

# Attaques par observation

Hélène Le Boudier, Ronan Lashermes et **Fabien Autrel**

2023



# Objectifs pédagogiques



Comprendre la problématique des attaques par observation.



Implémenter un HMAC.



L'attaquant peut :

- mesurer la consommation de courant ou le rayonnement électromagnétique ;
- simuler, récupérer la somme de tous les registres internes pour chaque instruction de :
  - `Verify_PIN`,
  - `Compare_arrays`.

Attaque par apprentissage en 2 étapes.

① Étape d'apprentissage :

- 1 circuit test,
- capable de changer le PIN secret du circuit test,
- nombre d'essais illimité,
- 1 réseau de neurones,
- lie les traces mesurées au PIN secret.

② Étape d'inférence :

- attaque sur le circuit cible,
- compare à l'apprentissage.



# Lancement de l'attaque

- Installation des paquets (ou utilisation de Nix) :
  - `pip3 install tensorflow,`
  - `pip3 install keras.`
- ⚠ PC puissant recommandé.
- Lancer l'apprentissage de l'attaque avec :



Terminal

```
pin_verif/> python3 ../L4/H-learning.py bin/
```

- Lancer l'attaque avec :

Terminal

```
pin_verif/> python3 ../L4/I-inference.py bin/
```

# Que faire ?

- Comment se protéger ?
- ⚠ La duplication accentue la fuite !



## Définition

- **MAC** (Message Authentication Code)
- But : seul le propriétaire de la clé peut vérifier l'**intégrité** du message.

## Construction :

- 1 une clé secrète  $K$ ,
- 2 un message  $M$ ,
- 3  $T = \text{MAC}(M, K)$ .



## Définition

Un **HMAC** est un type de MAC, calculé avec :

- une fonction de hachage cryptographique,
  - une clé secrète.
- 
- SHA-256.





# À vous de jouer !

## Terminal

```
pin_verif/> python3 ../L4/Z-compute-hmac.py
```

- Bibliothèque `hmac_lib` :
  - `sha256`
  - `hmac-sha256`
- Utiliser le HMAC pour faire votre comparaison de code PIN.





*inria*



*illustrations : Le Mooncat*