THALES
Building a future we can all trust

Side-Channel Attacks

Cyber In Occitanie

SRE

Who am I?

2018
Master in computer science

TELECOM SudParis

IP PARIS

2021
Thesis in Deep Learning for SCA

UNIVERSITÉ DE LYON

THALES
Building a future we can all trust

From 2021-…
Security evaluator

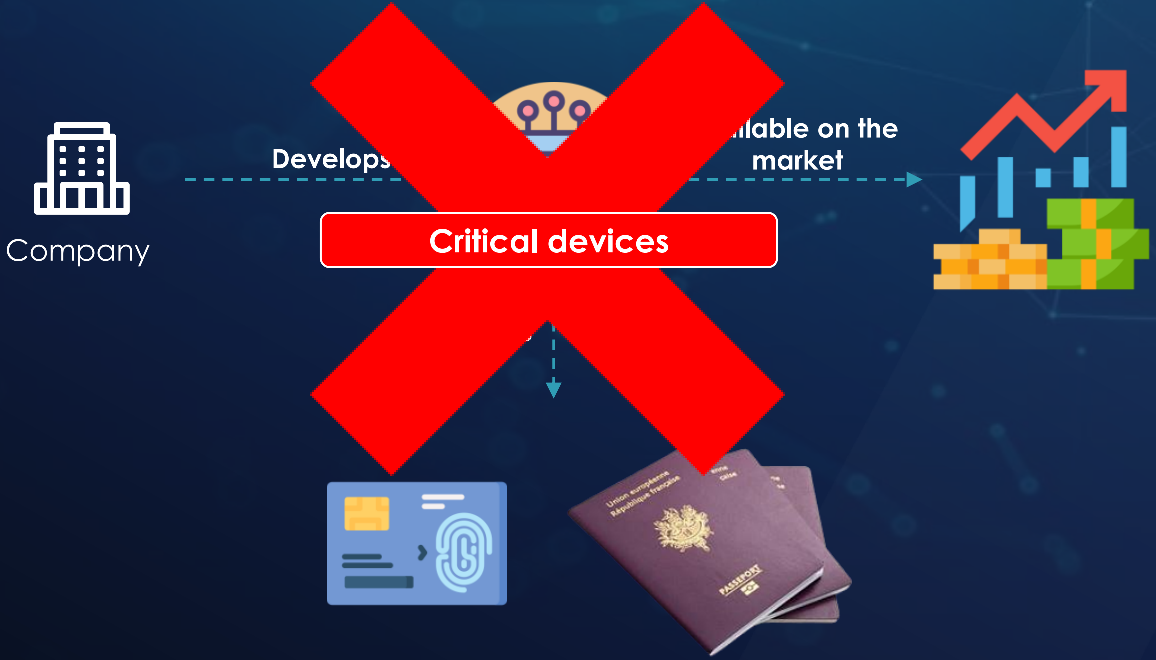
THALES
Building a future we can all trust

Agenda

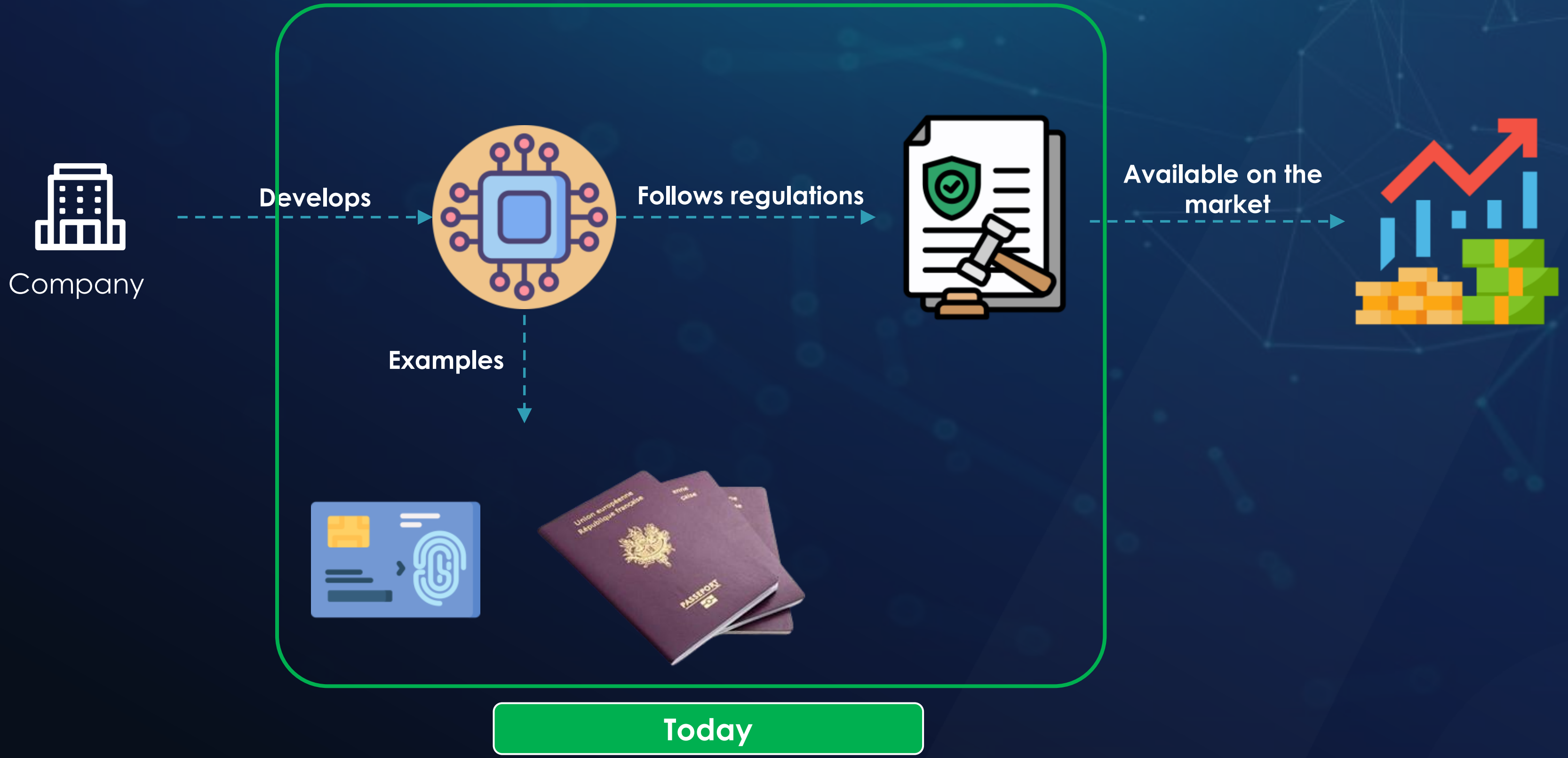
- Context
- Side-channel attacks
- **Statistical distinguishers**
- Demonstration

Context

Context



Context



Company

Develops

Follows regulations

Available on the market

Examples

Today

Context

Common Criteria for Information Technology Security Evaluation (or CC*)

- International standard (ISO/IEC 15408) for computer security certification
- Framework which helps developer for defining
  - The Target of Evaluation (**TOE**)
  - The security assets to protect (e.g. secret keys)
  - The secure functions related to the TOE (e.g. secure communication, authentication process)
  - The level of security assurance (EAL, Evaluation Assurance Level)
- Different security levels
  - **EAL 1**: This is the most basic level, focusing on functional testing to ensure the product performs as specified.
  - **EAL 2**: This level involves more in-depth testing, including structural analysis of the product's components.
  - **EAL 3**: It adds a methodical approach to testing and checking the product's security features.
  - **EAL 4**: This level requires a more rigorous design process, along with thorough testing and review of the product's security.
  - **EAL 5**: It introduces semi-formal methods for design and testing, increasing the level of assurance.
  - **EAL 6**: This level builds upon EAL5 with more formal verification techniques.
  - **EAL 7:** It is the highest level of assurance, requiring formal verification of the product's design and security features.

Functional testing and structural verifications

+ security testing (software attacks)

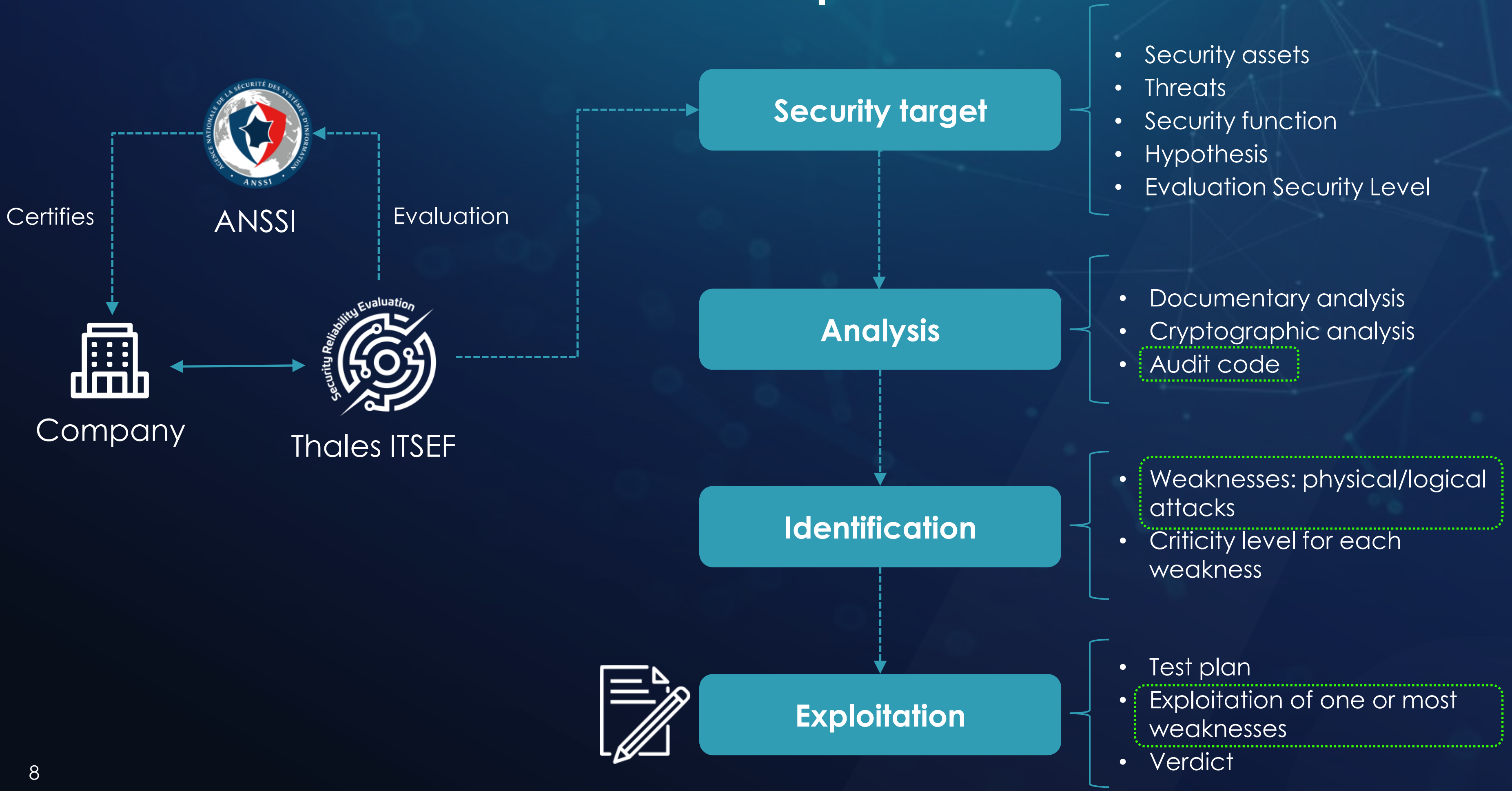
+ physical attacks

+ formal verifications

Who is responsible for making the evaluation process?

*https://www.commoncriteriaportal.org/index.cfm

Evaluation process



ANSSI

Certifies

Evaluation

Company

Thales ITSEF

Security target

- Security assets
- Threats
- Security function
- Hypothesis
- Evaluation Security Level

Analysis

- Documentary analysis
- Cryptographic analysis
- Audit code

Identification

- Weaknesses: physical/logical attacks
- Criticity level for each weakness

Exploitation

- Test plan
- Exploitation of one or most weaknesses
- Verdict

Physical attacks

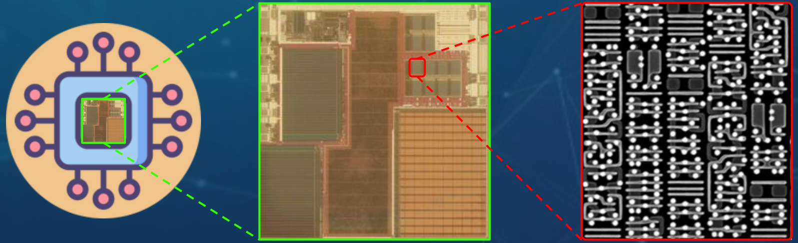
Physical attacks

Thales ITSEF

Evaluates

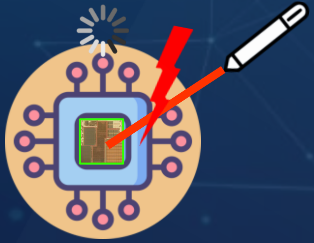
Today

Reverse ingineering / FIB probing routing



Invasive attacks

Temporarily / Permanently perturbations (fault attacks)



Semi-Invasive attacks

Side-channel attacks

Non-Invasive attacks

Side-channel Attacks

Side-channel attacks

Surveillance and Exploitation of an INVOLUNTARY leak from the system

- Related to the communication protocol
- Related to the implementation
- Related to the underlying hardware and physics

Common channels (not exhaustive)

- Errors returned [**SW**]
- Computation time (duration, cache access time, etc.) [**SW**]
- Instantaneous current consumption (transistor switching) [**HW**]
  - 1 -> 1 or 0 -> 0 consumes less than 1 -> 0 or 0 -> 1
- Electromagnetic radiation [**HW**]
- Photon emissions, temperature, noise, etc. [**HW**]

Objective

- Allows attacks on cryptographic algorithms (RSA, ECC, AES, DES, HASH...) to retrieve secret information (key, message)
- Allows attacks on AI embedded systems to retrieve NN architecture, weight value.
- Also helps to understand how the code works (reverse engineering)

Side-channel attacks

Historical example: Hagelin cipher machine

Context

- In 1956, Egyptian embassy used Hagelin cipher machine to protect communications.

Weakness

- The Hagelin machine used 7 wheels (or rotors), which were part of its cryptographic mechanism.
- These wheels needed proper positioning to ensure the encryption and decryption were accurate.
- The Hagelin machine required periodic resetting or reinitialization of their settings, including the positions of the wheels.

Strategy (MI5)

- Peter Wright (MI5, Scientific expert and counterintelligence officer) proposes installing a microphone in the encryption room.

Acoustic channel

- Acoustic cryptanalysis is a known technique where sounds produced by mechanical operations can reveal information about the machine's state.
- The noise emitted during the initialization allowed MI5 to determine the position of the wheels during encryption process.

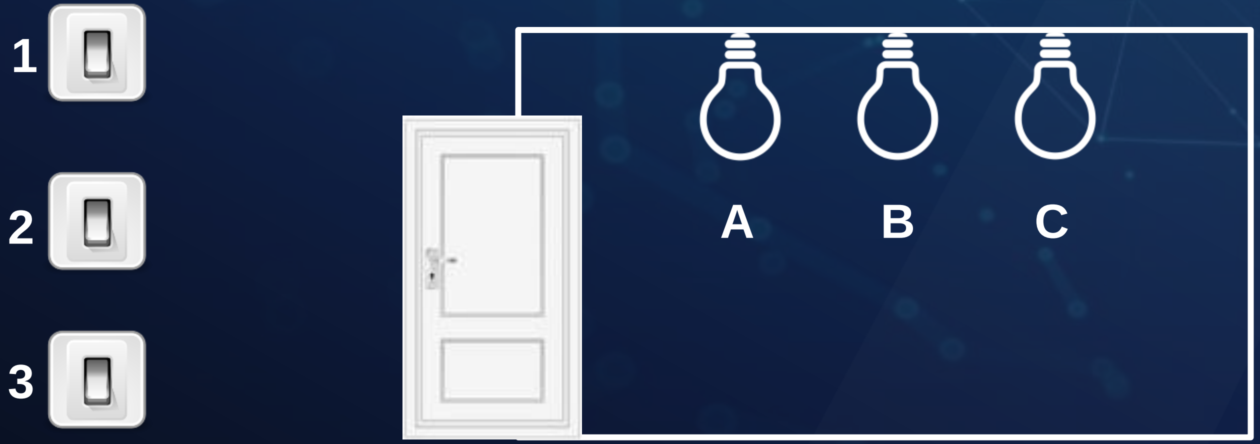
Result

- MI5 was be able to decrypt the secure communication of the Egyptian embassy.

Side-channel attacks

Short exercise: 3 switches, 3 light bulbs in another room

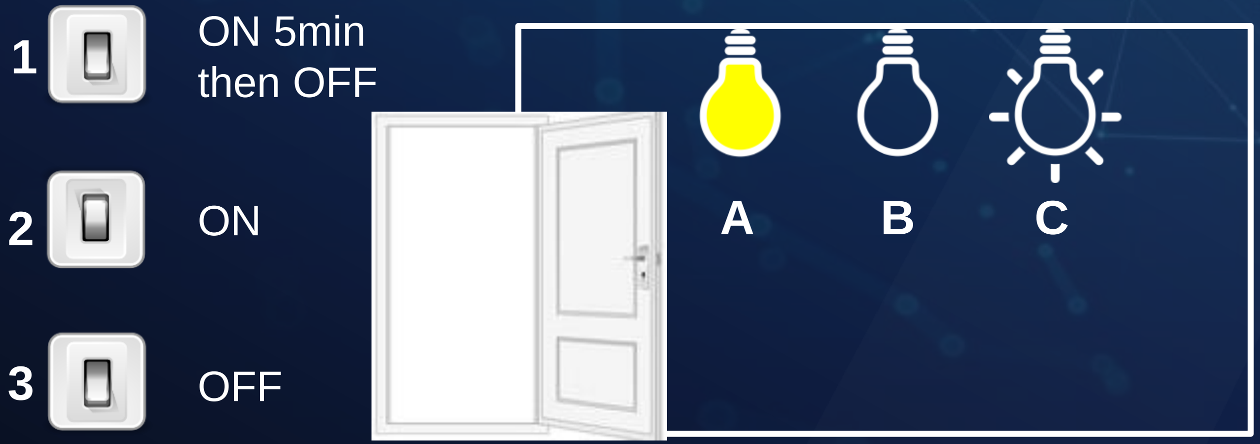
- Only one visit is possible
- Which switch turns on each light bulb?



Side-channel attacks

Short exercise: 3 switches, 3 light bulbs in another room

- Only one visit is possible
- Which switch turns on each light bulb?



1 ON 5min then OFF

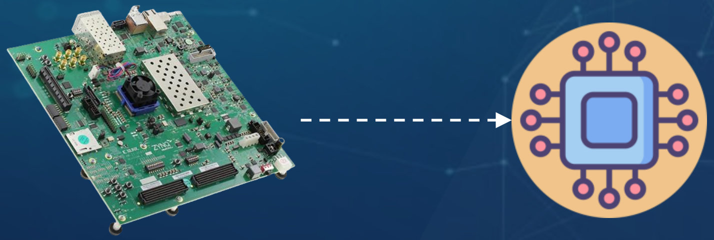
2 ON

3 OFF

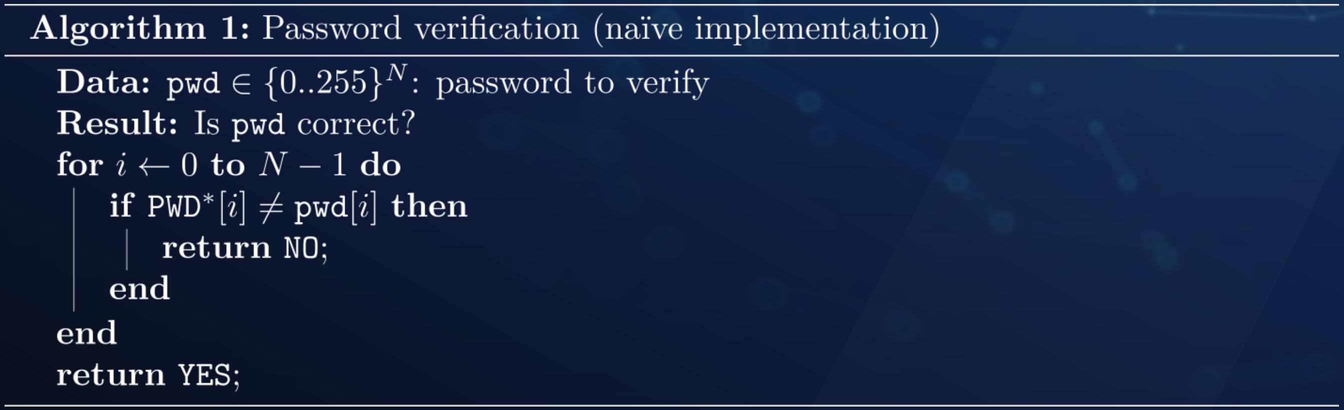
Side-channel attacks

Scenario of this talk

- **Target of Evaluation:** Embedded system
- **Security function:** Password verification
- **Security asset**: Password
- **Attack**: side-channel attack



Password verification process



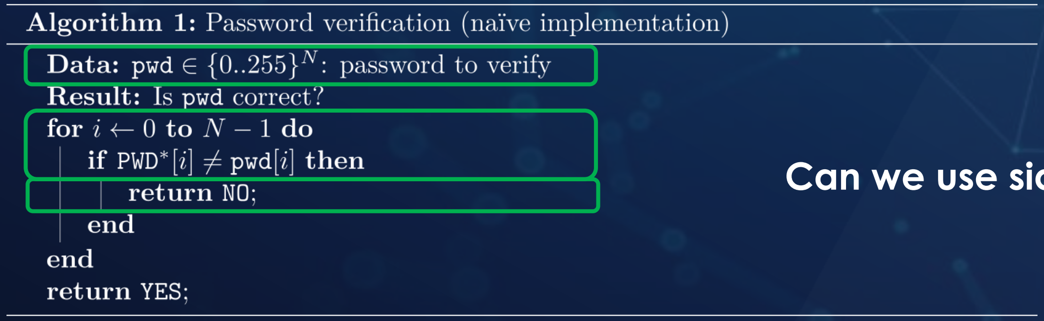
```
Algorithm 1: Password verification (naïve implementation)
Data: pwd $\in \{0..255\}^N$: password to verify
Result: Is pwd correct?
for $i \leftarrow 0$ to $N - 1$ do
    if PWD$^*[i] \neq$ pwd$[i]$ then
        return NO;
    end
end
return YES;
```

Side-channel attacks

Hypotheses

- No limit on the number of attempts
- PWD* is stored in memory
- Is brute-force possible?

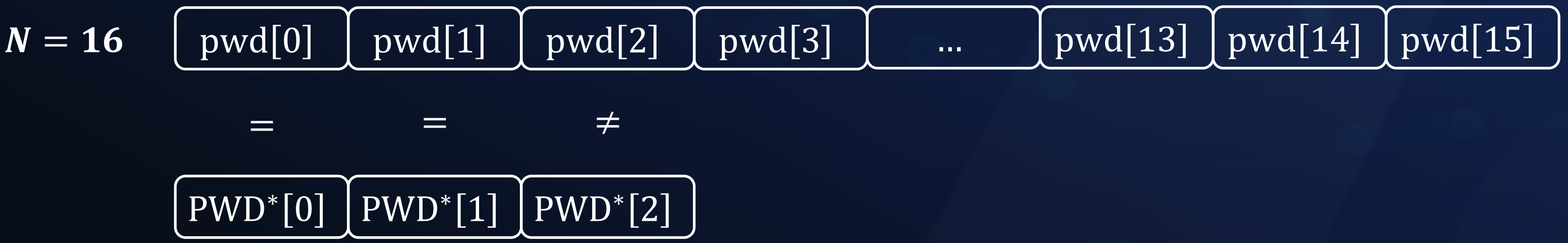
Password verification process



Algorithm 1: Password verification (naïve implementation)

```
Data: pwd ∈ $\{0..255\}^N$: password to verify
Result: Is pwd correct?
for $i \leftarrow 0$ to $N-1$ do
    if PWD*$[i] \neq$ pwd$[i]$ then
        return NO;
    end
end
return YES;
```

Can we use side-channel attacks?

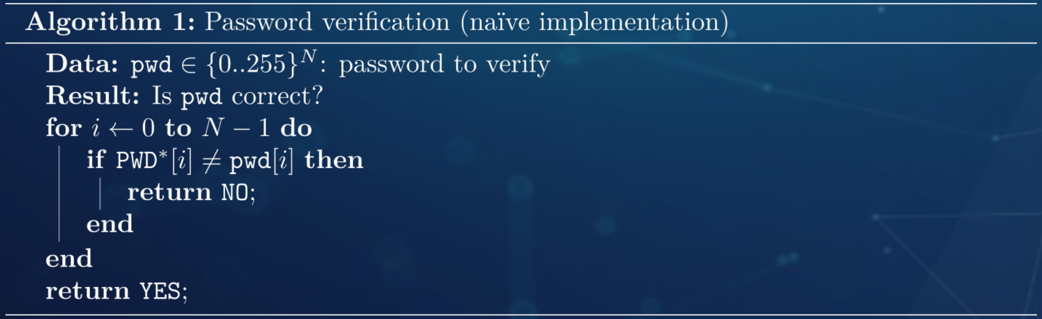


$N = 16$

pwd[0]	pwd[1]	pwd[2]	pwd[3]	...	pwd[13]	pwd[14]	pwd[15]
$=$	$=$	$\neq$					
PWD*[0]	PWD*[1]	PWD*[2]					

Side-channel attacks

Password verification process



```
Algorithm 1: Password verification (naïve implementation)
  Data: pwd $\in \{0..255\}^N$: password to verify
  Result: Is pwd correct?
  for $i \leftarrow 0$ to $N - 1$ do
      if PWD$^*[i] \neq$ pwd$[i]$ then
          return NO;
      end
  end
  return YES;
```

“Divide-and-conquer” strategy

- pwd is validated byte per byte
- Timing difference between the number of correctly verified bytes

In practice

- Assuming $PWD^*[i] \neq \mathrm{pwd}[i]$ operation takes $\approx 10\ ms$
- Number of requests = $2^8 \times 16 = 2^{12}$ instead of $(2^8)^{16}$ (brute-force)

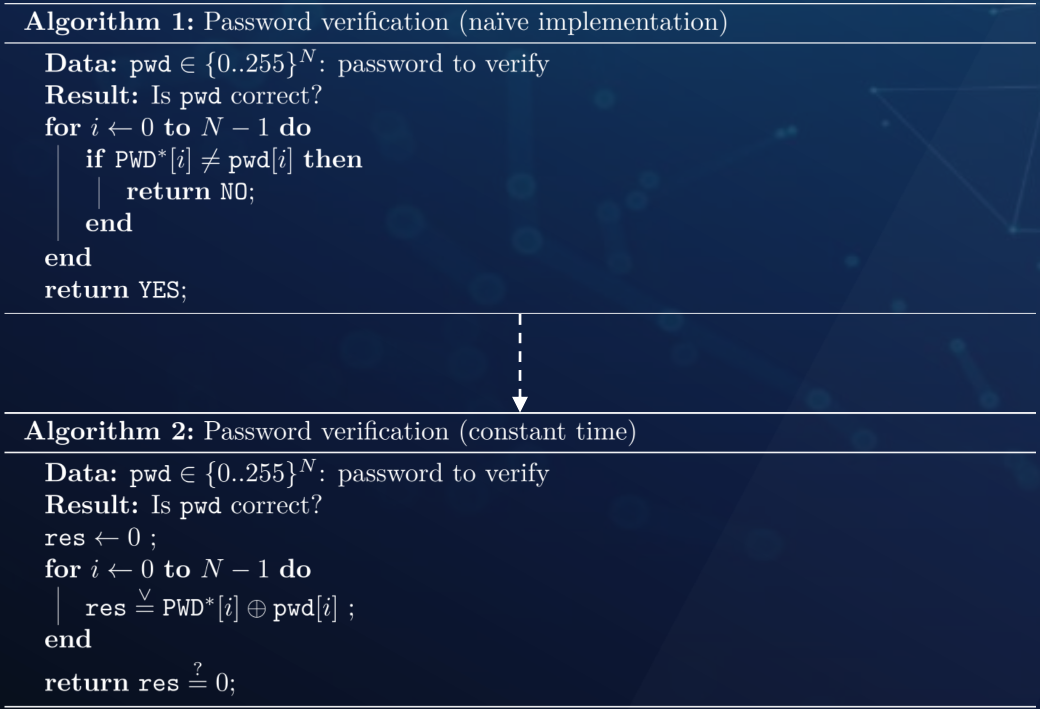
Countermeasure?

Side-channel attacks

Countermeasure?

- Developing an algorithm which is not time-dependent

Password verification process



```
Algorithm 1: Password verification (naïve implementation)
Data: $\texttt{pwd} \in \{0..255\}^N$: password to verify
Result: Is $\texttt{pwd}$ correct?
for $i \leftarrow 0$ to $N-1$ do
    if $\texttt{PWD}^*[i] \neq \texttt{pwd}[i]$ then
        return $\texttt{NO}$;
    end
end
return $\texttt{YES}$;
```

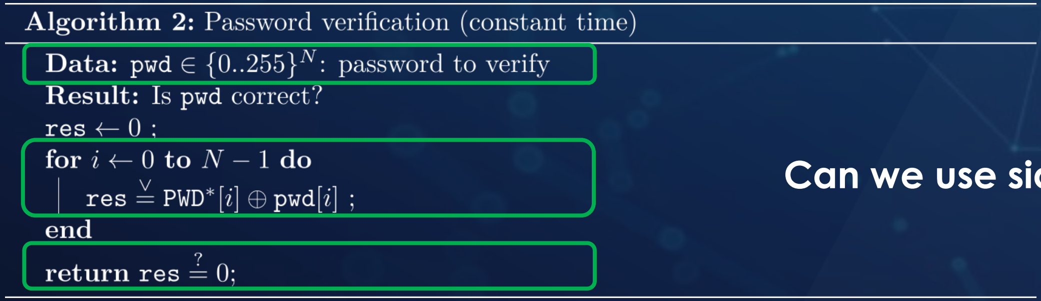
```
Algorithm 2: Password verification (constant time)
Data: $\texttt{pwd} \in \{0..255\}^N$: password to verify
Result: Is $\texttt{pwd}$ correct?
$\texttt{res} \leftarrow 0$ ;
for $i \leftarrow 0$ to $N-1$ do
    $\texttt{res} \overset{\vee}{=} \texttt{PWD}^*[i] \oplus \texttt{pwd}[i]$ ;
end
return $\texttt{res} \overset{?}{=} 0$;
```

Side-channel attacks

Hypothesis

- No limit on the number of attempts
- PWD is stored in memory
- Is brute-force possible?

Password verification process



```
Algorithm 2: Password verification (constant time)
Data: pwd $\in \{0..255\}^N$: password to verify
Result: Is pwd correct?
res $\leftarrow 0$ ;
for $i \leftarrow 0$ to $N - 1$ do
    res $\overset{\vee}{=}$ PWD$^*[i] \oplus$ pwd$[i]$ ;
end
return res $\overset{?}{=} 0$;
```

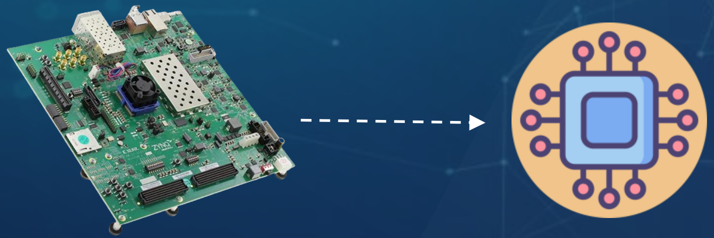
Can we use side-channel attacks?

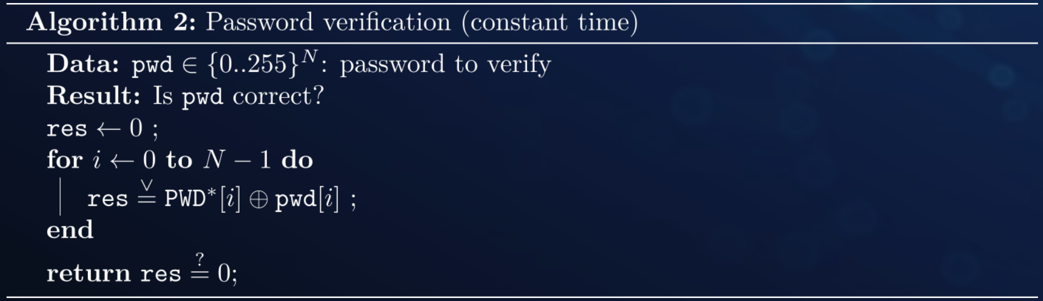
$N = 16$	pwd[0]	pwd[1]	pwd[2]	pwd[3]	...	pwd[13]	pwd[14]	pwd[15]
	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
	PWD*[0]	PWD*[1]	PWD*[2]	PWD*[3]	...	PWD*[13]	PWD*[14]	PWD*[15]

Side-channel attacks

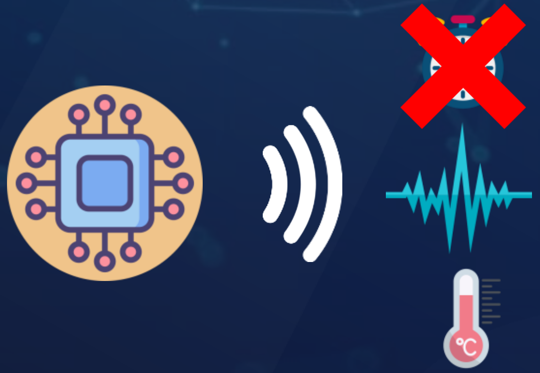
Scenario of this talk

- **Target of Evaluation:** Embedded system
- **Security function:** Password verification
- **Security asset**: Password
- **Attack**: side-channel attack





```
Algorithm 2: Password verification (constant time)
  Data: pwd ∈ $\{0..255\}^N$: password to verify
  Result: Is pwd correct?
  res ← 0 ;
  for $i \leftarrow 0$ to $N - 1$ do
      res $\overset{\vee}{=}$ PWD$^*[i] \oplus$ pwd$[i]$ ;
  end
  return res $\overset{?}{=} 0$;
```



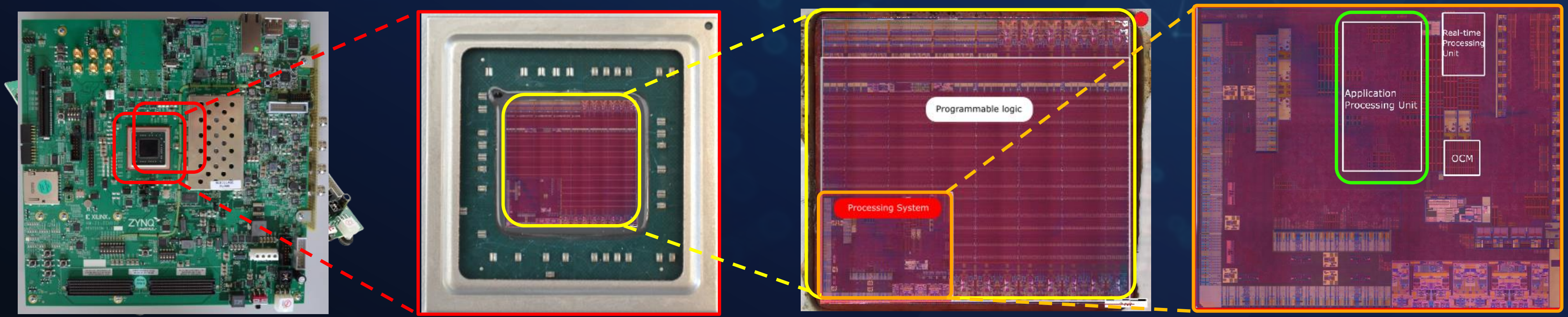
Side-channel attacks

Capture physical emanation

- **Source:** Electromagnetic signal
- **Equipment:** EM probe, oscilloscope, PC

Step 1

- **Physical identification**



The CPUs are here ☺

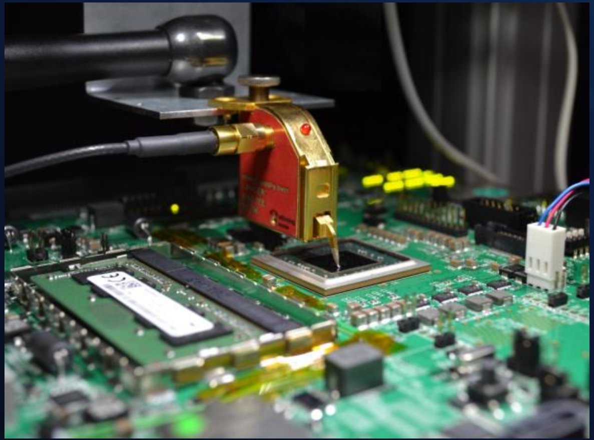
Side-channel attacks

Capture physical emanation

- **Source:** Electromagnetic signal
- **Equipment:** EM probe, oscilloscope, PC

Step 2

- **Setup preparation**



Side-channel attacks

Capture physical emanation

- **Source:** Electromagnetic signal
- **Equipment:** EM probe, oscilloscope, PC

Step 3

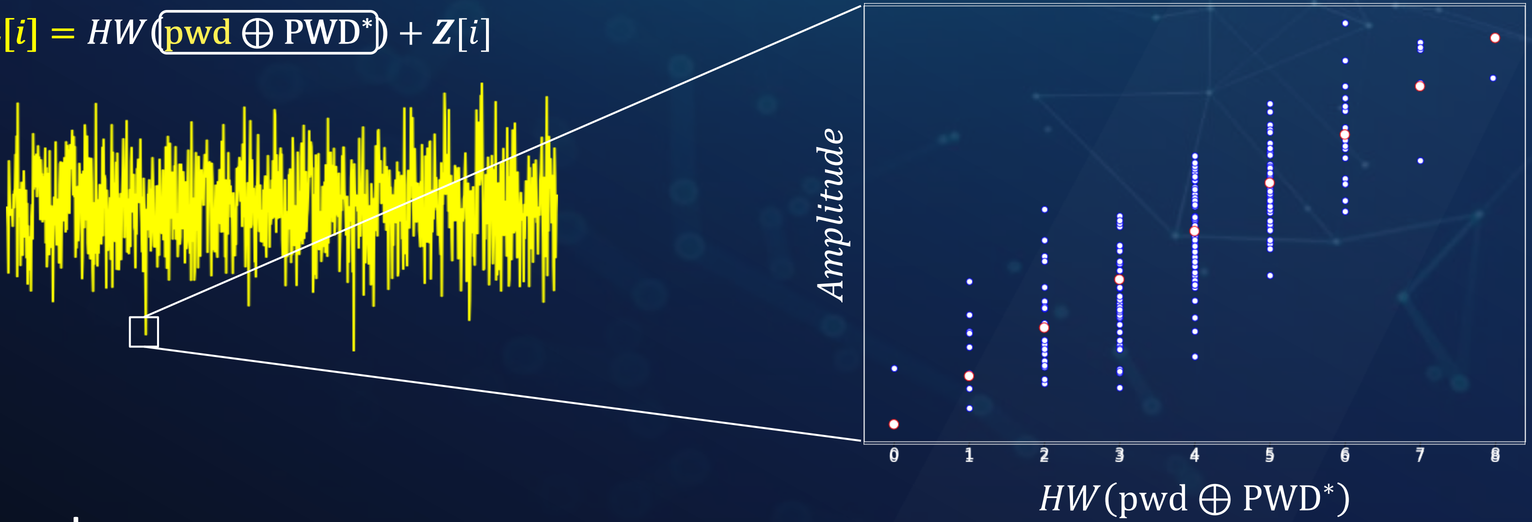
- **Signal acquisition**

Side-channel attacks

Leakage model ψ

- A leakage model is a function $\psi: \mathbb{F}_2^n \rightarrow \mathbb{R}$ which characterizes a dependency between a pair $(\mathrm{pwd}, \mathrm{PWD}^*)$ and the electromagnetic signal $\boldsymbol{L}$

$$\boldsymbol{L}[i] = HW\left(\boxed{\mathrm{pwd} \oplus \mathrm{PWD}^*}\right) + \boldsymbol{Z}[i]$$



Attacker's goal

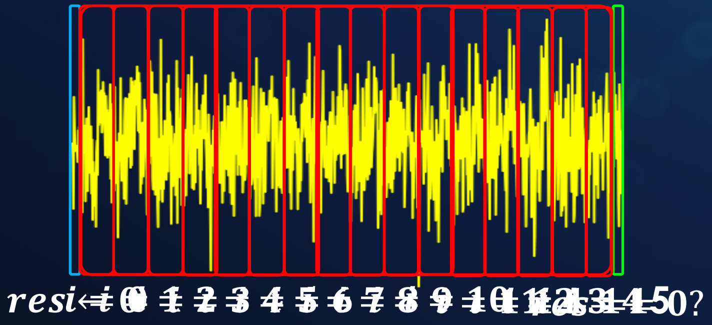
- Retrieving the dependency between pwd and PWD in a minimum amount of physical traces.

How can we identify such dependencies?

Side-channel attacks

Step 4: Signal analysis

- A signal characterizes the process conducted by the password verification



```
Algorithm 2: Password verification (constant time)
  Data: $\texttt{pwd} \in \{0..255\}^N$: password to verify
  Result: Is pwd correct?
  $\texttt{res} \leftarrow 0$ ;
  for $i \leftarrow 0$ to $N-1$ do
      $\texttt{res} \overset{\vee}{=} \texttt{PWD}^*[i] \oplus \texttt{pwd}[i]$ ;
  end
  return $\texttt{res} \overset{?}{=} 0$;
```

Next step?

- Which part of the signal is interesting for our study?
- Should we consider all the time samples?
- Do we need to restrict ourselves to a sub-portion of the signal?

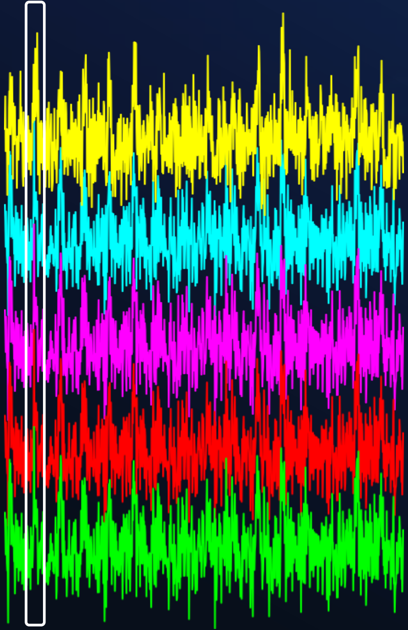
Side-channel attacks

Issues

- Lots of points in a physical signal are not dependent on the sensitive data.
- How can you identify them?

Step 5: Points of interest detection

$$\boldsymbol{SNR}[i] = \frac{\mathbb{V}_{\mathbf{pwd}}[\mathbb{E}[\boldsymbol{L}[i]\ |HW(\mathrm{pwd} \oplus \mathrm{PWD}^*)]}{\mathbb{E}_{\mathrm{pwd}}[\mathbb{V}[\boldsymbol{L}[i]]|HW(\mathrm{pwd} \oplus \mathrm{PWD}^*)]} = \frac{\mathbb{V}_{\mathbf{pwd}}[HW(\mathrm{pwd} \oplus \mathrm{PWD}^*)]}{\mathbb{V}[\boldsymbol{Z}[i]]}$$



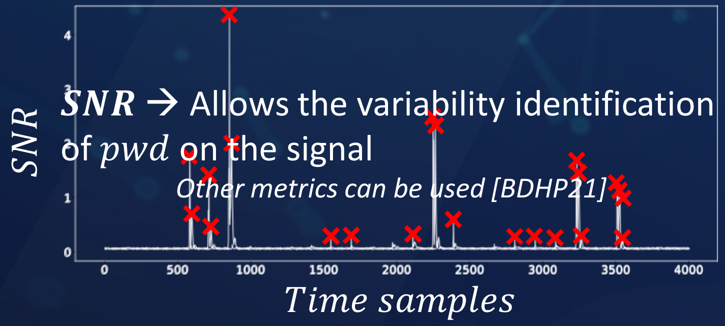
$$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \mathbf{Z}$$

$$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \mathbf{Z}$$

$$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \mathbf{Z} \ \sim \mathcal{N}_D(0, \Sigma_D)$$

$$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \mathbf{Z}$$

$$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \mathbf{Z}$$



$\boldsymbol{SNR}$ → Allows the variability identification of pwd on the signal

Other metrics can be used [BDHP21]

[BDHP21] It Started with Templates: The Future of Profiling in Side-Channel Analysis. Batina, L., et al. (2021). In: Avoine, G., Hernandez-Castro, J. (eds) Security of Ubiquitous Computing Systems.

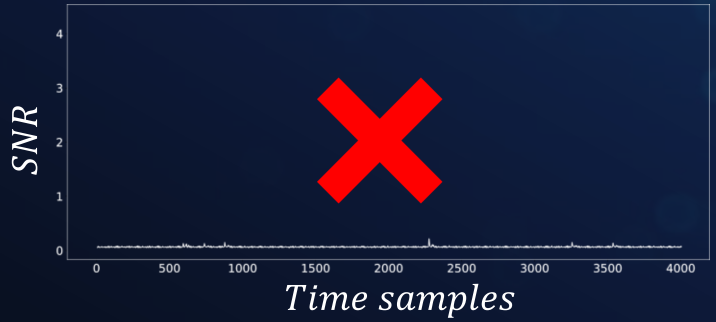
Side-channel attacks

Issues

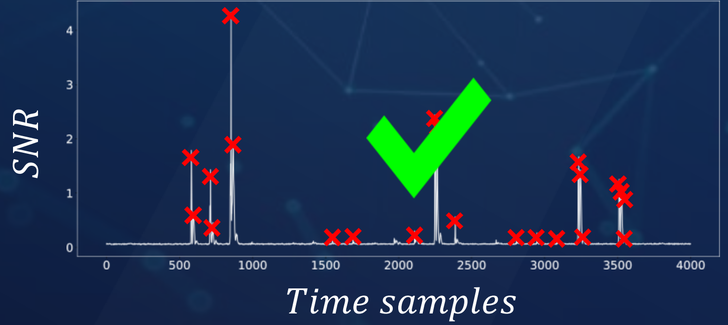
- Lots of points in a physical signal are not dependent on the sensitive data.
- How can you identify them?

Step 5: Points of interest detection

No depency with pwd



Dependencies with pwd



Next step?

- How can we exploit these points of interest (POIs)?

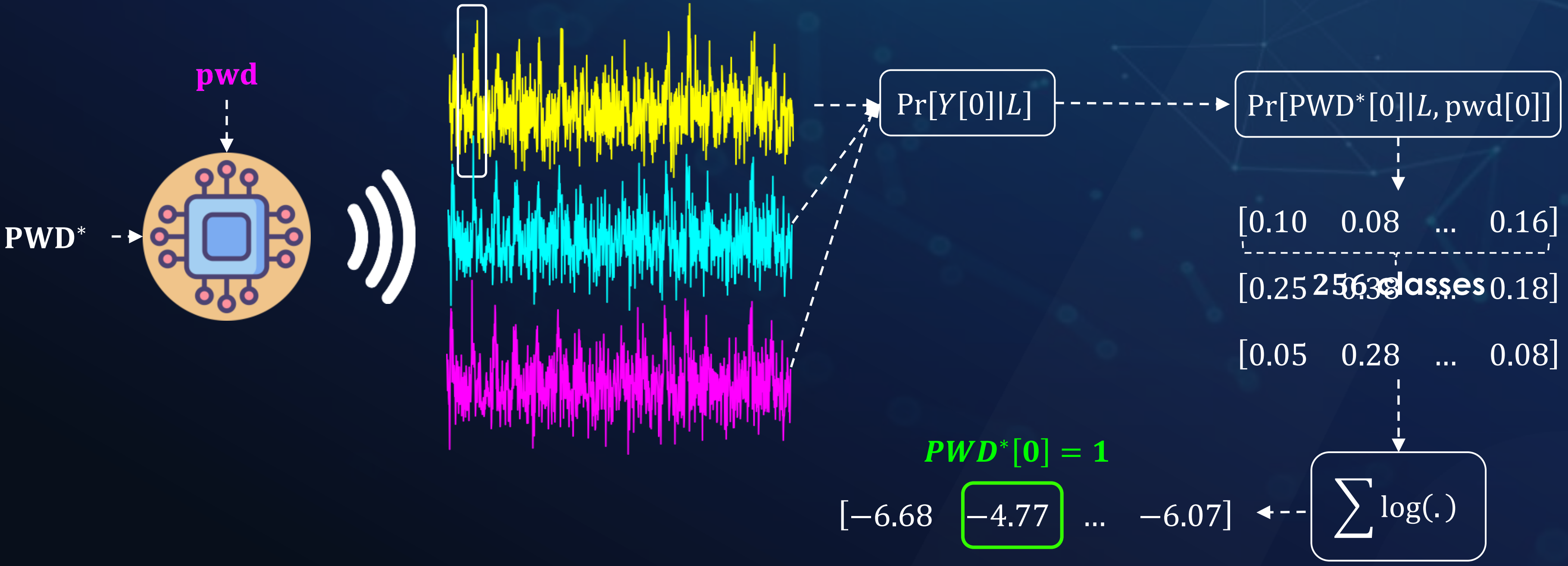
Statistical distinguishers

Statistical distinguishers

Optimal attack: Maximum likelihood [HRG14]

- Goal: retrieving information on PWD*
- Let denotes $Y[i] = \mathrm{pwd}[i] \oplus \mathrm{PWD}^*[i]$

$$\boldsymbol{L} = HW(Y[0]) + \mathbf{Z}$$



$\Pr[Y[0] | L]$

$\Pr[\mathrm{PWD}^*[0] | L, \mathrm{pwd}[0]]$

$$\underbrace{[0.10 \quad 0.08 \quad \dots \quad 0.16]}_{\textbf{256 classes}}$$

$$[0.25 \quad 0.38 \quad \dots \quad 0.18]$$

$$[0.05 \quad 0.28 \quad \dots \quad 0.08]$$

$\sum \log(.)$

$$[-6.68 \quad \boxed{-4.77} \quad \dots \quad -6.07]$$

$\boldsymbol{PWD^*[0] = 1}$

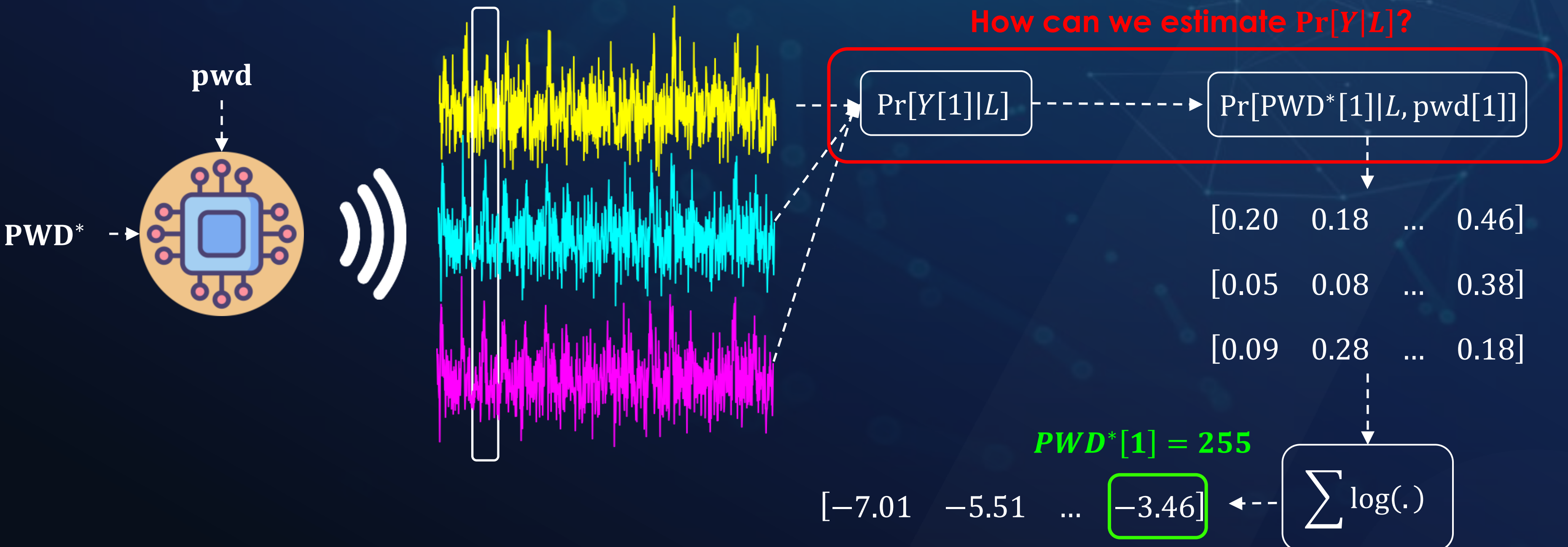
[HRG14] Good Is Not Good Enough – Deriving Optimal Distinguishers from Communication Theory. Heuser, A., et al. CHES 2014.

Statistical distinguishers

Optimal attack: Maximum likelihood [HRG14]

- Goal: retrieving information on PWD^*
- Let denotes $Y[i] = \mathrm{pwd}[i] \oplus \mathrm{PWD}^*[i]$

$$\boldsymbol{L} = HW(Y[1]) + \mathbf{Z}$$



How can we estimate $\Pr[Y|L]$?

$\mathbf{pwd}$

$\mathbf{PWD}^*$

$\Pr[Y[1]|L]$

$\Pr[\mathrm{PWD}^*[1]|L, \mathrm{pwd}[1]]$

$[0.20 \quad 0.18 \quad \dots \quad 0.46]$

$[0.05 \quad 0.08 \quad \dots \quad 0.38]$

$[0.09 \quad 0.28 \quad \dots \quad 0.18]$

$\sum \log(.)$

$\boldsymbol{PWD^*[1] = 255}$

$[-7.01 \quad -5.51 \quad \dots \quad \boxed{-3.46}]$

- **Success Rate**: Probability to succeed an attack within N_a attack traces

[HRG14] Good Is Not Good Enough – Deriving Optimal Distinguishers from Communication Theory. Heuser, A., et al. CHES 2014.

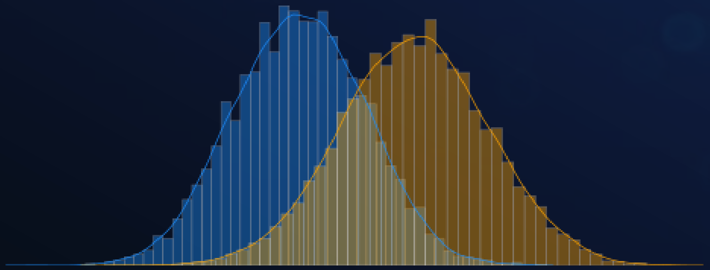
Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Existing solutions

Generative model



Discriminative model

Statistical distinguishers

Estimation of $\Pr[Y|L]$

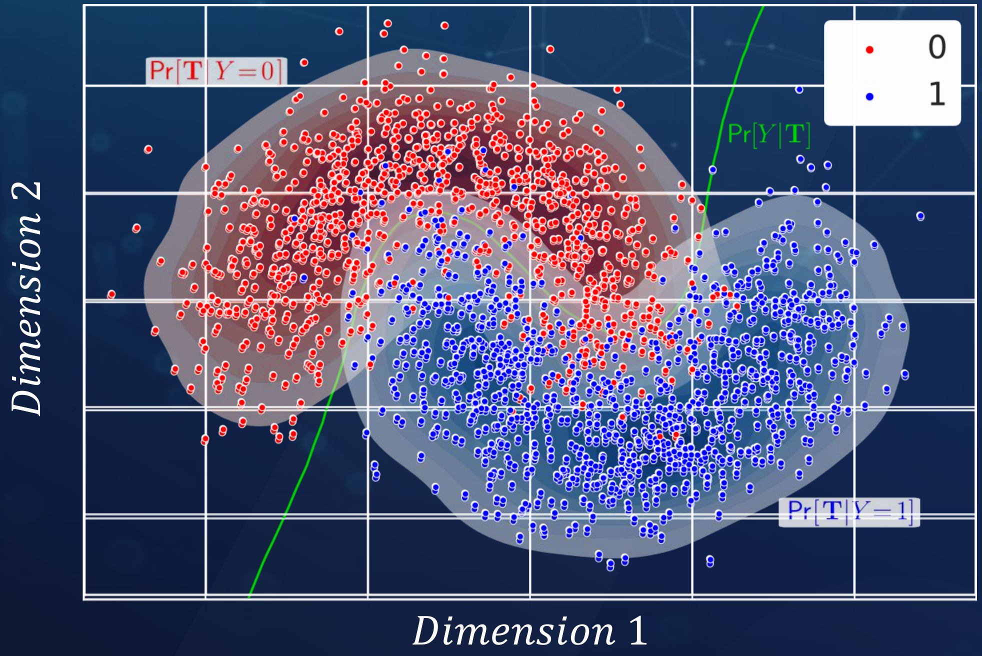
- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative approach

- Estimation of $\Pr[L|Y]$ to then deduce $\Pr[Y|L]$ (Bayes' theorem)
- Historical side-channel attacks [CRR03, SLP05]

Discriminative approach

- Estimation of $\Pr[Y|L]$ via the approximation of a decision boundary
- Approach using AI [MPP16, CDP17, MDP20]



[CRR03] Template attacks. Chari, S. *et al. CHES 2003*.
[SLP05] A stochastic model for differential side channel cryptanalysis. Schindler, W. et al. *CHES 2005*.
[MPP16] Breaking cryptographic implementations using deep learning techniques. Maghrebi, H. et al. *SPACE 2016*.
[CDP17] Convolutional neural networks with data augmentation against jitter-based countermeasures - profiling attacks without pre-processing. Cagli, E. *et al. CHES 2017*.
[MDP20] A comprehensive study of deep learning for side-channel analysis. Masure, L. et al. *TCHES 2020*.

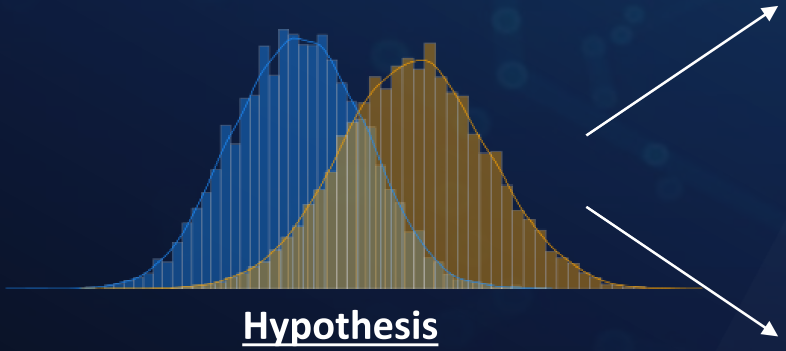
Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\Pr[L|Y]$



Template attack [CRR03]

Stochastic attack [SLP05]

[CRR03] Template attacks. Chari, S. *et al. CHES 2003*.
[SLP05] A stochastic model for differential side channel cryptanalysis. Schindler, W. et al. *CHES 2005*.

Statistical distinguishers

Estimation of $\Pr[Y|L]$

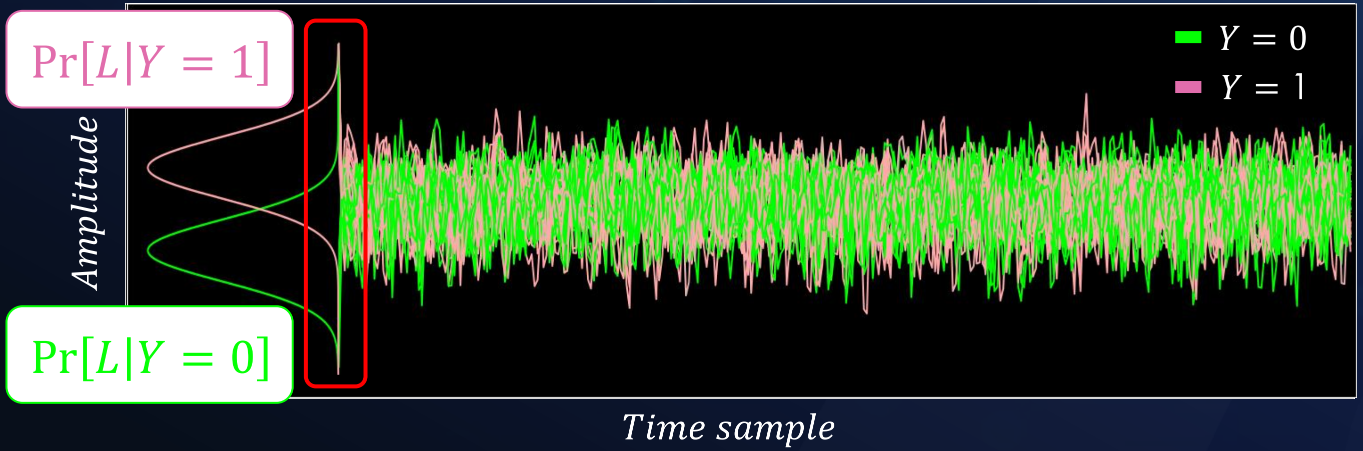
- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\Pr[L|Y]$

Toy example

- Goal: Approximating $\Pr[L|Y]$
- Hypothesis: $Y \in \{0,1\}$



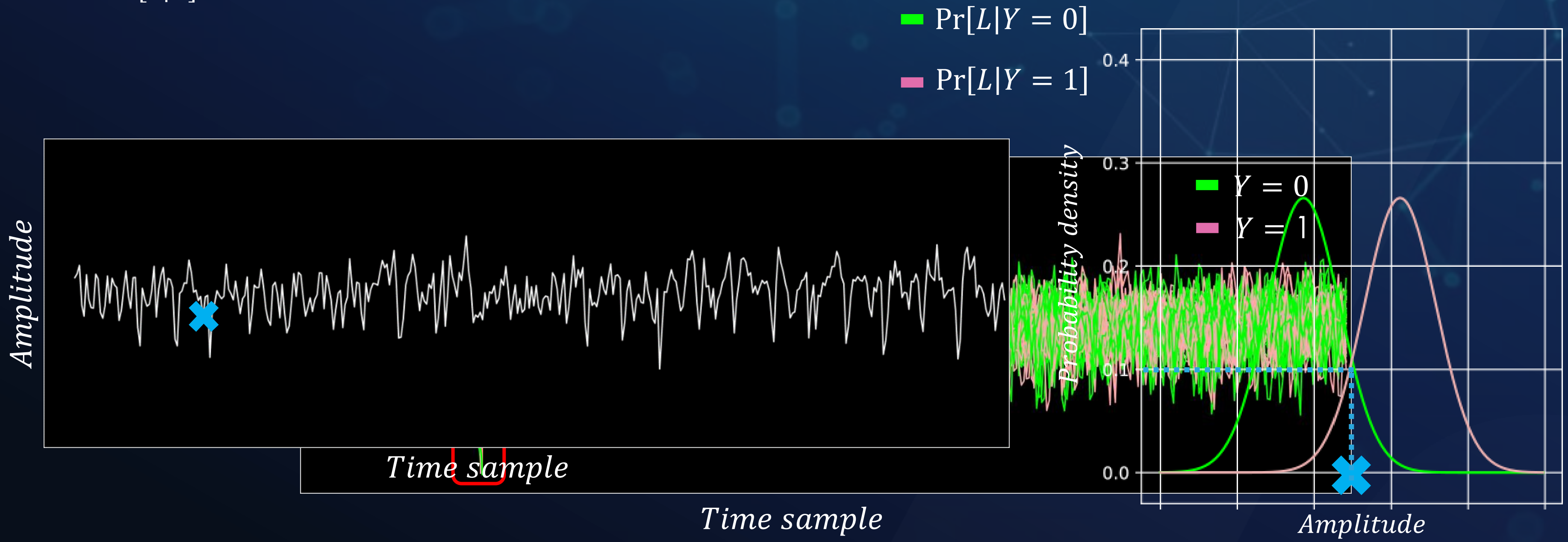
Statistical distinguishers

Estimation of $\mathbf{Pr}[Y|L]$

- Problem: $\mathbf{Pr}[Y|L]$ is unknown and device-dependent

Generative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\Pr[L|Y]$



Statistical distinguishers

Estimation of $\mathrm{Pr}[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\mathrm{Pr}[L|Y]$

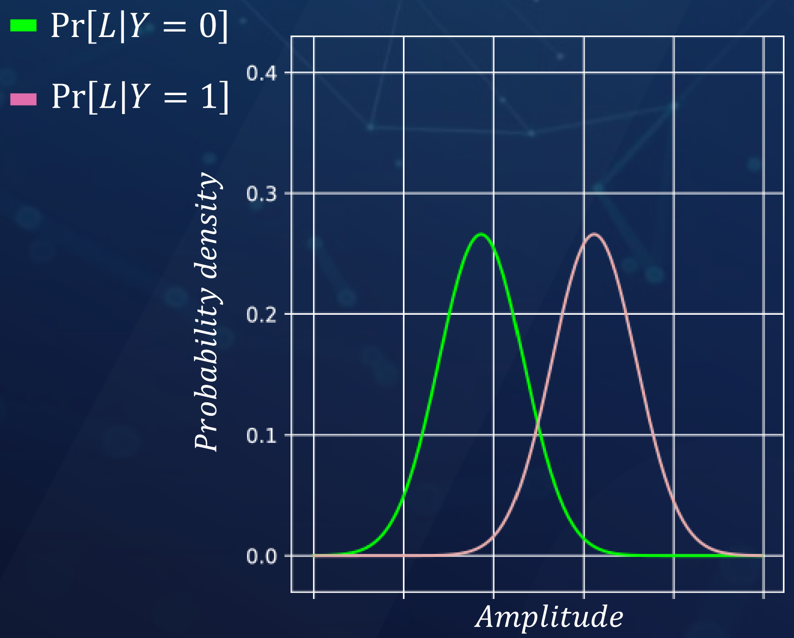
Steps

1) Acquire a set of N traces such that Y is unknown

2) For each trace l_i, we calculate $Pr[l_i|Y = 0]$ and $Pr[l_i|Y = 1]$

3) We compute the Maximum likelihood:

$$\hat{Y} = \underset{k\in\{0,1\}}{\operatorname{argmax}}\left(\sum_{i=0}^{N-1} log(\mathrm{Pr}[l_i|Y = k])\right)$$



Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\Pr[L|Y]$

Benefits

✓ Information on the exploited POIs

✓ Confident in the modeling of $\Pr[L|Y]$

✓ Multiple POIs can be exploited simultaneously

Limitations

✗ Strong hypothesis on the leakage model (Gaussian hypothesis)

✗ The success of attack performances depends on the POIs selection

How can we automate the process?

Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Generative

Discriminative

$$\Pr[L|Y] \xrightarrow{\text{Bayes' theorem}} \Pr[Y|L] = \Pr[L|Y] \times \boxed{\frac{\Pr[\boxed{Y}]}{\Pr[L]}}$$

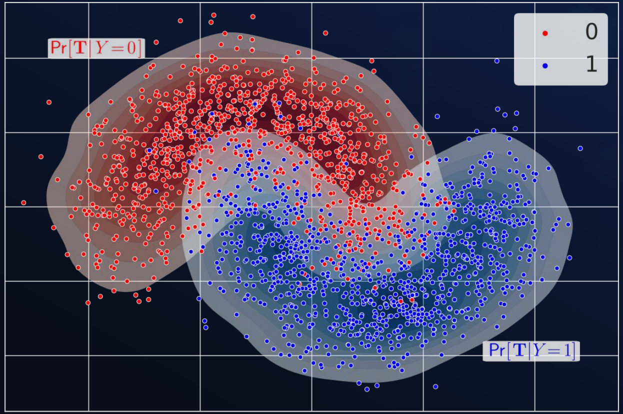
$$\Pr[Y|L] = c \cdot \Pr[L|Y]$$

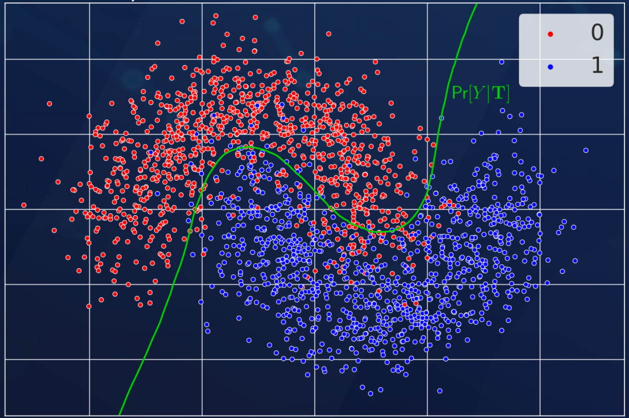
Uniformly distributed

Constant

Maximum likelihood

$$\hat{Y} = \underset{k\in\{0,1\}}{argmax}\left(\sum_{i=0}^{N-1} log(\Pr[l_i|Y=k])\right)$$





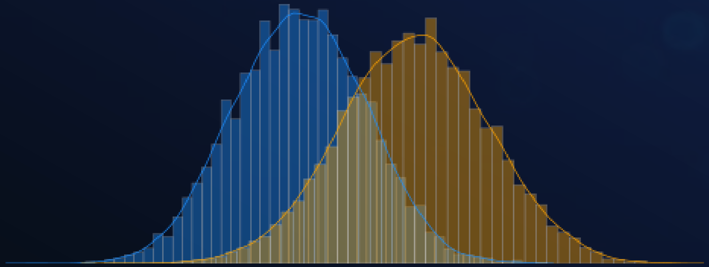
Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[Y|L]$ is unknown and device-dependent

Existing solutions

Generative model



Discriminative model

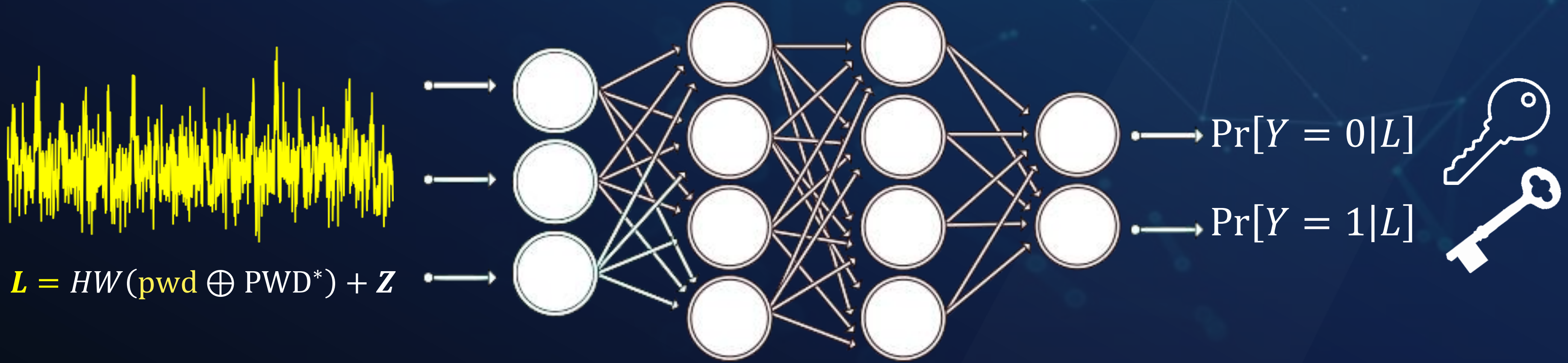
Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Discriminative approach

- Given a trace L to which it must associate a sensitive variable Y, the discriminative methods approximate the conditional probability $\Pr[Y|L]$



$\boldsymbol{L} = HW(\mathrm{pwd} \oplus \mathrm{PWD}^*) + \boldsymbol{Z}$
$\Pr[Y = 0|L]$
$\Pr[Y = 1|L]$

Training process

- Use of loss function to minimize with gradient descent (e.g. Negative Log-Likelihood)
- Not detailed in this talk (see [MDP20])

[MDP20] A comprehensive study of deep learning for side-channel analysis. Masure, L. et al. *TCHES 2020*.

Statistical distinguishers

Estimation of $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$

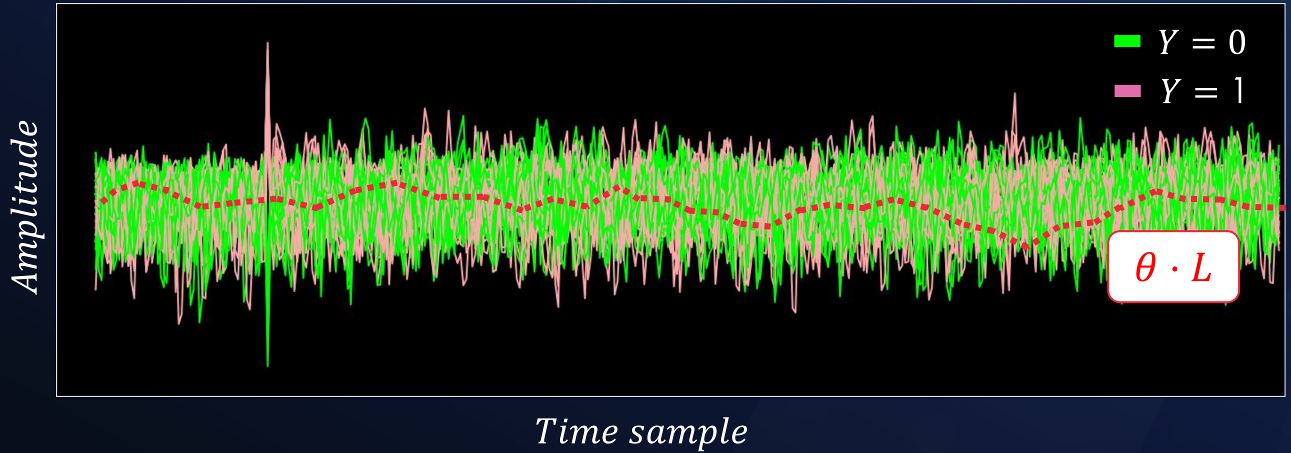
- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Discriminative approach

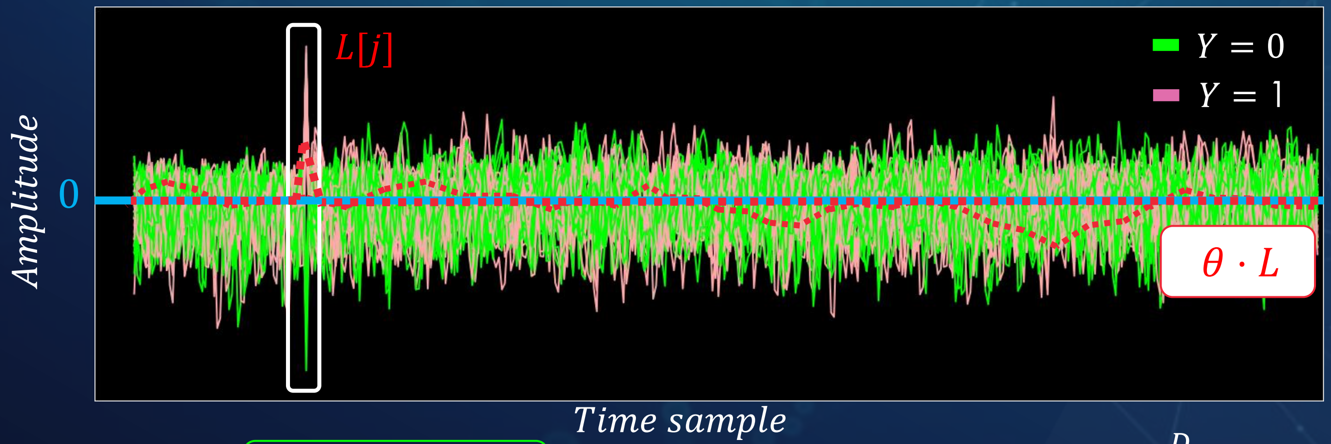
- Given a trace L to which it must associate a sensitive variable Y, the discriminative methods approximate the conditional probability $\Pr[Y|L]$

Toy example

- Goal: Approximating $\Pr[Y|L]$
- Hypothesis: $Y \in \{0,1\}$



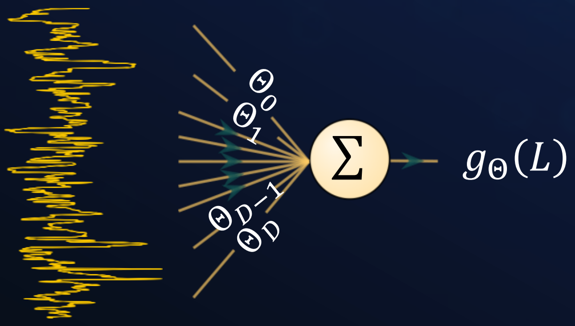
Statistical distinguishers

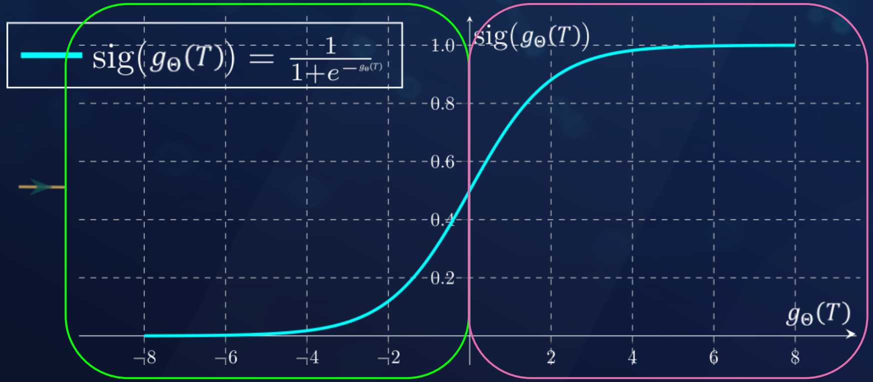


$$f_\Theta(L) = \begin{cases} 0 \text{ si } g_\Theta(L) < 0 \\ 1 \text{ si } g_\Theta(L) \geq 0 \end{cases}$$

such that

$$\boldsymbol{g_\Theta(L) = b +} \Theta \sum_{i=0}^{D} \cdot \Theta[j] L[i]$$





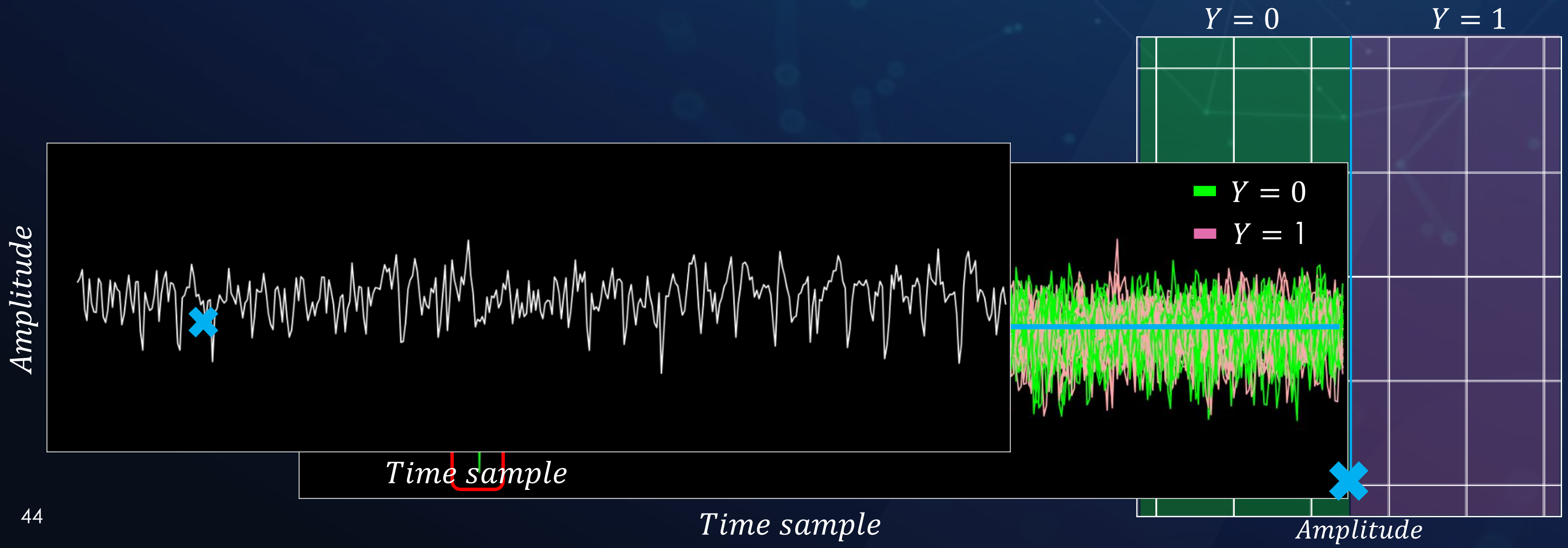
Statistical distinguishers

Estimation of $\mathbf{Pr}[Y|L]$

- Problem: $\mathbf{Pr}[Y|L]$ is unknown and device-dependent

Discriminative approach

- Given a trace L to which it must associate a sensitive variable Y, the discriminative methods approximate the conditional probability $\Pr[Y|L]$



Statistical distinguishers

Estimation of $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

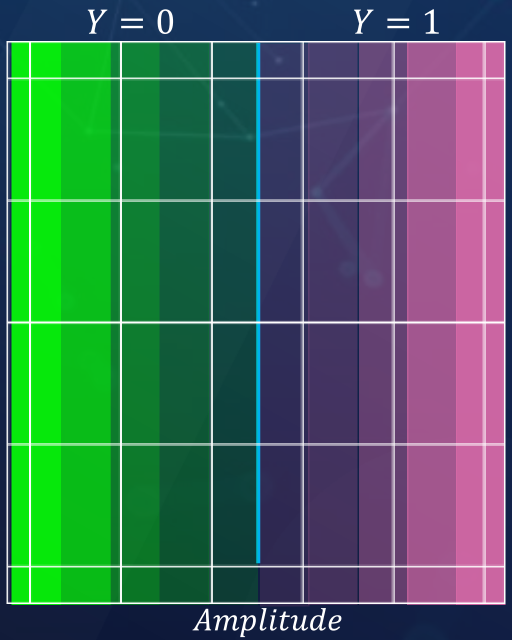
Discriminative approach

- Given a trace L to which it must associate a sensitive variable Y, the discriminative methods approximate the conditional probability $\Pr[Y|L]$

Steps

1) Acquire a set of N traces such that Y is unknown
2) For each trace l_i, we calculate $Pr[Y = 0|l_i]$ and $Pr[Y = 1|l_i]$
3) We compute the Maximum likelihood:

$$\hat{Y} = \underset{k\in\{0,1\}}{\operatorname{argmax}}\left(\sum_{i=0}^{N-1} log(\Pr[Y = k|l_i])\right)$$



Statistical distinguishers

Estimation of $\Pr[Y|L]$

- Problem: $\mathbf{Pr}[\boldsymbol{Y}|\boldsymbol{L}]$ is unknown and device-dependent

Discriminative approach

- Given a trace L to which it must associate a sensitive variable Y, the generative methods approximate the conditional probability $\Pr[L|Y]$

Benefits

✓ No hypothesis on the leakage model

✓ All the tasks (e.g. POIs selection) are automatized

✓ Multiple POIs can be exploited simultaneously

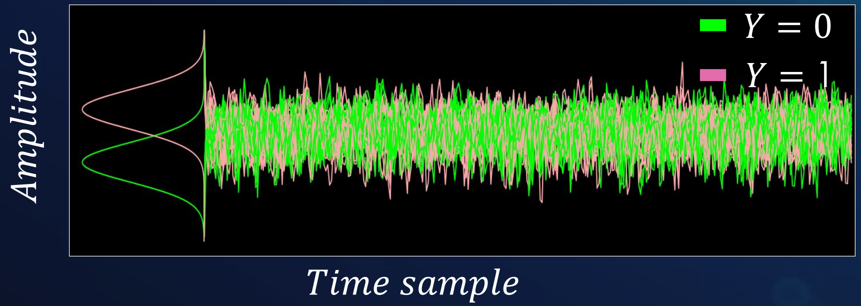
Limitations

✗ Neural networks can be seen as black-box tools

✗ Construction of adequate statistical model

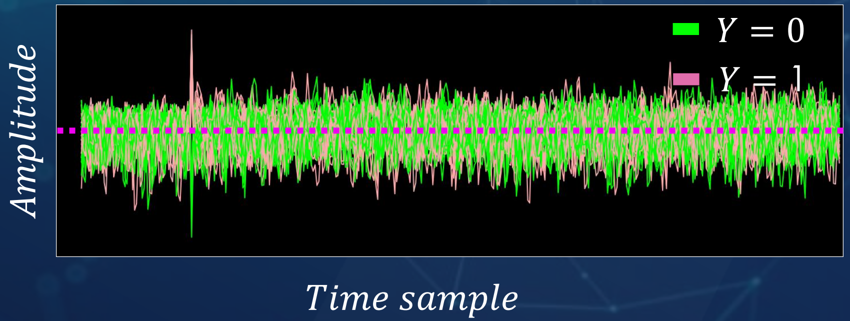
Statistical distinguishers

Generative



- ✗ Strong hypothesis on the leakage model (Gaussian hypothesis)
- ✗ The success of attack performances depends on the POIs selection
- ✓ Interpretability & Explainability
- ✓ Confident in the modeling of $\Pr[L|Y]$

Discriminative

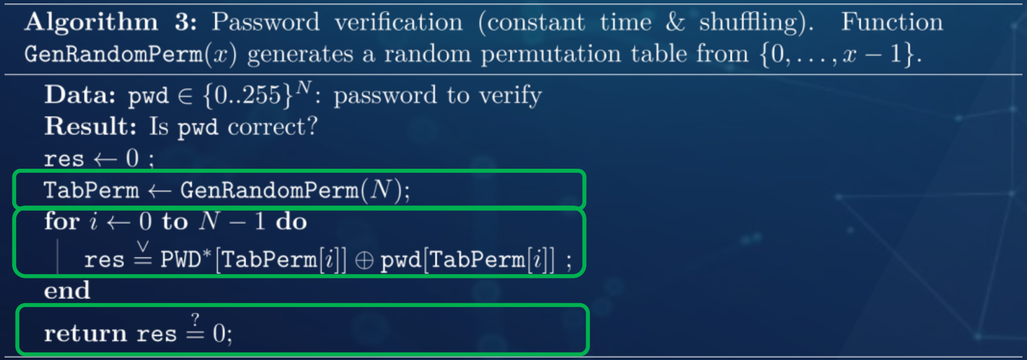


- ✓ Performance
- ✓ All the tasks (e.g. POIs selection) are automatized
- ✗ Neural networks can be seen as black-box tools
- ✗ Construction of adequate statistical model

Countermeasures

Countermeasures

Desynchronization



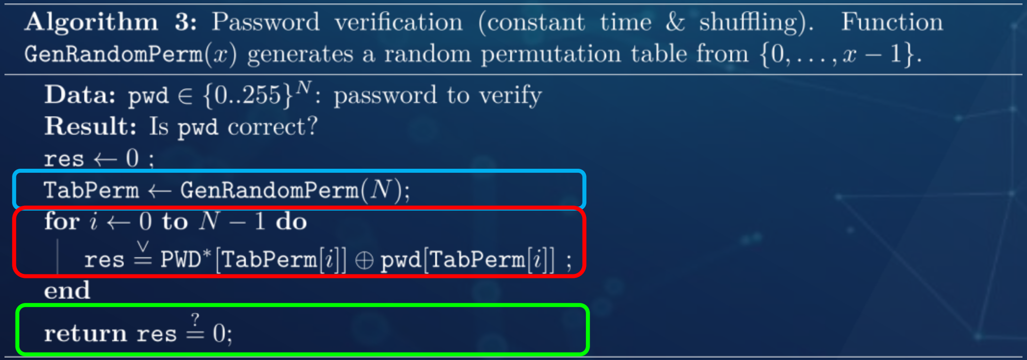
Algorithm 3: Password verification (constant time & shuffling). Function GenRandomPerm(x) generates a random permutation table from $\{0, \dots, x-1\}$.

```
Data: pwd $\in \{0..255\}^N$: password to verify
Result: Is pwd correct?
res $\leftarrow 0$ ;
TabPerm $\leftarrow$ GenRandomPerm($N$);
for $i \leftarrow 0$ to $N-1$ do
    res $\overset{\vee}{=}$ PWD$^*$[TabPerm[$i$]] $\oplus$ pwd[TabPerm[$i$]] ;
end
return res $\overset{?}{=}$ 0;
```

$N = 16$	5	8	10	0	...	1	7	3
	pwd[5]	pwd[8]	pwd[10]	pwd[0]	...	pwd[1]	pwd[7]	pwd[3]
	⊕	⊕	⊕	⊕	⊕	⊕	⊕	⊕
	PWD*[5]	PWD*[8]	PWD*[10]	PWD*[0]	...	PWD*[1]	PWD*[7]	PWD*[3]

Countermeasures

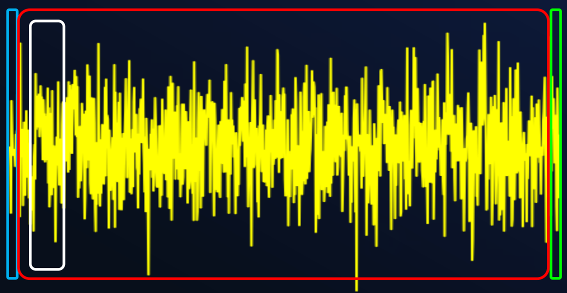
Desynchronization



Algorithm 3: Password verification (constant time & shuffling). Function GenRandomPerm(x) generates a random permutation table from $\{0, \dots, x-1\}$.

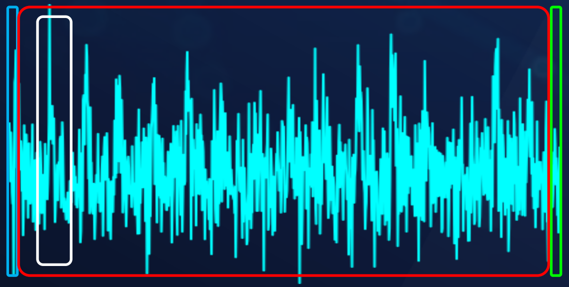
```
Data: pwd $\in \{0..255\}^N$: password to verify
Result: Is pwd correct?
res $\leftarrow 0$ ;
TabPerm $\leftarrow$ GenRandomPerm($N$);
for $i \leftarrow 0$ to $N-1$ do
    res $\overset{\vee}{=}$ PWD$^*$[TabPerm[$i$]] $\oplus$ pwd[TabPerm[$i$]] ;
end
return res $\overset{?}{=} 0$;
```

Query 1



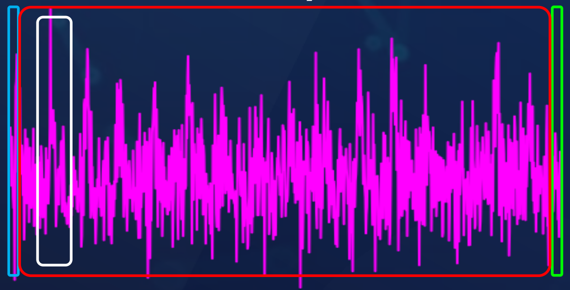
$HW(\text{pwd}[5] \oplus \text{PWD}^*[5]) + \mathbf{Z}$

Query 2



$HW(\text{pwd}[2] \oplus \text{PWD}^*[2]) + \mathbf{Z}$

Query 3



$HW(\text{pwd}[14] \oplus \text{PWD}^*[14]) + \mathbf{Z}$

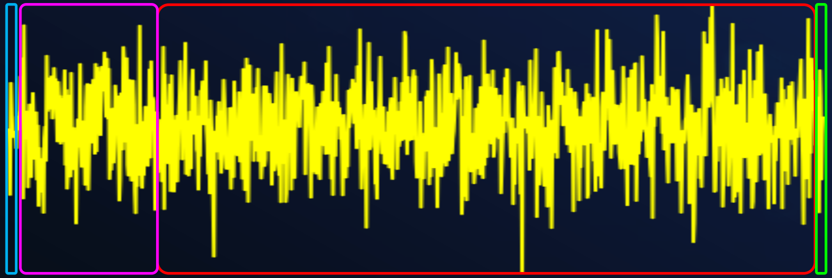
Countermeasures

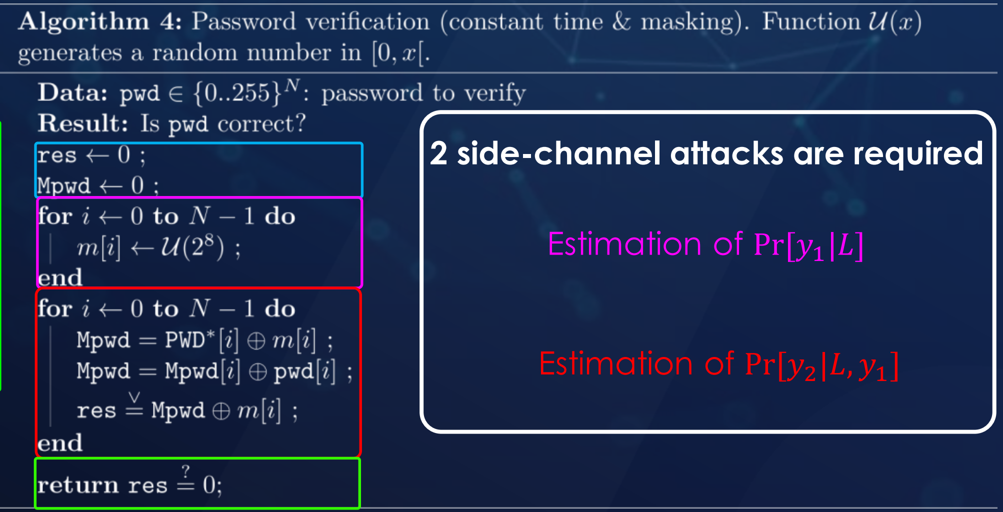
Masking

- Decomposition of a sensitive variable y into $(y_1, y_2, \dots y_n)$ such that $y_1, y_2, \dots, y_{n-1} \leftarrow \left(\mathcal{U}(2^8)\right)^n$ and $y_n \leftarrow y - (y_1 + y_2 + \dots + y_n)$

Example

- m is a N-byte random vector such that $y_1 = m$ and $y_2 = y \oplus m = \mathrm{pwd} \oplus \mathrm{PWD}^* \oplus m$
- m is refresh for each query





Algorithm 4: Password verification (constant time & masking). Function $\mathcal{U}(x)$ generates a random number in $[0, x[$.

```
Data: pwd ∈ $\{0..255\}^N$: password to verify
Result: Is pwd correct?
res ← 0 ;
Mpwd ← 0 ;
for $i \leftarrow 0$ to $N-1$ do
    $m[i] \leftarrow \mathcal{U}(2^8)$ ;
end
for $i \leftarrow 0$ to $N-1$ do
    Mpwd = PWD*[$i$] ⊕ $m[i]$ ;
    Mpwd = Mpwd[$i$] ⊕ pwd[$i$] ;
    res $\overset{\vee}{=}$ Mpwd ⊕ $m[i]$ ;
end
return res $\overset{?}{=}$ 0;
```

2 side-channel attacks are required

Estimation of $\Pr[y_1|L]$

Estimation of $\Pr[y_2|L, y_1]$

Personal recommendations

Machine-Learning

Online courses

- Andrew NG's course (Coursera): Machine Learning by Stanford University | Coursera, Deep Learning by deeplearning.ai | Coursera

Books

- **Shai Shalev-Shwartz and Shai Ben-David. Understanding Machine Learning: From Theory to Algorithms.**
- **Christopher M. Bishop and Hugh Bishop. Deep Learning: Foundations and Concepts.**

Open source Librairies:

- **Tensorflow, PyTorch**

International conferences

- NeurIPS, ICML, ECML-PKDD, CVPR, …

Side-channel attacks

Online courses

- Amir Moradi's course: https://www.youtube.com/@AmirMoradi_impsec/playlists

Books

- Embedded Cryptography 1 | Wiley
- Embedded Cryptography 2 | Wiley
- Embedded Cryptography 3 | Wiley

Public datasets:

- ASCAD, AES_HD, AES_RD, DPA contest, …

Open source Librairies:

- **For side-channel attacks**: SCALib – the Side-Channel Analysis Library: https://scalib.readthedocs.io/
- **For deep learning**: AISyLab's framework - GitHub - AISY_Framework: Deep Learning-based Framework for Side-Channel Analysis

International conferences

- CHES, Cascade, Crypto, Eurocrypt, Asiacrypt, …

Demonstration

THALES

Building a future we can all trust

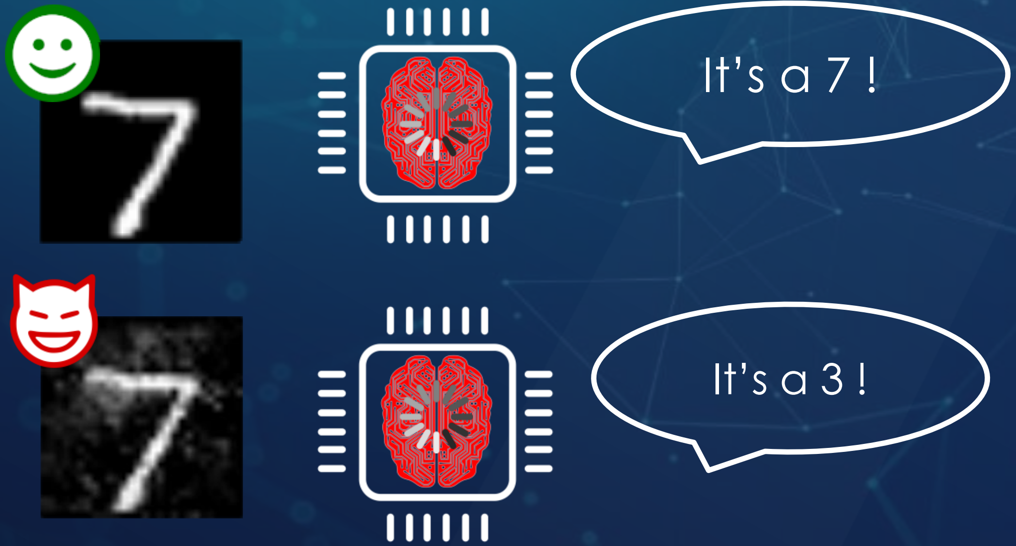
Physical Attacks against Neural Networks

Benefits from an adversary’s viewpoint

SRE

Evasion attack

Adversarial examples, KESAKO ?



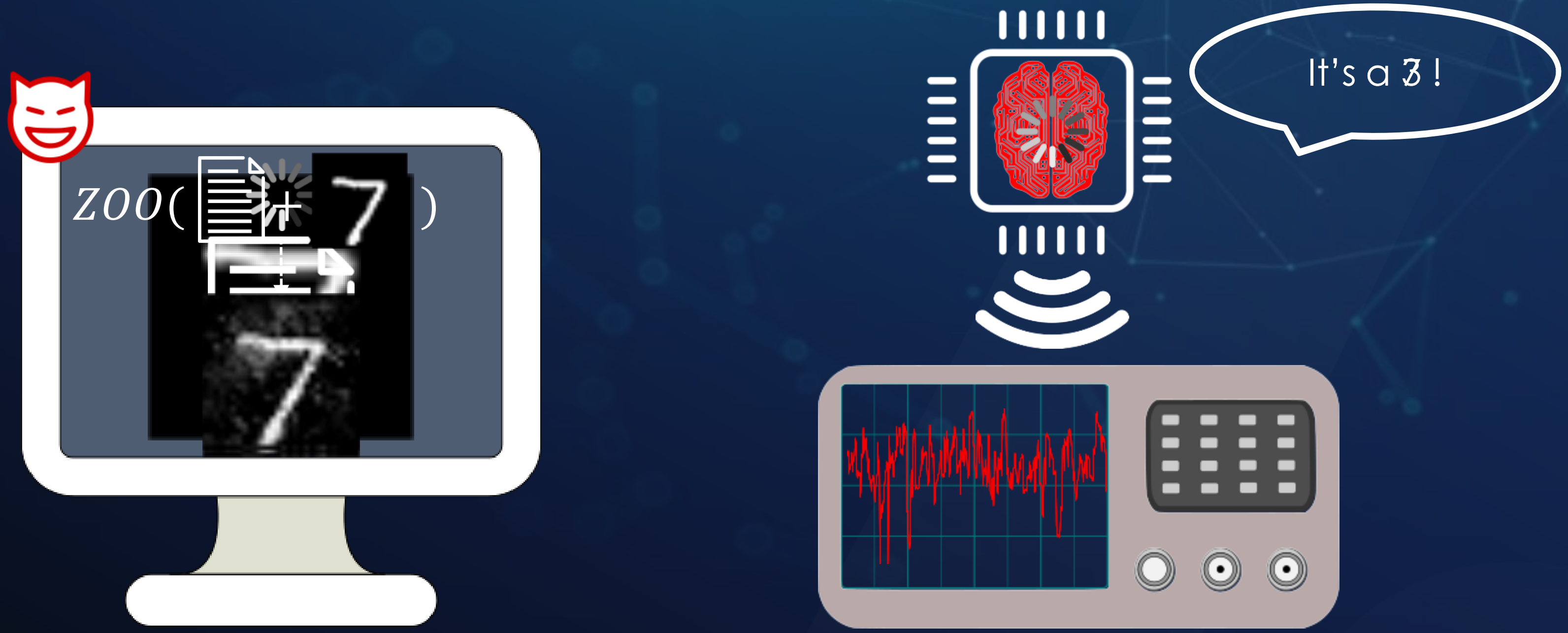
- ❑ Could be difficult to consider in practice
    - ❑ ***White-box scenario***: knowledge of the IA architecture, weights, activation functions, etc…
    - ❑ ***Black-box scenario***: partial knowledge on the AI system (e.g., *logits*)

Practical issue: How can we generate adversarial examples without any knowledge on the device?

Evasion attack

Practical issue: How can we generate adversarial examples without any knowledge on the device?
Our idea :

1) Extraction of the logits through the use of ***side-channel attacks***
2) Use the state-of-the art adversarial attacks (*e.g.* ZOO*)



* ‘‘ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models’’, Chen *et al.*, AISec ‘17

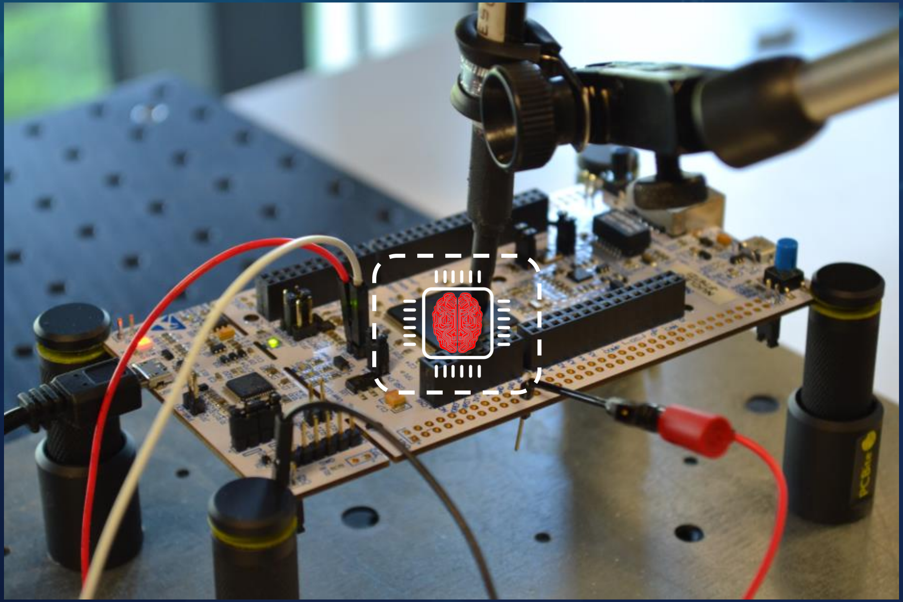
Evasion attack

Device: ARM Cortex-M7 MCU on a STM32F767 board (216 MHz)

Embedded AI: a 8-bit quantized denseNET (weights, activations, inputs) with NNOM* tool

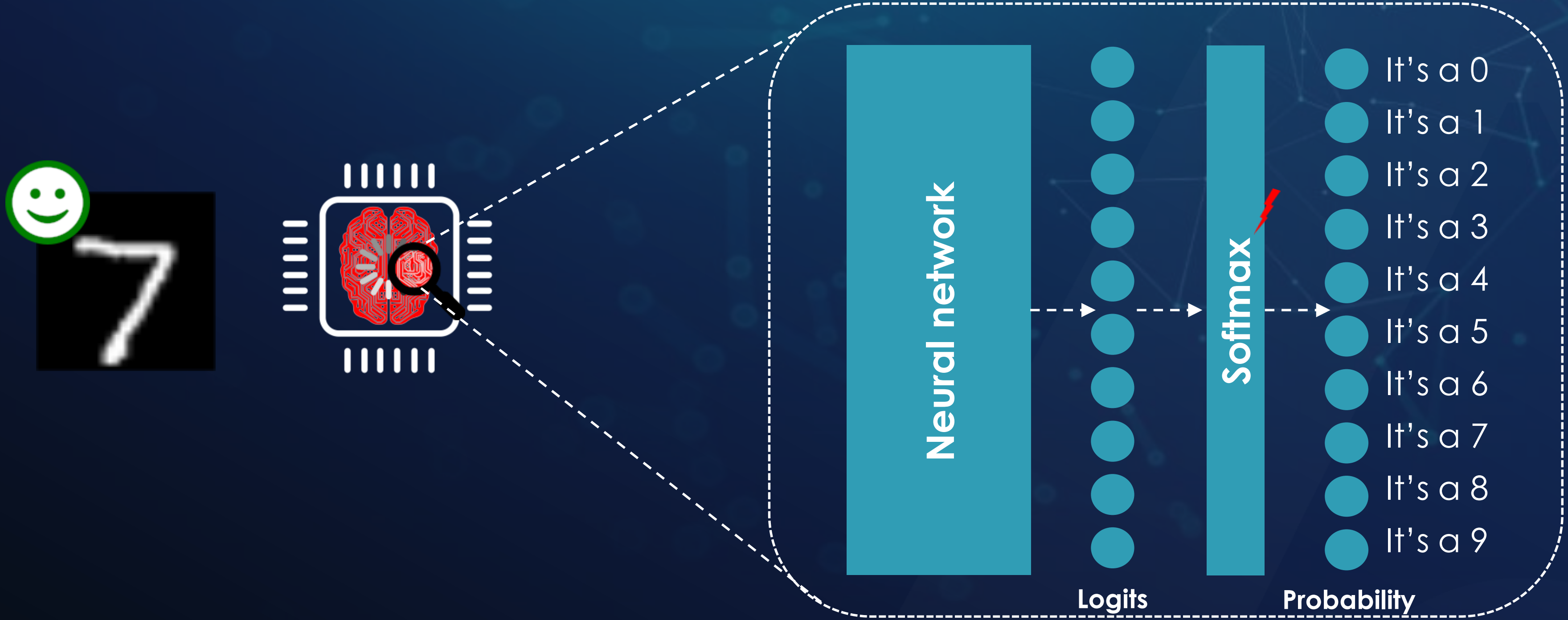
Classification problem: 10 classes (MNIST)

Channel: Electromagnetic signal (EMV-Technik RF-U 2,5 probe)



* GitHub - majianjia/nnom: A higher-level Neural Network library for microcontrollers.

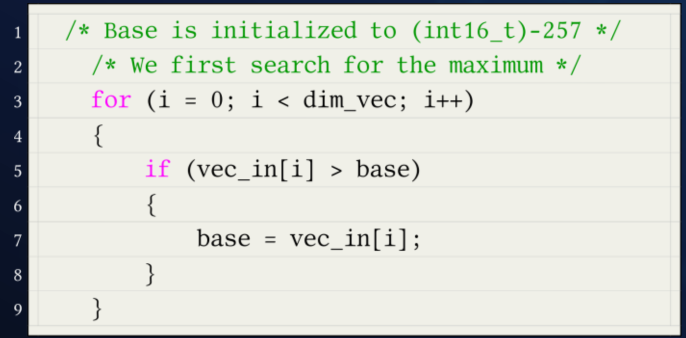
Evasion attack



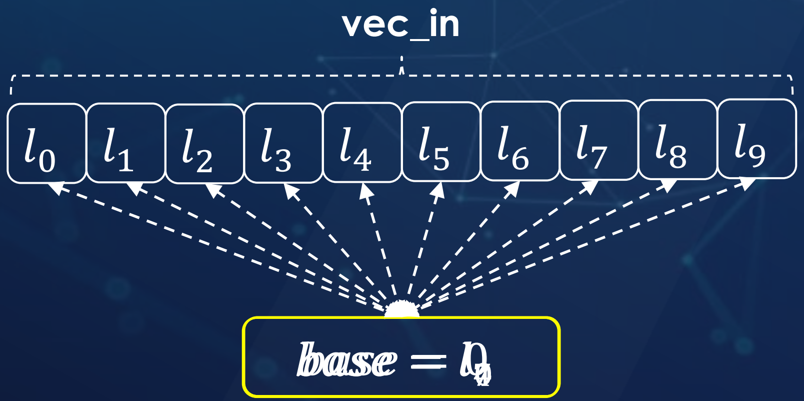
Evasion attack

Targeted function: Softmax function

C code related to the *softmax* function



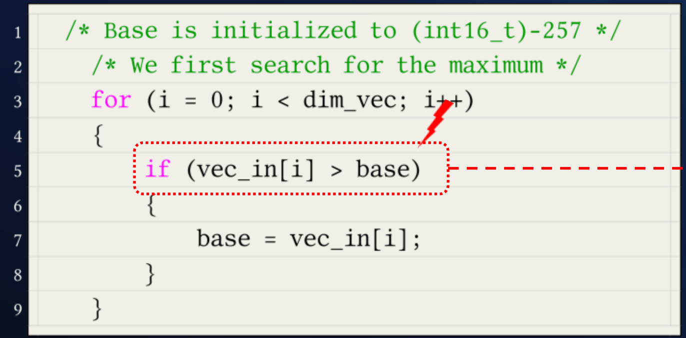
```
/* Base is initialized to (int16_t)-257 */
  /* We first search for the maximum */
  for (i = 0; i < dim_vec; i++)
  {
      if (vec_in[i] > base)
      {
          base = vec_in[i];
      }
  }
```



Evasion attack

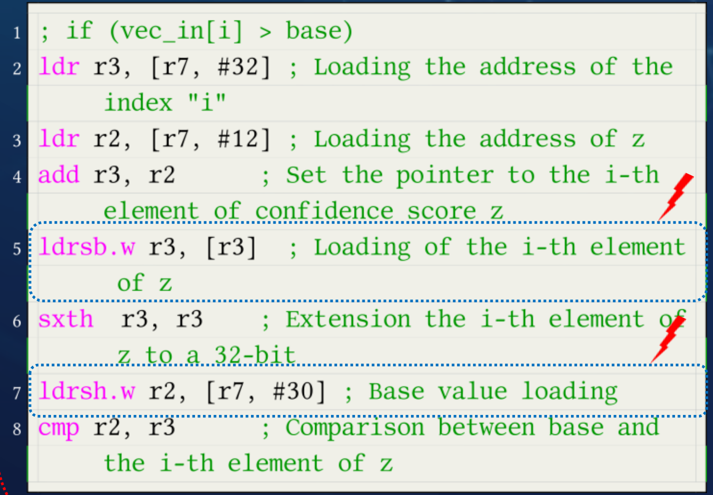
Targeted function: Softmax function

C code related to the *softmax* function



```
/* Base is initialized to (int16_t)-257 */
  /* We first search for the maximum */
  for (i = 0; i < dim_vec; i++)
  {
      if (vec_in[i] > base)
      {
          base = vec_in[i];
      }
  }
```

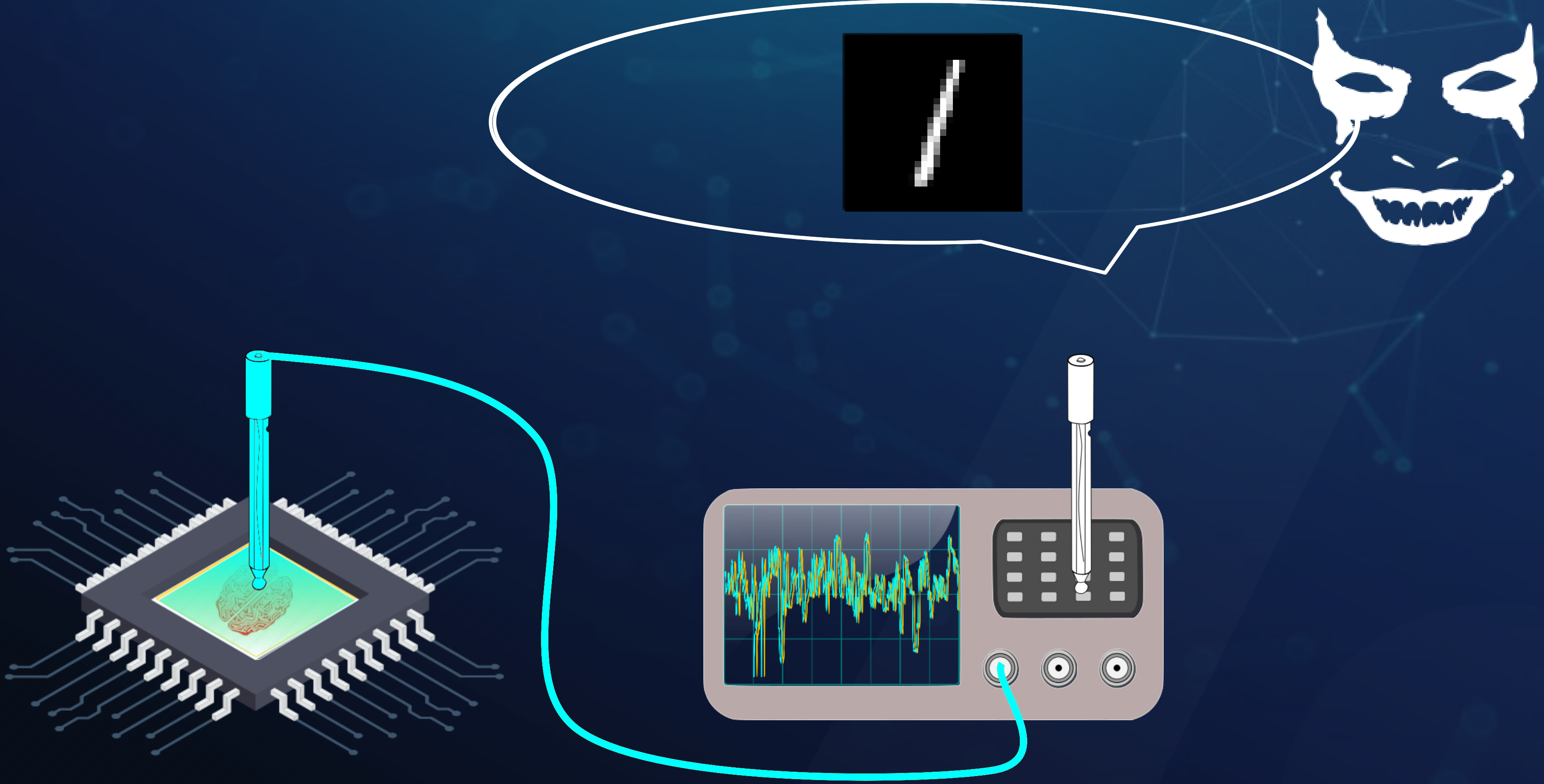
Assembly code



```
; if (vec_in[i] > base)
ldr r3, [r7, #32] ; Loading the address of the
    index "i"
ldr r2, [r7, #12] ; Loading the address of z
add r3, r2        ; Set the pointer to the i-th
    element of confidence score z
ldrsb.w r3, [r3]  ; Loading of the i-th element
    of z
sxth  r3, r3      ; Extension the i-th element of
    z to a 32-bit
ldrsh.w r2, [r7, #30] ; Base value loading
cmp r2, r3        ; Comparison between base and
    the i-th element of z
```

Evasion attack

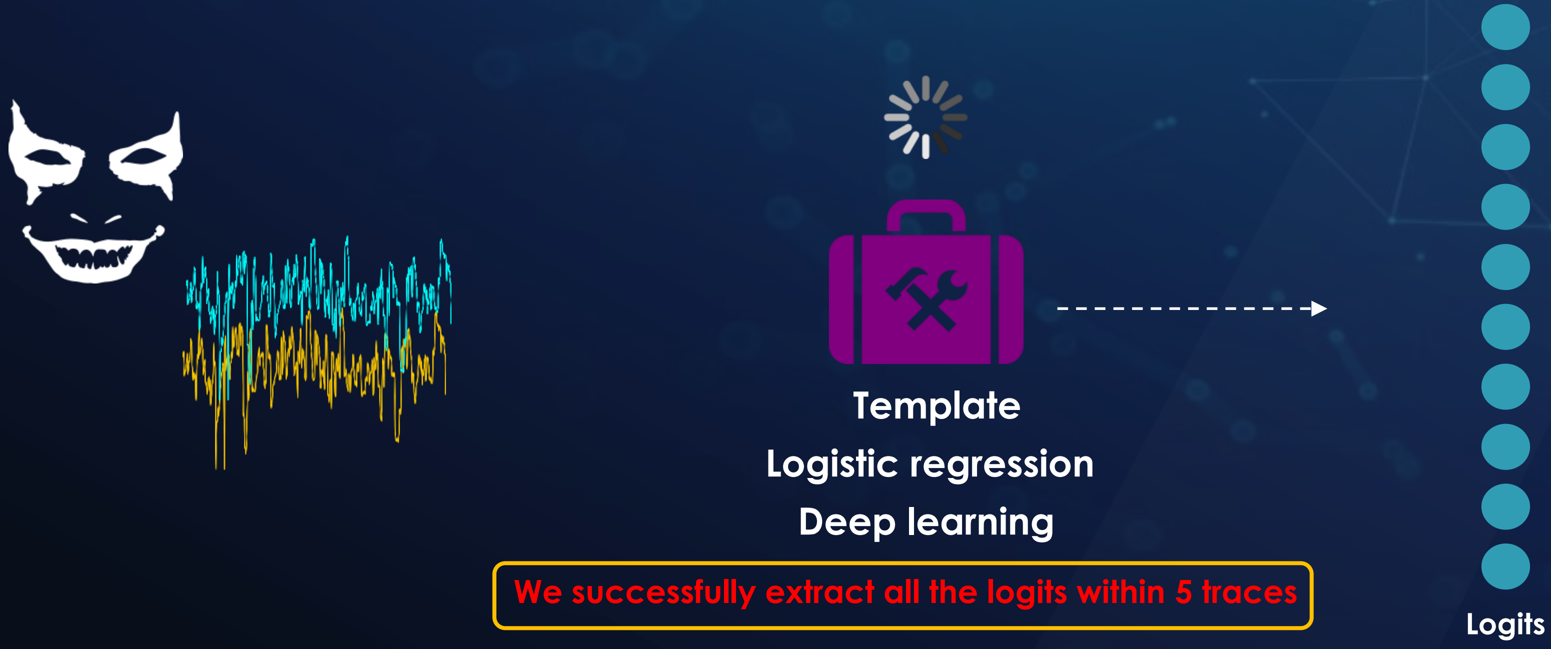
Side-Channel attacks



Evasion attack

Experimental results

- Profiled attack
- Black-box scenario (the attacker has no knowledge on the targeted AI)



Template

Logistic regression

Deep learning

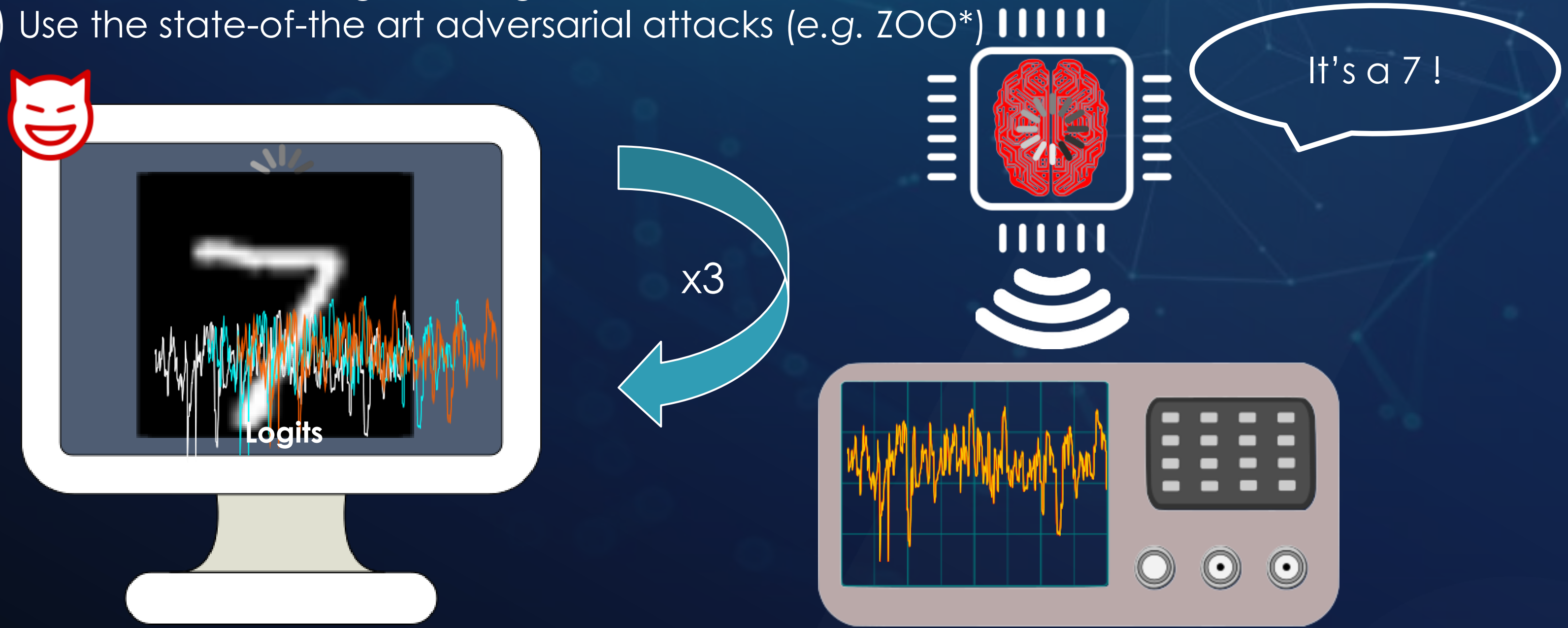
We successfully extract all the logits within 5 traces

Evasion attack

Practical issue: How can we generate adversarial examples without any knowledge on the device?

Our idea :

1) Extraction of the logits through the use of ***side-channel attacks***
2) Use the state-of-the art adversarial attacks (*e.g.* ZOO*)



* ''ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models'', Chen *et al.*, AISec '17

Evasion attack

Generate an adversarial example

- Application of the *Zeroth Order Optimization* (ZOO)* method

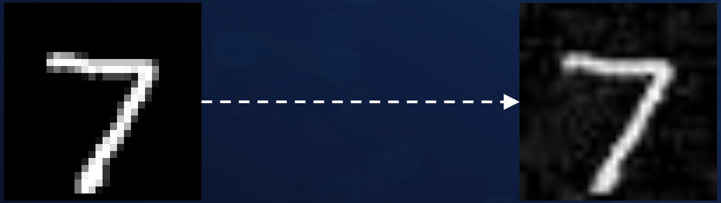
Recap (ZOO)

Let $\boldsymbol{X} \in \mathbb{R}^{\boldsymbol{n}}$ be an image and a wrong targeted class that an attacker wants to predict $y^* \in \mathcal{Y}$, she looks for an adversarial example $X^* \in \mathbb{R}^n$ such that the following relation is satisfied:

$$\boxed{\|\boldsymbol{X}^* - \boldsymbol{X}\|_2^2} + c \times \boxed{g_{obj}(\boldsymbol{X}^*, y^*)},$$

with $g_{obj}(X^*, y^*) = \max\left(\max_{y \neq y^*}\left(\log\left(F(X^*)\right)\right)[i] - \log\left(F(X^*)\right)[y^*], 0\right)$

Original image



Adversarial example

$$\boldsymbol{y^* = 7} \dashrightarrow \boldsymbol{y^* = 3}$$

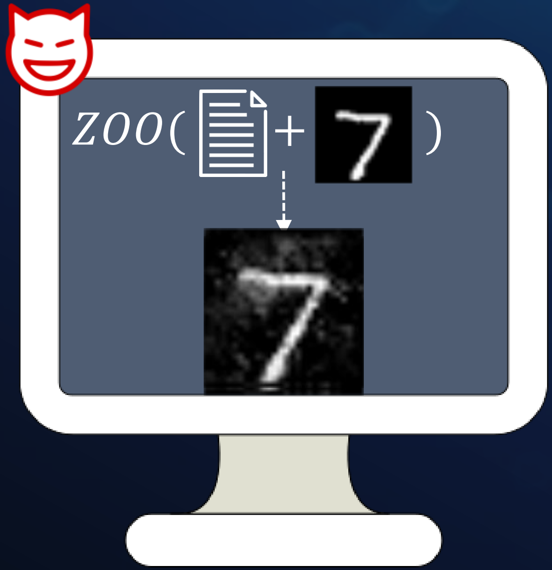
* ''ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models'', Chen *et al.*, AISec '17

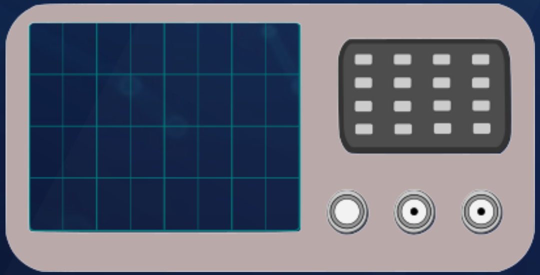
Evasion attack

Practical issue: How can we generate adversarial examples without any knowledge on the device?

Our idea :

1) Extraction of the logits through the use of ***side-channel attacks***
2) Use the state-of-the art adversarial attacks (*e.g.* ZOO*)





* ‘‘ZOO: Zeroth Order Optimization Based Black-box Attacks to Deep Neural Networks without Training Substitute Models’’, Chen *et al.*, AISec ’17

Thank you

Contact: gabriel.zaid@thalesgroup.com