PREMIÈRE
MINISTRE

Liberté
Égalité
Fraternité

Étude de cas : attaques par canaux auxiliaires

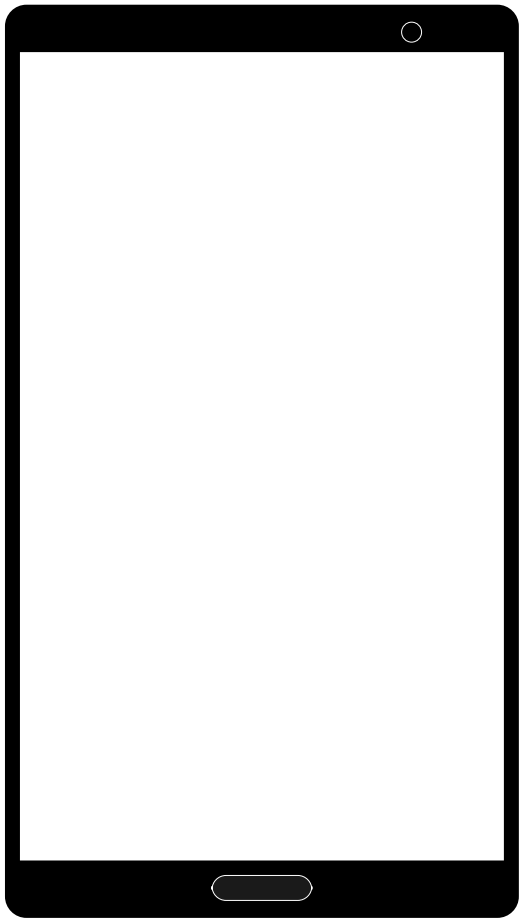
Thomas TROUCHKINE

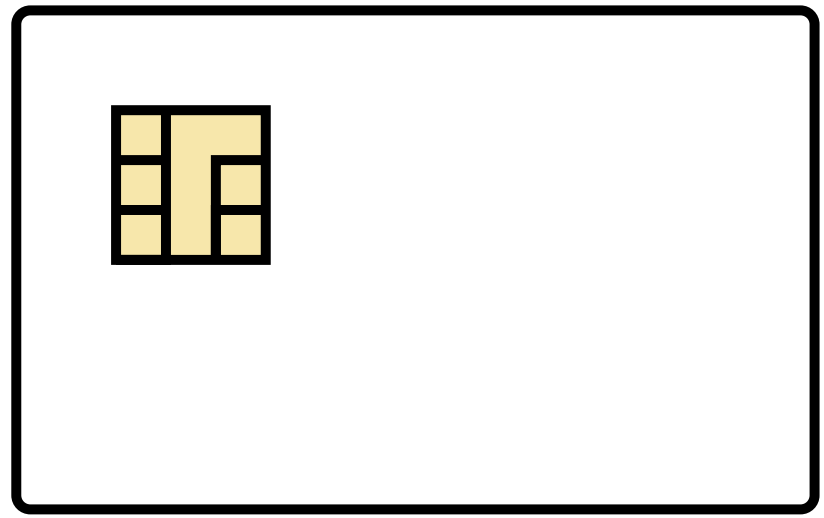
Agence nationale de la sécurité des systèmes d'information

October 13, 2022

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Vérification de code PIN





PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Vérification de code PIN

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      return false;
  }
  return true;
}
```

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

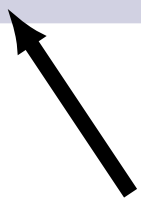
Vérification de code PIN

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      return false;
  }
  return true;
}
```

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Vérification de code PIN

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      return false;
  }
  return true;
}
```

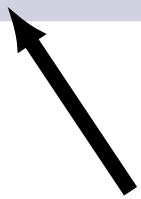


Boucle avec un temps d'exécution variable

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Vérification de code PIN

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      return false;
  }
  return true;
}
```



Boucle avec un temps d'exécution variable

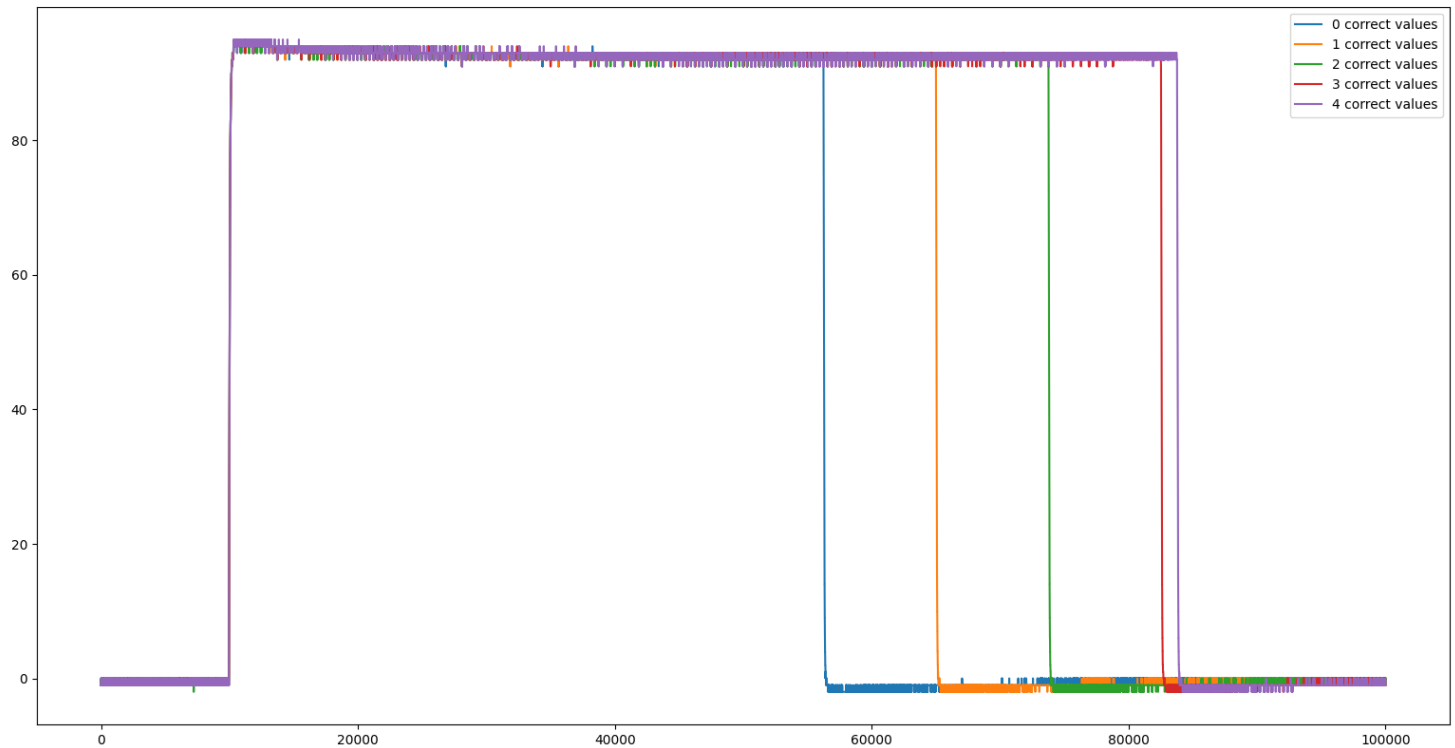
Temps d'exécution dépend du nombre de valeurs identiques

Démonstration

(Fuite en temps)

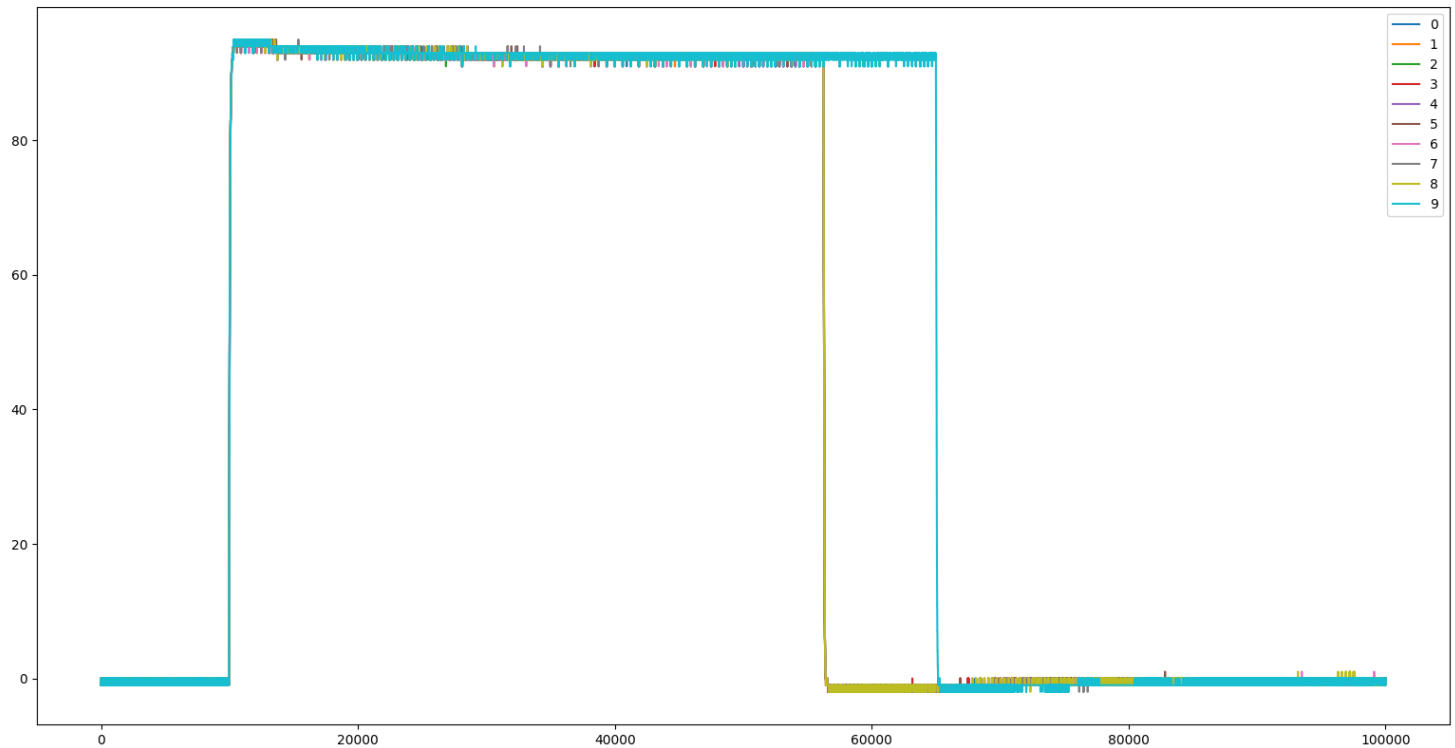
Attaque par analyse temporelle

Signal de trigger en fonction du nombre de valeurs correctes dans le PIN



Attaque par analyse temporelle

Signal de trigger en fonction de la première valeur du PIN



Attaque par analyse temporelle

Démonstration

(Attaque complète)

Attaque par analyse temporelle

Démonstration

(Sans signal de trigger depuis la carte attaquée)

Attaque par analyse temporelle

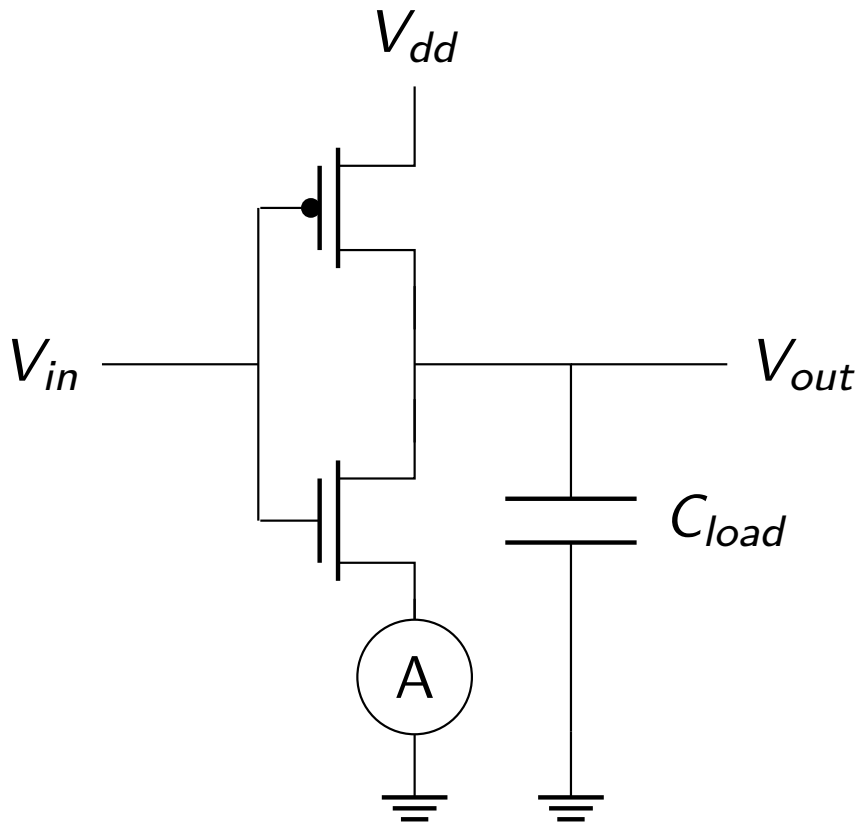
Démonstration

(Sans signal de trigger depuis la carte attaquée)

Échec

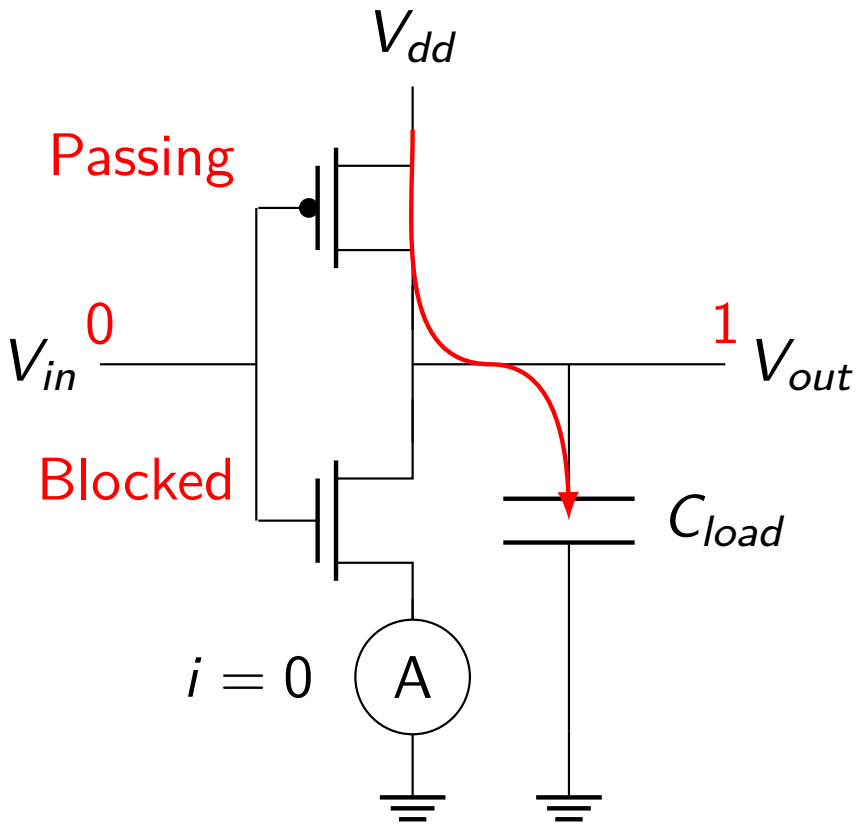
Attaque par analyse de courant

La consommation en énergie d'un circuit dépend des données qu'il manipule



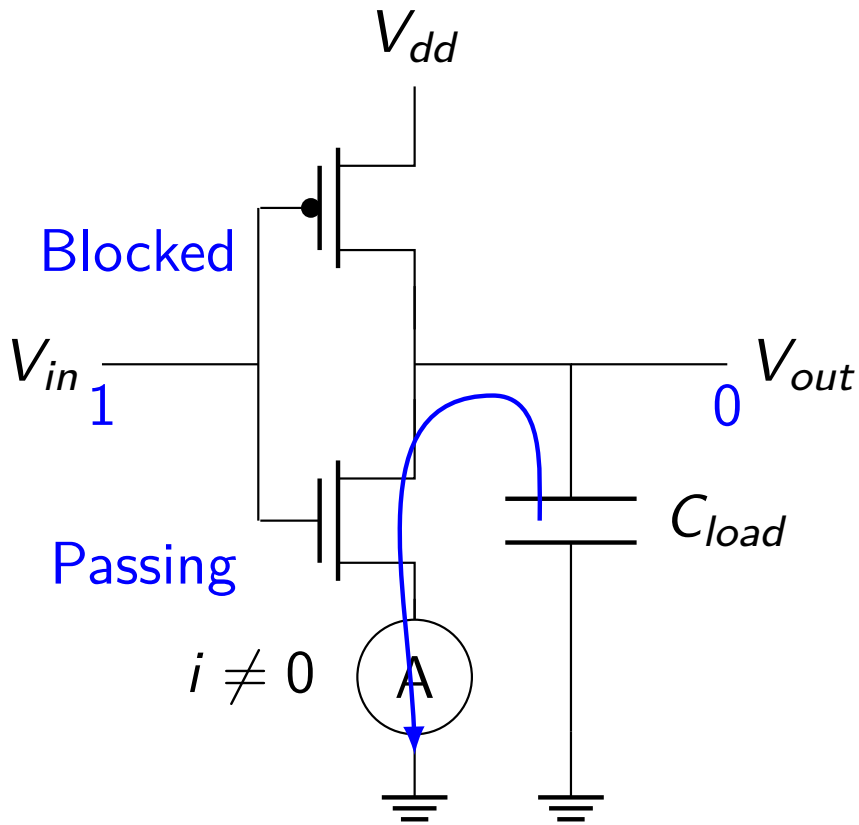
Attaque par analyse de courant

La consommation en énergie d'un circuit dépend des données qu'il manipule



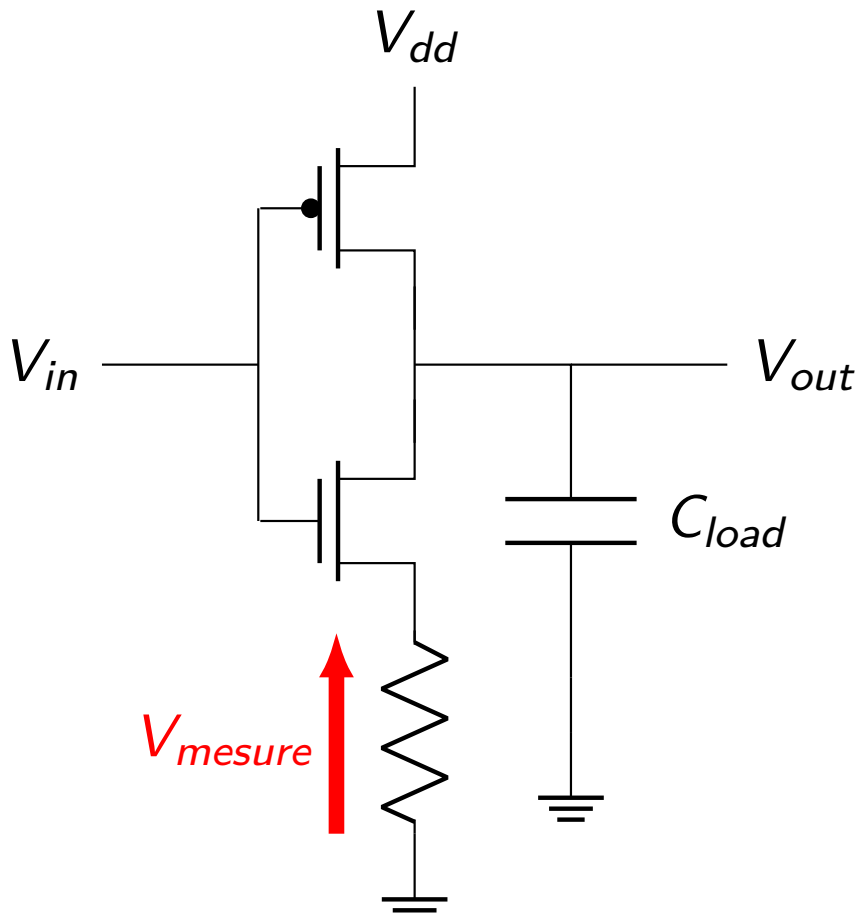
Attaque par analyse de courant

La consommation en énergie d'un circuit dépend des données qu'il manipule



Attaque par analyse de courant

La consommation en énergie d'un circuit dépend des données qu'il manipule



Attaque par analyse de courant

- On mesure la tension générée par l'ensemble des transistors du circuit
- De manière générale la tension observée peut s'exprimer telle que :

$V_{obs} = F(d) + B$

où

Attaque par analyse de courant

- On mesure la tension générée par l'ensemble des transistors du circuit
- De manière générale la tension observée peut s'exprimer telle que :

$V_{obs} = F(d) + B$

où

- d représente les données (ici la valeur du code PIN) qui nous intéresse

Attaque par analyse de courant

- On mesure la tension générée par l'ensemble des transistors du circuit
- De manière générale la tension observée peut s'exprimer telle que :

$V_{obs} = F(d) + B$

où

- d représente les données (ici la valeur du code PIN) qui nous intéresse
- F est la fonction qui modélise l'impact de d sur la tension observée

Attaque par analyse de courant

- On mesure la tension générée par l'ensemble des transistors du circuit
- De manière générale la tension observée peut s'exprimer telle que :

$V_{obs} = F(d) + B$

où

- d représente les données (ici la valeur du code PIN) qui nous intéresse
- F est la fonction qui modélise l'impact de d sur la tension observée
- B est le bruit lié au fonctionnement général du circuit
  - $B = C \pm R$ où C est la valeur nominale de consommation du circuit et R des fluctuations aléatoires liées au fonctionnement du circuit/environnement

Attaque par analyse de courant

Principe de l'attaque

- On veut distinguer le bon PIN des mauvais PIN
- En moyenne le bon PIN aura une consommation : $\tilde{V}_{obs}(PIN_{correct}) = \tilde{F}(PIN_{correct}) + C$
- En comparant cette trace de consommation moyenne avec celle obtenue en testant tous les PIN, on aura:
  - $\tilde{V}_{obs}(PIN_{test}) - \tilde{V}_{obs}(PIN_{correct}) = 0$ pour le bon PIN
  - $\tilde{V}_{obs}(PIN_{test}) - \tilde{V}_{obs}(PIN_{correct}) \neq 0$ pour les mauvais PIN

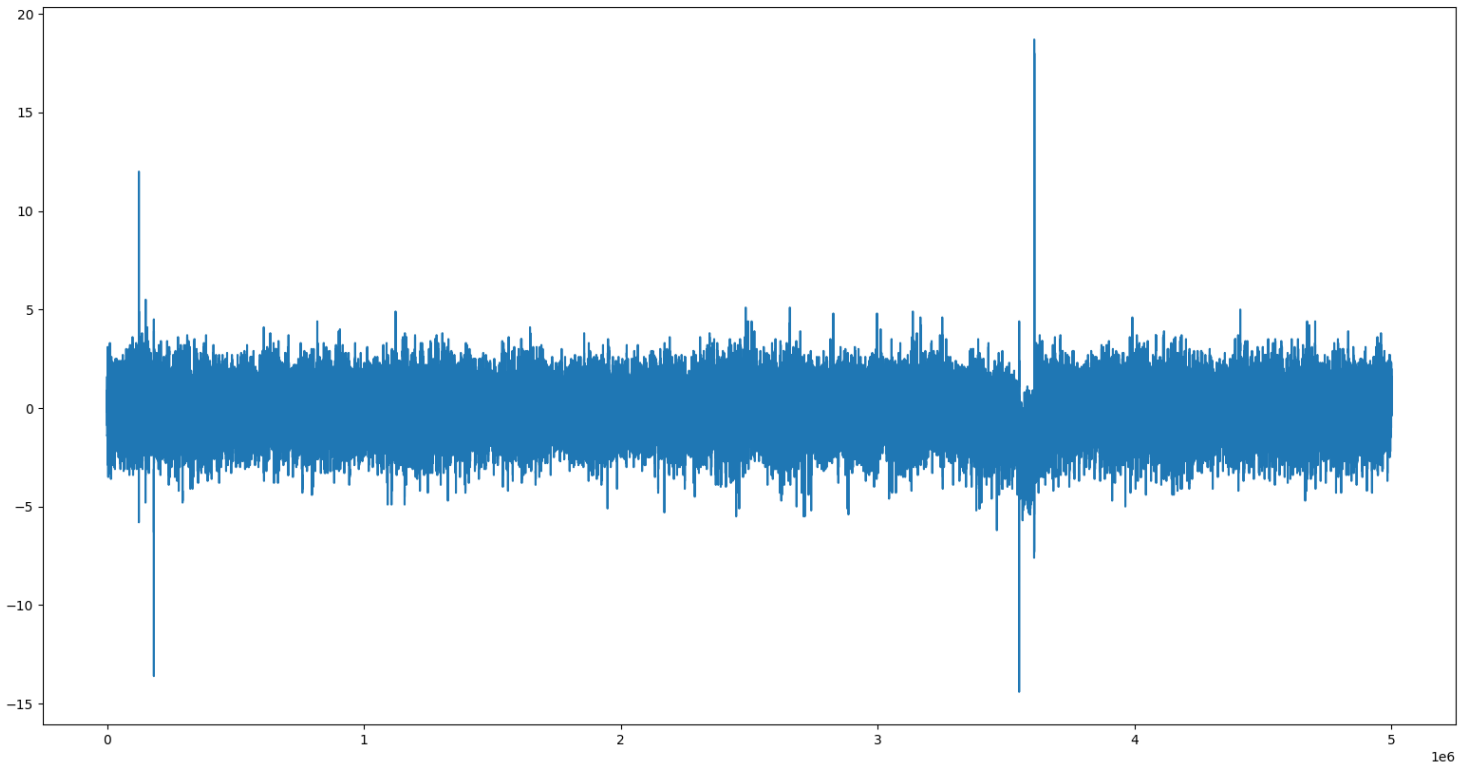
Attaque par analyse de courant

Démonstration

(Fuite en courant)

Attaque par analyse de courant

Différence de la consommation moyenne du circuit entre le bon PIN et le mauvais PIN (1ère valeur)



Attaque par analyse de courant

Problème : On a besoin d'une trace de référence avec le bon PIN

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Attaque par analyse de courant

Problème : On a besoin d'une trace de référence avec le bon PIN

- On a accès à un composant similaire sur lequel faire des acquisitions : Attaque par templating

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Attaque par analyse de courant

Problème : On a besoin d'une trace de référence avec le bon PIN

- On a accès à un composant similaire sur lequel faire des acquisitions : Attaque par templating
- On simule les traces sur ordinateur : Attaque par corrélation

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Contre-mesures

Contre les fuites temporelles : fonction en temps constant

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  bool pinOk = true;
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      pinOk = false;
  }
  return pinOk;
}
```

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Contre-mesures

Contre les fuites temporelles : fonction en temps constant

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  bool pinOk = true;
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      pinOk = false;
  }
  return pinOk;
}
```

Boucle toujours en temps variable !

PREMIÈRE MINISTRE
Liberté
Égalité
Fraternité

Contre-mesures

Contre les fuites temporelles : fonction en temps constant

```
bool verifyPin(char* buf1, char* buf2, unsigned char size){
  bool pinOk = true;
  bool dummy = false;
  for(unsigned char i=0; i<size; i++){
    if(buf1[i] != buf2[i])
      pinOk = false;
    else
      dummy = false;
  }
  return pinOk;
}
```

Questions?