LEDGER

Le hardware wallet.
Un coffre-fort personnel pour la finance décentralisée.

9 juin 2022

Ledger Donjon

L’équipe de sécurité produit chez Ledger

Approche offensive

- Recherche de vulnérabilité en continu et exploitation sur tous les produits conçus par Ledger
- Recherche de vulnérabilité sur le code et le matériel tiers (HSM, secure elements, etc.)

Défense en profondeur

- Automatisation : analyse statique, fuzzing, pour trouver de problèmes à grande échelle
- Revue de tout le code dès que la sécurité peut être affectée
- Définition de nouvelles couches de sécurité, pour rendre la vie des pirates plus difficile

Programme de bug bounty

- Prise en compte de tous les problèmes remontés
- Proposition de correctifs de sécurité

Agenda

1. Comment protéger ses clés ?
2. Support de la DeFi par les wallets
3. Architecture d’un hardware wallet
4. Comment casser un hardware wallet

Comment protéger ses clés ?

Où entreposer ses secrets ? À qui les confier ?

BINANCE

coinbase

Exchanges

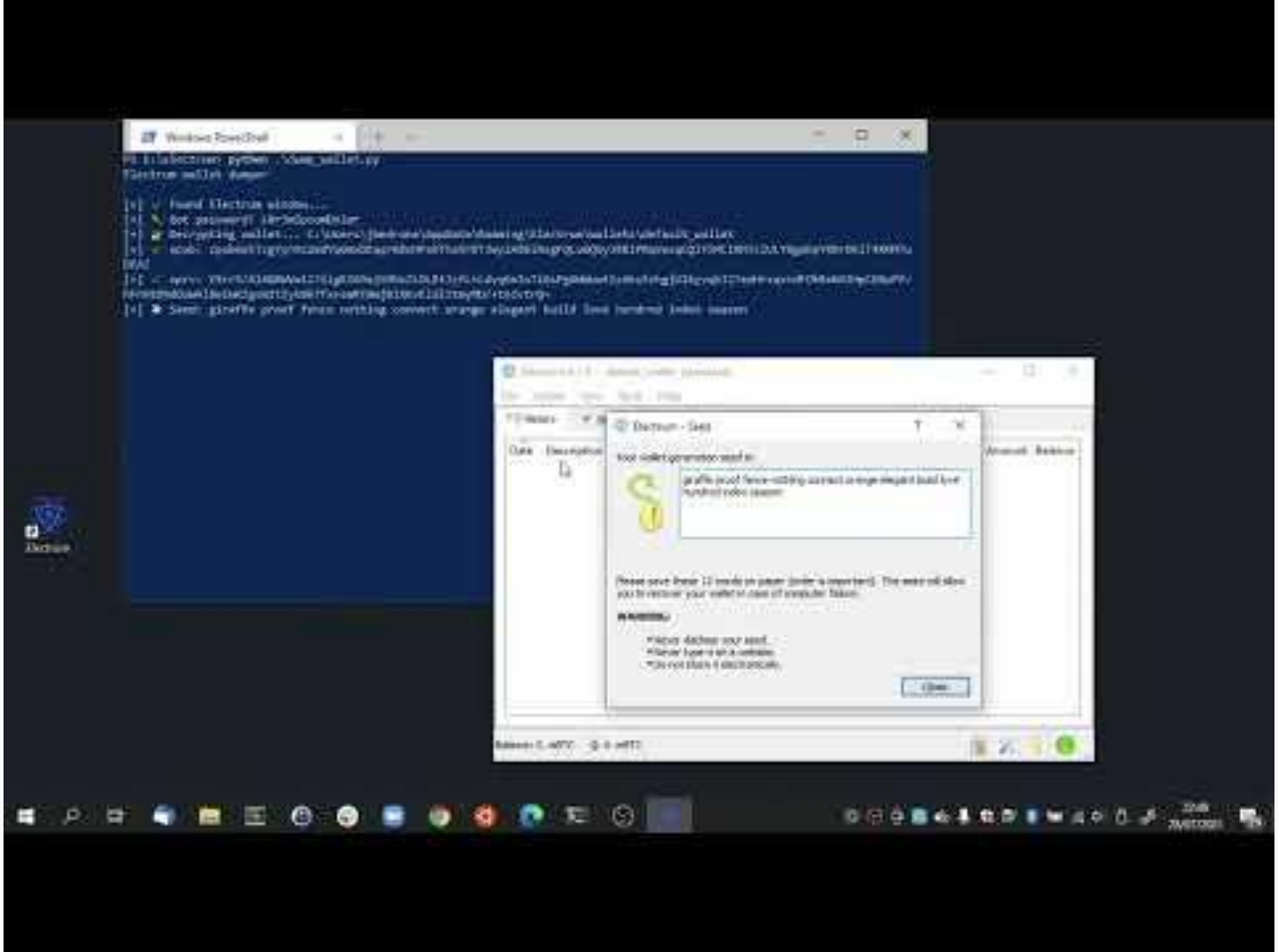
- Les exchanges stockent vos clés
- Délégation de sécurité
- “Not your keys, not your coins”
- Pas de contrôle total des transactions

Software Wallets

- L’application protège les clés
- Pratique : un simple logiciel
- Sécurité ?

Hardware Wallets

- Le matériel protège vos clés
- Résistance aux malware
- Résistance aux attaques physiques
- Compromis : plus sécurisé, mais moins pratique


Windows PowerShell
Electrum wallet dumper
[+] ✓ Found Electrum wallets...
[+] ✓ Got password: iAmSoCoolBro
[+] ✓ Decrypting wallet... C:\Users\[redacted]\AppData\Roaming\Electrum\wallets\default_wallet
[+] ✓ Wrote: [redacted]
[+] ➤ Seed: giraffe proof fence nothing connect orange elegant build love rundrop index season
Electrum
Electrum - Seed
your wallet generation seed is:
giraffe proof fence nothing connect orange elegant build love rundrop index season
Please save these 12 words on paper (order is important). This seed will allow you to recover your wallet in case of computer failure.
WARNING:
* Never disclose your seed.
* Never type it on a website.
* Do not store it electronically.
Close

Software wallets sur téléphone

Fonctionnalités de sécurité

- Isolation des applications
- Secure Enclave, parfois avec un Secure Element
- Chaîne de confiance depuis le démarrage

Faiblesses

- Surface d’attaque importante
- Pas de gestion des algorithmes utilisés dans la blockchain depuis l’enclave sécurisée

→ Vulnérabilités face aux malwares

Hardware Wallets

Un équipement dédié à la sécurité des cryptomonnaies

Fonctionnalité de sécurité

- Résistant contre les malware
- Affichage sécurisé : “What you see is what you sign”
- Protection contre les attaques physiques


DeFi ?
Ledger

DeFi avec les portefeuilles matériels

Intéractions avec un smart contract

- Transaction standard sur la blockchain :
  - Destinataire
  - Frais de transaction maximum, etc.
- Avec certaines particularités :
  - Montant nul, et **données** spécifiques au contrat

```
MethodID: 0x0b86a4c1
[0]:  000000000000000000000000cafe001067cdef266afb7eb5a286dcfd277f3de5
[1]:  000000000000000000000000000000000000000000000000cf2c4708747031eb8c
[2]:  000000000000000000000000000000000000000000000000e6b491daf5de49
[3]:  000000000000000000000000c02aaa39b223fe8d0a0e5c4f27ead9083c756cc2
[4]:  00000000000000000000000000000000000000000000000000000000000000a0
[5]:  0000000000000000000000000000000000000000000000000000000000000001
[6]:  000000000000000000004de5458ae80894a0924ac763c034977e330c565f1687
```

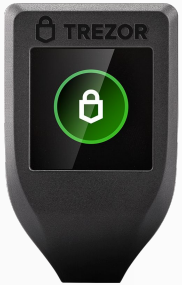
Swapping with ParaSwap

Blind signing

With ParaSwap Plugin

Sécurité des hardware wallets

Architecture matérielle



Trezor T

- **Un seul MCU** : ARM Cortex-M4 (STM32F427)
- 180 MHz
- 2 MB Flash
- 256 KB SRAM
- True RNG
- **Entièrement open source**

Coldcard Mk3

- STM32L496RGT6 (Cortex-M4, 80 MHz, 320 KB RAM)
- Secure Element : ATECC608A (“**Mémoire sécurisée**”)
- True RNG
- **Entièrement open source**

Ledger Nano X

- STM32WB55 (Cortex-M4, 32 MHz, 128 or 256 kB RAM)
- Secure Element: ST33J2M0 (60 MHz, 2MB Flash, 50 kB RAM)
- True RNG
- **Tout le code manipulant les secrets tourne sur le secure element**
- **Partiellement** open source

PWN2BTC™

Smart Contract & Crypto Hardware Exploitation Competitions

June 7th 2022

What is the best security for cryptocurrencies? What are the safest Smart Contracts? What makes the most secure military encrypted hardware? We asked ourselves this question and are now posing it as a contest.

Over the past few years we have seen some incredible feats of cryptocurrency, smart contract, and hardware wallet compromise. Are the items in production effective at protecting the value of what we have stored in them?

We're here to test that with the first annual PWN2BTC.

You Pwn it, you keep the Crypto!

We want you to have an opportunity to crack some of the best that the industry is offering. Show up to show off your hacker skills and see if the technology stands up to the marketing. Immunefi and Unciphered will be extending their expertise to organize each competition.

Unciphered LLC
@uncipheredLLC

WINNER WINNER CHICKEN DINNER.

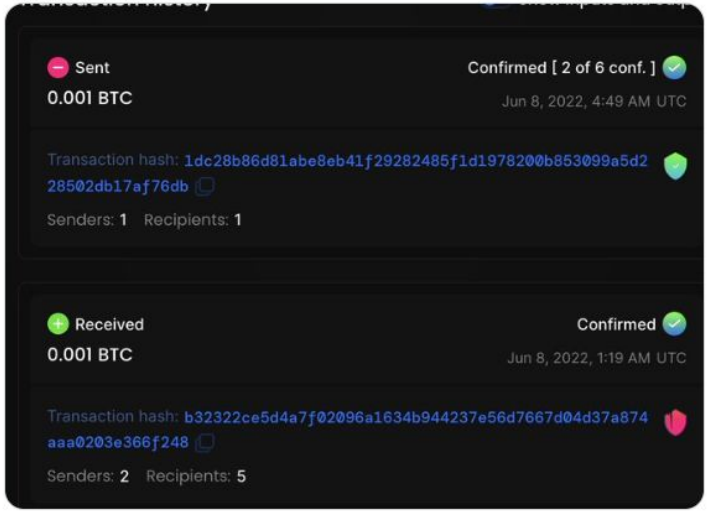
The first winner in the worlds first #PWN2BTC Hardware Competition is Team 0x414141

They cracked the Trezor One w/ firmware 1.11.1 & Max Pinsize: blockchair.com/bitcoin/addres...

Total Time: 3h 30min.

1st place finish for the Trezor One is $1,000USD.

Traduire le Tweet



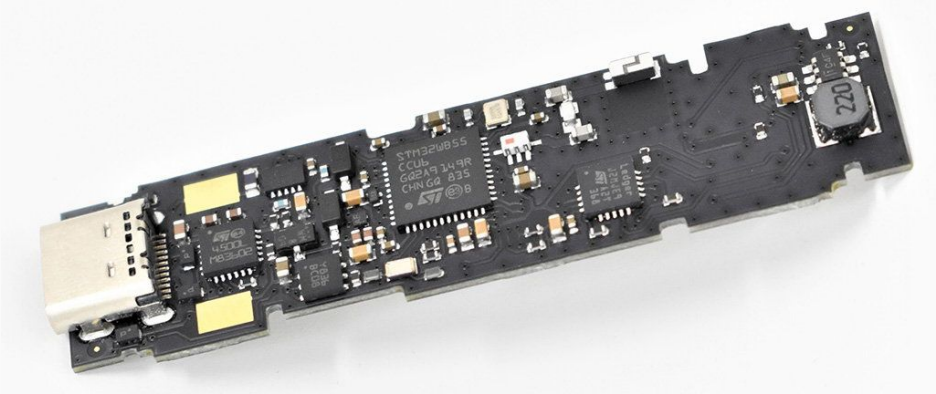
7:02 AM · 8 juin 2022 · Twitter Web App

Secure Element

Une sécurité physique certaine. Pour quelle sécurité logicielle ?

ST31H320 - Spécifications techniques

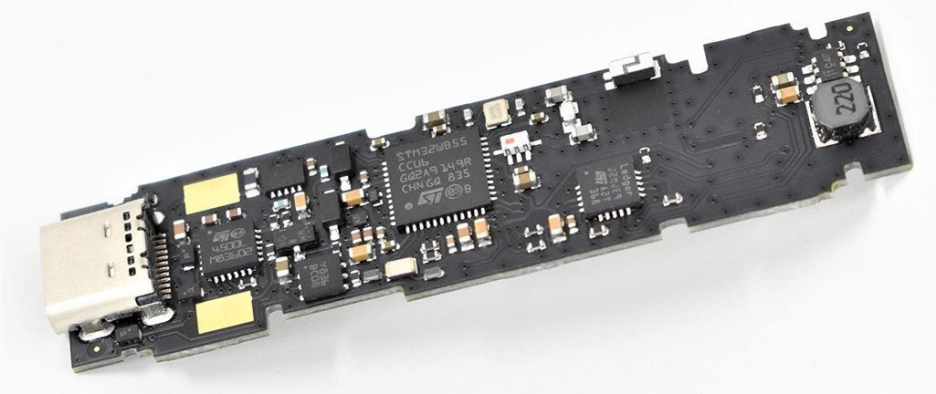
- CPU: Cortex M0+, 28 MHz
- 10 KB of RAM: 4K pour les apps, 1K pour la pile
- Pas d’interface de débogage
- Programmes exécutés depuis la mémoire flash



Secure Element

Une sécurité physique certaine. Pour quelle sécurité logicielle ?

ST31H320 - Développement et contraintes de sécurité



Processeur

- Trop lent pour le fuzzing
- Pas de mécanisme de sécurité récent : TrustZone, authentification des pointeurs…

Mémoire limitée :

- Pas de cookies de pile
- Pas d’allocateur dynamique
- Langages bas niveau uniquement : C

Programmes exécutés depuis la mémoire Flash : pas d’ASLR

Pas d’interface de débogage : comment comprendre les bugs ?

Tests de sécurité automatisés

Apporter la sécurité logicielle dans le monde de l'embarqué

Analyse statique dans la CI

- Coverity Scan
- CodeQL
- Clang Static Analyzer

Tests fonctionnels sur du vrai matériel

- Wallets contrôlés à distance
- Contrôle des boutons
- Capture du bus de l'écran pour déterminer l'affichage + OCR
- **… ne passe pas à l'échelle** (> 100 apps)

Issues: Project Scope | All In Project

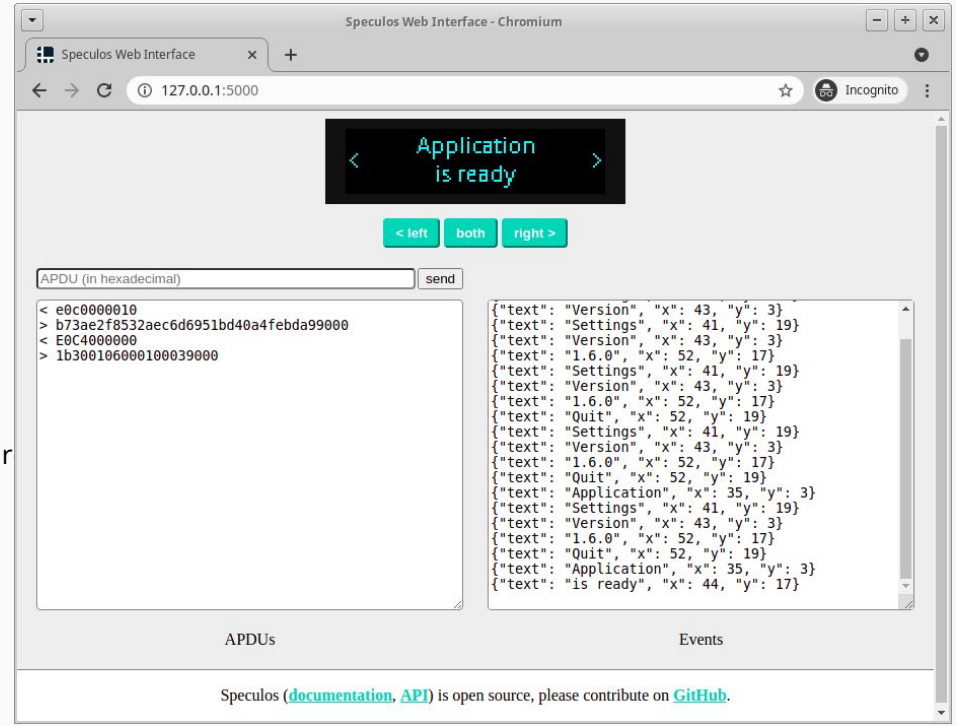
CID	Type	Impact	First Detected
308527	Untrusted loop bound	Medium	01/19/21
308530	Untrusted loop bound	Medium	01/19/21
308531	Untrusted loop bound	Medium	01/19/21
349037	Operands don't affect result	Medium	06/09/21
349038	Unsigned compared against 0	Medium	06/09/21
349039	Unsigned compared against 0	Medium	06/09/21
349040	Out-of-bounds access	High	06/09/21
349041	Out-of-bounds read	High	06/09/21
349042	Negative array index read	High	06/09/21
349043	Buffer not null terminated	High	06/09/21
349044	Buffer not null terminated	High	06/09/21
349045	Buffer not null terminated	High	06/10/21

Emulation d’un wallet

Débogage, test, fuzzing… Le tout sur un ordinateur

- Émulateur d’applications reposant sur QEMU
- Support de GDB (QEMU)
- Framework de test complet
- Fuzzing : rapidité et scalabilité
- Tests fonctionnels à l’échelle dans la CI
- Open source (https://speculos.ledger.com/)

→ Fuzzing + analyse statique : détection des corruptions mémoir



Pourquoi un Secure Element?

Beaucoup de contraintes de développement, pour quelle utilité ?

Attestation à distance

Le device est authentique, personne ne peut le contrefaire

Pas besoin d’”autocollants de sécurité”

Intégrité du code

Envoi sécurisé des applications

Les applications ne peuvent pas être modifiées

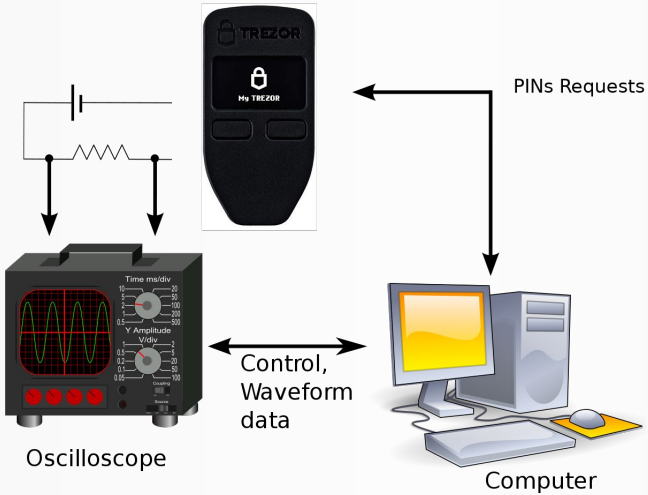
Vous avez perdu votre device ? Les fonds sont à l’abri

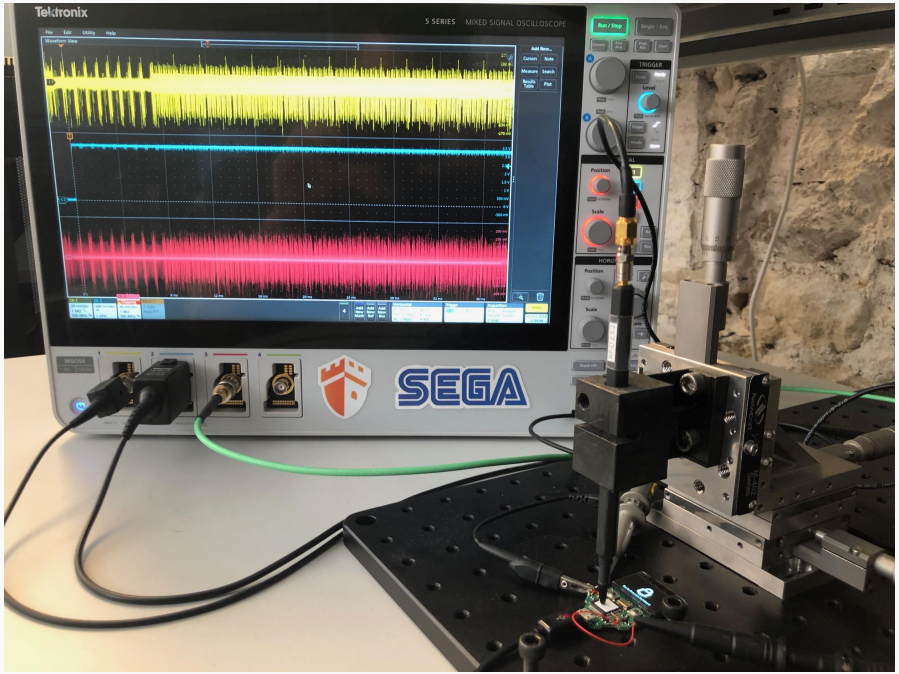
L’attaquant ne peut pas retrouver le code PIN, ni extraire vos clés

Attaques matérielles

Attaques par canaux auxilliaires

Exploitation des fuites d'information lors du fonctionnement du device





Attaques par faute sur des puces sécurisées

Protection des secrets dans l’embarqué

Microcontrôleurs

- Mémoire flash, fusibles pour empêcher la lecture
- Peu onéreux
- Pas de protection contre les attaques physiques

Secure Elements

- Protection contre les attaques physiques
- Évalués / certifiés par des laboratoires spécialisés
- Accès limité (JCVM, NDA…)

Microchip ATECC508A

- Mémoire sécurisée
- Utilisée pour sécuriser des objets connectés
- Quelle sécurité ?

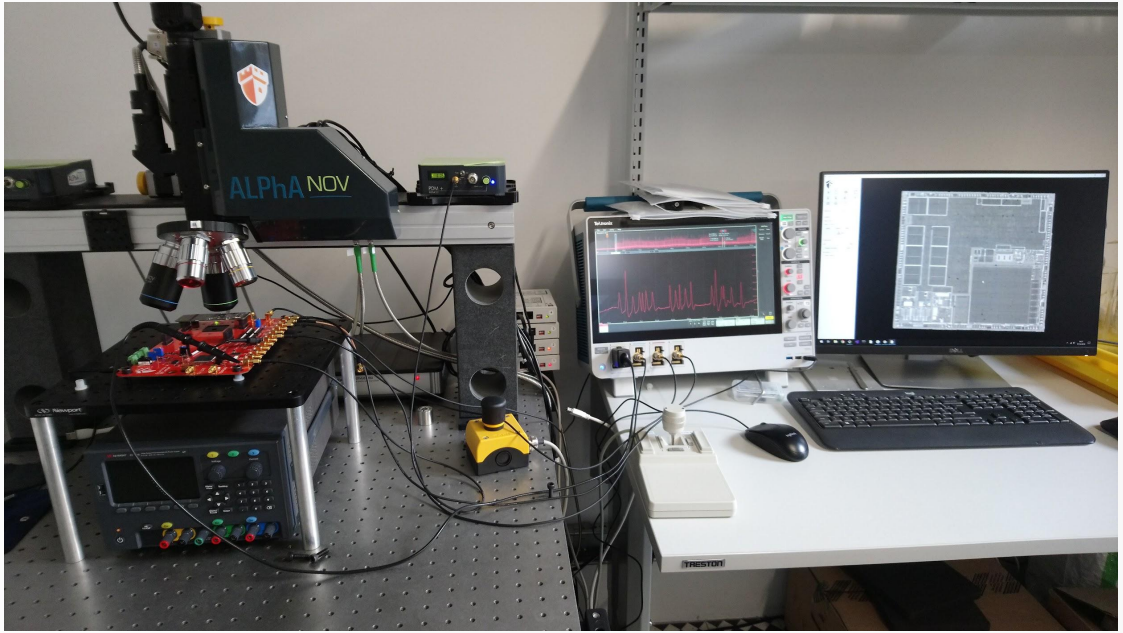
ATECC508A (Coldcard Mk2)

- Surface d’attaque limitée
- Firmware secret
- Protection contre *certaines* attaques
  - Glitches en tension
  - Bouclier métallique
- **Pas de protection contre les lasers**

Attaques laser

Modification du comportement d'un programme en fautant le matériel à l'aide d'un laser

Carte d'attaque (Donjon) et circuit sous le microscope et le faisceau laser



Banc de test, oscilloscope, logiciel d'attaque (Donjon)

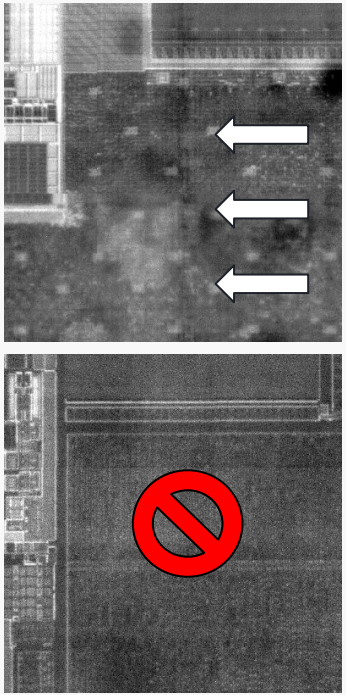
ATECC508A au microscope

Le silicium est transparent aux infrarouges

Les circuits intégrés sont photosensibles

Illuminer un transistor peut le rendre conducteur…

… ce qui va provoquer des erreurs lors de l’exécution !

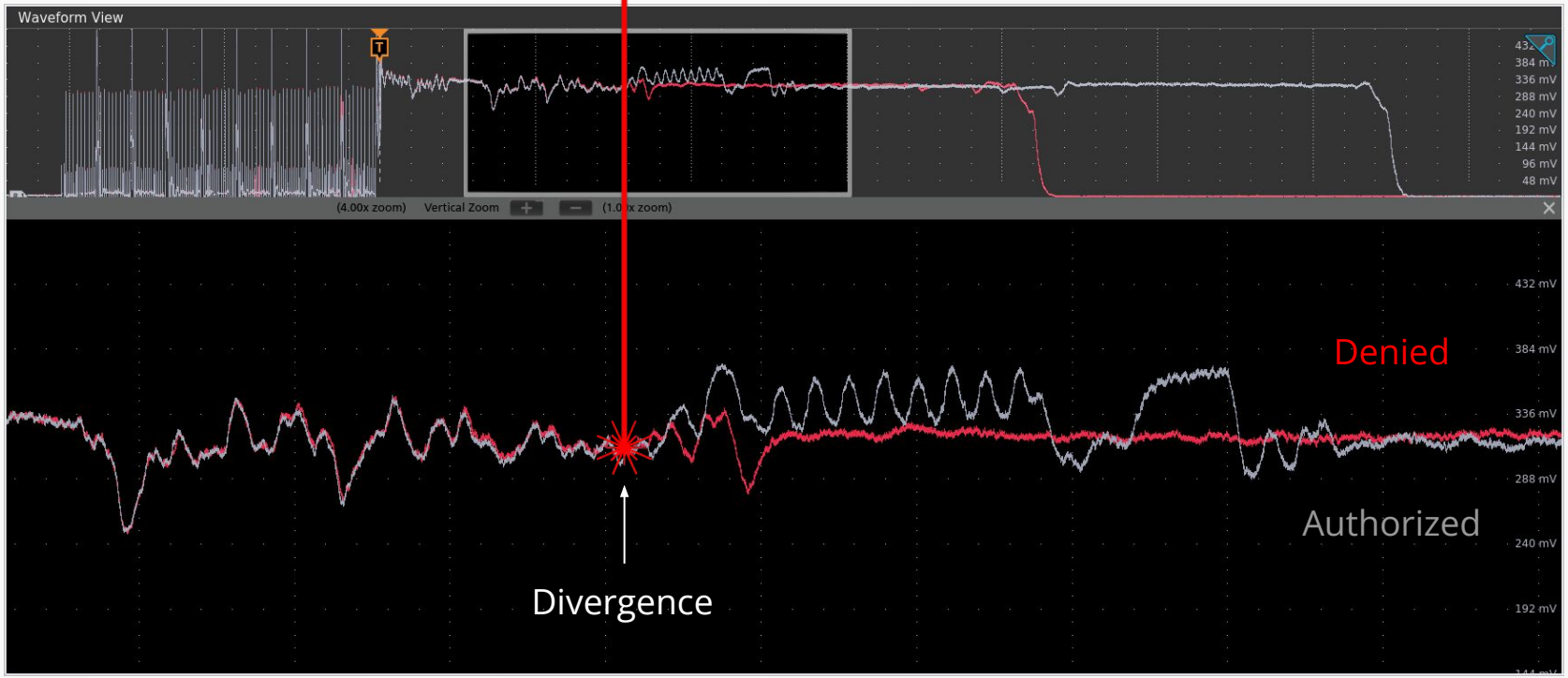


Code supposé

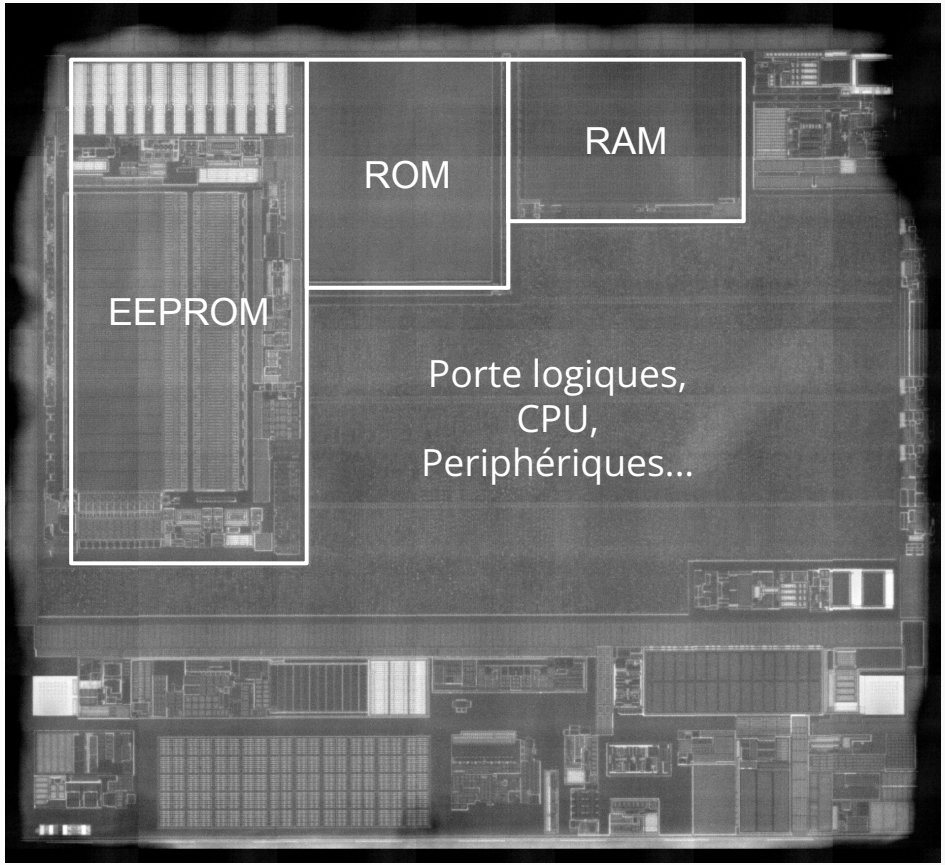
```
if (global_state.authenticated) {
  key_data = eeprom_read(private_key);

  signature = sign_transaction(key_data, transaction_data);
  i2c_send(SW_NO_ERROR + signature);
} else {
  // Comman  d requires authentication
  i2c_send(SW_CONDITIONS_NOT_SATISFIED);
}
```

Quand tirer?

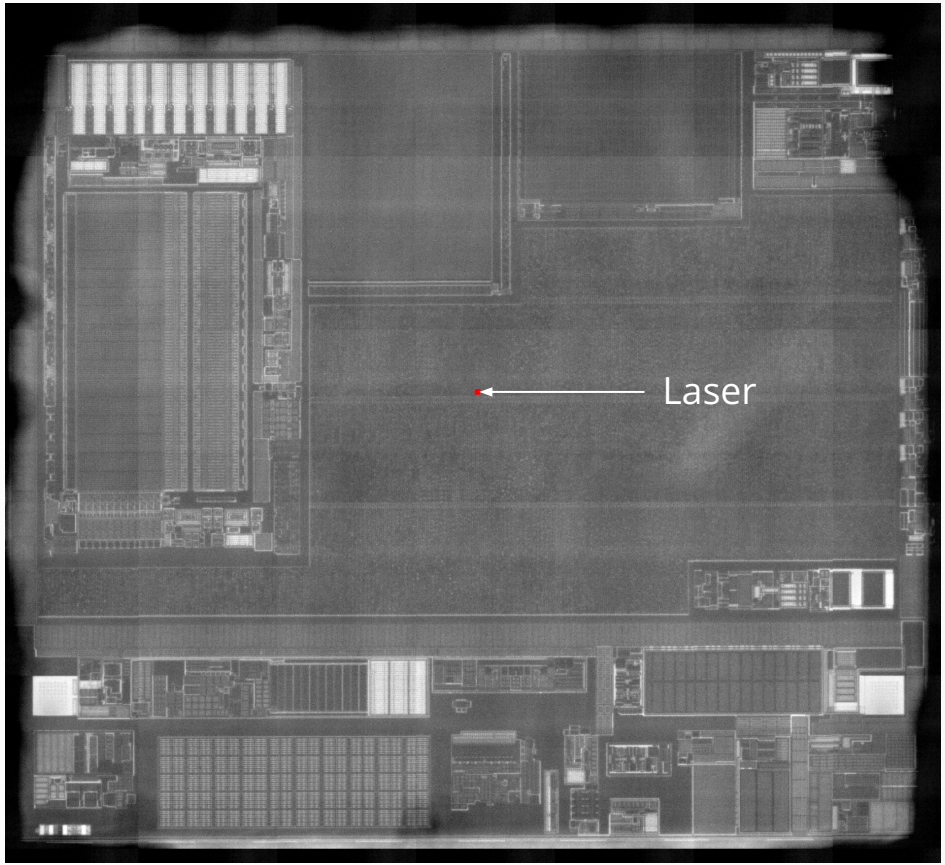
Instrumentation

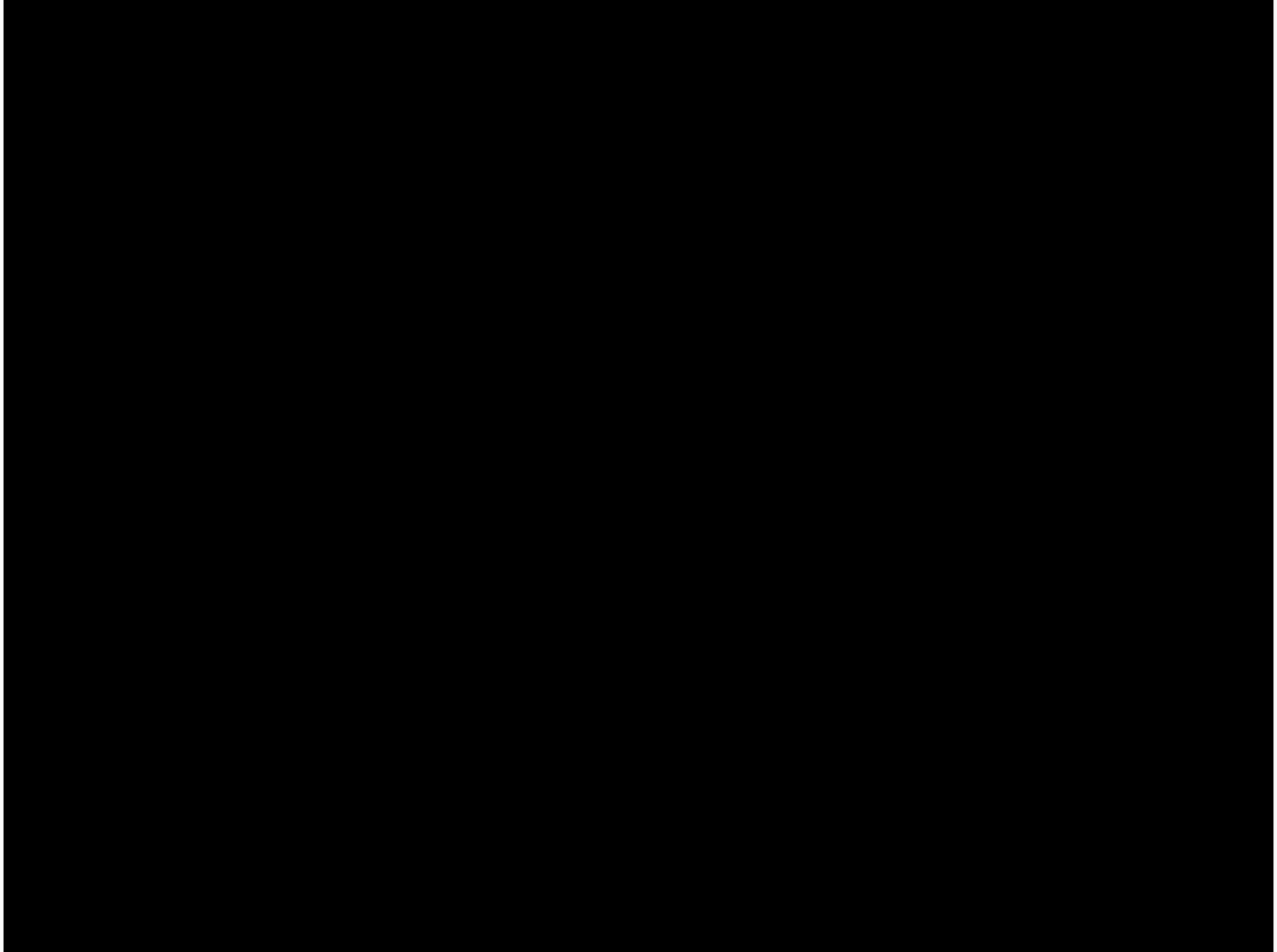


Où tirer ?



Où tirer ?





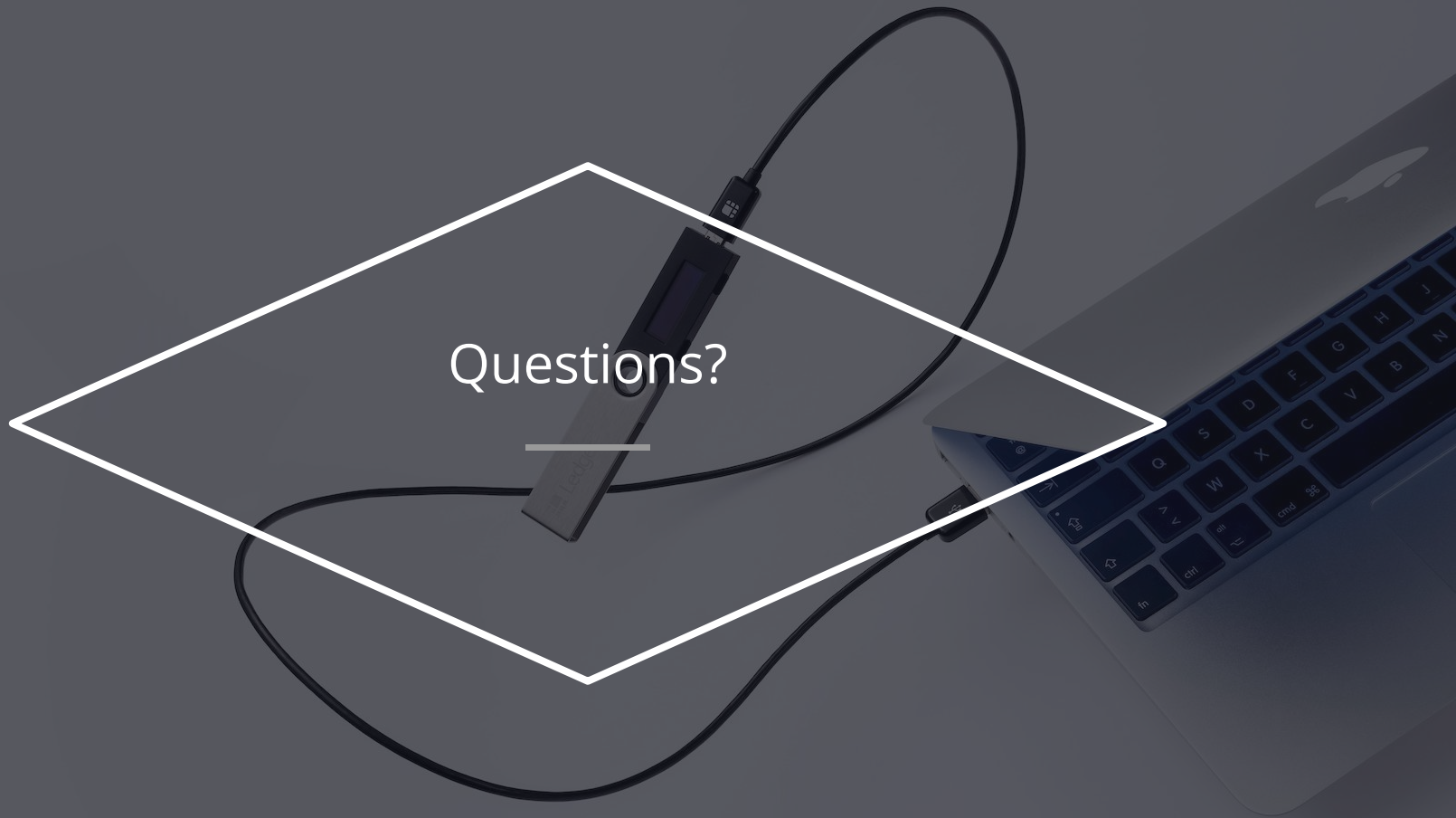
Ça marche vraiment ?

- Attaque présentée à Black Hat USA 2020. Casse Coldcard Mk2 (ATECC508A)
- Coldcard a mis à jour son chip : Coldcard Mk3 (ATECC608A)
- Seconde attaque présentée à Black Hat USA 2021. Casse Coldcard Mk3
- Coldcard a changé le chip du Coldcard Mk3 (ATECC608B)
- Présentation d’une nouvelle attaque à Black Hat Europe 2022.

Conclusion

- La sécurité est un élément crucial des wallets
- Les hardware wallets apportent le niveau de protection le plus élevé
  - … au détriment de la facilité d’utilisation
- Le niveau des attaques monte au vu des sommes en jeu
- Challenge : diversifier les usages malgré les contraintes fortes sur le matériel

Questions?

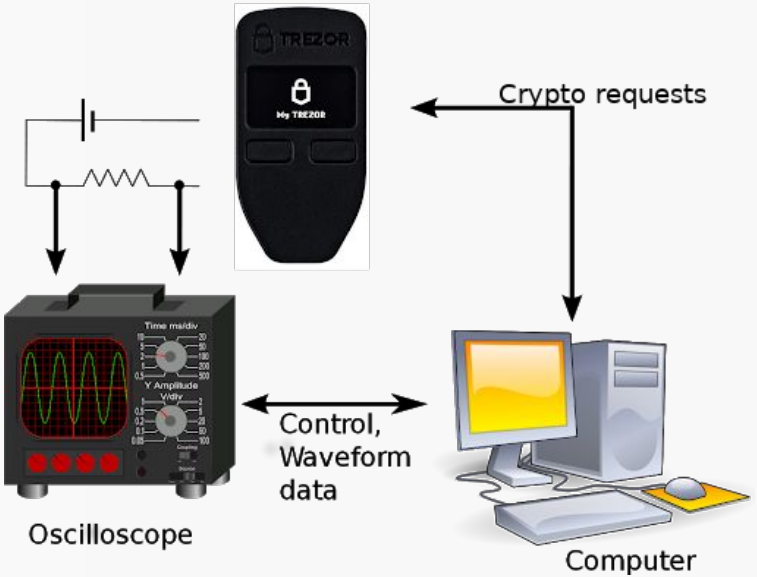


Side Channel Attacks

Exploiting physical leakage of information during device handling

Power consumption, electromagnetic emanations:

- Measure power consumption during cryptographic operations
- Record traces
- Post-process traces
- Conduct **Side Channel Analysis**

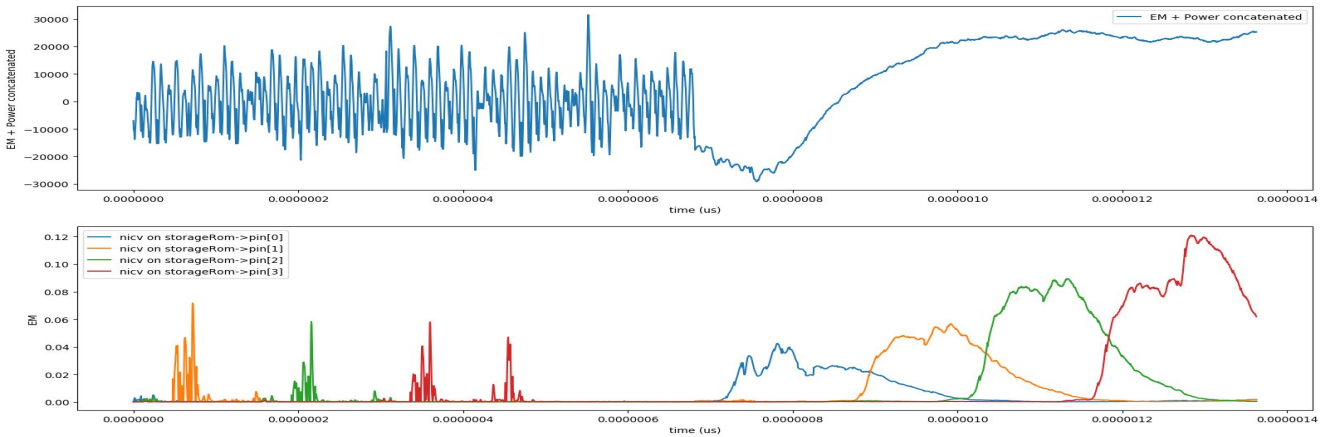


Side Channel Attack on Trezor PIN

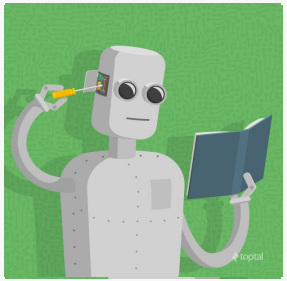
```
/* Check whether pin matches storage.  The pin must be
 * a null-terminated string with at most 9 characters.
 */
bool storage_containsPin(const char *presented_pin)
{
/* The execution time of the following code only depends on
the
* (public) input.  This avoids timing attacks.
*/
    char diff = 0;
    uint32_t i = 0;
    while (presented_pin[i]) {
        diff |= storageRom->pin[i] - presented_pin[i];
        i++;
    }
    diff |= storageRom->pin[i];
    return diff == 0;
}
```

Acquiring traces

- Power / EM single trace
- Trace synchronization
- Point of interests: detection depending on `storageRom->pin[i] - presented_pin[i]` for i <= 0 < 4



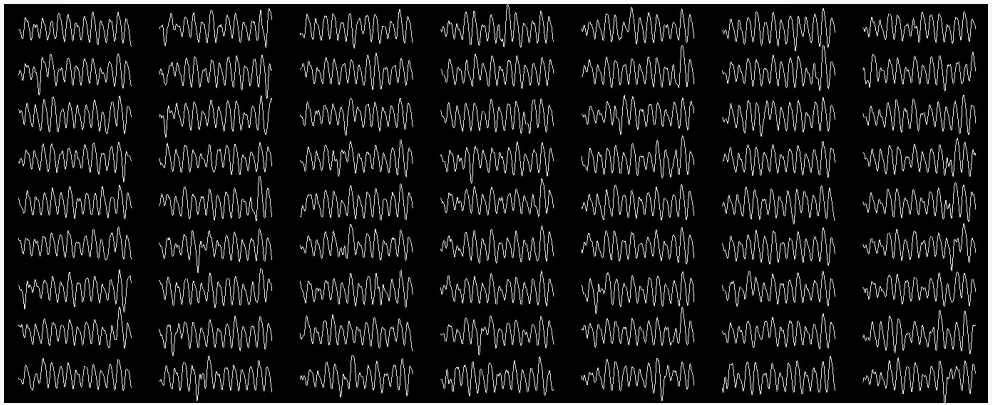
Machine Learning

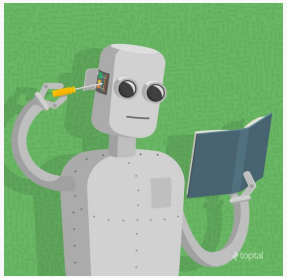


Machine Learning on PIN

PIN behaviour is learnt in a very similar way

digit=1 :
digit=2 :
digit=3 :
digit=4 :
digit=5 :
digit=6 :
digit=7 :
digit=8 :
digit=9 :





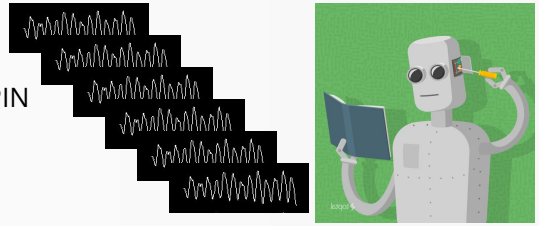
Device A

Device B

Breaking Trezor PIN

1st phase:

- Get device A, record many traces with random PIN
- Learn the behavior of the device

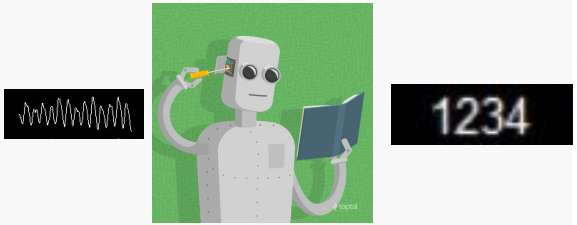


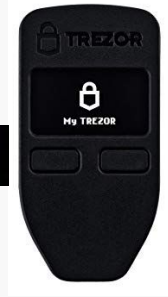
Device A

2nd phase:

- Get a physical access to the attacked device
- Enter random PIN, measure the power consumption of the device
- Ask to the ML algorithm the most likely PIN
- **Profit!**

On average, **5 tries to guess the correct PIN** (15 tries at most on Trezor)



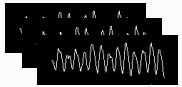


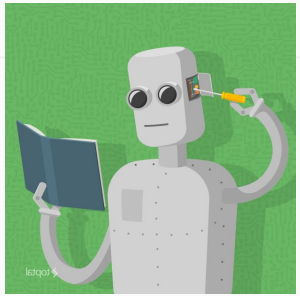
Device B

Side Channel Attacks: PIN verification function

Ledger

1. Get a device A, record many traces with random PIN





Device A

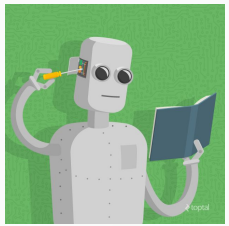
2. Learn the behavior of the device

---

3. Get a physical access to the attacked device

4. Enter random PIN, measure the power consumption of the device, ask to the MLA
   try the most likely PIN

1234



Device B

On average, 5 tries to guess the correct PIN
(15 tries at most on Trezor)

5. Enjoy