

# CPA — Comment ça fonctionne ?

Benjamin Grolleau

ENSEIRB-MATMECA, Bordeaux INP, France

benjamin.grolleau@bordeaux-inp.fr

**Abstract**—La Correlation Power Analysis (CPA) est aujourd’hui l’une des méthodes les plus efficaces pour extraire une clé secrète à partir de mesures de consommation d’un dispositif exécutant AES. Cette étude présente une attaque CPA complète appliquée au premier octet d’une clé AES-128, à partir d’un jeu de 100 traces de consommation synchronisées avec leurs plaintexts correspondants. Nous décrivons la construction du modèle de fuite fondé sur le poids de Hamming, la génération des hypothèses de clé, le centrage des données, puis le calcul des vecteurs de corrélation via le coefficient de Pearson. Les résultats montrent qu’un pic de corrélation clair permet d’isoler la vraie clé parmi les 256 hypothèses, et que la convergence de l’attaque apparaît dès quelques dizaines de traces. Enfin, nous introduisons une mesure de confiance fondée sur le contraste entre le meilleur et le second meilleur score, permettant d’évaluer la fiabilité du résultat obtenu.

## I. INTRODUCTION

Les attaques par analyse de consommation représentent aujourd’hui une menace majeure pour les implémentations matérielles d’algorithmes cryptographiques tels qu’AES. Parmi elles, la CPA — *Correlation Power Analysis* — est l’une des méthodes les plus puissantes et les plus utilisées. Contrairement à la DPA classique, qui repose sur une séparation binaire des traces selon un bit de fuite, la CPA exploite directement la corrélation statistique entre un modèle de fuite et les mesures physiques, ce qui la rend plus robuste au bruit et aux variations aléatoires.

Le principe consiste à construire, pour chaque hypothèse de clé, un modèle de fuite basé sur les données intermédiaires manipulées par l’algorithme (ici, la sortie de la SBox d’AES). Ce modèle est ensuite comparé aux traces de consommation mesurées à l’aide du coefficient de corrélation de Pearson. L’hypothèse de clé qui maximise cette corrélation est alors considérée comme la plus probable.

Dans ce document, nous détaillons la méthodologie permettant d’appliquer une attaque CPA sur un byte de la clé AES : génération des hypothèses, construction du modèle de fuite, normalisation, calcul des vecteurs de corrélation et analyse des scores. Nous introduisons également la notion de confiance associée aux résultats et examinons l’évolution de la corrélation en fonction du nombre de traces disponibles.

## II. DONNÉES D’ENTRÉE

Pour mener une attaque CPA sur l’implémentation AES étudiée, nous disposons de deux ensembles de données synchronisés : les traces de consommation mesurées et les textes d’entrée correspondants.

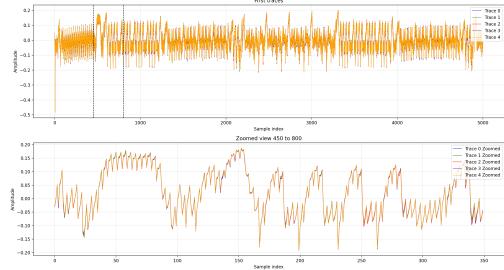


Figure 1. Exemple de traces de consommation utilisées pour l’attaque CPA.

Le premier ensemble est constitué de 100 traces de consommation — représenté en figure 1 — chacune contenant 5000 échantillons. Chaque trace représente l’évolution temporelle de la consommation électrique lors d’une exécution complète de l’algorithme AES. Toutes les exécutions utilisent la même clé secrète, mais des plaintexts différents, ce qui permet d’observer des variations dépendant des données intermédiaires manipulées par le chiffrement.

En parallèle, nous possédons 100 plaintexts de 16 octets, correspondant chacun à une trace. Cette correspondance un-à-un est essentielle : elle permet de reconstruire, pour chaque hypothèse de clé, les valeurs intermédiaires du premier tour d’AES et donc de générer le modèle de fuite nécessaire au calcul des corrélations.

Dans ce contexte, chaque trace peut être vue comme le résultat d’une fonction de fuite physique  $\mathcal{L}$ , appliquée à l’exécution de l’AES sur un texte d’entrée donné. On peut donc formaliser la relation entre plaintext, clé et trace de la manière suivante :

$$\text{trace}[i] = \mathcal{L}(\text{AES}(\text{secret\_key}, \text{textin}[i])) \quad (1)$$

## III. MÉTHODOLOGIE DE L’ATTAQUE CPA

L’objectif de la CPA est d’identifier la valeur de chaque octet de la clé AES en mesurant la corrélation entre un modèle de fuite théorique et les traces de consommation enregistrées. Contrairement à la DPA, qui repose sur une partition binaire des traces, la CPA exploite directement le coefficient de corrélation de Pearson, ce qui la rend plus robuste au bruit. Dans cette étude, nous ciblons le byte d’index 0, les autres octets pouvant être traités de manière analogue.

### A. Valeur intermédiaire ciblée

L'attaque CPA repose sur l'observation d'une fuite physique dépendant d'une opération interne d'AES manipulant à la fois les données d'entrée et la clé secrète. La première opération non linéaire de l'algorithme, l'application de la SBox durant la phase *SubBytes*, constitue une cible privilégiée car elle amplifie la dépendance entre plaintext et clé.

Pour un octet d'index  $i$ , la valeur intermédiaire manipulée par l'implémentation est :

$$v = \text{SBox}(\text{plaintext}[i] \oplus \text{key}[i]). \quad (2)$$

Cette transformation est idéale pour une attaque CPA pour plusieurs raisons :

- elle dépend directement de la clé secrète via l'opération XOR,
- elle s'applique à chaque exécution AES avec un plaintext différent, fournissant une variabilité statistique exploitable,
- l'opération SBox introduit une non-linéarité rendant les données intermédiaires plus sensibles au bruit contenu dans les mesures,
- dans les microcontrôleurs CMOS, l'écriture de  $v$  dans un registre déclenche une consommation proportionnelle au nombre de bits commutés, ce qui justifie l'utilisation d'un modèle de fuite fondé sur le poids de Hamming.

Dans cette étude, nous ciblons spécifiquement l'octet d'index 0. Pour chaque trace de consommation, cet octet est donc calculé à partir de  $\text{plaintext}[0]$ .

### B. Construction des hypothèses de fuite

Pour retrouver  $\text{key}[0]$ , nous testons les 256 hypothèses  $g \in [0x00; 0x01; \dots; 0xFF]$ .

Pour chaque trace  $k$ , nous calculons :

$$v_k(g) = \text{SBox}(\text{plaintext}[k][0] \oplus g), \quad (3)$$

puis on applique le modèle de fuite retenu, fondé sur le poids de Hamming :

$$\text{hyp}_g[k] = \text{HW}(v_k(g)). \quad (4)$$

Chaque hypothèse  $g$  génère ainsi un vecteur de taille égale au nombre de traces (ici 100). Ce vecteur représente la consommation que l'on s'attendrait à observer si la clé réelle correspondait effectivement à  $g$ .

La figure 2 montre l'hypothèse de fuite pour la vraie clé (0x2b) et une clé incorrecte (0x90).

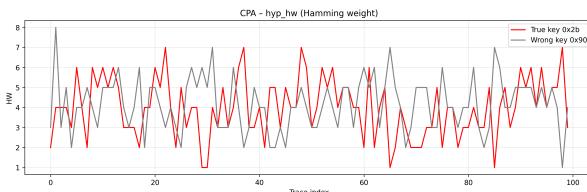


Figure 2. Modèle HW pour la vraie clé (rouge) et une clé incorrecte (gris).

### C. Centrage du modèle de fuite

Avant le calcul de la corrélation, les hypothèses de fuite doivent être recentrées autour de zéro. Ce pré-traitement élimine le biais constant présent dans le poids de Hamming et garantit que la corrélation ne dépend que des variations du modèle, et non de sa moyenne. Pour chaque hypothèse  $g$ , le centrage est défini par :

$$\text{hyp}_{g,c}[k] = \text{hyp}_g[k] - \text{mean}(\text{hyp}_g). \quad (5)$$

On note  $\text{hyp}_g[k]$  l'hypothèse pour le texte d'incice  $k$ , pour le guess  $g$ .  $\text{hyp}_{g,c}[k]$  représente cette même hypothèse mais centré ( $c$ ).

Ce centrage supprime le biais statique et conduit au signal illustré en figure 3.

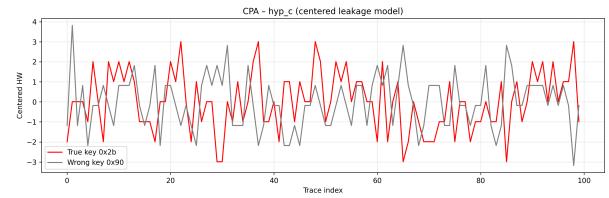


Figure 3. Hypothèses de fuite centrées pour la vraie clé et une clé incorrecte.

### D. Centrage des traces de consommation

De la même manière que pour les hypothèses de fuite, les traces de consommation doivent être centrées avant le calcul de la corrélation. Ce centrage est effectué indépendamment pour chaque échantillon temporel afin de supprimer la composante continue du signal de mesure (offset analogique, bruit de fond, dérive de l'alimentation, etc.).

Les traces sont également centrées colonne par colonne :

$$\text{traces}_c[k][t] = \text{traces}[k][t] - \text{mean}_k(\text{traces}[k][t]). \quad (6)$$

Ce prétraitement garantit que la corrélation ne dépend que des variations dynamiques liées à l'activité CMOS et que les différences d'offset entre traces n'influencent pas les résultats. Après ce centrage, chaque colonne temporelle présente une moyenne nulle, ce qui permet un calcul de Pearson strictement basé sur la similarité des formes temporelles et non sur leur amplitude absolue.

### E. Calcul du vecteur de corrélation

Une fois les hypothèses de fuite et les traces centrées, la CPA consiste à mesurer la similarité linéaire entre les variations du modèle et celles observées dans les traces de consommation. Cette similarité est quantifiée pour chaque hypothèse de clé  $g$  à l'aide du coefficient de corrélation de Pearson, calculé indépendamment pour chaque instant temporel  $t$ .

Pour une hypothèse donnée  $g$ , on définit donc le vecteur de corrélation  $\rho$  :

$$\rho(g, t) = \frac{\sum_k \text{hyp}_{g,c}[k] \cdot \text{traces}_c[k][t]}{\sqrt{\sum_k \text{hyp}_{g,c}[k]^2} \sqrt{\sum_k \text{traces}_c[k][t]^2}}. \quad (7)$$

Chaque hypothèse produit ainsi un vecteur  $\rho(g, t)$  de longueur 5000. Ce calcul présente plusieurs propriétés essentielles pour l'attaque :

- $\rho(g, t) \approx 0$  indique qu'il n'existe aucune relation linéaire entre la fuite modélisée et la consommation observée,
- $\rho(g, t)$  proche de 1 ou  $-1$  traduit une corrélation forte, révélant que l'instant  $t$  dépend fortement de la valeur intermédiaire ciblée,
- seule la bonne hypothèse de clé doit produire un pic net de corrélation autour du moment où AES manipule  $v$ .

L'ensemble des corrélations pour les 256 hypothèses est représenté sous forme de carte de chaleur en figure 4. Chaque ligne correspond à un guess, chaque colonne à un échantillon temporel.

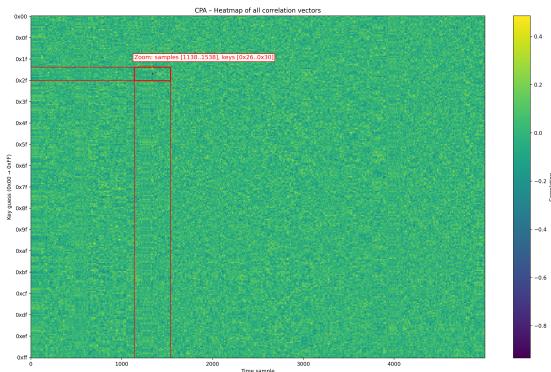


Figure 4. Carte de chaleur des corrélations pour les 256 hypothèses de clé. Une zone structurée n'apparaît que pour la clé correcte.

On peut observer ce fameux pique de chaleur qui est encadré, montrant qu'une corrélation a été trouvée.

#### F. Analyse du vecteur de corrélation

Une fois les corrélations calculées pour chaque hypothèse, il est possible d'examiner plus finement la forme des vecteurs  $\rho(g, t)$  afin d'identifier la clé correcte. La figure 5 représente, pour les 256 hypothèses de clé, leur vecteur de corrélation associé. Les 255 clés incorrectes sont affichées en gris, tandis que la clé correcte est mise en évidence en rouge.

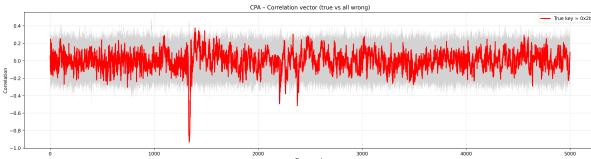


Figure 5. Vecteur de corrélation pour la vraie clé (rouge) et les 255 hypothèses incorrectes (gris).

On observe que :

- les hypothèses incorrectes produisent des vecteurs bruités, centrés autour de zéro, sans structure identifiable,
- aucune d'entre elles ne présente de motif temporel cohérent, ce qui traduit l'absence totale de relation linéaire entre leur modèle de fuite et les traces,

- en revanche, la clé correcte génère un vecteur qui se distingue nettement du bruit.

Un zoom sur la région temporelle comprenant la fuite est présenté en figure 6.

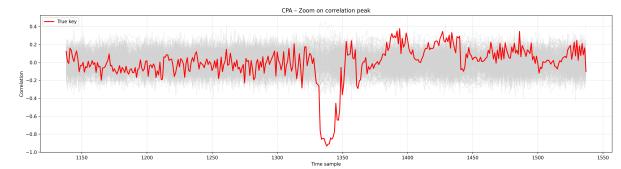


Figure 6. Zoom sur la zone de corrélation maximale révélant la fuite.

Ce pic constitue la signature attendue dans une attaque CPA :

- il apparaît exactement à l'instant où AES manipule la valeur intermédiaire  $v$ ,
- son amplitude est nettement supérieure à celle des hypothèses incorrectes,
- son signe n'a pas d'importance (corrélation positive ou négative), car seule l'amplitude compte.

Ainsi, même en présence de bruit, la clé correcte émerge clairement, ce qui illustre la puissance de la CPA et son efficacité sur des traces réelles.

#### G. Définition du score et sélection de la clé

Une fois les vecteurs de corrélation  $\rho(g, t)$  calculés pour les 256 hypothèses de clé, nous associons à chaque vecteur une valeur unique afin de comparer efficacement les hypothèses. En CPA, ce résumé est simplement le maximum absolu de la corrélation :

$$\text{score}(g) = \max_t |\rho(g, t)|. \quad (8)$$

Ce score correspond à la plus forte corrélation observée entre le modèle de fuite de l'hypothèse  $g$  et les traces mesurées.

- si  $g$  est incorrect, les valeurs de  $\rho(g, t)$  restent proches de zéro, donc  $\text{score}(g)$  est faible,
- si  $g$  est la bonne clé, un pic significatif apparaît autour de l'instant où AES manipule la valeur intermédiaire ciblée, donnant un score nettement plus élevé.

La figure 7 présente la courbe des scores CPA pour toutes les hypothèses entre 0x00 et 0xFF. La clé correcte (0x2b) produit un pic dominant, largement supérieur aux mauvaises clés.

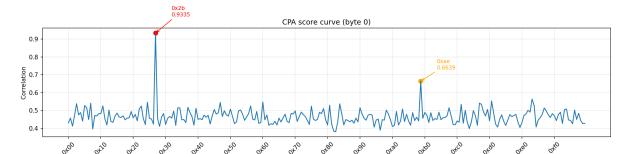


Figure 7. Score CPA pour les 256 hypothèses de clé : la clé correcte produit un pic dominant.

Cette étape conclut l'attaque CPA pour un byte donné : une seule valeur se détache de manière claire et robuste, ce qui

confirme l'efficacité du modèle de fuite et la qualité du calcul de corrélation.

#### IV. TAUX DE CONFIANCE DES RÉSULTATS

L'analyse du score permet d'identifier l'hypothèse de clé la plus probable, mais il est également utile de quantifier la confiance associée à ce résultat. En effet, il arrive que le score maximal soit bien supérieur au bruit — ce qui indique une estimation très fiable — mais dans certains cas plusieurs hypothèses présentent des scores proches, rendant le choix moins certain.

La figure 7 illustre, pour le byte analysé, que l'hypothèse  $0x2b$  génère un pic de corrélation nettement dominant. Cependant, un second candidat ( $0xb2$ ) se distingue légèrement du bruit de fond, bien que beaucoup plus faiblement.

Pour formaliser cette notion d'écart entre les deux meilleurs candidats, nous définissons un taux de confiance fondé sur le contraste entre le meilleur score :  $\text{best\_score}$ , et le second meilleur :  $\text{second\_score}$ .

Nous introduisons d'abord une mesure de contraste :

$$\text{contrast} = \frac{\text{best\_score} - \text{second\_score}}{\max(\text{second\_score}, , 10^{-15})}, \quad (9)$$

puis nous ramenons cette grandeur dans l'intervalle  $[0, 1]$  afin d'obtenir un taux de confiance exploitable :

$$\text{confidence} = \min(\max(\text{contrast}, , 0), , 0.90). \quad (10)$$

Une valeur proche de 1 indique que l'hypothèse de clé correcte se détache nettement — ce qui est le cas lorsque la CPA produit un pic clair et isolé — tandis qu'une valeur faible reflète une incertitude due à une séparation statistique insuffisante entre les deux meilleurs candidats.

#### V. TAUX DE CONVERGENCE

Pour évaluer la stabilité et la fiabilité de l'attaque CPA, il est intéressant d'observer comment la corrélation évolue lorsque l'on augmente progressivement le nombre de traces utilisées. L'objectif est de vérifier à partir de combien de traces la bonne clé devient clairement dominante par rapport aux autres hypothèses.

Comme pour la DPA, nous testons différents sous-ensembles de tailles croissantes :

$$N = \{k = 10|k = 1; 2; \dots; 10\} \quad (11)$$

Pour chaque valeur de  $N$ , on recalcule entièrement les scores CPA pour les 256 hypothèses. On obtient donc une courbe de convergence montrant comment le score de la vraie clé se détache progressivement du bruit généré par les mauvaises clés.

Dans la pratique, on observe généralement :

Pour un faible nombre de traces ( $N < 20$ ), la corrélation est trop faible, et aucune hypothèse ne se distingue réellement. Le score reste noyé dans le bruit.

À partir de 30 traces, la corrélation de la vraie clé commence à se démarquer et dominer de manière faible.

Au delà de 30 traces, la clé correcte devient clairement la meilleure hypothèse. Le pic de corrélation est stable, reproduit et nettement supérieur aux autres.

Cette évolution est présentée figure 8, où la clé  $0x2b$  finit par s'imposer de manière systématique au-delà d'un certain seuil de traces.

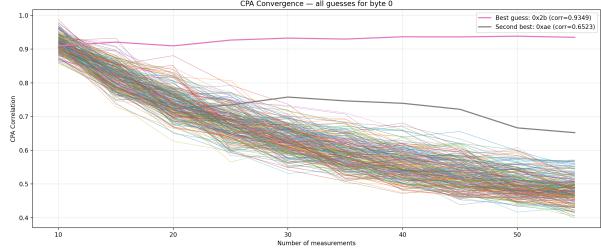


Figure 8. Convergence des scores CPA en fonction du nombre de traces utilisées. La clé correcte finit par dominer clairement lorsque  $N$  devient suffisamment grand.

#### VI. CONCLUSION

Dans ce document, nous avons réalisé une attaque CPA complète sur un byte de la clé AES-128, en exploitant la corrélation entre un modèle de fuite théorique et les traces de consommation mesurées. Grâce au modèle fondé sur le poids de Hamming et au coefficient de Pearson, la clé correcte apparaît nettement, avec un pic de corrélation localisé au moment où la SBox est évaluée. Les résultats montrent également que l'attaque converge rapidement : quelques dizaines de traces suffisent pour voir émerger la bonne hypothèse.

La mesure de confiance introduite permet quant à elle de quantifier la séparation entre la meilleure clé et les autres hypothèses, ce qui est utile pour détecter les cas ambigus ou les erreurs de classement.

Enfin, la sortie du programme (figure 9) confirme l'identification correcte de l'ensemble de la clé, ainsi que les valeurs de score, de confiance et la cohérence des résultats obtenus.

== CPA BYTE SUMMARY ==						
Byte	Guess	Correct	Status	Confidence	Second Best	
0	0x2b	0x2b	OK	92.40%		
1	0x7e	0x7e	OK	100.00%		
2	0x15	0x15	OK	88.12%		
3	0x16	0x16	OK	85.76%		
4	0x28	0x28	OK	88.86%		
5	0xae	0xae	OK	84.90%		
6	0xd2	0xd2	OK	86.72%		
7	0xa6	0xa6	OK	72.50%		
8	0xab	0xab	OK	67.33%		
9	0xf7	0xf7	OK	63.53%		
10	0x15	0x15	OK	89.13%		
11	0x88	0x88	OK	80.76%		
12	0x09	0x09	OK	95.56%		
13	0xcf	0xcf	OK	98.42%		
14	0x4f	0x4f	OK	94.11%		
15	0x3c	0x3c	OK	80.67%		

Figure 9. Sortie du programme CPA montrant le byte de clé reconstruit, son score et le taux de confiance associé.