Randomness of random in Cisco ASA

Ryad BENADJILA (CryptoExperts)*
ryad.benadjila@cryptoexperts.com

Arnaud EBALARD (ANSSSI)
arnaud.ebalard@ssi.gouv.fr

Real World Crypto – March 28th 2023

*Work performed while at ANSSI.
The beginning of the story ...

Work on development projects

▶ X-509 parser [x509-parser]
▶ Elliptic Curve Cryptography library libecc [libecc]

Tests on a >250 millions X.509 certificates set led to ...

| >250 millions X.509 Certs (TLS campaign) |
The beginning of the story ...

Work on development projects

- X-509 parser [x509-parser]
- Elliptic Curve Cryptography library libecc [libecc]

Tests on a >250 millions X.509 certificates set led to ...

<table>
<thead>
<tr>
<th>&gt;250 millions X.509 Certs (TLS campaign)</th>
</tr>
</thead>
<tbody>
<tr>
<td>82k dup. ECDSA nonces</td>
</tr>
<tr>
<td>113k dup. ECDSA keys</td>
</tr>
<tr>
<td>313k Cisco ASA ECDSA self-signed certs</td>
</tr>
<tr>
<td>200k Cisco ASA RSA self-signed certs</td>
</tr>
<tr>
<td>82k dup. ECDSA nonces</td>
</tr>
<tr>
<td>113k dup. ECDSA keys</td>
</tr>
<tr>
<td>12k dup. RSA modulus (≈6%)</td>
</tr>
</tbody>
</table>

ECDSA nonce reuse with same key ⇒ private key compromised
The beginning of the story ...

Work on development projects

- X-509 parser [x509-parser]
- Elliptic Curve Cryptography library libecc [libecc]

Tests on a >250 millions X.509 certificates set led to ...

- >250 millions X.509 Certs (TLS campaign)
- 313k Cisco ASA ECDSA self-signed certs
- 82k dup. ECDSA nonces (≈ 26%)
- 113k dup. ECDSA keys (≈ 36%)
- 12k dup. RSA modulus (≈ 6%)

ECDSA nonce reuse with same key ⇒ private key compromised!

ANSSI – CryptoExperts Randomness of random in Cisco ASA 2/20
The beginning of the story ...

Work on development projects

- X-509 parser [x509-parser]
- Elliptic Curve Cryptography library libecc [libecc]

Tests on a >250 millions X.509 certificates set led to ...

>250 millions X.509 Certs (TLS campaign)

- 313k Cisco ASA ECDSA self-signed certs
- 200k Cisco ASA RSA self-signed certs
- 82k dup. ECDSA nonces (≈ 26%)
- 113k dup. ECDSA keys (≈ 36%)
- 12k dup. RSA modulus (≈ 6%)
The beginning of the story …

Work on development projects

- X-509 parser [x509-parser]
- Elliptic Curve Cryptography library libecc [libecc]

Tests on a >250 millions X.509 certificates set led to …

- >250 millions X.509 Certs (TLS campaign)
- 313k Cisco ASA ECDSA self-signed certs
- 200k Cisco ASA RSA self-signed certs
- 82k dup. ECDSA nonces (≈ 26%)
- 113k dup. ECDSA keys (≈ 36%)
- 12k dup. RSA modulus (≈ 6%)

ECDSA nonce reuse with same key ⇒ private key compromised!
Iterative key recovery

Over 313k X.509 ASA ECDSA self-signed certificates with 216k unique keys

ANSSI – CryptoExperts
Some background on RNG fails ...

History

[CVE-2008-0166] 05/2008: predictable Debian OpenSSL RNG
⇒ Broken SSH/SSL RSA/DSA keys
[PS3EPICFAIL] 12/2010: Epic Fail ECDSA on the Sony PS3
⇒ Nonce reuse, compromise of the firmware signature key
[PSANDQQS] 08/2012: Mining your Ps and Qs (modulus GCD)
⇒ Compromised RSA keys on many embedded devices
[NSBTCFAIL] 01/2013: Recovering BTC private keys
⇒ Nonce reuse, crypto-wallet ECDSA key compromise
Some background on RNG fails …

History

[CVE-2008-0166] 05/2008: predictable Debian OpenSSL RNG ⇒ Broken SSH/SSL RSA/DSA keys

[PS3EPICFAIL] 12/2010: Epic Fail ECDSA on the Sony PS3 ⇒ Nonce reuse, compromission of the firmware signature key

[PSANDQS] 08/2012: Mining your Ps and Qs (modulus GCD) ⇒ Compromised RSA keys on many embedded devices

[NSBTCFAIL] 01/2013: Recovering BTC private keys ⇒ Nonce reuse, crypto-wallet ECDSA key compromission


What about understanding and fixing last one for real? 😊 😁

[CVE-2023-20107] Cisco ASA low entropy keys
Distribution per month, broken / total

Over 313k certs ECDSA ASA

Date (year − month)

# of certs

\cdot 10^4

0

0.5

1

1.5

2

2013-01
2014-01
2015-01
2016-01
2017-01
2018-01
2019-01
2020-01
2021-01
2022-01

CVE-2019-1715

\textbf{broken} (key recovered) certs with \textit{notBefore} this month

certs with \textit{notBefore} in this month

ANSSI – CryptoExperts Randomness of random in Cisco ASA
Cisco Adaptative Security Appliance (ASA)

- Firewall
- VPN (IPsec / TLS)
- IDS/IPS
- ...

Hardware devices: easily available for a decent price!
Virtual appliances ASAv
- Firmware shared with HW
- Difference: no Cavium

Virtual appliances ASAv images available
- Instrumentation details
- Not provided here (lack of time)
- Full length article with this to appear soon!
Cisco Adaptative Security Appliance (ASA)

- Firewall
- VPN (IPsec / TLS)
- IDS/IPS
- ...

Hardware devices: easily available for a decent price!
Cisco Adaptative Security Appliance (ASA)

- Firewall
- VPN (IPsec / TLS)
- IDS/IPS
- ...

Virtual appliances ASAv
- Firmware shared with HW
- Difference: no Cavium

Hardware devices: easily available for a decent price!
Virtual appliances ASAv images available
Cisco Adaptative Security Appliance (ASA)

- Hardware devices: easily available for a decent price!
- Virtual appliances ASAv images available

- Instrumentation details not provided here (lack of time)
- Full length article with this to appear soon
## 5506-X stats

### Black box approach (through scripting)

<table>
<thead>
<tr>
<th>Firmware</th>
<th>RSA modulus</th>
<th>ECDSA r nonce</th>
<th>ECDSA x key</th>
<th>#generated</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.6.2-23</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6.3-20</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6.4-34</td>
<td>•</td>
<td></td>
<td>•</td>
<td>15</td>
</tr>
<tr>
<td>9.6.4-36</td>
<td>•</td>
<td></td>
<td>•</td>
<td>15</td>
</tr>
<tr>
<td>9.6.4-40</td>
<td>•</td>
<td></td>
<td>•</td>
<td>15</td>
</tr>
<tr>
<td>9.6.4-41</td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6.4-42</td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.6.4-45</td>
<td>•</td>
<td></td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>9.7.1-4</td>
<td></td>
<td></td>
<td></td>
<td>160</td>
</tr>
<tr>
<td>9.8.1</td>
<td></td>
<td></td>
<td></td>
<td>60</td>
</tr>
<tr>
<td>9.8.2</td>
<td>•</td>
<td></td>
<td>•</td>
<td>60</td>
</tr>
<tr>
<td>9.8.3</td>
<td>•</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9.8.4-10</td>
<td>•</td>
<td></td>
<td></td>
<td>10</td>
</tr>
<tr>
<td>9.8.4-41</td>
<td>•</td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.9.1</td>
<td>•</td>
<td></td>
<td>•</td>
<td>30</td>
</tr>
<tr>
<td>9.9.2-85</td>
<td>•</td>
<td></td>
<td>•</td>
<td>30</td>
</tr>
<tr>
<td>9.10.1-44</td>
<td>•</td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.12.4</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.12.4-35</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.13.1-12</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.14.3-18</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.15.1-15</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.16.2-14</td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>9.16.2</td>
<td></td>
<td></td>
<td></td>
<td>45</td>
</tr>
</tbody>
</table>

- ● collisions shared between firmware versions
- ▲ = isolated collisions
- ■ = collisions emerging with same certificate time
- Same color = collision values shared across versions
- Empty box = no observable collisions, inconclusive
- Versions **highlighted** are vulnerable and NOT concerned by CVE-2019-1715

ANSSI – CryptoExperts

Randomness of random in Cisco ASA
The RNG players in Cisco ASA

Key generation algorithms

Deterministic RNG engines:
- instantiate
- generate
- reseed

Entropy sources and lifters

- Many primitives per layer
- Many combinations of these! (depending on ASA(v) version)
- Disclaimer: focus on important parts (not exhaustive)
Entropy sources and lifters

Key generation algorithms

Deterministic RNG engines:
- instantiate, generate, reseed

Entropy sources and lifters

Cheap sources:
- Unseeded `rand()`
- `gettimeofday()` (regular and rounded to 10ms)
- `rdtsc`
- ASLR buffers
- Uninitialized buffers
  ...

Better sources:
- `rdrand`, `rdseed`
- Cavium ring oscillators
- Perf counters...

Heavily relies on CPU type (all) and hypervisor (ASA v).
Entropy sources and lifters

Deterministic RNG engines:
- instantiate, generate, reseed

Key generation algorithms

Cheap sources:
- Unseeded rand()
- gettimeofday() (regular and rounded to 10ms)
- rdtsc
- ASLR buffers
- Uninitialized buffers
...

Better sources:
- rdrand, rdseed
- Cavium ring oscillators
- Perf counters
...

Heavily relies on CPU type (all) and hypervisor (ASAv)
Entropy sources and lifters

Deterministic RNG engines:
- instantiate,
generate,
reseed

Key generation algorithms

Entropy sources and lifters:

Cheap sources:
- Unseeded `rand()`
- `gettimeofday()` (regular and rounded to 10ms)
- `rdtsc`
- ASLR buffers
- Unitialized buffers
...

Better sources:
- `rdrand`, `rdseed`
- Cavium ring oscillators
- Perf counters
...

Entropy lifters:
- LFSR (32 bits seeds)
- LCG (32 bits seeds)
- BSAFE lifter (see below)
...

RSA Labs BSAFE CERT-C 2.7.2.0

Heavily relies on CPU type (all) and hypervisor (ASAv)
Deterministic generators

Key generation algorithms

Deterministic RNG engines: instantiate, generate, reseed

Entropy sources and lifters

NIST SP-800 90A DRBGs:
- CTR-DRBG (AES-256)
- Hash-DRBG (SHA-512)

Deterministic RNGs are as good as their entropy sources!
Deterministic generators

Key generation algorithms

Deterministic RNG engines: instantiate, generate, reseed

- NIST SP-800 90A DRBGs:
  - CTR-DRBG (AES-256)
  - Hash-DRBG (SHA-512)

- OpenSSL MD_RAND:
  - SHA-1 based PRG
  - Content of in/out buffer impacts state

Entropy sources and lifters
Deterministic generators

Deterministic RNG engines: instantiate, generate, reseed

NIST SP-800 90A DRBGs:
- CTR-DRBG (AES-256)
- Hash-DRBG (SHA-512)

OpenSSL MD_RAND:
- SHA-1 based PRG
- Content of in/out buffer impacts state

- D stands for Deterministic
- Deterministic RNGs are as good as their entropy sources!
Key generation details

Deterministic RNG engines:
- instantiate, generate, reseed

Entropy sources and lifters

Key generation algorithms

RSA key generation:
- FIPS 186-4 seeded generation of \( P \) and \( Q \)
- Seed of 28 bytes
- Explains why GCD attacks fail!

ECDSA key generation:
- Classical OpenSSL (P-256 curve)
  - 32 bytes reduced modulo the order

ECDSA nonce generation, among:
- BSAFE based (see later)
- Hash-DRBG based (see later)
Key generation details

Entropy sources and lifters

Deterministic RNG engines: instantiate, generate, reseed

Key generation algorithms

RSA key generation:
- FIPS 186-4 seeded generation of $P$ and $Q$
- Seed of 28 bytes
- Explains why GCD attacks fail!

ECDSA key generation:
- Classical OpenSSL (P-256 curve)
- 32 bytes reduced modulo the order
**Key generation details**

**Key generation algorithms**

- **RSA key generation:**
  - FIPS 186-4 seeded generation of $P$ and $Q$
  - Seed of 28 bytes
  - Explains why GCD attacks fail!

- **ECDSA key generation:**
  - Classical OpenSSL (P-256 curve)
  - 32 bytes reduced modulo the order

- **ECDSA nonce generation, among:**
  1. BSAFE based (see later)
  2. Hash-DRBG based (see later)

**Deterministic RNG engines:**

- Instantiate, generate, reseed

**Entropy sources and lifters**
Calls to the DRBG and random generate

Deterministic generator (CTR-DRBG or MD_RAND)

Entropy sources and lifters

Deterministic generator

generate

Hash-DRBG

instantiate

16 → 3 × 16 → 3 × 16 → 3 × 16 → 3 × 16 → 28 → 32 → 8

BSAFE part 1

RSA seed

RSA key

(FIPS 186-4 $P$, $Q$)

ECDSA key

BSAFE part 2

Generate ECDSA nonce

Variant #1 for ECDSA nonce

Variant #2 for ECDSA nonce
Calls to the DRBG and random generate

Deterministic generator (CTR-DRBG or MD_RAND)

Entropy sources and lifters

Hash-DRBG

16 → 3 × 16 → 3 × 16 → 3 × 16 → 3 × 16 → 28 → 32 → 8

RSA key

(FIPS 186-4 P, Q)

RSA seed

Variant #1 for ECDSA nonce

Variant #2 for ECDSA nonce
Calls to the DRBG and random generate

Deterministic generator (CTR-DRBG or MD_RAND)

Entropy sources and lifters

Deterministic generator

generate

 Satoshi

16

3 x 16

3 x 16

3 x 16

3 x 16

28

32

8

RSA key

(FIPS 186-4 P, Q)

RSA seed

ECDSA key

ANSSI – CryptoExperts

Randomness of random in Cisco ASA
Calls to the DRBG and random generate

Deterministic generator (CTR-DRBG or MD_RAND) instantiate

Entropy sources and lifters

Deterministic generator generate

BSAFE part 1

RSA key (FIPS 186-4 $P, Q$)

RSA seed

BSAFE part 2

ECDSA key

Generate ECDSA nonce

Variant #1 for ECDSA nonce
Calls to the DRBG and random generate

Deterministic generator (CTR-DRBG or MD_RAND) instantiate

Entropy sources and lifters

Deterministic generator generate

Deterministic generator generate

Entropy sources and lifters

Deterministic generator generate

Hash-DRBG instantiate

Variant #2 for ECDSA nonce

BSAFE part 1

RSA seed

RSA key (FIPS 186-4 P, Q)

ECDSA key

Generate ECDSA nonce

BSAFE part 2

Generate ECDSA nonce

Variant #1 for ECDSA nonce

ANSSI – CryptoExperts

Randomness of random in Cisco ASA

12/20
BSAFE lifter for ECDSA nonce

BSAFE part 1 (16 bytes)

BSAFE part 2 (8 bytes)

Used as next SHA-1 IV

SHA-1(-)

SHA-1(IV, -)

SHA-1(− ∥ 00...00)

SHA-1(− ∥ 00...00)

20 bytes

20 bytes

mod 0xb20d...e5 (20 bytes NIST prime)

mod 0xb20d...e5 (20 bytes NIST prime)

20 bytes

20 bytes

Nonce (32 bytes)
BSAFE lifter for ECDSA nonce

Cumbersome but deterministic derivation.
At most 24 bytes of entropy when drawing 32 bytes nonce.
Overview of instantiated mechanisms

**Used mechanisms**
- CTR-DRBG used for RSA seed, ECDSA key
- ECDSA nonce using BSAFE with seeds from CTR-DRBG

**CTR-DRBG Instantiate**
- DRBG Personalization string:
  - Fixed "CiscoSSL DRBG60"
  - time from boot rounded to 10ms
- Entropy/nonce:
  - 40/20 bytes from MD_RAND ...
  - ... seeded by LFSR ...
  - ... seeded by 32 bits RDTSC.

**CTR-DRBG Generate calls**
- Addin: counter + time from boot rounded to 10ms
Key aspects of a tricky keygenning

Estimated complexity

- $2^{32}$ possible LFSR seeds
- $\approx 2^{13}$ possible tuples for the 15 rounded time values

⇒ Exhaustive search for $\approx 2^{45}$ (w/ heavy DRBG calls)

Meet in the middle solution

- Patch the binary with a known fixed seed, do some stats on the timings as independent variables (valid approach)
- Take the most probable paths to reduce complexity, generate enough target certs and validate approach
Pros: complexity reduced to $\approx 2^{13}$ for stats gathering
ASA® v9.10.1.44

Timing statistics using patched binary (fixed seed) + envelope reduction

- **Pros:** complexity reduced to $\approx 2^{37.5}$ for validation PoC on unpatched binary by reusing these envelope stats
- **Cons:** only 1.7% of possible certs remains accessible
ASAv v9.10.1.44
Timing statistics using patched binary (fixed seed)

Pros: complexity reduced to $\approx 2^{37.5}$ for validation PoC on unpatched binary by reusing these envelope stats

Cons: only 1.7% of possible certs remains accessible

Validation of the 1.7% breaking rate
8 days on a “basic” 16 cores server
## ASAv firmware analysis: overview of results

<table>
<thead>
<tr>
<th>Firmware</th>
<th>RSA modulus</th>
<th>ECDSA nonce</th>
<th>ECDSA key</th>
<th>Comment</th>
<th>Keygen time complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASAv9.6.4-36</td>
<td></td>
<td></td>
<td></td>
<td>HASH-DRBG seeded by LFSR seeded by 32 bits rdtsc, used for nonce. CTR-DRBG is seeded by MD_RAND, itself seeded by HASH-DRBG itself seeded by a LFSR, itself seeded by rdtsc rounded to 32 bits</td>
<td>$2^{32}$ (nonce)</td>
</tr>
<tr>
<td>ASAv9.8.1</td>
<td></td>
<td></td>
<td></td>
<td>CTR-DRBG “saved” by addin with true gettimeofday(), HASH-DRBG seeded by a LFSR itself seeded by rdtsc rounded to 32 bits</td>
<td>$2^{32}$ (nonce)</td>
</tr>
<tr>
<td>ASAv9.8.2</td>
<td></td>
<td></td>
<td></td>
<td>MD_RAND seeded by rand(), ASLR in input buffers for MD_RAND (nonce), BSAFE seeded by MD_RAND</td>
<td>$\approx 2^{33}$</td>
</tr>
<tr>
<td>ASAv9.8.3</td>
<td></td>
<td></td>
<td></td>
<td>CTR-DRBG seeded by rand() BSAFE seeded by CTR_DRBG</td>
<td>$\approx 2^{16}$</td>
</tr>
<tr>
<td>ASAv9.9.1</td>
<td></td>
<td></td>
<td></td>
<td>MD_RAND seeded by rand(), ASLR in input buffers for MD_RAND (nonce), BSAFE seeded by MD_RAND</td>
<td>$\approx 2^{33}$</td>
</tr>
</tbody>
</table>
| ASAv9.10.1-44 |             |             |           | CTR-DRBG seeded by MD_RAND seeded by LFSR seeded by 32 bits rdtsc. Bad gettimeofday is also used. | Full: $\approx 2^{45}$ PoC: $\approx 2^{37.5}$

**Legend:**
- **Fully broken with a PoC keygen**
- **Broken with a PoC keygen with higher time complexity**
- **Fragile** entropy sources, harder to exploit (but seems feasible)
- **Broken as a side effect of nonce breaking**

Versions highlighted are vulnerable and NOT concerned by previous CVE-2019-1715
Conclusion

What we learned already knew.

- Fail instead of fallback to a bad entropy source
- Consider worst code path, remove if unacceptable/unsure
- Mix multiple sources instead of using a single one
- DRBG specific
  - DRBG security depends on instantiate() source
  - Poor addins for DRBG generate() calls is risky
  - Reseeding often is a requirement [DRBG-ANALYSIS]

Final thoughts

- Good looking keys, etc $\Rightarrow$ good random
- Good DRBG/PRNG $\Rightarrow$ good random
- Full 50 pages article to come for SSTIC 2023 in June


Greg Zaverucha and Dan Shumow “Are Certificate Thumbprints Unique?”,

Cisco Adaptive Security Appliance Software and Firepower Threat Defense Software Low-Entropy Keys Vulnerability,
https://sec.cloudapps.cisco.com/security/center/content/CiscoSecurityAdvisory/cisco-sa-asa5500x-entropy-6v9bHVYP, March 2023