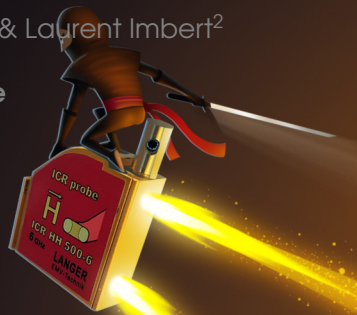
A Side Journey to Titan

Revealing and Breaking NXP's P5x ECDSA Implementation on the Way

Thomas Roche[1] & Victor Lomne[1] & Camille Mutschler[1,2] & Laurent Imbert[2]
[1]**NinjaLab, Montpellier, France**
[2]**LIRMM, Univ. Montpellier, CNRS, France**



October 15, 2021

Journees SIF

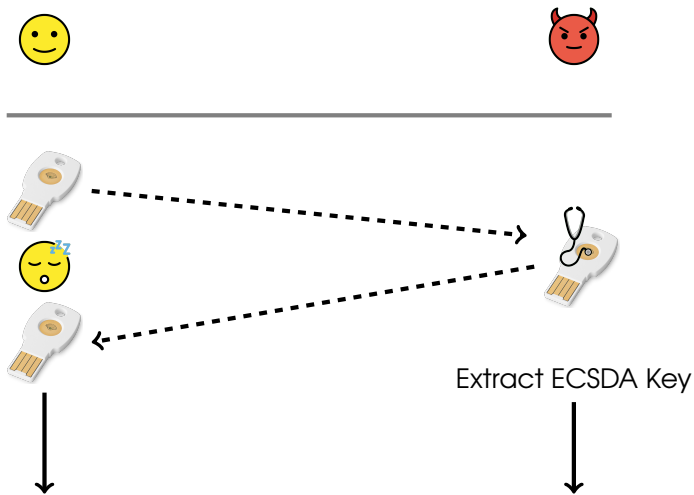
Product Description

- ***Google Titan Security Key***: hardware FIDO U2F token
- Hardware token to be used as 2FA for your Google account
  and many other services supporting FIDO U2F protocol
- 3 versions:
  - Left: micro-USB, NFC and BLE interfaces
  - Middle: USB type A and NFC interfaces
  - Right: USB type C interface

FIDO U2F Protocol

- FIDO U2F: open standard for two-factor authentication
  - Hosted by FIDO alliance (historically developed by Google, Yubico and NXP)
- FIDO U2F protocol works in two steps - for each account:
  - *Registration* $\rightarrow$ ECDSA key pair generation
  - *Authentication* $\rightarrow$ ECDSA signature

Side-Channel Attack Scenario



Side-Channel Attack Scenario

Trusted hardware

Titan Security Keys are designed to make the critical cryptographic operations performed by the security key strongly resistant to compromise during the entire device lifecycle, from manufacturing through actual use.

The firmware performing the cryptographic operations has been engineered by Google with security in mind. This firmware is sealed permanently into a secure element hardware chip at production time in the chip production factory. The secure element hardware chip that we use is designed to resist physical attacks aimed at extracting firmware and secret key material.

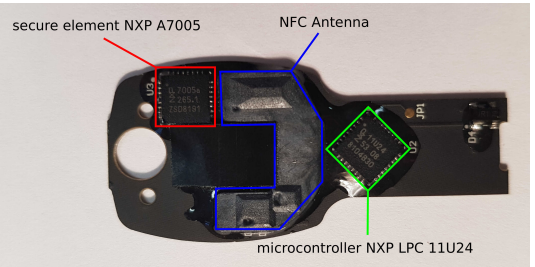
These permanently-sealed secure element hardware chips are then delivered to the manufacturing line which makes the physical security key device. Thus, the trust in Titan Security Key is anchored in the sealed chip as opposed to any other later step which takes place during device manufacturing.

From https://cloud.google.com/blog/products/identity-security/titan-security-keys-now-available-on-the-google-store

Google Titan Security Key Teardown

- Recto: HW manufacturer is **Feitian**

- Verso: secure element is ***NXP A7005a***



Other FIDO U2F products based on ***NXP A700x*** chip:

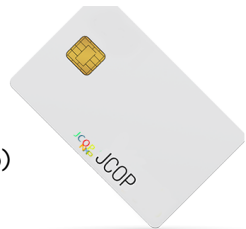
- Yubico Yubikey Neo
- Feitian K9, K13, K21, K40

Similarities with other NXP Products

- Several NXP JavaCard smartcards (JCOP) can be purchased on the web
- Those similar to ***NXP A700X*** are based on ***NXP P5x*** chips

NXP J3D081

JCOP 2.4.2 R2
CC EAL5+ (2015)

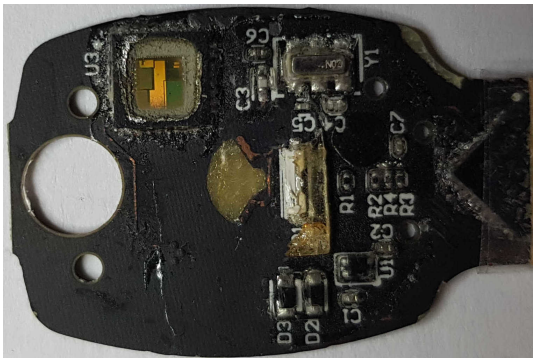


- NXP J3D081_M59_DF and variants
- NXP J3A081 and variants
- NXP J2E081_M64 and variants
- NXP J3D145_M59 and variants
- NXP J3D081_M59 and variants
- NXP J3E145_M64 and variants
- NXP J3E081_M64_DF and variants

Rhea
2nd largest moon of Saturn after ***Titan***

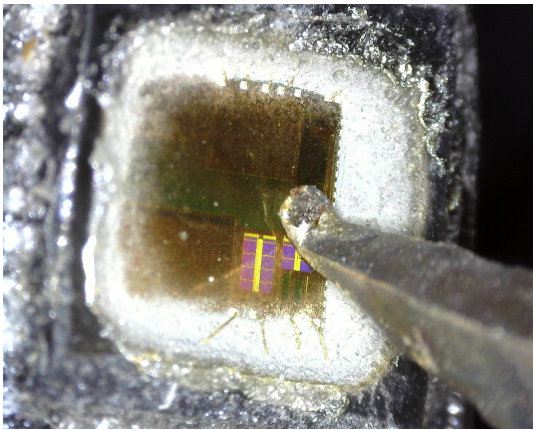
Titan / Rhea Package Openings

- ***Titan's NXP A700X***:
  - wet chemical opening
    aluminium tape + fuming nitric acid
  - ***Google Titan Security Key*** still alive !
- ***Rhea***:
  - mecanical opening
    scalpel + acetone
  - ***Rhea*** still alive !



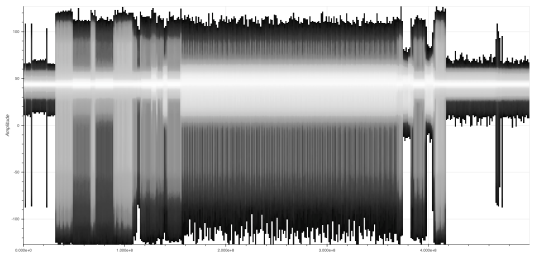
EM Side-Channel Acquisition Setup (about 10k€)

- **EM sensor:** Langer ICR HH 500-6 (diam. 500μm, freq. BW 2MHz to 6GHz)
- **Manual micro-manipulator:** Thorlabs PT3/M 3 axes (X-Y-Z)
- **Oscilloscope:** PicoScope 6404D, freq. BW 500MHz, SR 5GSa/s

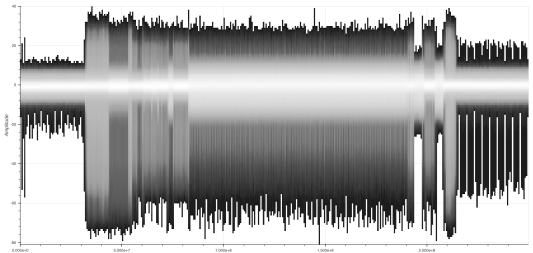


EM activity of ECDSA signature on Titan / Rhea

- ***Titan***

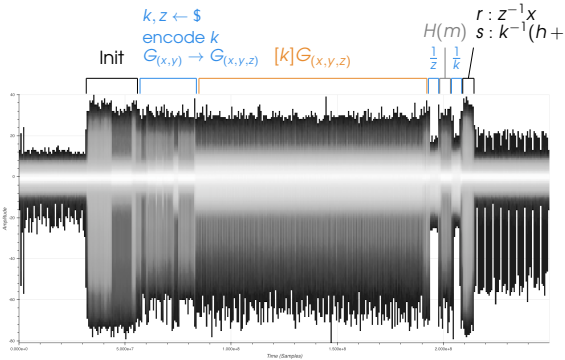


- ***Rhea***



- ECDSA signature EM activities on ***Titan*** and ***Rhea*** look very similar !

Rhea – Signature Alg. – EM Radiations

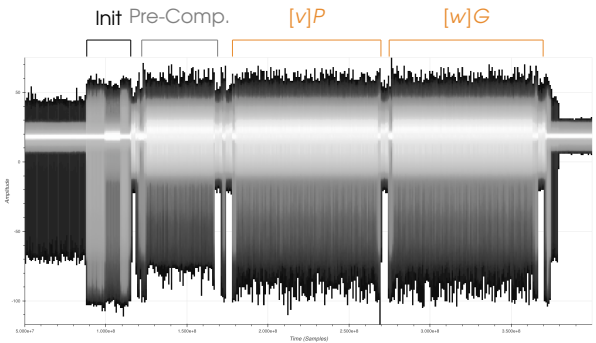


E = P256, SHA-256

Scalar Multiplication $[k]G$

- Constant time algorithm
  Double-and-Add-Always
- 128 iterations for a 256-bit nonce k
  2 bits of k by iteration
- Randomized point on projective coordinates

Rhea – Verification Alg. – EM Radiations



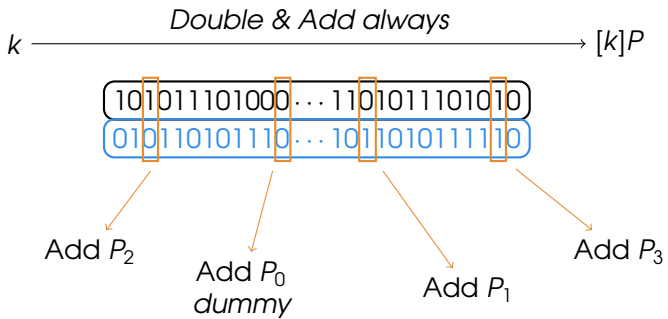
Scalar Multiplications $[v]P$ and $[w]G$

- Scalar mult. are **not constant time** simple *Double-and-Add*
- Expensive pre-comp. before $[v]P$
- The scalar multiplication algorithm is a **left-to-right *comb method*** (of width 2)
- Scalars are not blinded.

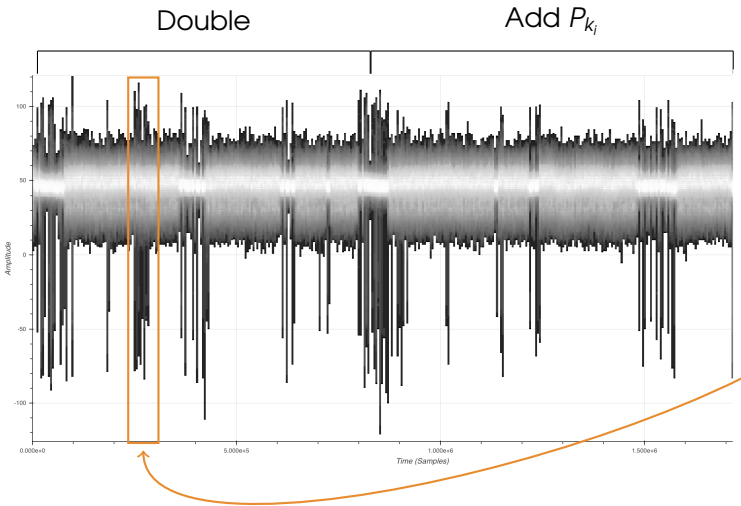
Scalar Mult. w. Left-To-Right Comb method (width 2)

Precomp: $P_0, P_1 = P, P_2 = [2^{128}]P, \ P_3 = P_2 + P_1$

$k =$ $1010111101000 \cdots 1101011101010$ $0101101011110 \cdots 1011010111110$

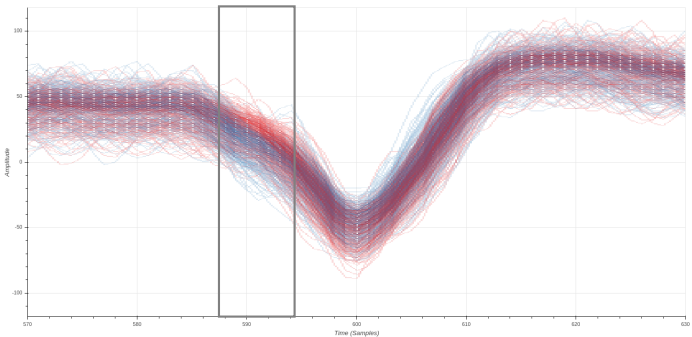


Rhea – Single Iteration – Leakage Area



Rhea – Leakage Illustration

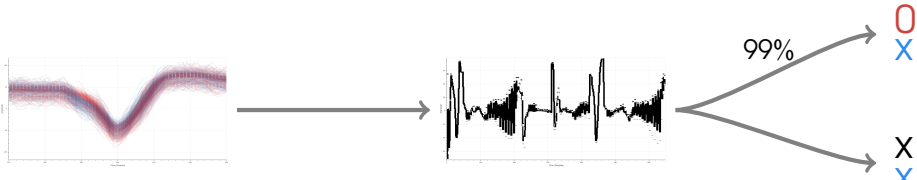
1000 Superposed Iterations – Zoom in Leakage Area



$k_i = 0$ x

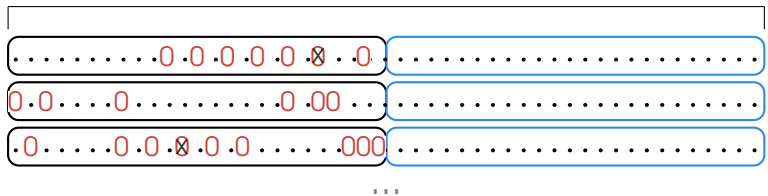
$k_i = 1$ x

Single Trace Matching



256-bit Nonces

Identified 0 bits
27.5 by nonce
in average



Hidden Number Problem

- Recovering an ECDSA secret key given some partial knowledge on the nonces can be expressed as a Hidden Number Problem (HNP/EHNP)
- HNP and EHNP can be defined as games with an oracle
- The oracle reveals x and $f_m(\alpha x)$ for several random values of x

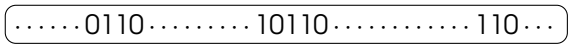
  The player should find the hidden value α
- HNP: f_m discloses the m most significant bits of αx

  $1101001010\cdots\cdots\cdots\cdots\cdots\cdots$
- EHNP: f_m discloses m bits of αx, not necessarily consecutive

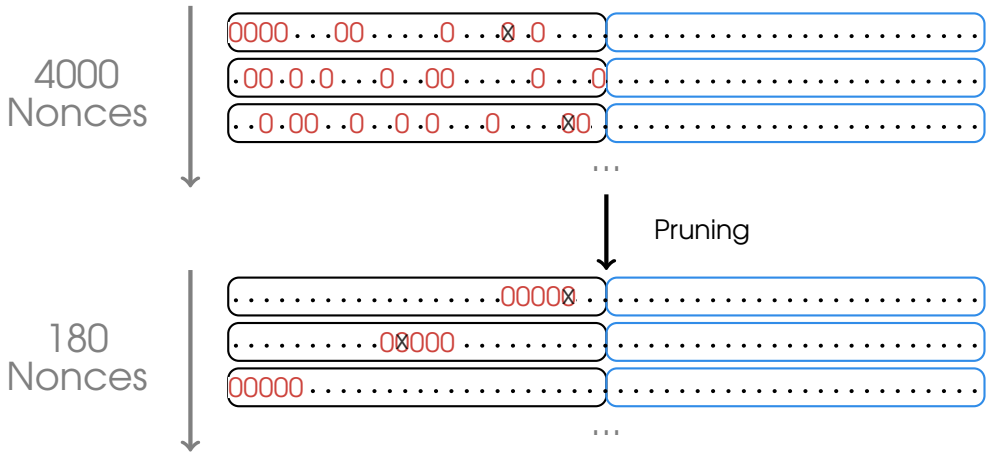
  $\cdots 1 \cdots 01 \cdots 0 \cdots 101 \cdots 0 \cdots 10 \cdots$

Solving (E)HNP

- (E)HNP can be reduced to instances of lattice-based problems (SVP, CVP) that may be solved using lattice reduction techniques (LLL, BKZ)
- # oracle queries and # known bits dictate the size of the lattice and the probability of success
- In practice EHNP can be solved when the m bits revealed by the oracle form blocks of sufficiently many consecutive bits



Rhea – Nonces Selection



Brute-Forcing the Key

- LLL reduction (for 80 signatures) takes about 100s
- 5 errors among 180 available signatures

$\hookrightarrow$ Brute-force attack on random subsets

Final Attack

- Acquisition of 4000 traces: $\sim 4h$
- Trace Processing: $\sim 4h$
- Brute-force attack: $\sim 20min$

Touchdown on Titan

- Use Rhea parameters for PoI extraction
- Pruning: from 6000 signatures to 156
- 7 errors among 156 available signatures

$\hookrightarrow$ Brute-force attack on random subsets

Final Attack

- Acquisition of 6000 traces: $\sim 6h$
- Trace Processing: $\sim 6h$
- Brute-force attack: $\sim 30min$

Impact on Google Titan Security Key

- Proposed attack allows to physically extract an ECDSA private key linked to application account secured by FIDO U2F

- But high attack requirements:
  - Adversary already stolen victim's account login & password
  - Physical access to ***Google Titan Security Key*** during 10 hours
  - Adversary has access to:
    - Chemical lab
    - Expensive SCA setup
    - Custom softwares

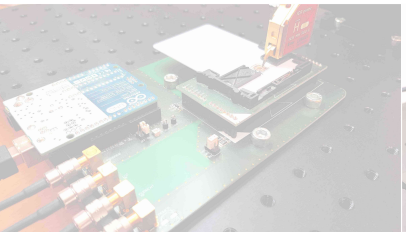
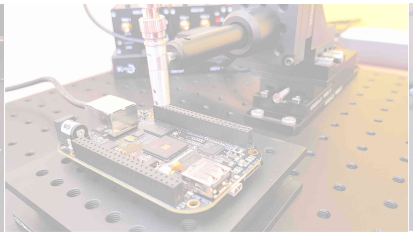
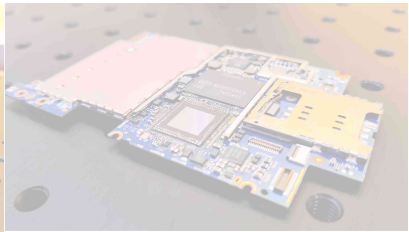
- **So still safer to use *Google Titan Security Key* than nothing !**

Attack Mitigations

- Hardening the NXP P5x cryptographic library:
  - Blinding of the scalar
  - Re-randomizing table lookup of precomputed points in comb implementation at each new access
- Use FIDO U2F counter to detect clones:
  - U2F counter *may* be used for detecting cloned U2F devices
  - Only limit validity of attack
  - Dependent on FIDO U2F server implementation

NinjaLab

Improve the Security of your Cryptographic Implementation

https://ninjalab.io
contact@ninjalab.io

NinjaLab
161 rue Ada
btiment 4 - CC477
34095 MONTPELLIER CEDEX 5
FRANCE