The Return of Coppersmith's Attack: Practical Factorization of Widely Used RSA Moduli

ACM CCS'17, Dallas, TX, USA

Matus Nemec[1,2] Marek Sys[1] Petr Svenda[1] Dusan Klinec[3,1] Vashek Matyas[1]

[1] Masaryk University
Brno, Czech Republic

[2] Ca' Foscari University
Venice, Italy

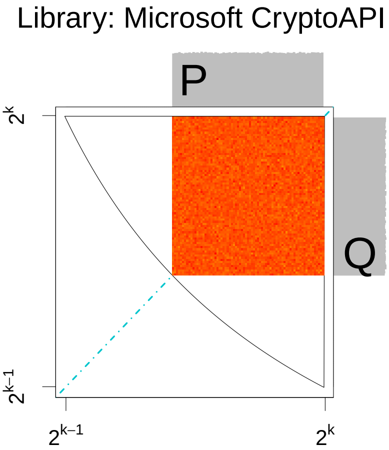
[3] Enigma Bridge
Cambridge, UK

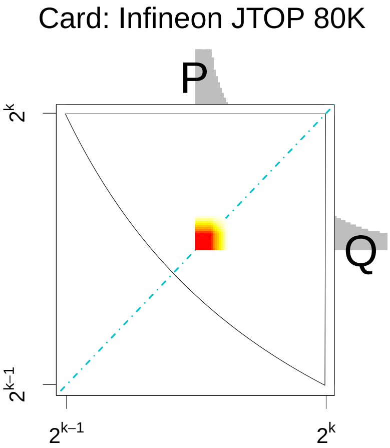
Contributions

- **Structure of RSA primes** in library of Infineon Technologies
- Application of **Coppersmith's factorization method**
- Analysis of **impacted domains**, including **eID, TPM, tokens** and other NIST FIPS 140-2 and CC EAL 5+ **certified devices**
- Lessons learned and mitigation

The Million-Key Question: Investigating the Origins of RSA Public Keys

USENIX Security 2016

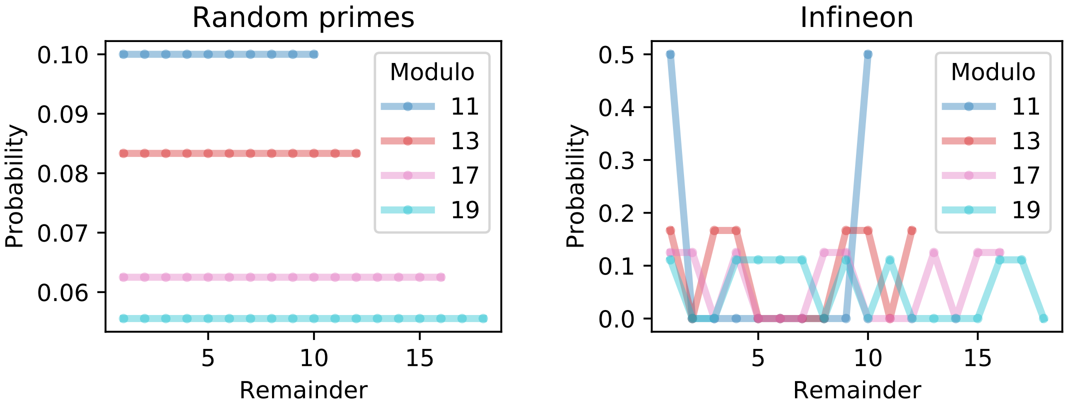




Motivation

The Million-Key Question: Investigating the Origins of RSA Public Keys
USENIX Security 2016

Distribution of RSA keys modulo small primes:



Vulnerable RSA keys

$$N = p * q$$

$$p_{ideal} = \text{random prime}$$

$$p_{Infineon} = (k * M + 65537^{a} \bmod M);\ \ a, k \in \mathbb{Z}$$

$$M = 2 * 3 * 5 * 7 * \cdots * P_n$$

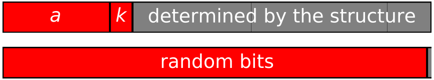
Consequences of the structure:

1. Fingerprint
2. Entropy loss
3. Factorization is possible

Entropy in a prime

Infineon:

Random:



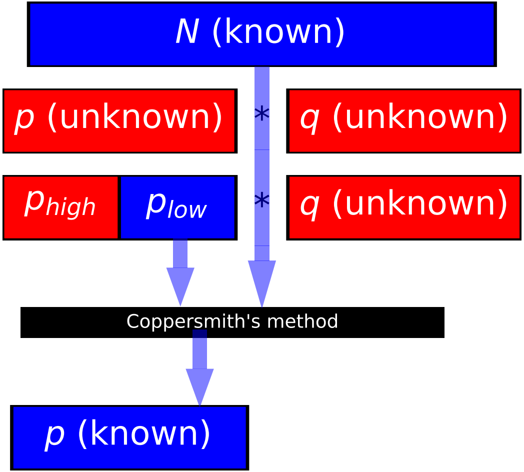
Infineon: *a* | *k* | determined by the structure

Random: random bits

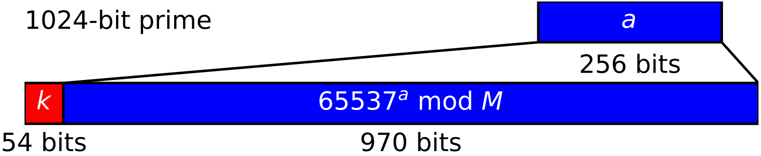
Coppersmith's method as a black-box tool

Coppersmith: Partial knowledge of private key ⇒ full private key

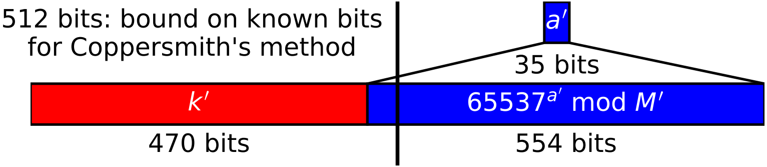
1. Modulus
2. Unknown factors
3. Partial knowledge of prime p
4. Apply Coppersmith's attack



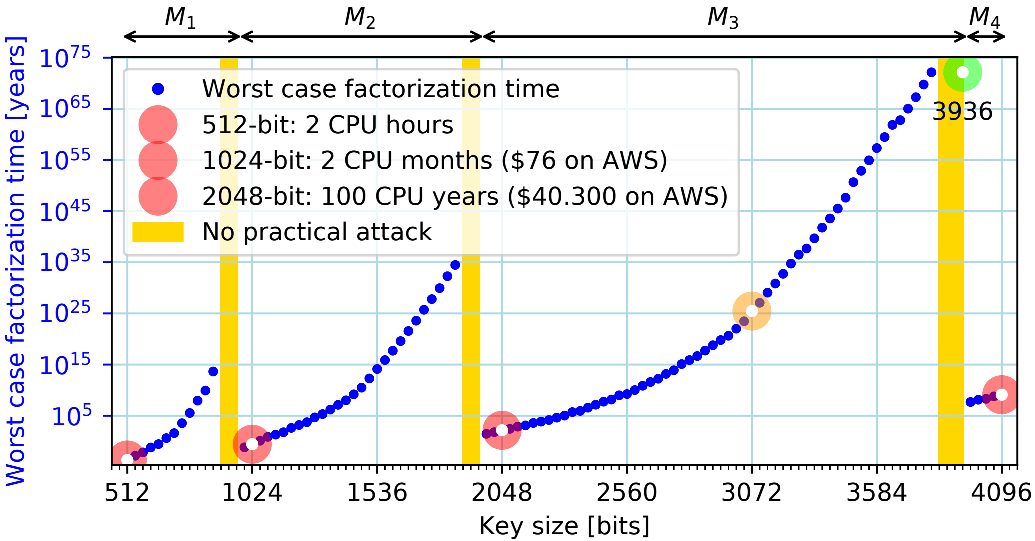
- $p = \mathbf{k} * M + 65537^{\mathbf{a}} \bmod M$
- Guess **a** and compute **k** using Coppersmith's method



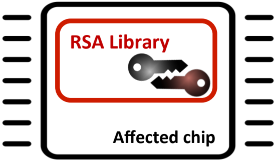
- **a** is still too large – find a smaller M' (divisor of M)



Attack complexity



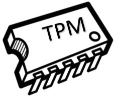
Impacts



Identity documents (eID, eHealth cards)

Authentication tokens

Programmable smartcards



Trusted Platform Modules (Data encryption, Platform integrity)

Message protection (S-MIME, PGP)

Software signing

Secure browsing* (TLS/HTTPS)

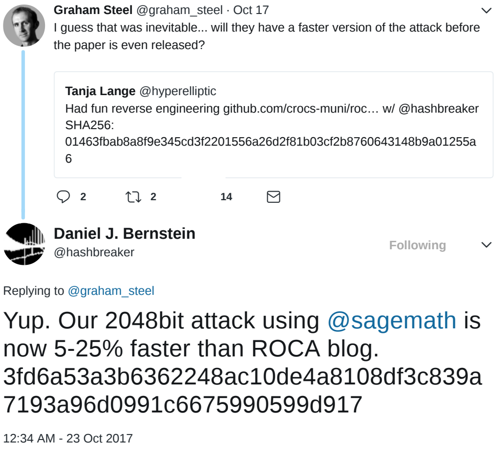
*only a small number of vulnerable keys found

Mitigations

- Test public keys for fingerprint at roca.crocs.fi.muni.cz
- Revoke certificates of weak keys (services become unavailable)
- Change algorithm, e.g. ECC (must update infrastructure)
- Generate new, secure keys:
  - Firmware update (uncommon), replace the device (costly)
  - Import a secure keypair (requires trusted environment)
- **Temporarily** switch to less affected key lengths (e.g., 3936-bit)
  - Significantly reduced security level, attack may improve
- Additional risk management when a vulnerable key is detected

Attack timeline

- End of Jan: Proof of Concept attack
- Feb 1st: Infineon notified
- Oct 10th: Microsoft Patch Tuesday
- Oct 16th: Public disclosure
- Oct 23rd: Tanja Lange & Daniel J. Bernstein announced a faster attack
- Vulnerable devices from 2007 found
- Oct 30th: Full paper published



Graham Steel @graham_steel · Oct 17

I guess that was inevitable... will they have a faster version of the attack before the paper is even released?

Tanja Lange @hyperelliptic

Had fun reverse engineering github.com/crocs-muni/roc… w/ @hashbreaker
SHA256:
01463fbab8a8f9e345cd3f2201556a26d2f81b03cf2b8760643148b9a01255a
6

Daniel J. Bernstein
@hashbreaker

Replying to @graham_steel

Yup. Our 2048bit attack using @sagemath is now 5-25% faster than ROCA blog.
3fd6a53a3b6362248ac10de4a8108df3c839a
7193a96d0991c6675990599d917

12:34 AM - 23 Oct 2017

Conclusions

- Optimizations may weaken security
- Secret design ⇒ delayed discovery of flaws ⇒ increased impacts
- Reconsider the certification process
- Prevent a single point of failure
  - Secure multi-party computation
  - Collaborative RSA

Thank you for your attention