Master Parisien de Recherche en Informatique

Cryptanalyse par canaux auxiliaires :
attaques et contre-mesures

Mélissa Rossi Dabi

Septembre 2016

Encadrants: Sonia Belaïd et Renaud Dubois
Responsable Pédagogique: Pierre Senellart
Rapporteur: Damien Vergnaud

Plan de la présentation

Théorie des attaques par canaux auxiliaires

Attaques de différentes implémentations d'AES

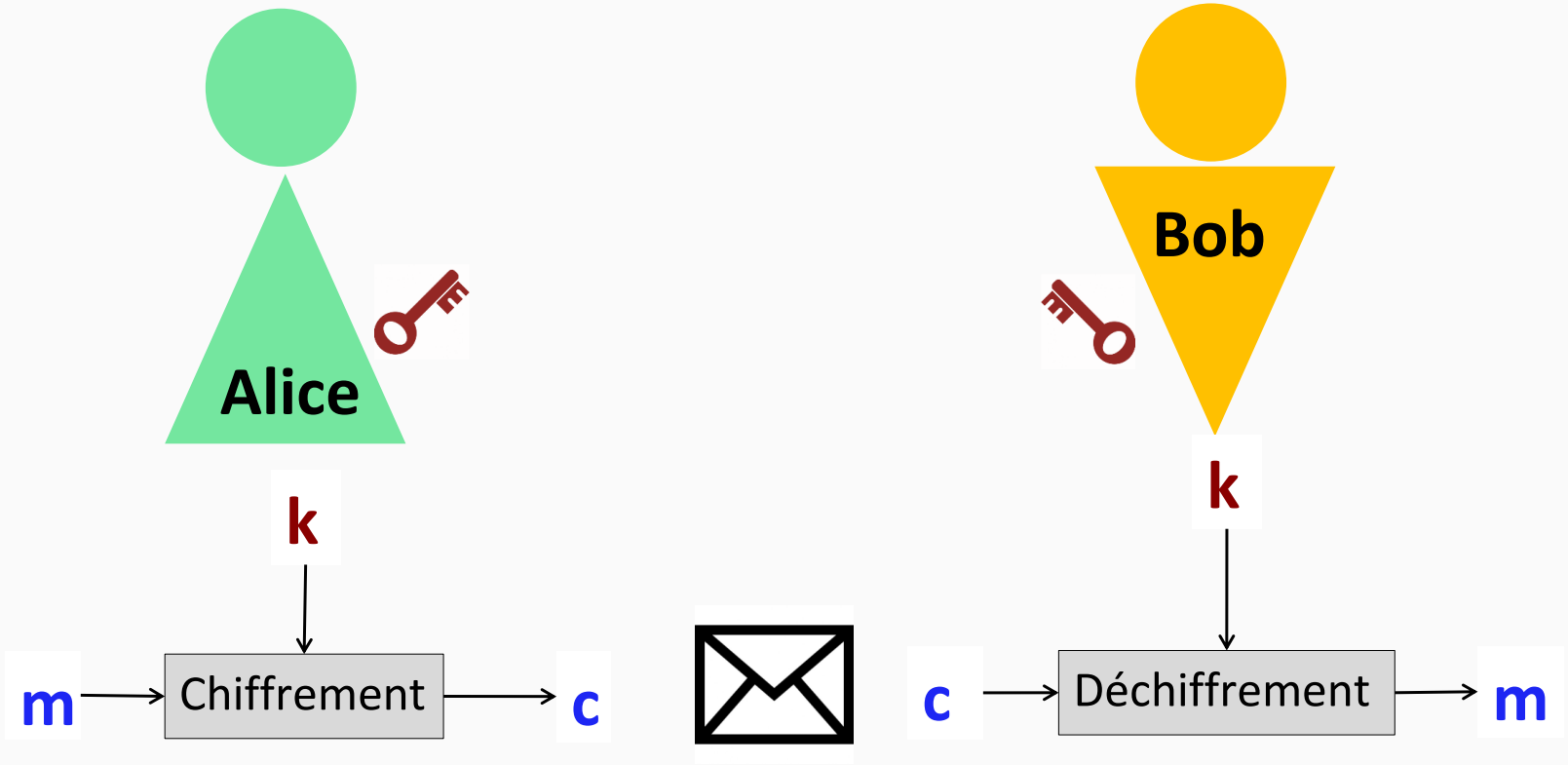
- Attaque d'un AES non sécurisé
- Une protection : le masquage
- Attaque du DPA Contest V4.2

Amélioration de la protection d'une implémentation d'AES

- Proposition
- Evaluation du schéma proposé

Théorie des attaques par canaux auxiliaires

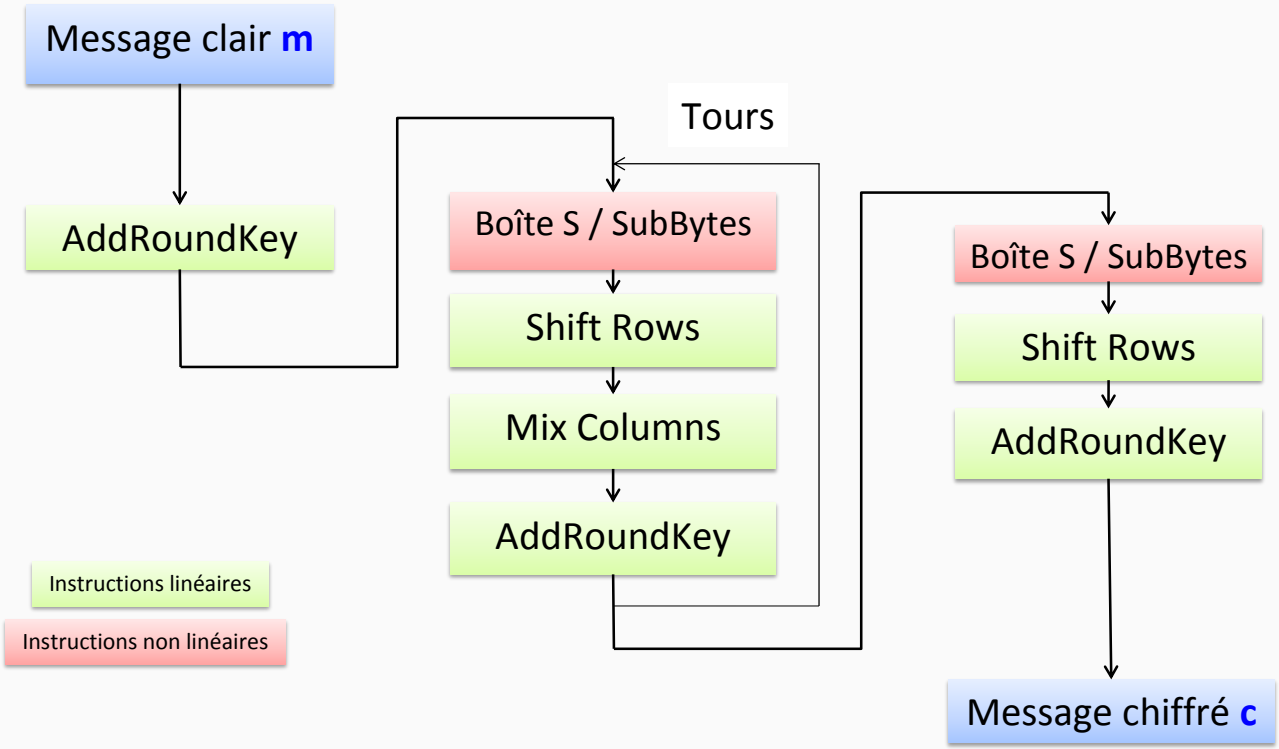
Cryptographie symétrique



Confidentialité

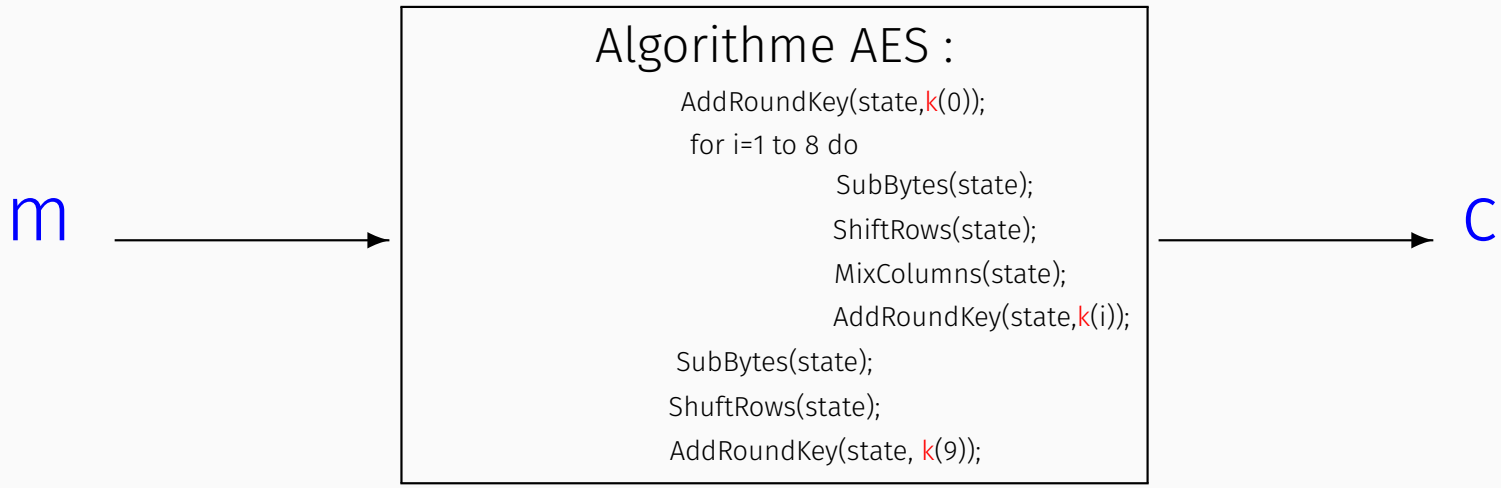
Algorithme AES :

```
AddRoundKey(state,k(0));
for i=1 to 8 do
        SubBytes(state);
        ShiftRows(state);
        MixColumns(state);
        AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```



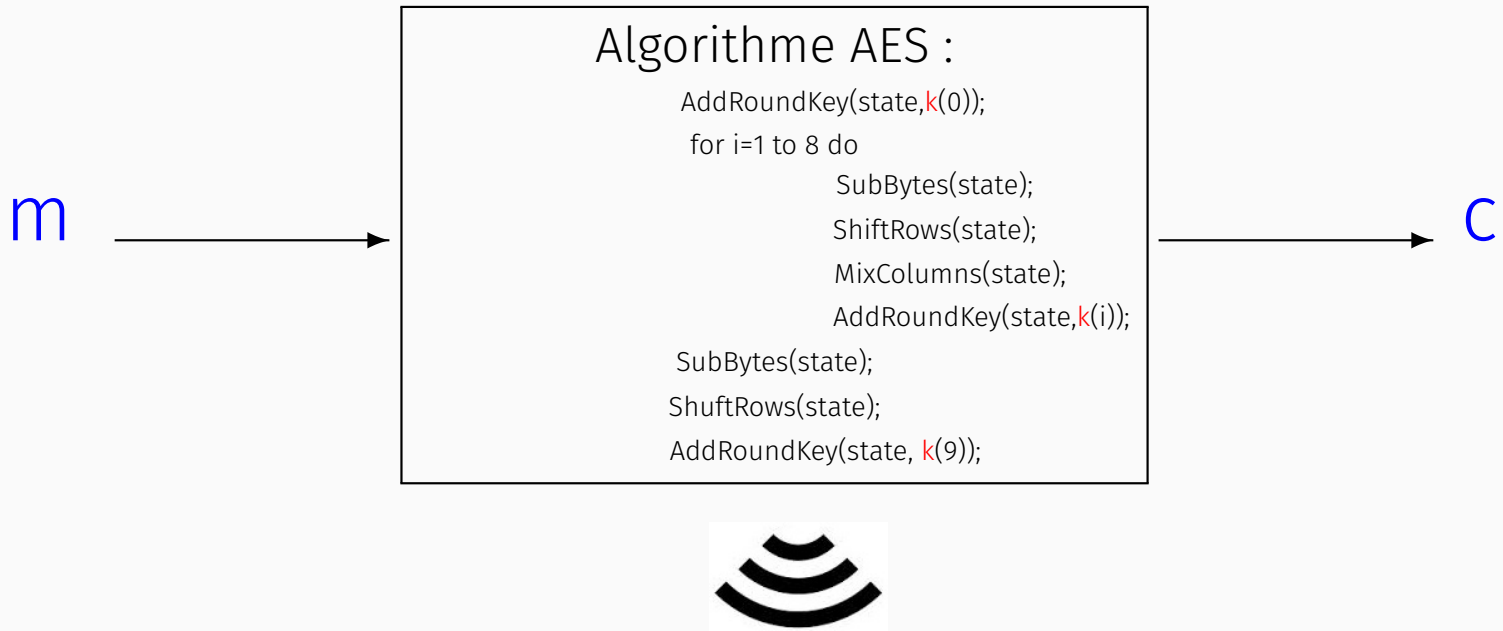
Cryptanalyse

Cryptanalyse boîte noire : $(m, c) \rightarrow k$



Cryptanalyse

Cryptanalyse par canaux auxiliaires : $(m, c, mesures\ physiques) \rightarrow k$



Cryptanalyse

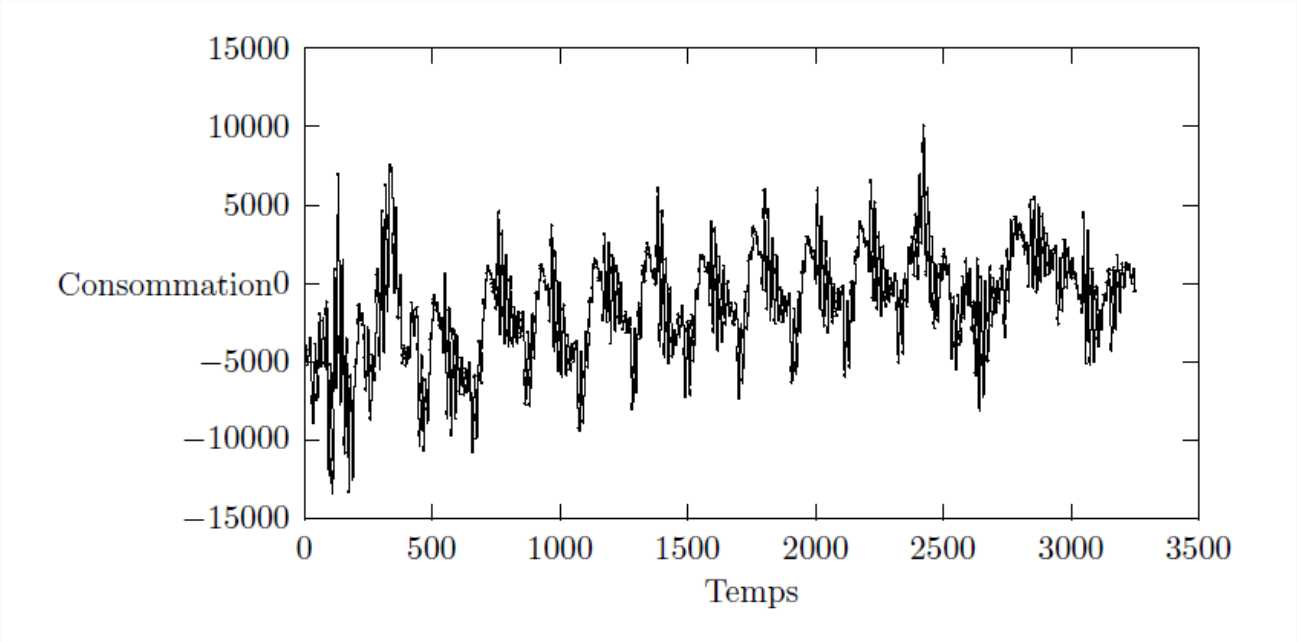
Cryptanalyse par canaux auxiliaires : $(m, c, mesures\ physiques) \rightarrow k$

m

Algorithme AES :

```
AddRoundKey(state,k(0));
for i=1 to 8 do
        SubBytes(state);
        ShiftRows(state);
        MixColumns(state);
        AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

c



Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\forall M \in [1, N] : W_M(t) = f(X_{M,K})$

Algorithme AES :

```
AddRoundKey(state,k(0));
 for i=1 to 8 do
                SubBytes(state);
                ShiftRows(state);
                MixColumns(state);
                AddRoundKey(state,k(i));
 SubBytes(state);
ShuftRows(state);
 AddRoundKey(state, k(9));
```

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

```
AddRoundKey(state,k(0));
for i=1 to 8 do
        SubBytes(state);
        ShiftRows(state);
        MixColumns(state);
        AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

→ Prédictions $X_{M,\tilde{K}}$

→ Mesure des traces $W_M(t)$

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

→ $\begin{pmatrix} X_{1,0} & \cdots & X_{1,\tilde{K}} & \cdots & X_{1,255} \\ X_{2,0} & \cdots & X_{2,\tilde{K}} & \cdots & X_{2,255} \\ \vdots & & & & \vdots \\ X_{N,0} & \cdots & X_{N,\tilde{K}} & \cdots & X_{N,255} \end{pmatrix}$

→ $\begin{pmatrix} W_0(t) \\ W_1(t) \\ \vdots \\ W_N(t) \end{pmatrix}$

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque DPA (Differential Power Analysis)

Cible de l'attaque : K fragment de la clé secrète (Diviser pour régner)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = f(X_{M,K})}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

$$\begin{pmatrix} X_{1,0} & \cdots & X_{1,\tilde{K}} & \cdots & X_{1,255} \\ X_{2,0} & \cdots & X_{2,\tilde{K}} & \cdots & X_{2,255} \\ \vdots & & & & \vdots \\ X_{N,0} & \cdots & X_{N,\tilde{K}} & \cdots & X_{N,255} \end{pmatrix}$$

$$\begin{pmatrix} W_0(t) \\ W_1(t) \\ \vdots \\ W_N(t) \end{pmatrix}$$

Distingueur

Trouve $\tilde{K}$ qui vérifie le mieux le modèle : $\forall M$ $W_M(t) \approx f(X_{M,\tilde{K}})$

Exemple de modèle :

$$W_M(t) \begin{cases} \geq 0 \text{ si } S[M \oplus K] \bmod 2 = 0 \\ \leq 0 \text{ si } S[M \oplus K] \bmod 2 = 1 \end{cases}$$

Attaque CPA (Correlation Power Analysis)

Modèle $\forall M \in [1, N] : W_M(t) = \lambda \cdot \phi(X_{M,K}) + \beta$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

→ Prédictions $V_{M,\tilde{K}} = \phi(X_{M,\tilde{K}})$

→ Mesure des traces $W_M(t)$

→ Distingueur

Trouve $\tilde{K}$ tel que $W_M(t) \approx \lambda \cdot V_{M,\tilde{K}} + \beta$ pour tous M

Un ou plusieurs instants d'intérêts t	Une valeur intermédiaire $X_{M,K}$	Une fonction ϕ

Attaque CPA (Correlation Power Analysis)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = \lambda \cdot \phi(X_{M,K}) + \beta}$

Algorithme AES :

Instant attaqué

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

→ Prédictions $V_{M,\tilde{K}} = \phi(X_{M,\tilde{K}})$

→ Mesure des traces $W_M(t)$

→ Corrélation

Trouve $\tilde{K}$ tel que $W_M(t) \approx \lambda \cdot V_{M,\tilde{K}} + \beta$ pour tous M

$$\rho(\tilde{K}, t) = \frac{Cov((V_{i,\tilde{K}})_{1 \leq i \leq N}, W_i(t)_{1 \leq i \leq N})}{\sigma_{(V_{i,\tilde{K}})_{1 \leq i \leq N}} \cdot \sigma_{(W_i(t))_{1 \leq i \leq N}}}$$

(Coefficient de Pearson)

Attaque CPA (Correlation Power Analysis)

Modèle $\boxed{\forall M \in [1, N] : W_M(t) = \lambda \cdot \phi(X_{M,K}) + \beta}$

Algorithme AES :

Instant attaqué →

```
AddRoundKey(state,k(0));
for i=1 to 8 do
    SubBytes(state);
    ShiftRows(state);
    MixColumns(state);
    AddRoundKey(state,k(i));
SubBytes(state);
ShuftRows(state);
AddRoundKey(state, k(9));
```

→ Prédictions $V_{M,\tilde{K}} = \phi(X_{M,\tilde{K}})$ ↘

→ Mesure des traces $W_M(t)$ ↗

Corrélation

Trouve $\tilde{K}$ qui maximise $\rho(\tilde{K}, t)$

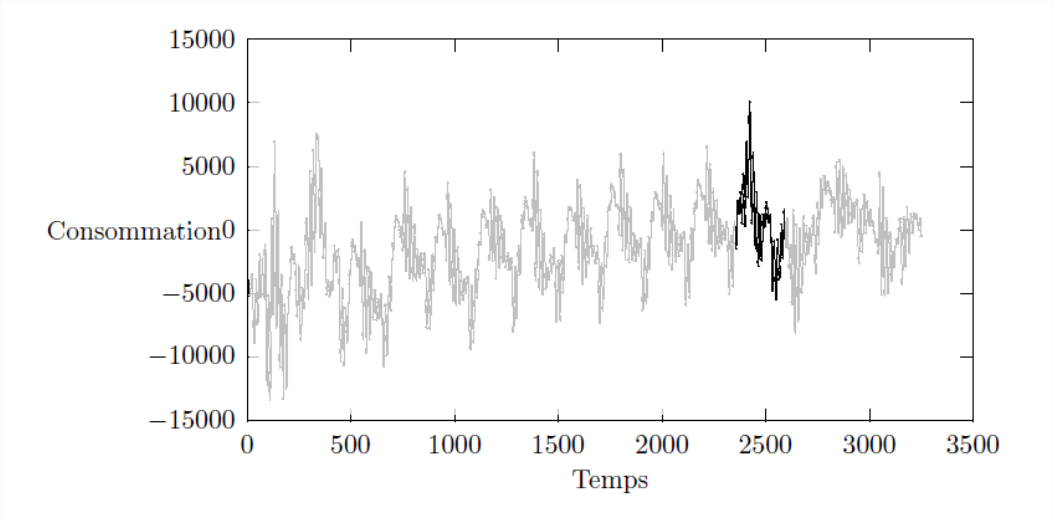
$$\rho(\tilde{K}, t) = \frac{Cov((V_{i,\tilde{K}})_{1 \leq i \leq N}, W_i(t)_{1 \leq i \leq N})}{\sigma_{(V_{i,\tilde{K}})_{1 \leq i \leq N}} \cdot \sigma_{(W_i(t))_{1 \leq i \leq N}}}$$

(Coefficient de Pearson)

Attaques de différentes implémentations d'AES

Attaque d'un AES non sécurisé

Traces : **DPA Contest V2** (Concours lancé par Télécom Paristech)

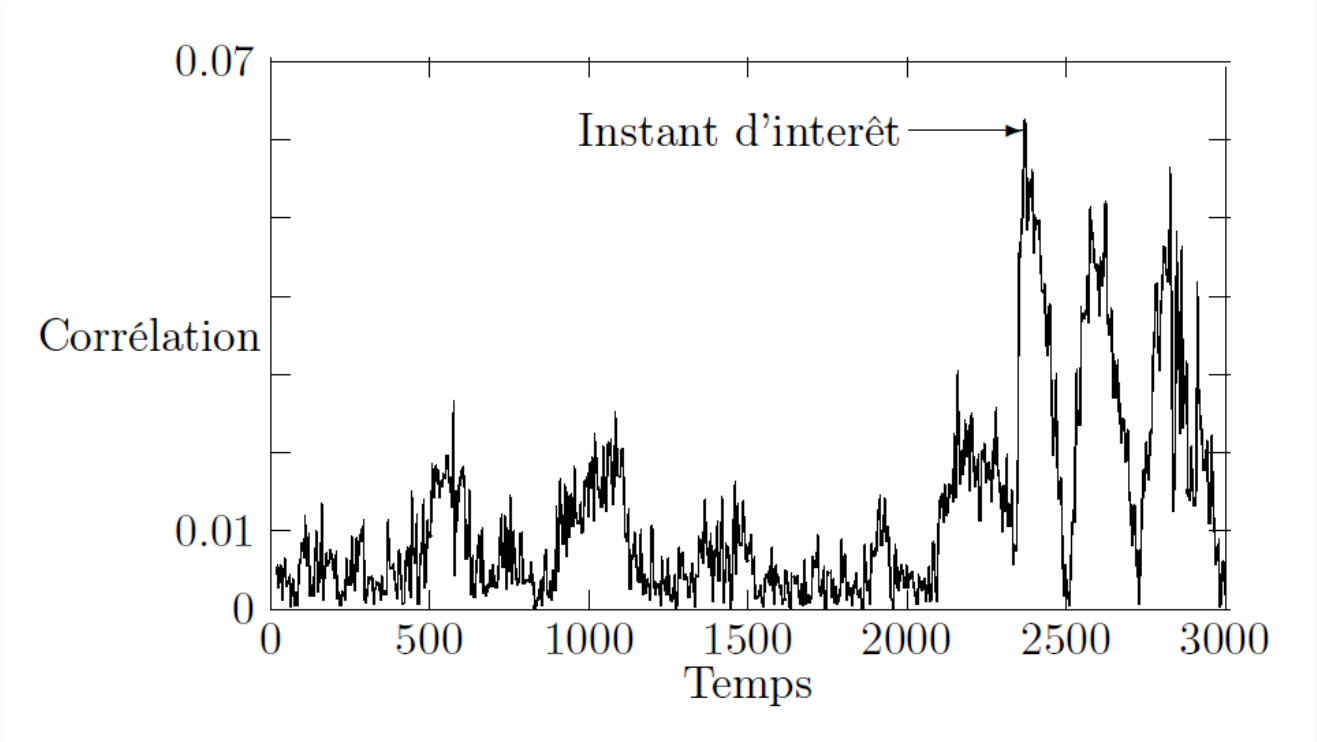


Attaque CPA en **distance de Hamming** au dernier tour.

Instants d'intérêts	Valeur intermédiaire	Fonction ϕ
?	Dernière boîte S	Distance de Hamming

Deux possibilités pour trouver t

- *Apprentissage* : calcul de $\rho(K, t)$ pour tous t



- *Attaque aveugle* : attaque menée à tous les instants du dernier tour et compteur d'occurrences de clés

Résultats

Résultats obtenus[1] lors de l'attaque du DPA Contest V2

Type d'attaque	Nombre de traces	Octets de clé retrouvés
Avec apprentissage	20000	98%
Aveugle	20000	90%
Aveugle	10000	84%
Aveugle	5000	59%

[1] Sans l'amélioration décrite dans le rapport

Une protection : le masquage

Éviter que les valeurs intermédiaires ne dépendent directement des valeurs sensibles.

- **Démultiplication** des variables avec ajout d'aléa
- Tout le long de l'algorithme

Une protection : le masquage

Éviter que les valeurs intermédiaires ne dépendent directement des valeurs sensibles.

- **Démultiplication** des variables avec ajout d'aléa
- Tout le long de l'algorithme

Masquage booléen d'ordre 1

Un masquage booléen d'ordre 1 est l'opération qui remplace une donnée x appelée *donnée sensible* par 2 données y_0 et y_1 où

$$\begin{cases} y_0 \leftarrow \$ \\ y_1 = x \oplus y_0 \end{cases}$$

Perte d'efficacité

Plusieurs implémentations de masquage booléen d'ordre 1 existent mais sont **coûteuses**.

Coût d'une implémentation : taille de code, nombre de cycles de processeurs, nombre d'aléas générés, surface

Comparaison d'efficacité pour l'implémentation de Rivain et Prouff[2] :

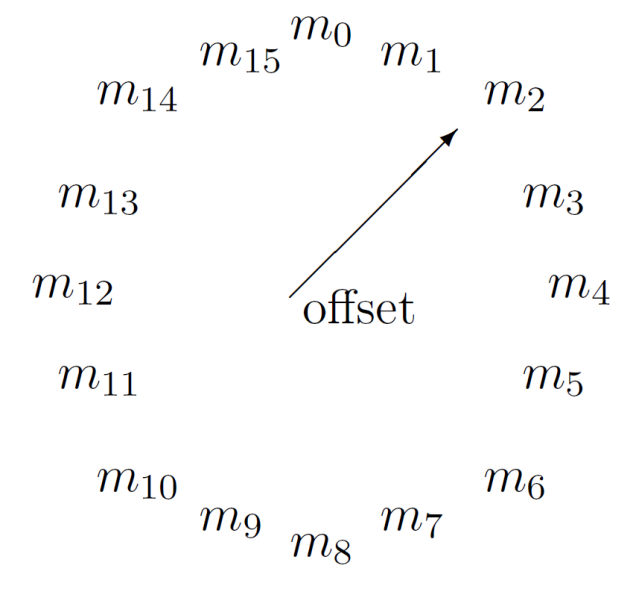
Algorithme	Nombre de cycles	RAM (octets)
AES sans protection	$3 \cdot 10^3$	32
AES masqué	$129 \cdot 10^3$	73

[2] En language assembleur : *8051-based 8-bit architecture*

Masquage RSM (Rotating S-box Masking)

Ensemble **fixe choisi de 16 masques** $(m_i)_{0 \leq i \leq 15}$.
Offset: indice du masque utilisé.

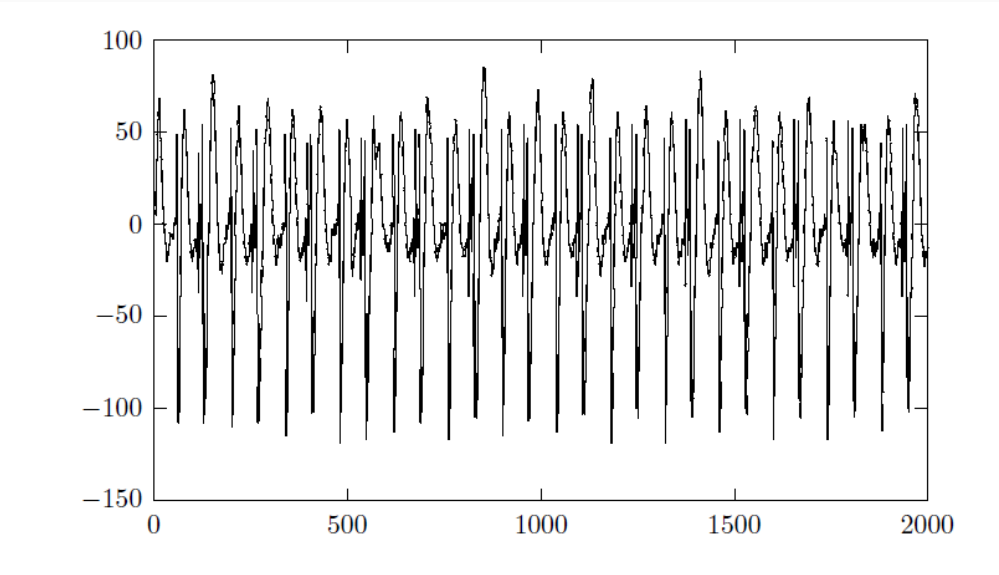
- Tiré aléatoirement en début de chiffrement.
- A chaque fois que l'algorithme a besoin de générer un nouveau masque, l'*offset* est ensuite incrémenté modulo 16.



Concours actuel (lancé en 2015):

Implémentation assembleur d'un AES-RSM software amélioré

- Chaque octet de l'état possède son propre offset. 16 barillets
- *Shuffle* : Mélange d'ordre de calcul sur les boîtes S
- 4 fois plus de cycles que pour un AES non protégé [3]



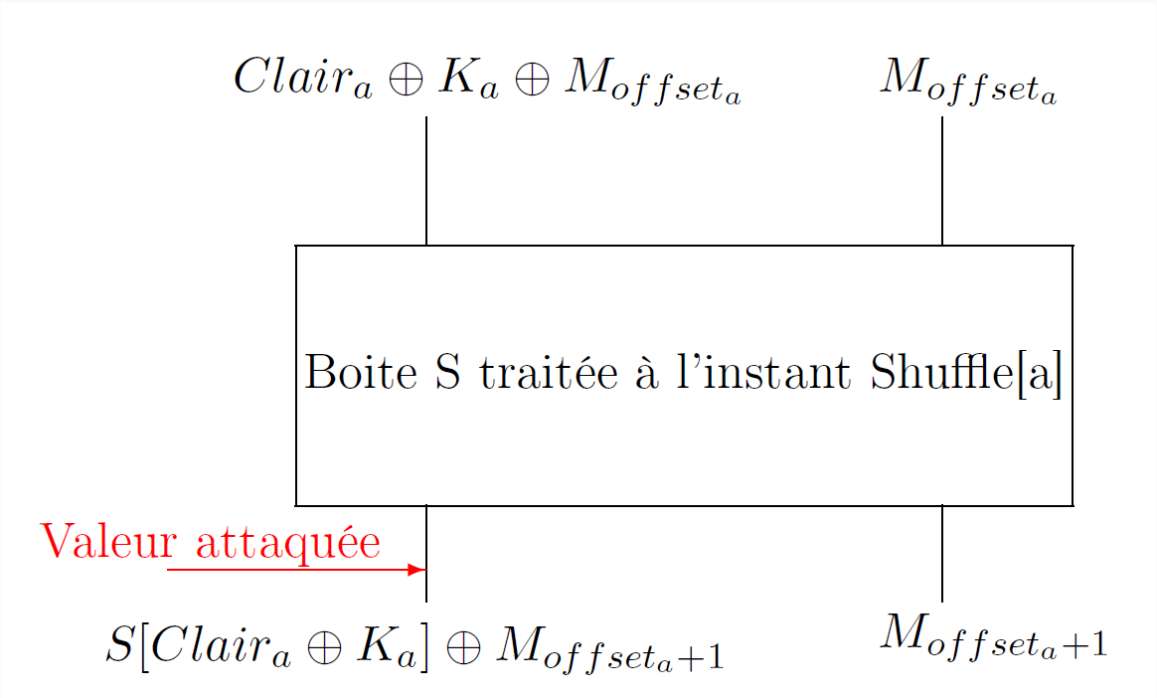
[3] Pour un processeur Atmega328P

Attaque CPA + attaque auxiliaire

Instants d'intérêts	Valeur intermédiaire	Fonction ϕ
Apprentissage	*Mix Columns*	Poids de Hamming

Attaque CPA + attaque auxiliaire

Instants d'intérêts	Valeur intermédiaire	Fonction ϕ
Apprentissage	*Mix Columns*	Poids de Hamming



Retrouver les offsets : une attaque non profilée

Attaque *horizontale* : déduit de l'information grâce à une trace

1. **11 incrémentations de l'offset** $\rightarrow$ Fuite du poids de Hamming
2. On repère le tour pour lequel l'écart est le plus fort entre deux valeurs consécutives.

Exemple :

Tour	0	1	2	3	4	5	6	7	8	9	10
Conso	200	190	254	**500**	**-5**	36	59	80	125	165	189

Taux de réussite : 60%

Résultats

L'attaque CPA + attaque horizontale non profilée : **publiée en juillet sur la plate-forme du DPA Contest**[4].

Type	Nb de traces pour >80%	Temps/Trace (ms)
Non Profilée (DPA Contest)	**188**	**1000**
Record établi en mars	175	3500

Classée 2ème au moment de sa publication

[4] http://www.dpacontest.org/v4/42_hall_of_fame.php

Résultats

L'attaque CPA + attaque horizontale non profilée : **publiée en juillet sur la plate-forme du DPA Contest**[4].

Type	Nb de traces pour >80%	Temps/Trace (ms)
Non Profilée (DPA Contest)	**188**	**1000**
Record établi en mars	175	3500
Record établi fin juillet	14	60

Classée 2ème au moment de sa publication

[4] http://www.dpacontest.org/v4/42_hall_of_fame.php

Amélioration de la protection d'une implémentation d'AES

Modification du code assembleur du DPA Contest V4.2 pour le protéger contre l'attaque précédente **en conservant son efficacité**.

Performances obtenues[5] :

Algorithme	Nombre de cycles	Aléa généré (bits)
V4.2 non modifié	$16 \cdot 10^3$	192
V4.2 modifié	$17 \cdot 10^3$	192

[5] Sur un processeur ATmega328p

Empêcher les attaques horizontales sur l'offset

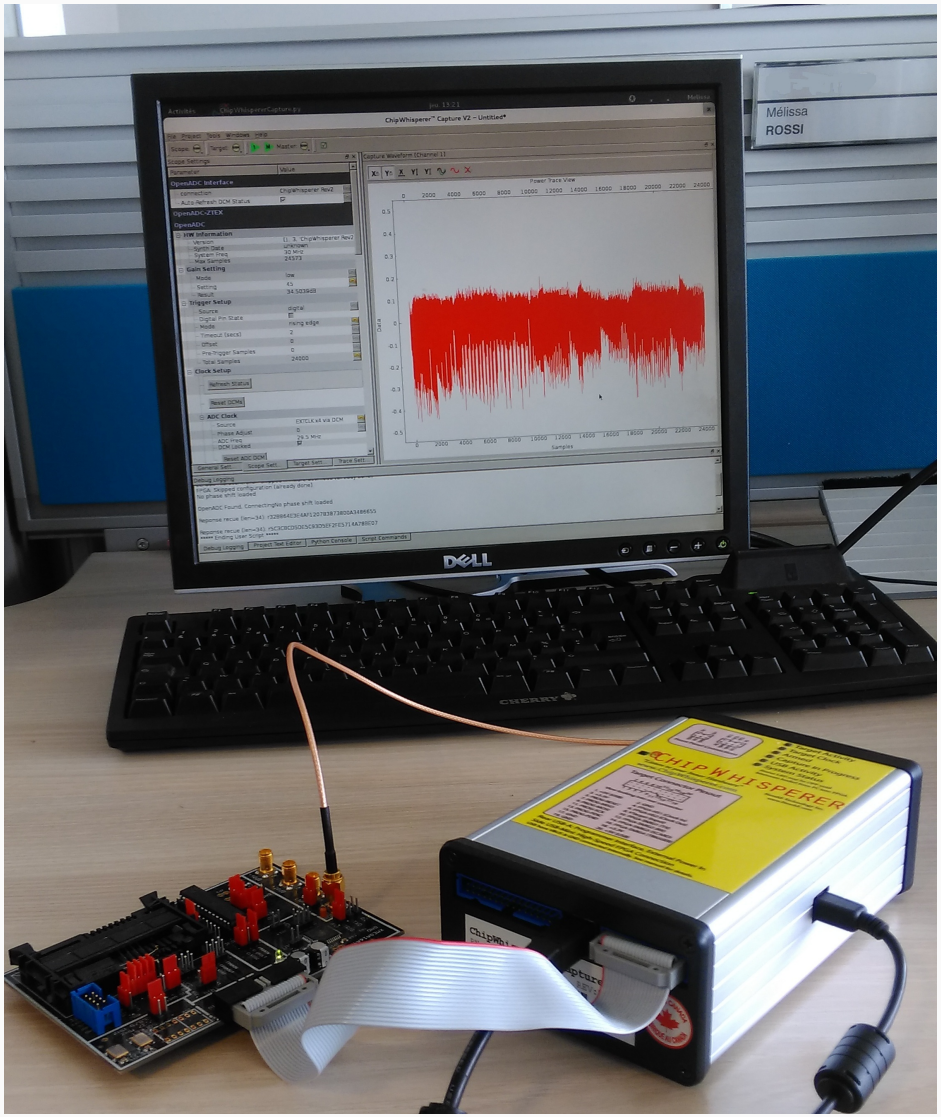
Manipulation d'un *offset* $\rightarrow$ Poids de Hamming vulnérable

Idée : **Poids constant**

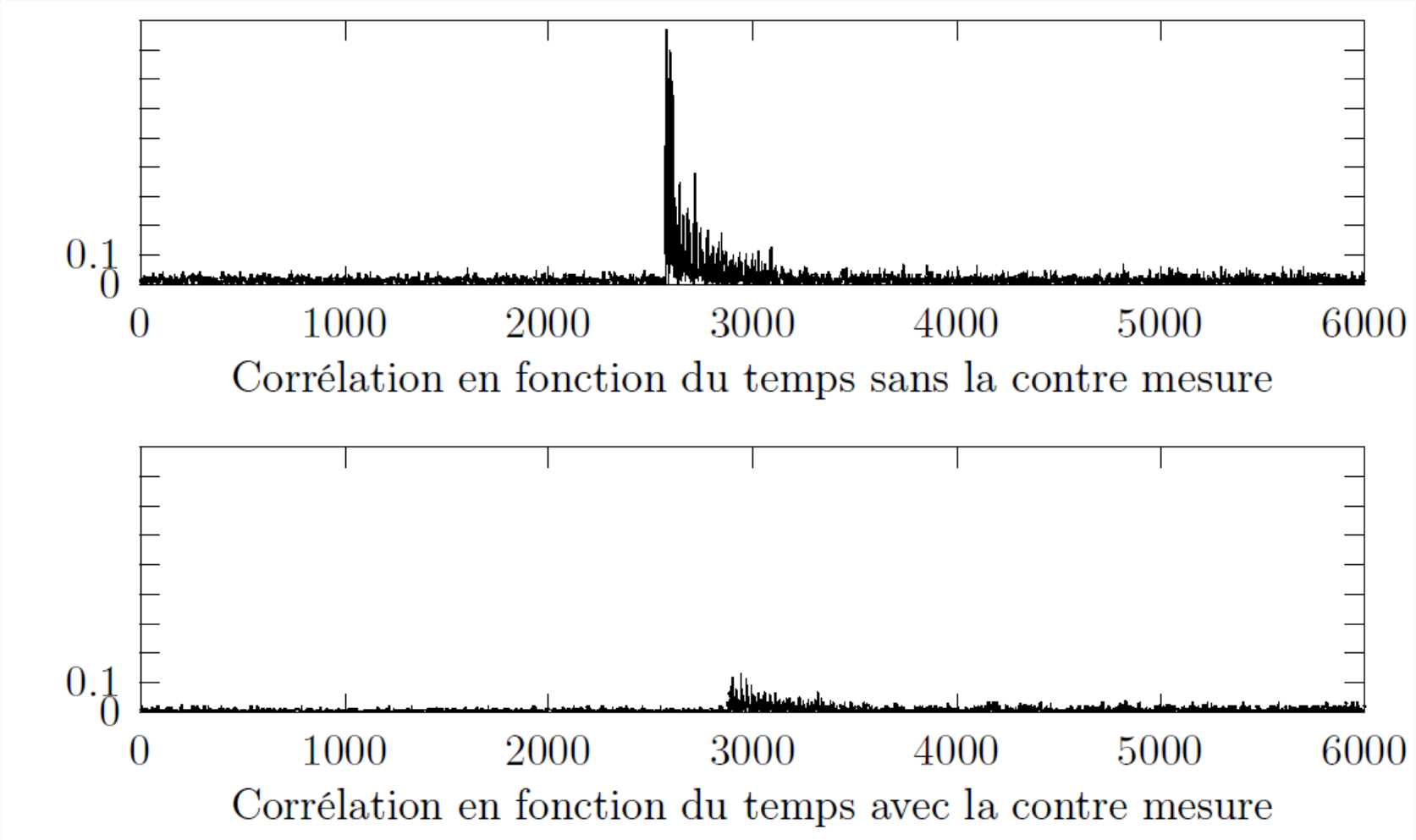
$0 \rightarrow 000111$	$2 \rightarrow 010011$	$9 \rightarrow 011010$	$11 \rightarrow 100110$
$15 \rightarrow 001011$	$14 \rightarrow 010101$	$5 \rightarrow 011100$	$13 \rightarrow 101001$
$6 \rightarrow 001101$	$12 \rightarrow 010110$	$7 \rightarrow 100011$	$4 \rightarrow 101010$
$10 \rightarrow 001110$	$3 \rightarrow 011001$	$1 \rightarrow 100101$	$8 \rightarrow 101100$

1. Fonction permettant de passer d'un *offset* au suivant sans repasser par les indices.
2. Modification des adresses et valeurs utilisées

ChipWhisperer



Apprentissage de l'incrémentation d'un offset (en 4000 traces)



CHES Challenge

CHES Challenge[6] : Concours d'attaques (juillet 2016 - août 2016) organisé pour la conférence CHES 2016

→ Schémas soumis aux attaquants pour tester leur résistance

Version	Nombre d'attaquants	Failles exploitées
Sans modification	4	*Offset* et *Shuffle* retrouvés et **attaque CPA**
Avec modification	3	Attaque par template au chargement de la clé **avant le chiffrement**

[6] https://ctf.newae.com/flags/

CHES Challenge[6] : Concours d'attaques (juillet 2016 - août 2016) organisé pour la conférence CHES 2016

→ Schémas soumis aux attaquants pour tester leur résistance

Version	Nombre d'attaquants	Failles exploitées
Sans modification	4	*Offset* et *Shuffle* retrouvés et **attaque CPA**
Avec modification	3	Attaque par template au chargement de la clé **avant le chiffrement**

Best Student CHES Challenge 2016

[6] https://ctf.newae.com/flags/

Perspectives

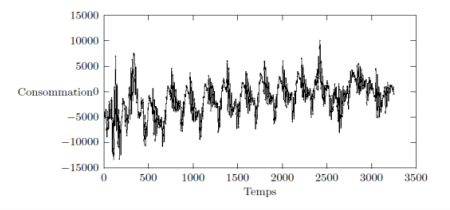
- Échange avec les membres du DPA Contest
- Étude de la sécurité d'une autre contre-mesure de masquage parmi celles publiées à CHES 2016[7].

[7] Par Dahmun Goudarzi et Matthieu Rivain

DPA Contest

CPA d'ordre 1

DPA Contest V2

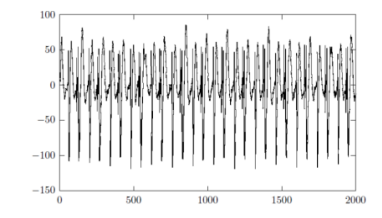


CPA d'ordre 1 avec attaque horizontale

Publication de l'attaque sur la plateforme du DPA Contest

DPA Contest V4.2

RSM et Shuffle

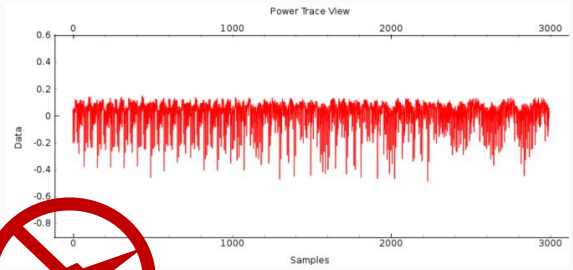


CHES Challenge

DPA Contest V4.2 modifié

Soumission aux attaquants

Mesures avec ChipWhiperer



Best Student prize

?

CPA d'ordre 1 avec attaque horizontale

Explication du coefficient de Pearson

$$\forall i\ X_i = \lambda \cdot\ Y_i + \beta \Leftrightarrow X_i - \overline{X} = \lambda \cdot (\ Y_i - \overline{Y})$$

Si on considère les vecteurs $\mathsf{U} = X - \overline{X}$ et $\mathsf{V} = \ Y - \overline{Y}$

La linéarité se mesure avec le cosinus de l'angle formé par les deux vecteurs. Plus ce cosinus se rapproche de +1 ou -1, plus la linéarité est vérifiée.

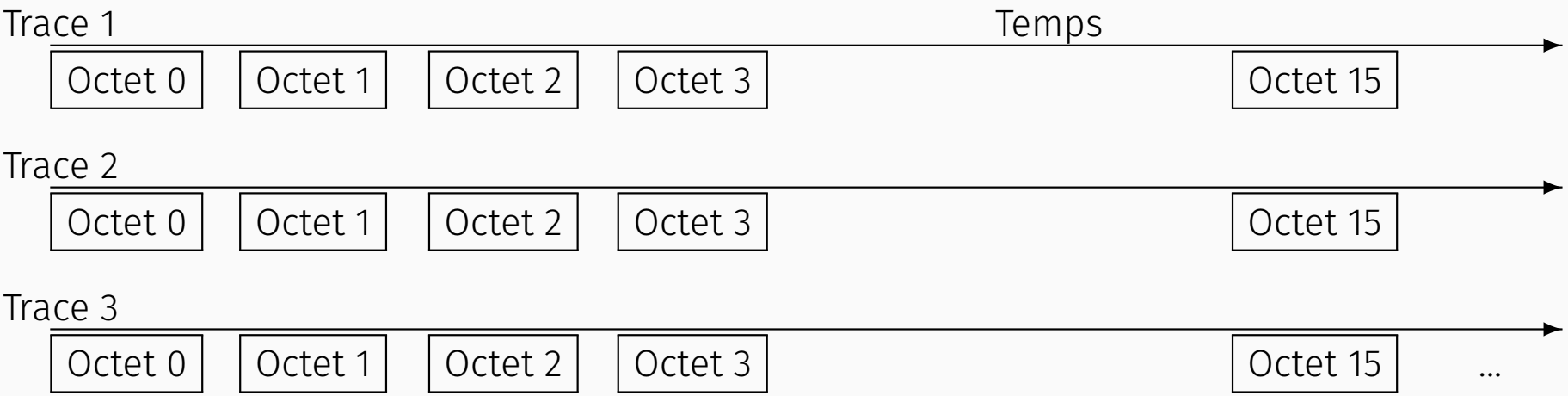
$$cos(\mathsf{U}, \mathsf{V}) = \frac{(\mathsf{U}|\mathsf{V})}{||\mathsf{U}||\cdot||\mathsf{V}||} = \frac{\sum(X_i - \overline{X})\cdot(\ Y_i - \overline{Y})}{\sum(X_i - \overline{X})^2 \cdot \sum(\ Y_i - \overline{Y})^2}$$

$$= \frac{cov(X, Y)}{\sigma_X \cdot \sigma_Y} = \rho(X,\ Y)$$

Shuffle : **Randomisation de l'ordre de traitement de l'état** pour une opération sensible donnée.

- Possible lorsque les octets de l'état sont traités d'une manière software les uns après les autres.

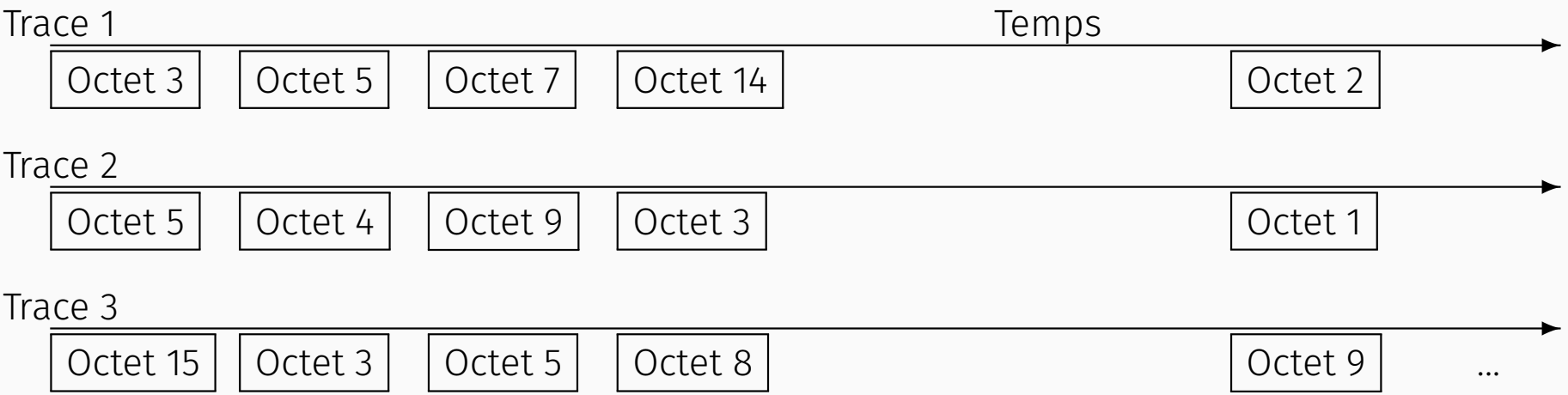
Exemple sans shuffle :



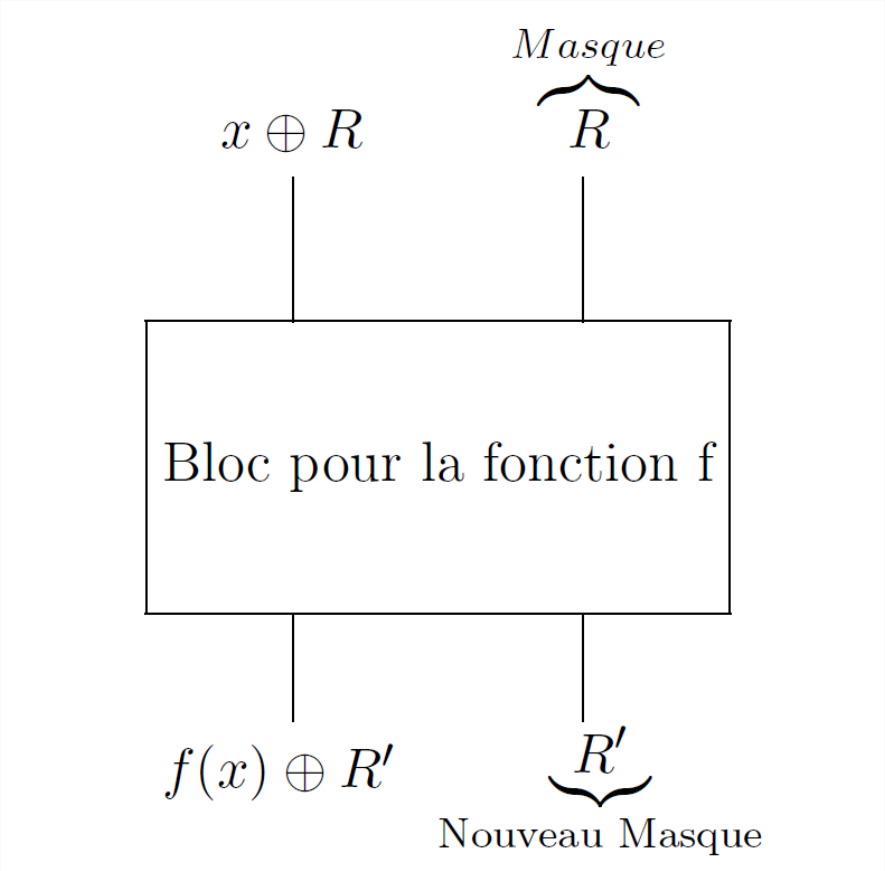
Shuffle : **Randomisation de l'ordre de traitement de l'état** pour une opération sensible donnée.

- Possible lorsque les octets de l'état sont traités d'une manière software les uns après les autres.

Exemple avec shuffle :



Dédoublement de la valeur d'entrée x : $x \oplus R$ (le masqué) et R (le masque).
Par exemple, pour un bloc f (Boîte S, *Shift-Rows, Mix Columns*) :



Contraintes Masquage d'ordre 1

L'intérieur d'un bloc **ne doit pas manipuler le $\oplus$ des deux entrées ni celui des deux sorties**.

- Opérations **linéaires** pour le $\oplus$: application au masqué et au masque en parallèle.
- Opérations **non linéaires**: plusieurs implémentations
  Exemple "look-up tables" pour boîtes S: pré-calcul des 256 S-box $S_i^*(\cdot) = S(\cdot \oplus i)$.
  L'opération consiste à définir un nouveau masque R' et à prendre $S^*_{x \oplus R}(R) \oplus R'$.

Retrouver les offsets : attaque profilée

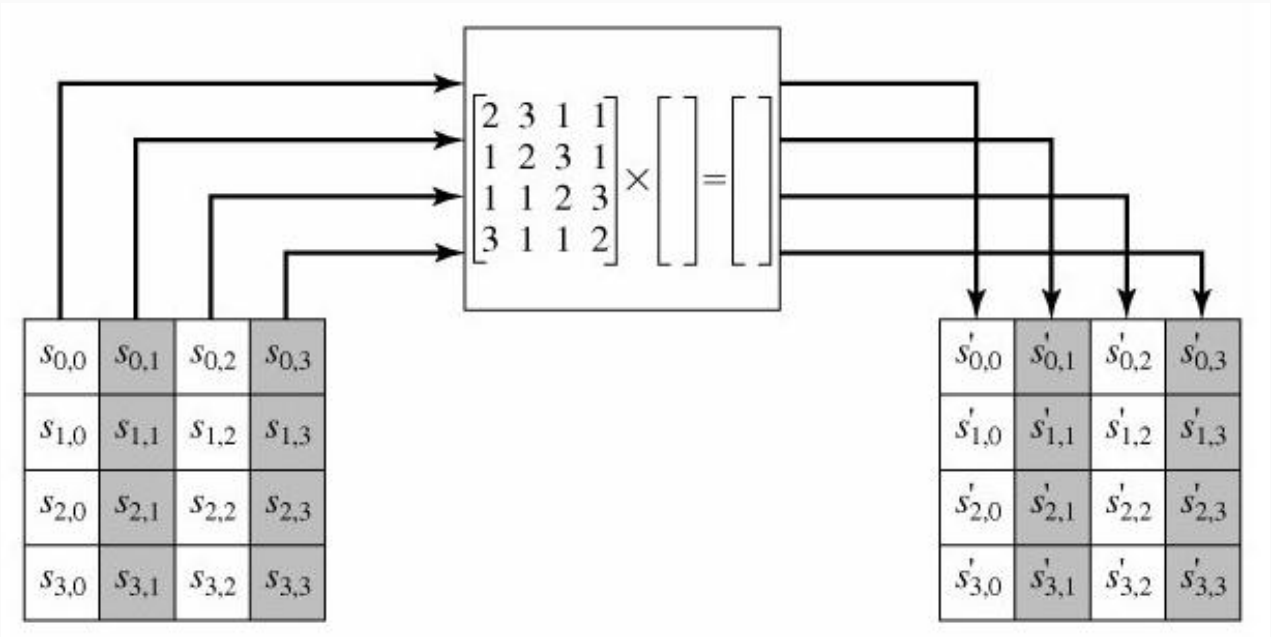
Une **attaque profilée** ou **attaque par template** : phase d'apprentissage sur un échantillon de courbes dont on connaît les *offsets* et *shuffle*.

- Template d'apprentissage qui stocke des spectres de Fourier moyens pour chaque tour et pour chaque octet.

Cette méthode a permis de retrouver l'*offset* avec **99% de réussite**.

Protection des autres étapes du premier tour

- *Mix Columns* modifié pour traiter les rangs de manière aléatoire. Moins coûteux que de randomiser l'ordre des octets.



- Les *Add Round Keys* et *Add Round Masks* randomisés avec le Shuffle

Exemples d'architectures de cartes