Snake Oil Crypto:

How I stopped to worry and started to love crypto

Team CIRCL
https://www.d4-project.org/

2019/11/27

D4 project

Jean-Louis Huynen

Outline

- Cryptography 101,
- Cryptography and Network captures,
- D4 passiveSSL Collection,
- Leveraging OpenPGP metedata,
- Checking for weak crypto.

Cryptography 101

- **Plaintext** P: Text in clear,
- **Encryption** E: Process of disguising the plaintext to hide its content,
- **Ciphertext** C: Result of the Encryption process,
- **Decryption** D: Process of reverting encryption, transforming C into P,
- **Encryption Key** EK: Key to encrypt P into C,
- **Decryption Key** DK: Key to decrypt C into P,
- **Cryptanalysis**: Analysis of C to recover P without knowing K.

Cryptography Services

- **Confidentiality** : Ensure the secrecy of the message except for the **intended** recipient,
- **Authentication** : Proving a party's identity,
- **Integrity** : Verifying that data transmitted were not altered,
- **Non-repudiation** : Proving that the sender sent a given message.

Type of Encryption Applications

- **In-transit encryption**: protects data while it is transferred from one machine to another,
- **At-rest encryption**: protects data stored on one machine.

Kerckhoffs's Principle

> *It [cipher] should not require secrecy, and it should not be a problem if it falls into enemy hands.*

There is no security in obscurity.

Attackers model I

Black Box - Attackers may only see inputs / outputs:

- **Ciphertext-Only Attackers (COA) :** see only the ciphertext,
- **Known-Plaintext Attackers (KPA):** see ciphertext and plaintext,
- **Chosen-Plaintext Attacker (CPA):** encrypt plaintext, and see ciphertext,
- **Chosen-Ciphertext Attakers (CCA):** encrypt plaintext, decrypt ciphertext.

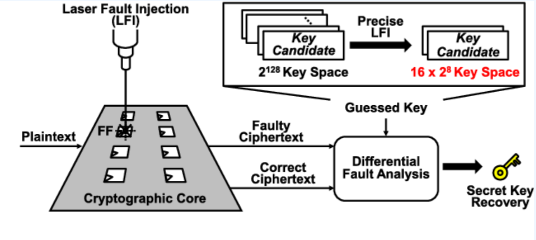
Attackers model II

Grey Box - Attackers see cipher's implementation:

- **Side-Channel Attacks:** study the behavior of the implementation (eg. tpm-fail[24420]),

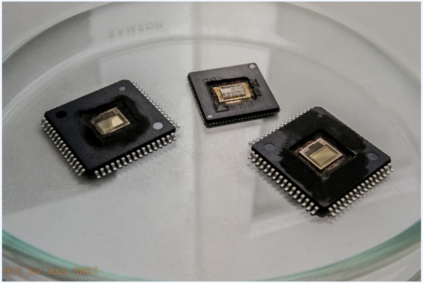
> We discovered timing leakage on Intel firmware-based TPM (fTPM) as well as in STMicroelectronics' TPM chip. Both exhibit secret-dependent execution times during cryptographic signature generation. While the key should remain safely inside the TPM hardware, we show how this information allows an attacker to recover 256-bit private keys from digital signature schemes based on elliptic curves.

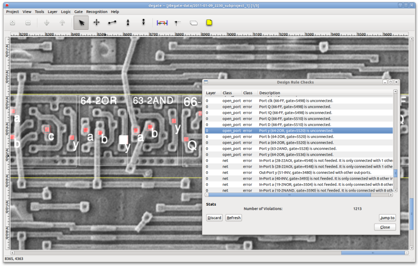
- **Invasive Attacks:**
  - injecting faults[MFS+18],



Attackers model IV

- decapping chips[1], reverse engineering[2,3], etc.





[1] https://siliconpron.org/wiki/doku.php?id=decap:start
[2] http://siliconzoo.org
[3] http://degate.org

Cryptography and Network captures

D4 passiveSSL Collection

Leveraging OpenPGP metedata

Checking for weak crypto

Snake Oil Crypto[4] - Problem Statement

IoT devices **are often the weakest devices** on a network:

- Usually the result of cheap engineering,
- sloppy patching cycles,
- sometimes forgotten–not monitored,
- few hardening features enabled.

We feel a bit safer when they use TLS, but should we?

[4] `https://github.com/d4-project/snake-oil-crypto`

Keep a log of links between:

- x509 certificates,
- ports,
- IP address,
- client (ja3),
- server (ja3s),

  "JA3 is a method for creating SSL/TLS client fingerprints that should be easy to produce on any platform and can be easily shared for threat intelligence."[5]

Pivot on additional data points during Incident Response

[5] https://github.com/salesforce/ja3

Snake Oil Crypto - Objectives

Collect and **store** x509 certificates and TLS sessions:

- Public keys type and size,
- moduli and public exponents,
- curves parameters.

Detect anti patterns in crypto:

- Moduli that share one prime factor,
- Moduli that share both prime factors, or private exponents,
- Small factors,
- Nonces reuse / common preffix or suffix, etc.

Focus on low hanging fruits that appeal to attackers

Researchers have shown that several devices generated their keypairs at boot time without enough entropy[6]:

```
prng.seed(seed)
p = prng.generate_random_prime()
// prng.add_entropy()
q = prng.generate_random_prime()
n = p*q
```

Given n=pq and n' = pq' it is trivial to recover the shared p by computing their **Greatest Common Divisor (GCD)**, and therefore **both private keys**[7].

[6] Bernstein, Heninger, and Lange: http://facthacks.cr.yp.to/
[7] http://www.loyalty.org/~schoen/rsa/

In Snake-Oil-Crypto we compute GCD[8] between:

- between certificates having the same issuer,
- between certificates having the same subject,
- on keys collected from various sources (PassiveSSL, Certificate Transparency, shodan, censys, etc.),

“Check all the keys that we know of for vendor X”

[8] using Bernstein’s Batch GCD algorithm

Snake Oil Crypto - MISP feed

2019-11-08	**Name:** crypto-material **References:** 0 **Referenced by:** 6 uses Object 13800 (network: x509) uses Object 13801 (network: x509) uses Object 13802 (network: x509) uses Object 13803 (network: x509) uses Object 13804 (network: x509) uses Object 13805 (network: x509)		
2019-11-08	Other	**p:** text	12732045980491482532629620809854872609730718866846479950748763992511013869872655864815736531245765416842653133761646084269424192867704218331356123978614869
2019-11-08	Other	**q:** text	None
2019-11-08	Other	**rsa-modulus-size:** text	1024
2019-11-08	Other	**type:** text	RSA

Snake Oil Crypto - MISP feed

The MISP feed:

- **Allows** for checking automatic checking by an IDS on hashed values,
- **contains** thousands on broken keys from a dozen of vendors,
- **will be accessible upon request (info@circl.lu).**

In the future:

- **Automatic** the vendor checks by performing TF-IDF on x509's subjects,
- **automatic** vendors notification.

- ✓ sensor-d4-tls-fingerprinting [9]: **Extracts** and **fingerprints** certificates, and **computes** TLSH fuzzy hash.
- ✓ analyzer-d4-passivessl [10]: **Stores** Certificates / PK details in a PostgreSQL DB.
- snake-oil-crypto [11]: **Performs** crypto checks, push results in MISP for notification
- lookup-d4-passivessl [12]: **Exposes** the DB through a public REST API.

[9] github.com/D4-project/sensor-d4-tls-fingerprinting
[10] github.com/D4-project/analyzer-d4-passivessl
[11] github.com/D4-project/snake-oil-crypto
[12] github.com/D4-project/lookup-d4-passivessl

Get in touch if you want to join/support the project, host a passive SSL sensor or contribute

- Collaboration can include research partnership, sharing of collected streams or improving the software.
- Contact: info@circl.lu
- https://github.com/D4-Project - https://twitter.com/d4_project

References I

- *Tpm-fail: TPM meets timing and lattice attacks*, 29th USENIX Security Symposium (USENIX Security 20) (Boston, MA), USENIX Association, August 2020.
- Ross J. Anderson, *Security engineering: A guide to building dependable distributed systems*, 2 ed., Wiley Publishing, 2008.
- Jean-Philippe Aumasson, *Serious cryptography: A practical introduction to modern encryption*, no starch press, 2017.
- Dieter Gollmann, *Computer security (3. ed.)*, Wiley, 2011.
- Kohei Matsuda, Tatsuya Fujii, Natsu Shoji, Takeshi Sugawara, Kazuo Sakiyama, Yu-Ichi Hayashi, Makoto Nagata, and Noriyuki Miura, *A 286 f2/cell distributed bulk-current sensor and secure flush code eraser against laser fault injection attack on cryptographic processor*, IEEE Journal of Solid-State Circuits **53** (2018), no. 11, 3174–3182.

References II

Alfred J. Menezes, Scott A. Vanstone, and Paul C. Van Oorschot, *Handbook of applied cryptography*, 1st ed., CRC Press, Inc., Boca Raton, FL, USA, 1996.