Elliptic Curve Cryptography in Practice

Joppe W. Bos

Joint work with
J. Alex Halderman, Nadia Heninger, Jonathan Moore,
Michael Naehrig, Eric Wustrow

Microsoft Research

Elliptic Curves in Practice – An Incomplete Overview

1933: Hasse, estimate of the number of points on an elliptic curve

$$|\#E(\mathbf{F}_p) - (p+1)| \leq 2\sqrt{p}$$

1985: Schoof, deterministic polynomial time algorithm for counting points on elliptic curves

1985-1987: Lenstra Jr., elliptic curves can be used to factor integers

Miller & Koblitz, elliptic curves can be used to instantiate *public-key cryptography*

2000: Standard for ECC by Certicom

2006: NIST standard for ECDSA

2006: RFC 4492, ECC in Transport Layer Security (TLS)

2009: RFC 5656, ECC in Secure Shell (SSH)

2009: Nakamoto, Bitcoin

OpenSSL Cryptography and SSL/TLS Toolkit

© OpenBSD

bitcoin

Elliptic Curves in Practice – An Incomplete Overview

1933: Hasse, estimate of the number of points on an elliptic curve

$$|\#E(\mathbf{F}_p) - (p+1)| \leq 2\sqrt{p}$$

1985: Schoof, deterministic polynomial time algorithm for counting points on elliptic curves

1985-1987: Lenstra Jr., elliptic curves can be used to factor integers

Miller & Koblitz, elliptic curves can be used to instantiate *public-key cryptography*

2000: Standard for ECC by Certicom

2006: NIST standard for ECDSA

2006: RFC 4492, ECC in Transport Layer Security (TLS)

2009: RFC 5656, ECC in Secure Shell (SSH)

2009: Nakamoto, Bitcoin

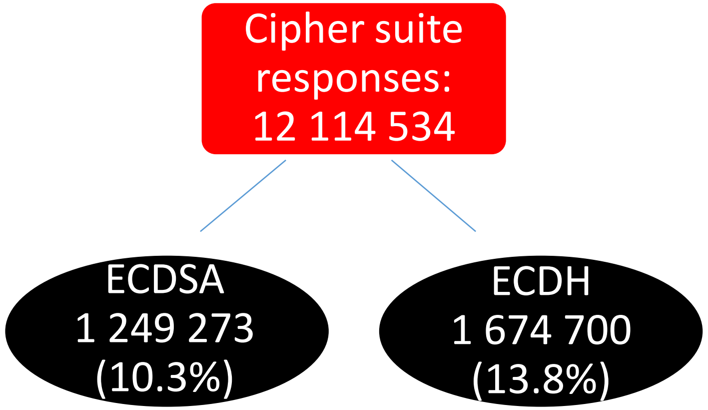
© OpenBSD

Questions

- What is the current state of existing elliptic curve deployments in several different applications?
- Can we find problems that might signal the presence of cryptographic vulnerabilities in ECC?

Secure Shell (SSH)

Scan the complete public IPv4 space
(October 2013) for SSH host keys (port 22)



Secure Shell (SSH)

Scan the complete public IPv4 space (October 2013) for SSH host keys (port 22)

Cipher suite responses:
12 114 534

- ECDSA
  1 249 273
  (10.3%)
- ECDH
  1 674 700
  (13.8%)

- ❖ 1 672 458 (99.8%) supported ecdh-sha2-nistp{256,384,521}
- ❖ 25 (0.001%) supported ecdh-sha2-nistp{521,384,256}

Client offered only EC cipher suites

- 458 689 DSA public key responses
- 29 648 RSA public key responses
- 7 935 empty key responses

<u>Hosts included several kinds of routers and embedded devices</u>

Huawei and Mikrotik.

Secure Shell (SSH)

Scan the complete public IPv4 space (October 2013) for SSH host keys (port 22)

Cipher suite responses:
12 114 534

ECDSA
1 249 273
(10.3%)

ECDH
1 674 700
(13.8%)

- 1 672 458 (99.8%) supported ecdh-sha2-nistp{256,384,521}
- 25 (0.001%) supported ecdh-sha2-nistp{521,384,256}

Client offered only EC cipher suites

- 458 689 DSA public key responses
- 29 648 RSA public key responses
- 7 935 empty key responses

<u>Hosts included several kinds of routers and embedded devices</u>

Huawei and Mikrotik.

Repeated keys (cloud hosting providers)

- shared SSH infrastructure that is accessible via multiple IP address
- mistake during VM deployment

Digital Ocean:
"The SSH host keys for some Ubuntu-based systems could have been duplicated by DigitalOcean's snapshot and creation process."

5614 hosts served the public key from Digital Ocean's setup guide

<u>Default keys present in the hardware or poor entropy on boot</u>

Juniper Web Device Manager, the Juni FemtoAP, and ZTE Wireless Controller

Transport Layer Security (TLS)

Scan the complete public IPv4 space for TLS cipher suits (port 443)

Total hosts: 30.2M

ECDH(E) 2.2M (7.2%)

98% **nistp256**

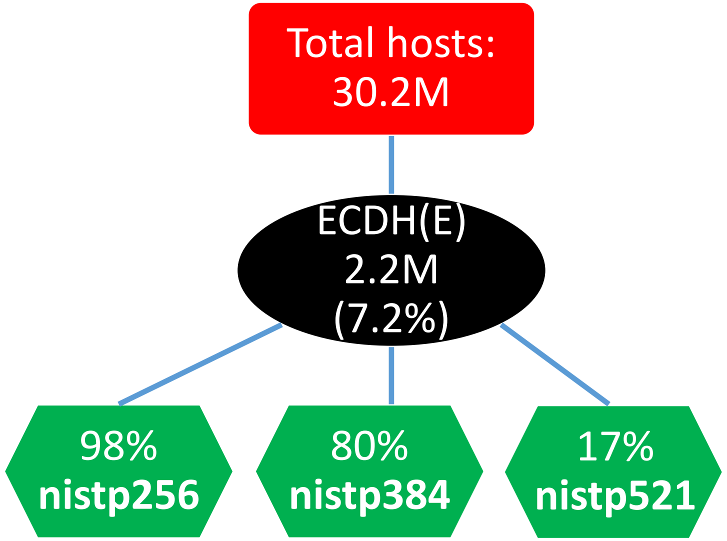
80% **nistp384**

17% **nistp521**

1.7 million hosts supported > 1 curve

Transport Layer Security (TLS)

Scan the complete public IPv4 space for TLS cipher suits (port 443)



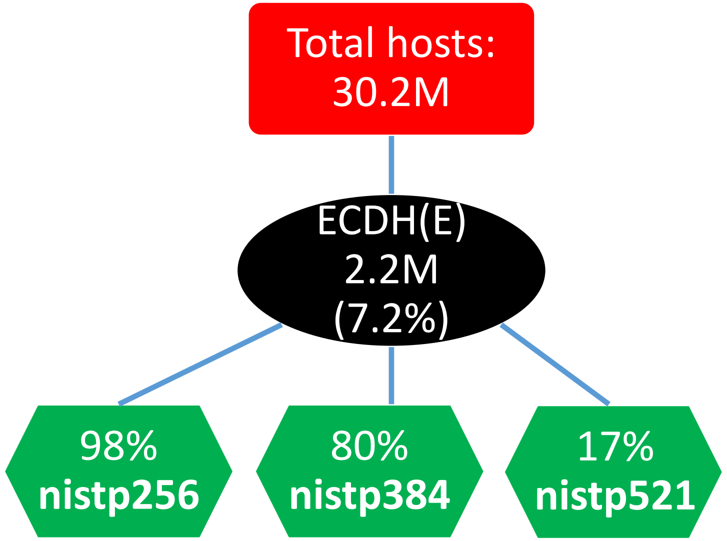
1.7 million hosts supported > 1 curve

- **354 767** hosts secp{256,384,521}r1
- **190** hosts secp{521,384,256}r1

Hosts prefer lower computation and bandwidth costs over increased security

Transport Layer Security (TLS)

Scan the complete public IPv4 space for TLS cipher suits (port 443)



1.7 million hosts supported > 1 curve

- **354 767** hosts secp{256,384,521}r1
- **190** hosts secp{521,384,256}r1

Hosts prefer lower computation and bandwidth costs over increased security

Many duplicated keys are from small set of subnets, most likely nothing wrong: single shared host, but

- *A single key* presented by 2000 hosts
- 1800 Netasq devices presented the same NISTp256 key for ECDHE key exchange

buying this device allows one to decrypt traffic from all other devices

Bitcoin

From asymmetric crypto point of view Bitcoin relies exclusively on ECDSA

August 2013: Bitcoin block chain (#252 450)

- Extracted **22M** transactions (26GB plaintext file)
- **46M** signatures
- **46M** ECDSA keys
  - **15.3M** unique

March 2014: **> 12.4 million** bitcoins in circulation
estimated value: **> 8.4 billion** USD

Bitcoin

From asymmetric crypto point of view Bitcoin relies exclusively on ECDSA

August 2013: Bitcoin block chain (#252 450)

- Extracted **22M** transactions (26GB plaintext file)
- **46M** signatures
- **46M** ECDSA keys
  - **15.3M** unique

March 2014: **> 12.4 million** bitcoins in circulation
estimated value: **> 8.4 billion** USD

Interesting choice:
not NIST P-256 but "special" sec256k1

secp256k1: $p \equiv 1 \pmod 6$, there exists $\zeta \in \mathbf{F}_p$, such that $\zeta^6 = 1$

$$\psi : E \rightarrow E, (x, y) \rightarrow (\zeta x, -y)$$

Fast scalar multiplication $\psi(P) = \lambda P$
for an integer $\lambda^6 \equiv 1 (\mathrm{mod}\ n)$

R. P. Gallant, R. J. Lambert, and S. A. Vanstone. Faster point multiplication on elliptic curves with efficient endomorphisms. CRYPTO 2001

Bitcoin

Elliptic Curve Digital Signatures (d, Q, m)

$k \in \mathbf{F}_n^{\times}$, $kG = (x, y)$, $r = x \bmod n$

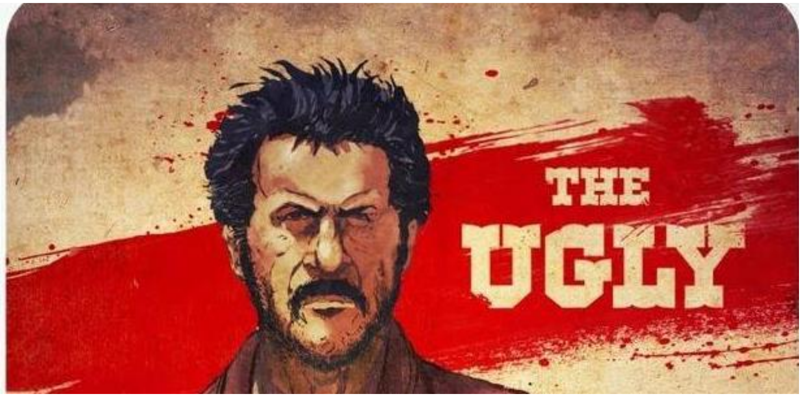
$\mathrm{s} = k^{-1}(\mathrm{Hash}(m) + dr) \bmod n$, Signature: (r, s)

We require $r \neq 0 \neq s$ and k is a per-message secret since

if (r, s_1) and (r, s_2) then $k \equiv (s_2 - s_1)^{-1}(e_1 - e_2) \pmod{n}$

$$d \equiv r^{-1}\big(ks - \mathrm{Hash}(m)\big) \pmod{n}$$

Bitcoin



Elliptic Curve Digital Signatures (d, Q, m)

$k \in \mathbf{F}_n^{\times}, \qquad kG = (x, y), \qquad r = x \bmod n$

$s = k^{-1}(\mathrm{Hash}(m) + dr) \bmod n, \qquad$ Signature: (r, s)

We require $r \neq 0 \neq s$ and k is a per-message secret since

if (r, s_1) and (r, s_2) then $k \equiv (s_2 - s_1)^{-1}(e_1 - e_2) \pmod{n}$

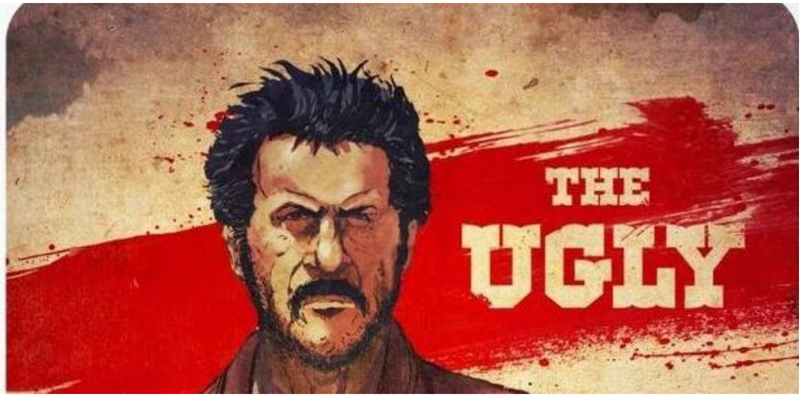
$$d \equiv r^{-1}\big(ks - \mathrm{Hash}(m)\big) \pmod{n}$$

We looked for duplicated nonces in the signatures
158 unique public keys had used the same signature nonce *r* value in more than one signature
→ making it possible to compute these users' private keys
Currently 0.00031217 BTC = 0.21 USD left on these accounts

Address: 1HKywxiL4JziqXrzLKhmB6a74ma6kxbSDj

March to October 2013: 59 BTC ≈ 40076 USD has been stolen from 10 of these addresses

Bitcoin



Elliptic Curve Digital Signatures (d, Q, m)

$k \in \mathbf{F}_n^\times$, $\quad kG = (x, y)$, $\quad r = x \bmod n$

$s = k^{-1}(\mathrm{Hash}(m) + dr) \bmod n$, $\quad$ Signature: (r, s)

We require $r \neq 0 \neq s$ and k is a per-message secret since

if (r, s_1) and (r, s_2) then $k \equiv (s_2 - s_1)^{-1}(e_1 - e_2) \pmod n$

$$d \equiv r^{-1}\big(ks - \mathrm{Hash}(m)\big) \pmod n$$

We looked for duplicated nonces in the signatures
158 unique public keys had used the same signature nonce *r* value in more than one signature
→ making it possible to compute these users' private keys
Currently 0.00031217 BTC = 0.21 USD left on these accounts

Address: 1HKywxiL4JziqXrzLKhmB6a74ma6kxbSDj

March to October 2013: 59 BTC ≈ 40076 USD has been stolen from 10 of these addresses

Possible cause

Poor entropy? At least 3 keys are known to be generated by implementations with Javascript's RNG problem

Conclusions

✓ **ECC is well-deployed and used in practice**

Statistics

Elliptic curves are used in practice

- > 1 out of 10 in SSH
- > 1 out of 14 in TLS
- 100% of all keys in Bitcoin
- However, hosts prefer lower computation and bandwidth costs over increased security

Conclusions

✓ **ECC is well-deployed and used in practice**

Statistics

Elliptic curves are used in practice

- > 1 out of 10 in SSH
- > 1 out of 14 in TLS
- 100% of all keys in Bitcoin
- However, hosts prefer lower computation and bandwidth costs over increased security

✓ **ECC is not immune to insufficient entropy and software bugs**

Cryptographic sanity check

- We found many instances of repeated public SSH and TLS keys
- Bitcoin: there are many signatures sharing ephemeral nonces
  This lead to the theft of a at least 59 BTC

See our paper in Financial Cryptography and Data Security 2014 and on eprint:

http://eprint.iacr.org/2013/734