sendo que o uso de *logs* pode se dar de forma proativa ou responsiva. No primeiro caso, pode-se detectar varreduras, ataques de força bruta, e similares, o que permite a identificação do componente sob ataque. No último, pode-se identificar quais sistemas foram comprometidos e/ou que senhas foram vazadas, por exemplo, permitindo a resposta ao incidente. A verificação externa da abrangência e correta implementação dos mecanismos de *log* é extremamente dificultada, devendo, portanto, ser realizada prioritariamente de modo interno ao sistema avaliado. Podemos, contudo, realizar algumas inferências sob o estado atual dos sistemas governamentais de resposta a incidentes como um todo. Apesar de estarmos constantemente realizando varreduras nas redes destes órgãos e seguidas tentativas de conexão a diversos servidores, não recebemos nenhuma notificação de nenhum destes órgãos, o que pode indicar que o nível de monitoração não esteja adequado. Nossa hipótese é reforçada pelo fato de que recebemos notificação da RNP/CSIRT local por sermos origem deste tipo de ação na rede, o que indica que o efetivo monitoramento das redes seria capaz de denunciar a ação de um atacante. Mais alarmante ainda, não recebemos nenhuma notificação mesmo quando acessamos o arquivo de senhas *shadow* de alguns servidores, o o que pode indicar que o acesso à estes arquivos não eram monitorados a despeito da importância destes.

5. Uma Visão Geral dos Serviços de Rede

Para prover uma visão geral das práticas de segurança adotadas pelos órgãos governamentais, estendemos nossa avaliação para incluir o *deploy* e a configuração de serviços de rede, além das aplicações *web*. Realizamos uma varredura dos servidores expostos à Internet através da solução *shodan* e mapeamos os principais serviços disponibilizados e as configurações utilizadas por estes. Devido ao grande número de *hosts* conectados à Internet brasileira, limitamos nossas análises aos 1617 domínios presentes na lista de domínios *.br* do portal de dados abertos [de Dados Abertos 2018]. A Figura 6 sumariza os serviços e protocolos identificados.

De um modo geral, a maioria dos *hosts* expostos à Internet implementa servidores *web*, o que é esperado, dado a necessidade de prover informações a população. Contudo, esperava-se que a maioria destes servidores implementassem mecanismos de comunicação segura `HTTPS`, o que não foi observado na prática. Nota-se que o número de servidores aceitando conexões do tipo `HTTP` é muito superior ao número de servidores aceitando `HTTPS`, de modo que, mesmo se descon-

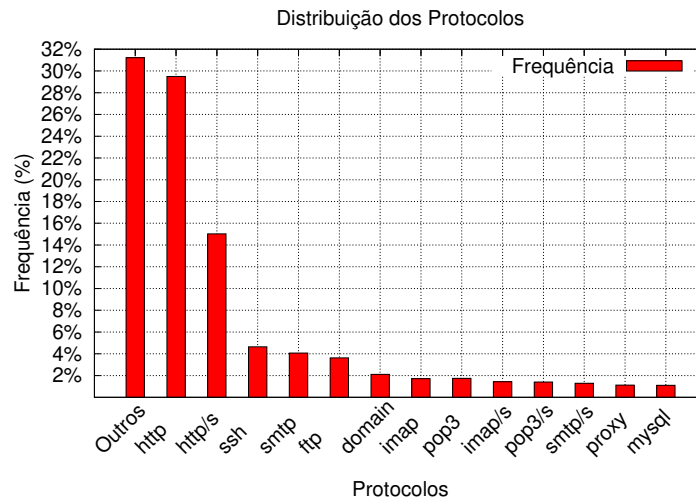


Figura 6. Serviços e protocolos identificados. Muitos servidores ainda estão configurados de forma insegura, permitindo apenas tráfego não-cifrado (HTTP na porta 80) ou o encaminhamento de e-mails de qualquer origem (SMTP na porta 25).

siderarmos os servidores que implementam apenas a redireção de tráfego HTTP para HTTPS na porta 80, ainda temos um número expressivo de servidores que aceitam apenas conexões não criptografadas. Ressaltamos que, de acordo com os padrões atuais [Standard 2018, Google 2018], mesmo informações disponíveis publicamente e do tipo somente-leitura são consideradas sensíveis, pois a simples informação de qual usuário acessou qual tipo de dado pode revelar informações significativas sobre este. Infelizmente, os erros de configuração não se limitam apenas a falta de suporte ao tráfego cifrado, mas estão presentes também quando este suporte é implementado. Dentre os servidores configurados para aceitar conexões criptografadas, 10% tinham os certificados SSL vencidos na data de escaneamento (Outubro/2019).

Além de servidores de páginas *web*, outros serviços também são afetados por má práticas de configuração. Dentre os servidores disponibilizando serviços de e-mail, 50.7% aceitam conexões externas não-autenticadas na porta 25. Desde 2012, recomenda-se o bloqueio da porta 25 para o envio de emails por parte dos clientes para se evitar a propagação de mensagens spam [Antispam.br 2012]. Apesar disso, pudemos nos conectar normalmente a estes servidores. De modo similar, dentre os servidores provendo serviços de nomes de domínios (DNS), 47% respondem a requisições externas, o que pode propiciar ataques do tipo amplificação [cert.gov 2013]. Finalmente, 90.5% dos servidores de acesso remoto via SSH estavam configurados na porta padrão e aceitando autenticação por senha e não por chave pública, sugerindo que estes estariam sendo executados com as configurações padrão.

6. Discussão

Adotar as melhores práticas de desenvolvimento de sistemas é essencial para torná-los mais robusto e menos suscetíveis a falhas e a apresentar vulnerabilidades. Deve-se, por exemplo, verificar e validar todas as entradas providas, uma prática não exaustiva nos sistemas avaliados. Além disso, deve-se utilizar mecanismos de segurança explícitos, como recursos de controle de acesso e autenticação para acesso a arquivos, e não mecanismos de segurança por obscuridade, como a codificação de URLs, como observado em diversos casos.

Adotar as melhores práticas de deployment de sistemas é essencial para não tornar serviços vulneráveis. Deve-se evitar usar configurações padrão, não otimizadas para o caso de uso específico. Deve-se, ainda, remover componentes não utilizados, diminuindo, assim, a superfície de ataque.

Realizar testes de segurança periódicos é fundamental para a identificação tanto de novas

vulnerabilidades oriundas de novas descobertas quando da introdução de falhas de configuração oriundas da atualização de serviços. A execução deste tipo de teste poderia ter identificado as vulnerabilidades apontadas neste trabalho.

Antecipar-se aos novos vetores de ataque é essencial para manter a segurança dos sistemas governamentais a longo prazo. Enquanto os sistemas atuais se mostram frágeis mesmo em relação às ameaças conhecidas, novos ataques vem sendo constantemente desenvolvidos. Atualmente, ataques do tipo XSS, por exemplo, tem evoluído para ataques do tipo *data-only* [Lekies et al. 2017], sendo ainda mais difíceis de serem prevenidos, detectados e eliminados.

Avaliar a maturidade das práticas de segurança nas diferentes esferas governamentais é essencial para um melhor entendimento de quais pontos precisam ser fortalecidos e quais práticas devem ser prioritariamente adotadas. Empiricamente, dado o número de vulnerabilidades encontradas em nosso estudo, acreditamos que as esferas municipais tem serviços mais vulneráveis do que as esferas estaduais e federal. Um hipótese plausível para suportar este fato seria os diferentes patamares de investimento em cada uma destas esferas. A verificação desta percepção e hipótese é deixada como trabalho futuro.

Limitações. Este trabalho se propõe a apresentar um panorama da segurança dos serviços de rede, portanto, particularidades e vulnerabilidades outras que as presentes no *framework* OWASP não são reportadas, ainda que estas possam ter impactos significativos na segurança dos sistemas. Ademais, por este trabalho ser suportado pelos resultados providos pelas ferramentas de escaneamento, a identificação de *hosts* também é limitada pela capacidade destas.

Trabalhos Futuros. Acreditamos que a avaliação contínua dos níveis de segurança é essencial para a melhoria das condições operacionais dos sistemas por prover *feedback* sobre a eficácia das configurações implementadas. Desta forma, para o futuro, planejamos repetir estas análises periodicamente de modo a acompanhar a evolução dos sistemas. Desejavelmente, o escopo das avaliações será ampliado para a identificação de outros erros de configurações e falhas.

7. Conclusão

Neste trabalho, avaliamos a segurança dos serviços e da infraestrutura das redes governamentais expostas na Internet. Revisitamos as principais vulnerabilidades OWASP e identificamos a ocorrência destas em aplicações implementadas por diferentes órgãos, o que indica que muitos destes ainda são incapazes de se proteger mesmo contra ataques e vulnerabilidades conhecidas. Realizamos também um escaneamento das redes e identificamos que grande parte dos servidores ainda se limita a aceitar dados não-criptografados, seja em serviços *web*, seja em serviços de e-mail e nomes de domínio. Diante destas descobertas e da crescente dependência do meio digital, recomendamos que as ações de segurança sejam priorizadas e as medidas tomadas revisadas para a melhoria da segurança dos serviços providos pelos órgãos públicos e, conseqüentemente, dos cidadãos.

Referências

- Abid, M. R., Bahri, S., Haddouti, H., and Gerndt, M. (2013). C-gov: E-gov services on the cloud for african rural communities. ICEGOV '13. ACM.
- Antispam.br (2012). Gerência da porta 25. <https://antispam.br/admin/porta25/definicao/>.
- Botacin, M., Kalysch, A., and Grégio, A. (2019). The internet banking [in]security spiral: Past, present, and future of online banking protection mechanisms based on a brazilian case study. ARES '19. ACM.
- Campos, R. and Marques, C. G. (2007). The evolution and the future of the e-gov. EATIS '07. ACM.
- Catakoglu, O., Balduzzi, M., and Balzarotti, D. (2017). Attacks landscape in the dark side of the web. SAC '17. ACM.
- cert.gov (2013). Dns amplification attacks. <https://www.us-cert.gov/ncas/alerts/TA13-088A>.

- CTIR.gov (2019). Estatísticas resultantes do trabalho de detecção, triagem, análise e resposta a incidentes cibernéticos. <https://emnumeros.ctir.gov.br/>.
- da Cruz, R. J. and Aranha, D. F. (2016). Security Analysis of Brazilian Banking Apps in the Android platform. In *XVI SBSEG*. SBC.
- da Economia, M. (2019). Governo digital. <https://www.governodigital.gov.br/EGD/historico-1/historico>.
- da União, C. G. (2017). Confira os detalhes da política nacional de dados abertos. http://www.governoaberto.cgu.gov.br/noticias/2016/copy_of_disponivel-2a-fase-da-consulta-publica-do-decreto-do-202a200emarco-civil202c-da-internet.
- de Dados Abertos, P. B. (2018). Domínios gov.br. <http://dados.gov.br/dataset/dominios-gov-br>.
- ExploitDB (2019). Google hacking database. <https://www.exploit-db.com/google-hacking-database>.
- Folha (2019). Site do procon de sp deixa informações pessoais vulneráveis. <https://agora.folha.uol.com.br/grana/2019/08/site-do-procon-de-sp-deixa-informacoes-pessoais-vulneraveis.shtml>.
- Google (2018). Por que usar o https? <https://developers.google.com/web/fundamentals/security/encrypt-in-transit/why-https>.
- Gupta, S. and Gupta, B. B. (2015). Php-sensor: A prototype method to discover workflow violation and xss vulnerabilities in php web applications. *CF '15*. ACM.
- Lekies, S., Kotowicz, K., Groß, S., Vela Nava, E. A., and Johns, M. (2017). Code-reuse attacks for the web: Breaking cross-site scripting mitigations via script gadgets. *CCS '17*. ACM.
- OWASP (2017). Top10 web vulnerabilities. https://www.owasp.org/images/7/72/OWASP_Top_10-2017_%28en%29.pdf.pdf.
- Pentest-Tools (2019). Website scanner. <https://pentest-tools.com/website-vulnerability-scanning/website-scanner>.
- Pitchford, M. (2018). The end of the develop-first, test-later approach to software development. <https://www.embedded.com/the-end-of-the-develop-first-test-later-approach-to-software-development/>.
- Prandini, M. and Ramilli, M. (2011). Security considerations about the adoption of web 2.0 technologies in sensitive e-government processes. *ICEGOV '11*. ACM.
- Shodan (2019). The search engine for the internet of things. <https://www.shodan.io/>.
- Standard, T. H.-O. (2018). Why https for everything? <https://https.cio.gov/everything/>.
- Thai, N. D. and Hieu, N. H. (2019). A framework for website security assessment. *ICCCM 2019*. ACM.
- Transparência, P. (2017). O que é e como funciona. <http://www.portaltransparencia.gov.br/sobre/o-que-e-e-como-funciona>.
- Verble, J. (2014). The nsa and edward snowden: Surveillance in the 21st century. *SIGCAS Comput. Soc.*, 44(3):14–20.
- Wang, B., Li, X., de Aguiar, L. P., Menasche, D. S., and Shafiq, Z. (2017). Characterizing and modeling patching practices of industrial control systems. *SIGMETRICS '17 Abstracts*. ACM.
- Zhang, H. and Zhang, X. (2018). Sql injection attack principles and preventive techniques for php site. *CSAE '18*. ACM.