On the Physical Security of Cryptographic Implementations

Soutenance de thèse de Matthieu Rivain

Oberthur Technologies & Université du Luxembourg

Plan

1. Introduction
2. Attaques par canaux auxiliaires
   - Modèle
   - Méthodes classiques
   - Masquage et attaques d'ordre supérieur
3. Schémas de masquage
   - Schémas de premier ordre
   - **Cryptanalyse d'un schéma d'ordre supérieur**
   - Schémas d'ordre supérieur
4. Attaques par fautes
   - **Sécurisation du RSA par double chaîne d'additions**
   - Autres travaux
5. Conclusion

Plan

1. Introduction
2. Attaques par canaux auxiliaires
   - Modèle
   - Méthodes classiques
   - Masquage et attaques d'ordre supérieur
3. Schémas de masquage
   - Schémas de premier ordre
   - **Cryptanalyse d'un schéma d'ordre supérieur**
   - Schémas d'ordre supérieur
4. Attaques par fautes
   - **Sécurisation du RSA par double chaîne d'additions**
   - Autres travaux
5. Conclusion

Introduction à la cryptographie

- But de la cryptographie : sécuriser les communications
- Services de sécurité :
  - confidentialité
  - intégrité
  - authenticité
  - signature numérique

Introduction à la cryptographie

- But de la cryptographie : sécuriser les communications
- Services de sécurité :
  - confidentialité
  - intégrité
  - authenticité
  - signature numérique
- Exemple : les transactions bancaires

Introduction à la cryptographie

- Algorithmes de chiffrement publics paramétrés par une *clé secrète*

Introduction à la cryptographie

- Algorithmes de chiffrement publics paramétrés par une *clé secrète*
- Cryptographie symétrique
  - une clé secrète SK pour chiffrer et déchiffrer
  - chiffrements par blocs *e.g.* DES, AES

Introduction à la cryptographie

- Algorithmes de chiffrement publics paramétrés par une *clé secrète*
- Cryptographie symétrique
  - une clé secrète SK pour chiffrer et déchiffrer
  - chiffrements par blocs *e.g.* DES, AES
- Cryptographie asymétrique
  - une clé publique PK et une clé privée SK
  - chiffrement avec PK, déchiffrement avec SK
  - signature avec SK, vérification avec PK
  - *e.g.* cryptosystème RSA

Introduction à la cryptographie

- Algorithmes de chiffrement publics paramétrés par une *clé secrète*
- Cryptographie symétrique
  - une clé secrète SK pour chiffrer et déchiffrer
  - chiffrements par blocs *e.g.* DES, AES
- Cryptographie asymétrique
  - une clé publique PK et une clé privée SK
  - chiffrement avec PK, déchiffrement avec SK
  - signature avec SK, vérification avec PK
  - *e.g.* cryptosystème RSA
- La sécurité repose sur le secret de la clé SK

Problème du stockage des clés

Problème

Où stocker une clé secrète ?

Problème du stockage des clés

Problème

Où stocker une clé secrète ?

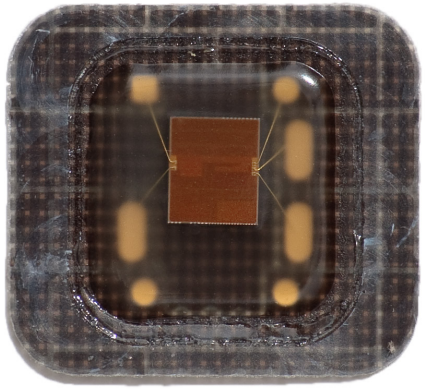
Solution

Dans une carte à puce !

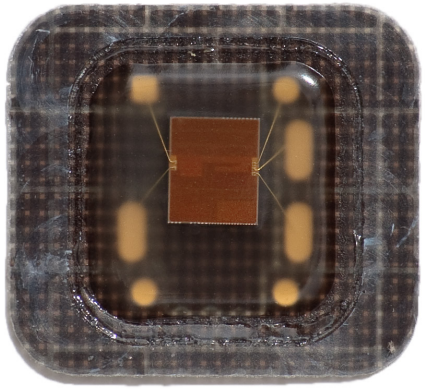
- pas besoin de mémoriser la clé
- support facile à transporter
- sécurité : la clé ne sort jamais de la carte

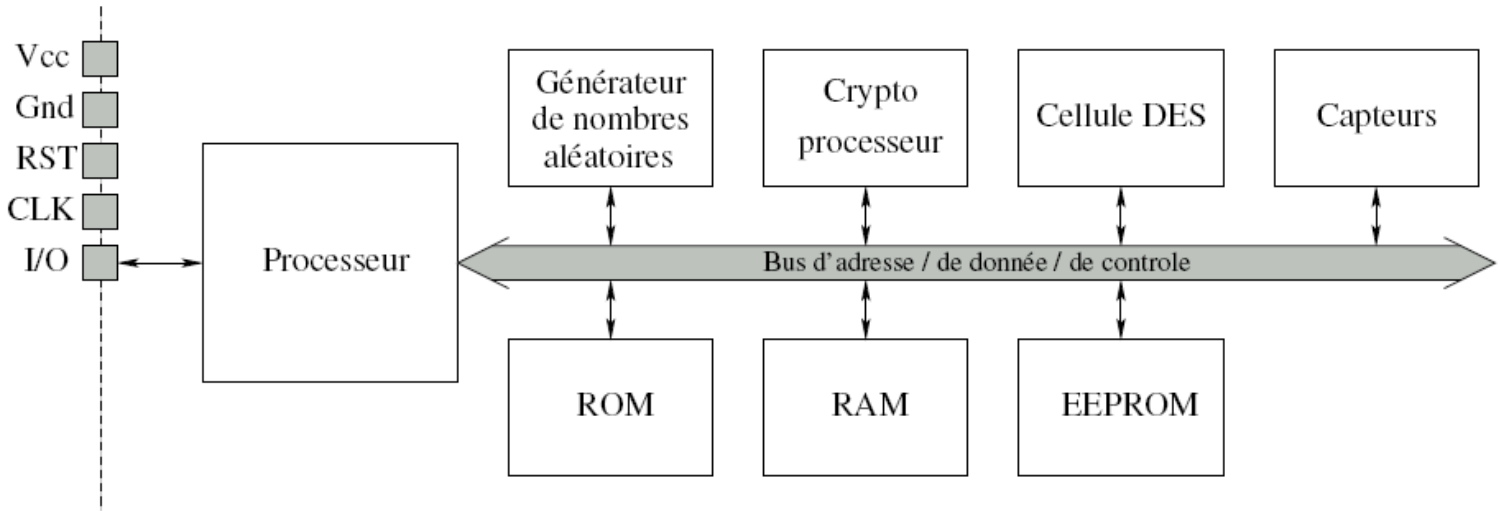
La carte à puce

La carte à puce



La carte à puce





Plan

1 Introduction

2 Attaques par canaux auxiliaires
- Modèle
- Méthodes classiques
- Masquage et attaques d'ordre supérieur

3 Schémas de masquage
- Schémas de premier ordre
- **Cryptanalyse d'un schéma d'ordre supérieur**
- Schémas d'ordre supérieur

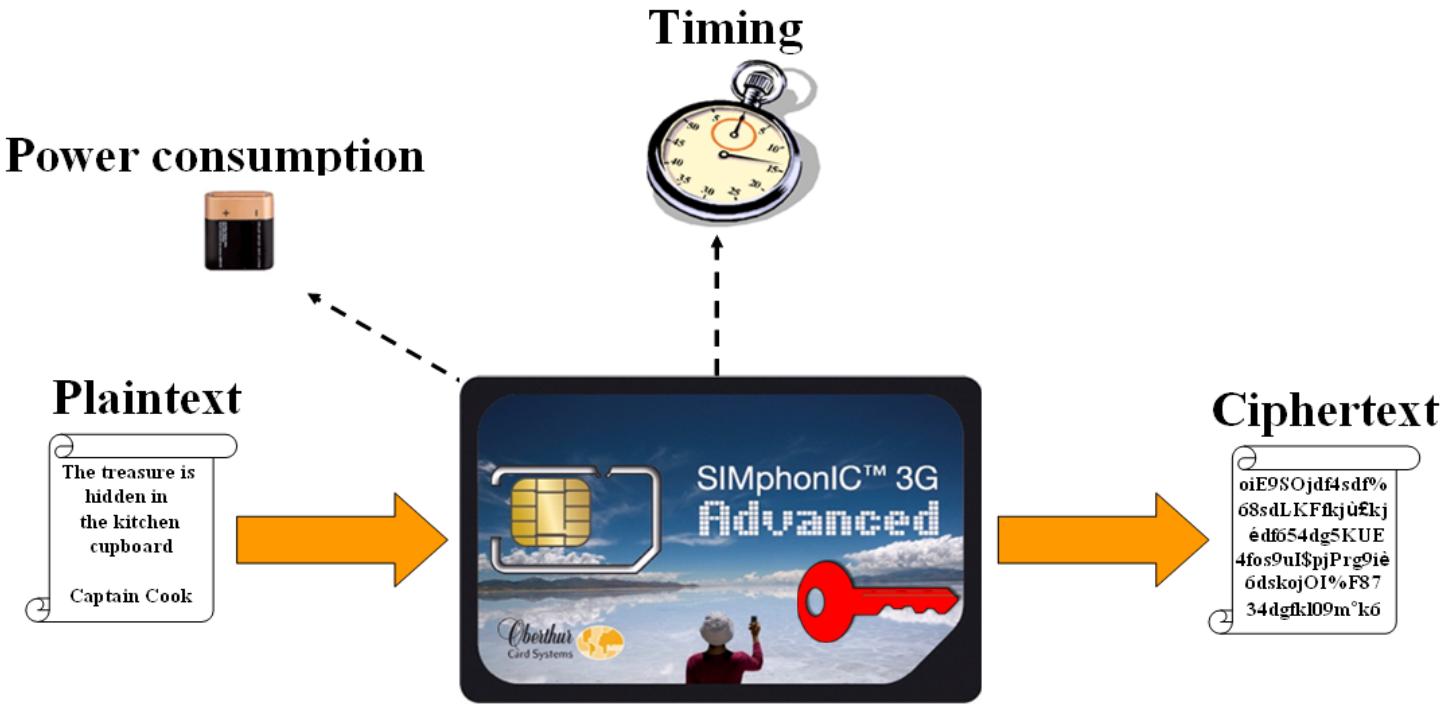
4 Attaques par fautes
- **Sécurisation du RSA par double chaîne d'additions**
- Autres travaux

5 Conclusion

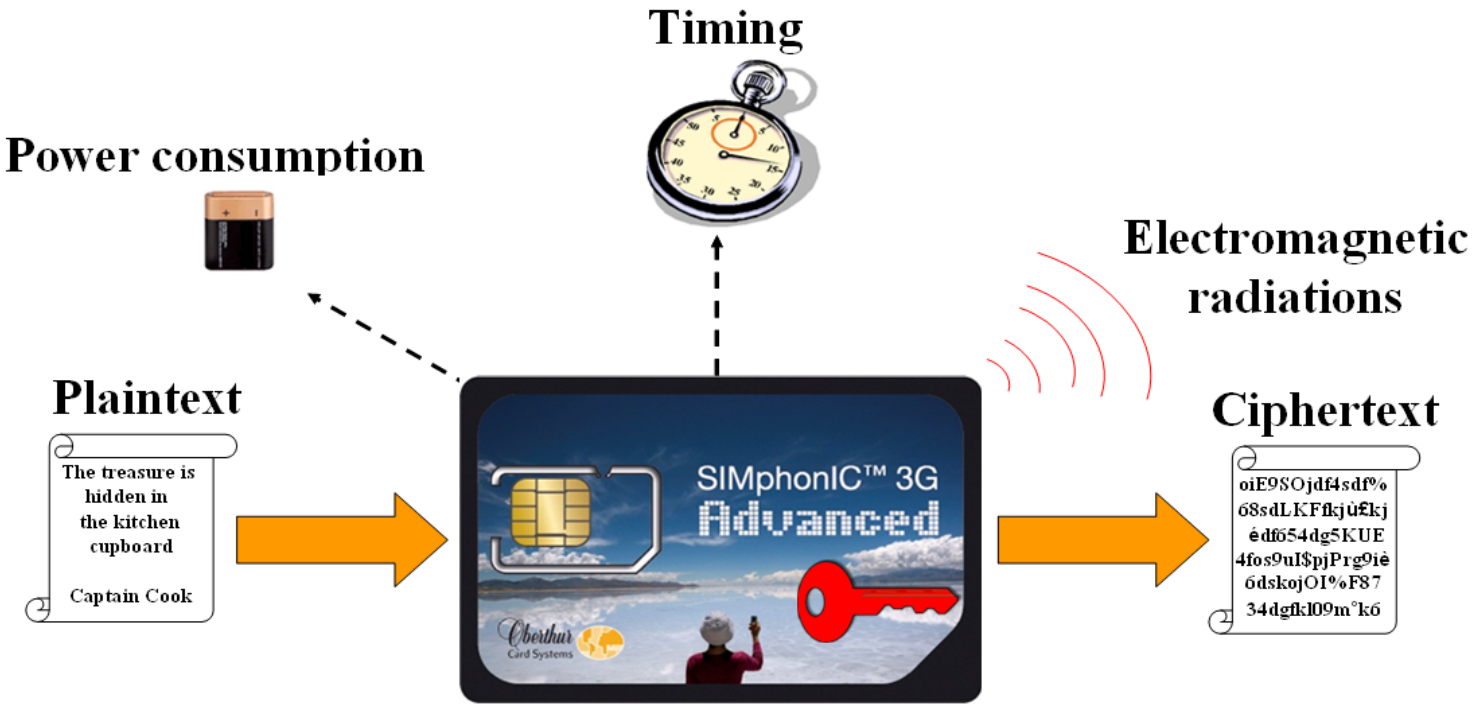
Attaques par canaux auxiliaires

Attaques par canaux auxiliaires

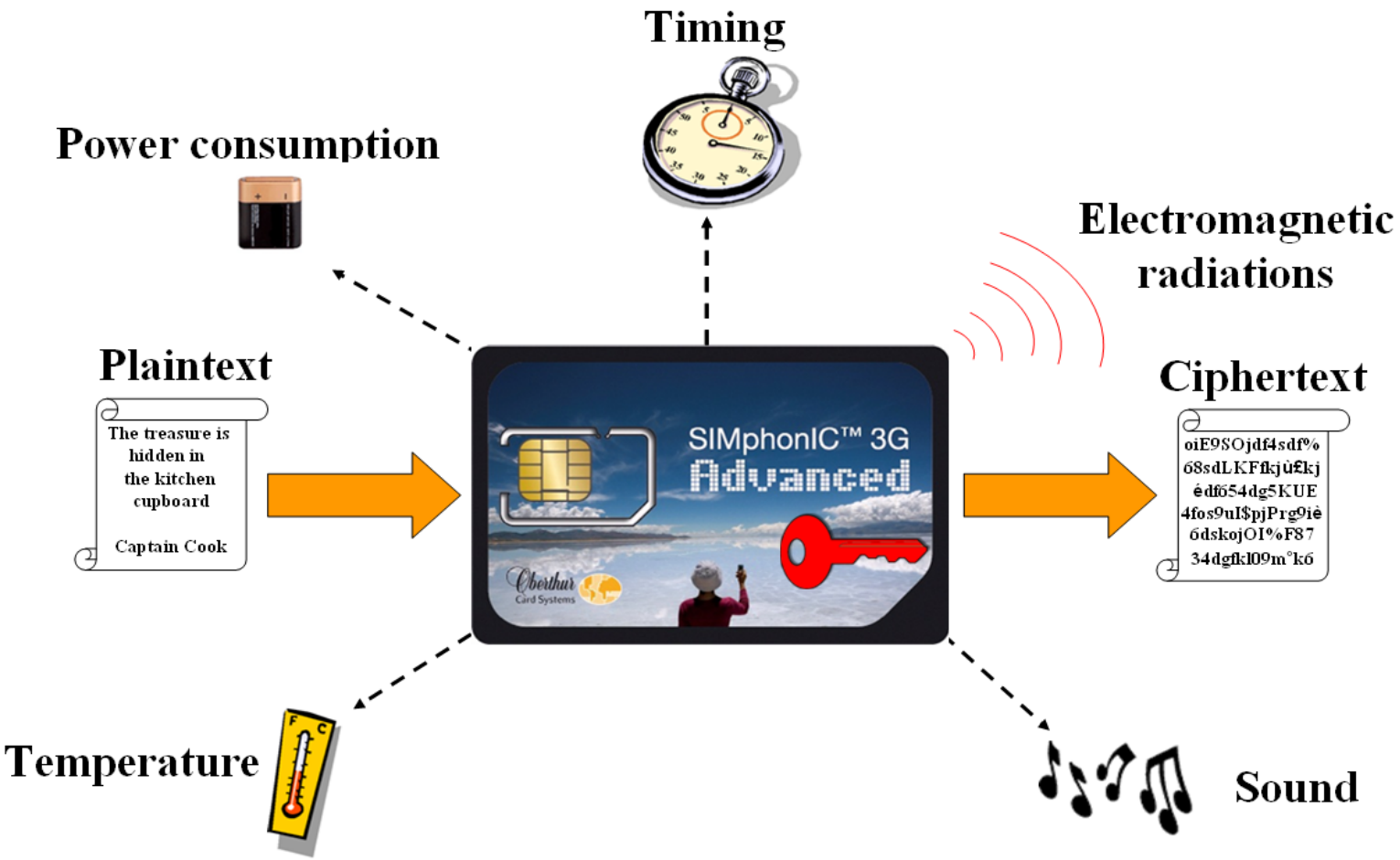
Attaques par canaux auxiliaires



Attaques par canaux auxiliaires

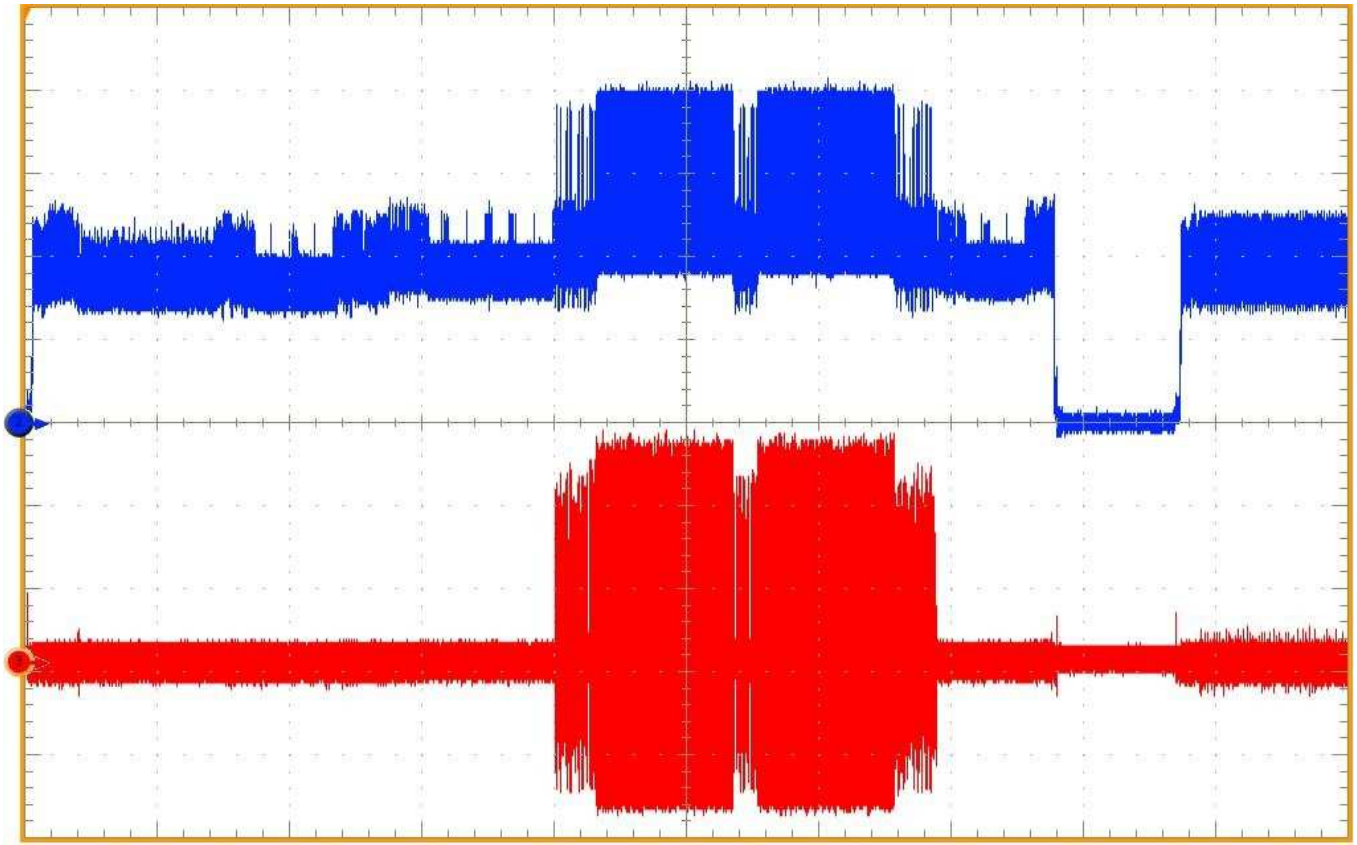


Attaques par canaux auxiliaires

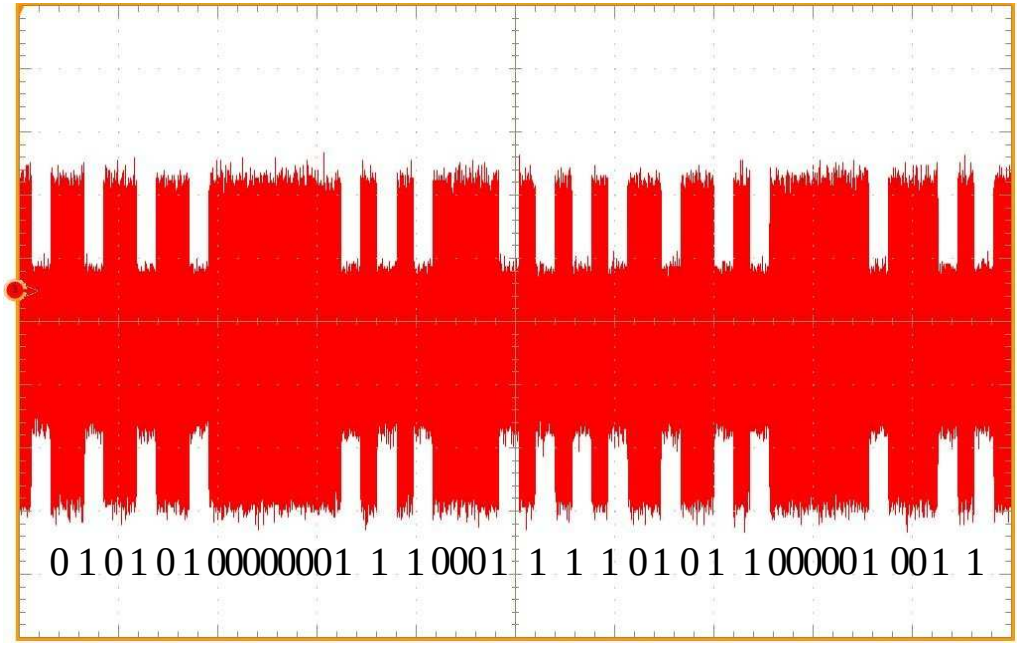


Attaques par canaux auxiliaires

- Exemple : consommation électrique et ondes électromagnétiques pendant un calcul RSA

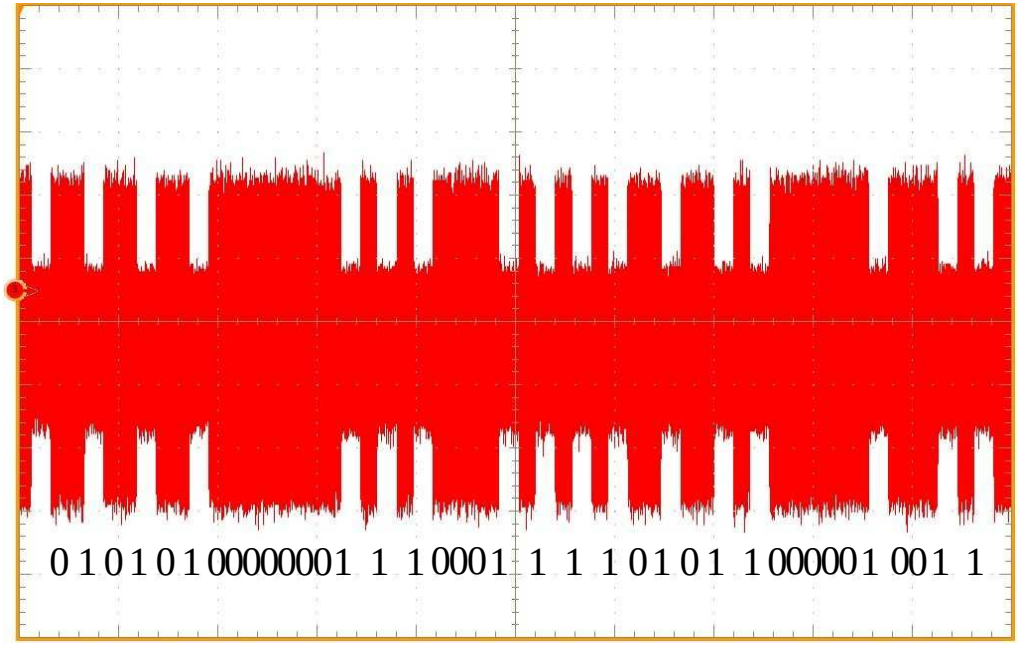


Analyse simple (SPA)



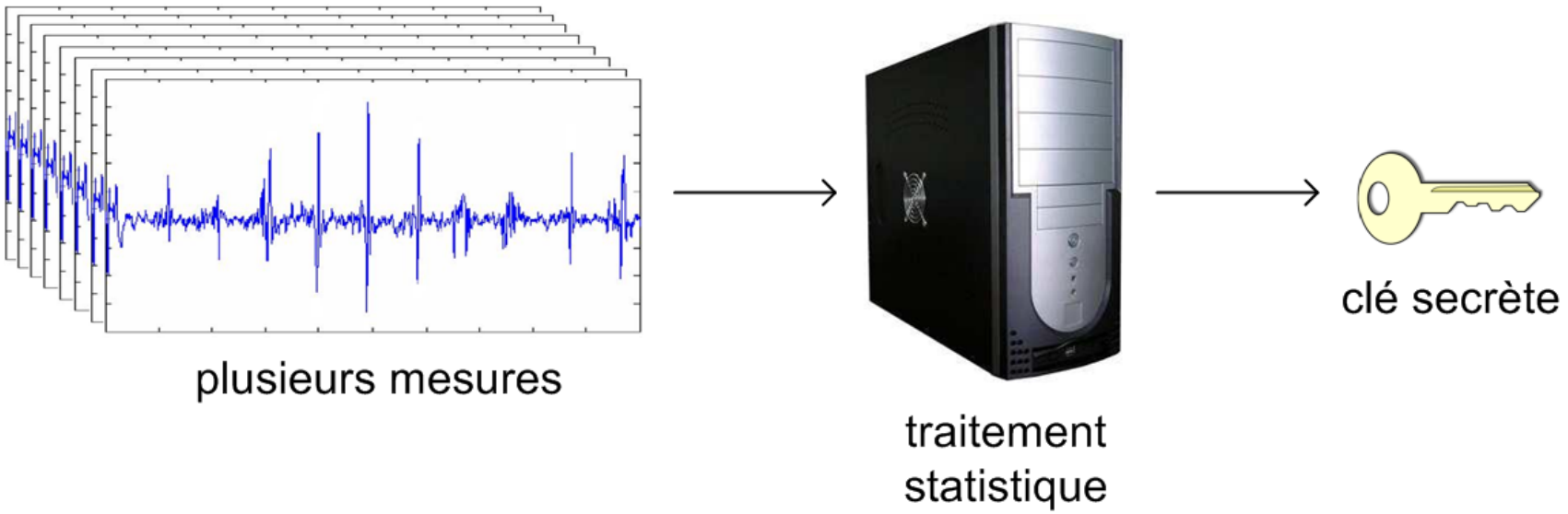
- Analyse le flot d'opérations
- Une seule mesure suffit (en général) pour retrouver la clé

Analyse simple (SPA)



- Analyse le flot d'opérations
- Une seule mesure suffit (en général) pour retrouver la clé
- Contre-mesure : implémenter les algorithmes de manière *atomique*

Analyse statistique



- Cible les données manipulées
- Traitement statistique sur plusieurs mesures
- Difficile à contrecarrer

Modèle

- Variable intermédiaire : $Z = f(X, K)$
  - X : valeur publique
  - K : valeur secrète

Modèle

- Variable intermédiaire : $Z = f(X, K)$
  - X : valeur publique
  - K : valeur secrète
- Fuite physique : L
  - L dépend statistiquement de Z :

$$L = \varphi(Z) + B$$

Modèle

- Variable intermédiaire : $Z = f(X, K)$
  - X : valeur publique
  - K : valeur secrète
- Fuite physique : L
  - L dépend statistiquement de Z :

$$L = \varphi(Z) + B$$

- Application d'un distingueur :

$$(d_k)_{k \in \mathcal{K}} \leftarrow \mathrm{D}(L, X)$$

Modèle

- Variable intermédiaire : $Z = f(X, K)$
  - X : valeur publique
  - K : valeur secrète
- Fuite physique : L
  - L dépend statistiquement de Z :

$$L = \varphi(Z) + B$$

- Application d'un distingueur :

$$(d_k)_{k \in \mathcal{K}} \leftarrow \mathrm{D}(L, X)$$

- On s'attend à avoir : $K = \mathrm{argmax}_{k \in \mathcal{K}}\ d_k$

Analyse différentielle (DPA)

- Utilise le *coefficient de corrélation linéaire* comme distingueur [KJJ99,BCO04]

Analyse différentielle (DPA)

- Utilise le *coefficient de corrélation linéaire* comme distingueur [KJJ99,BCO04]
- Pour chaque k :
  - prédiction $f(X, k)$ sur $Z = f(X, K)$

Analyse différentielle (DPA)

- Utilise le *coefficient de corrélation linéaire* comme distingueur [KJJ99,BCO04]
- Pour chaque k :
  - prédiction $f(X, k)$ sur $Z = f(X, K)$
  - modèle $\hat{\varphi} \circ f(X, k)$ pour $L = \varphi \circ f(X, K) + B$

Analyse différentielle (DPA)

- Utilise le *coefficient de corrélation linéaire* comme distingueur [KJJ99,BCO04]
- Pour chaque k :
  - prédiction $f(X, k)$ sur $Z = f(X, K)$
  - modèle $\hat{\varphi} \circ f(X, k)$ pour $L = \varphi \circ f(X, K) + B$
  - corrélation entre le modèle et la fuite :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), L\right]$$

Analyse différentielle (DPA)

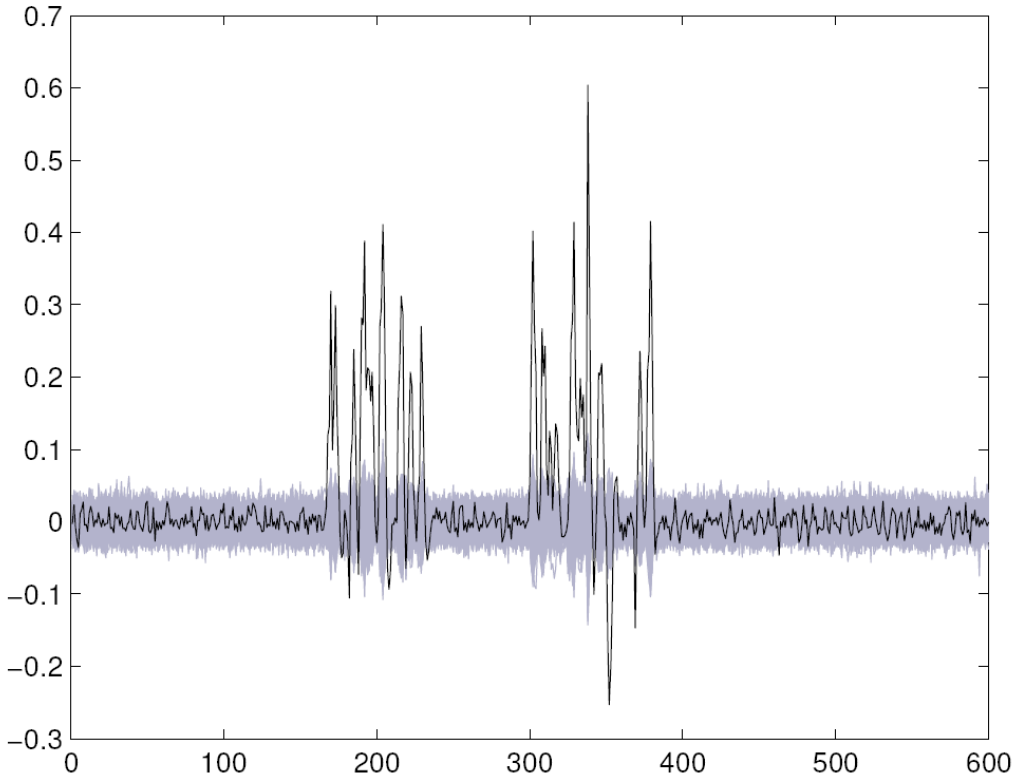
- Utilise le *coefficient de corrélation linéaire* comme distingueur [KJJ99,BCO04]
- Pour chaque k :
  - prédiction $f(X, k)$ sur $Z = f(X, K)$
  - modèle $\hat{\varphi} \circ f(X, k)$ pour $L = \varphi \circ f(X, K) + B$
  - corrélation entre le modèle et la fuite :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), L\right]$$

- Une forte corrélation est attendue pour $k = K$

Analyse différentielle (DPA)

- Attaque DPA sur une implémentation de l'AES sur carte à puce



Attaques profilées

- utilise la *vraisemblance* comme distingueur [CRR02]
- pour chaque k :

$$d_k \quad = \quad \mathrm{P}\left[K = k | L, X\right]$$

Attaques profilées

- utilise la *vraisemblance* comme distingueur [CRR02]
- pour chaque k :

$$\begin{aligned} d_k &= \mathrm{P}\left[K = k | L, X\right] \\ &= \alpha\, \mathrm{P}\left[L | Z = f(X, k)\right] \end{aligned}$$

Attaques profilées

- utilise la *vraisemblance* comme distingueur [CRR02]
- pour chaque k :

$$\begin{aligned} d_k &= \mathrm{P}\left[K = k | L, X\right] \\ &= \alpha\, \mathrm{P}\left[L | Z = f(X, k)\right] \end{aligned}$$

- requiert la connaissance des distributions $\mathrm{P}\left[L | Z = z\right]$
  - peut être obtenue par une *phase de profilage*

Attaques profilées

- utilise la *vraisemblance* comme distingueur [CRR02]
- pour chaque k :

$$\begin{aligned} d_k &= \mathrm{P}\left[K = k | L, X\right] \\ &= \alpha\, \mathrm{P}\left[L | Z = f(X, k)\right] \end{aligned}$$

- requiert la connaissance des distributions $\mathrm{P}\left[L | Z = z\right]$
  - peut être obtenue par une *phase de profilage*
- adversaire plus puissant $\Longrightarrow$ attaque plus efficace

Taux de succès de ces attaques

- Taux de succès d'une attaque :

$$\mathrm{P}\left[K = \mathrm{argmax}_{k \in \mathcal{K}}\ d_k\right]$$

  - extension à l'ordre o [SMY09]

Taux de succès de ces attaques

- Taux de succès d'une attaque :

$$\mathrm{P}\left[K = \mathrm{argmax}_{k \in \mathcal{K}}\ d_k\right]$$

  - extension à l'ordre o [SMY09]
- Analyse du taux de succès des attaques DPA & profilées à SAC 2008 [Riv08]

Taux de succès de ces attaques

- Taux de succès d'une attaque :

$$\mathrm{P}\left[K = \mathrm{argmax}_{k \in \mathcal{K}}\ d_k\right]$$

  - extension à l'ordre o [SMY09]
- Analyse du taux de succès des attaques DPA & profilées à SAC 2008 [Riv08]
  - sous l'hypothèse du bruit Gaussien
  - le vecteur $(d_0, \ldots, d_{|\mathcal{K}|-1})$ suit une loi Gaussienne multivariée

Taux de succès de ces attaques

- Taux de succès d'une attaque :

$$\mathrm{P}\left[K = \mathrm{argmax}_{k \in \mathcal{K}}\ d_k\right]$$

  - extension à l'ordre o [SMY09]
- Analyse du taux de succès des attaques DPA & profilées à SAC 2008 [Riv08]
  - sous l'hypothèse du bruit Gaussien
  - le vecteur $(d_0, \ldots, d_{|\mathcal{K}|-1})$ suit une loi Gaussienne multivariée
  - méthode efficace pour calculer le taux de succès

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]
- Pour chaque k :
  - information mutuelle entre le modèle et la fuite :

$$d_k = \mathrm{I}(\hat{\varphi} \circ f(X, k), L)$$

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]
- Pour chaque k :
  - information mutuelle entre le modèle et la fuite :

$$d_k = \mathrm{I}(\hat{\varphi} \circ f(X, k), L)$$

- Motivation : l'IM détecte toute dépendance statistique (pas uniquement linéaire)

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]
- Pour chaque k :
  - information mutuelle entre le modèle et la fuite :

$$d_k = \mathrm{I}(\hat{\varphi} \circ f(X, k), L)$$

- Motivation : l'IM détecte toute dépendance statistique (pas uniquement linéaire)
- Avantage : plus de généricité

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]
- Pour chaque k :
  - information mutuelle entre le modèle et la fuite :

$$d_k = \mathrm{I}(\hat{\varphi} \circ f(X, k), L)$$

- Motivation : l'IM détecte toute dépendance statistique (pas uniquement linéaire)
- Avantage : plus de généricité
- Questions :
  - comment estimer l'IM ?
  - quel est l'intérêt de la MIA par rapport à la DPA ?

Analyse par information mutuelle

- Nouvelle approche : utilisation de l'*information mutuelle* comme distingueur [GBT08,Aum07]
- Pour chaque k :
  - information mutuelle entre le modèle et la fuite :

$$d_k = \mathrm{I}(\hat{\varphi} \circ f(X, k), L)$$

- Motivation : l'IM détecte toute dépendance statistique (pas uniquement linéaire)
- Avantage : plus de généricité
- Questions :
  - comment estimer l'IM ?
  - quel est l'intérêt de la MIA par rapport à la DPA ?
- Réponses à ACNS 2009 [PR09]

Masquage

- Ajout de valeurs aléatoires M_i à chaque variable sensible :

$$M_0 = Z \oplus M_1 \oplus \cdots \oplus M_d$$

Masquage

- Ajout de valeurs aléatoires M_i à chaque variable sensible :

$$M_0 = Z \oplus M_1 \oplus \cdots \oplus M_d$$

- La *variable masquée* M_0 et les *masques* M_i sont manipulés séparément

Masquage

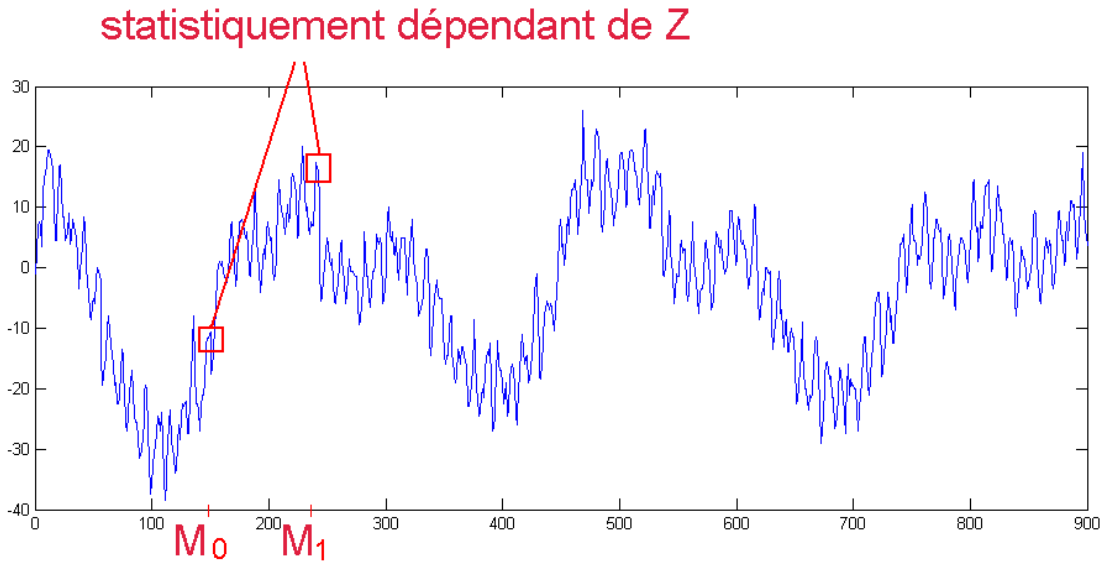
- Ajout de valeurs aléatoires M_i à chaque variable sensible :

$$M_0 = Z \oplus M_1 \oplus \cdots \oplus M_d$$

- La *variable masquée* M_0 et les *masques* M_i sont manipulés séparément
- Variables intermédiaires indépendantes de toute variable sensible
- Les attaques décrites ne marchent plus

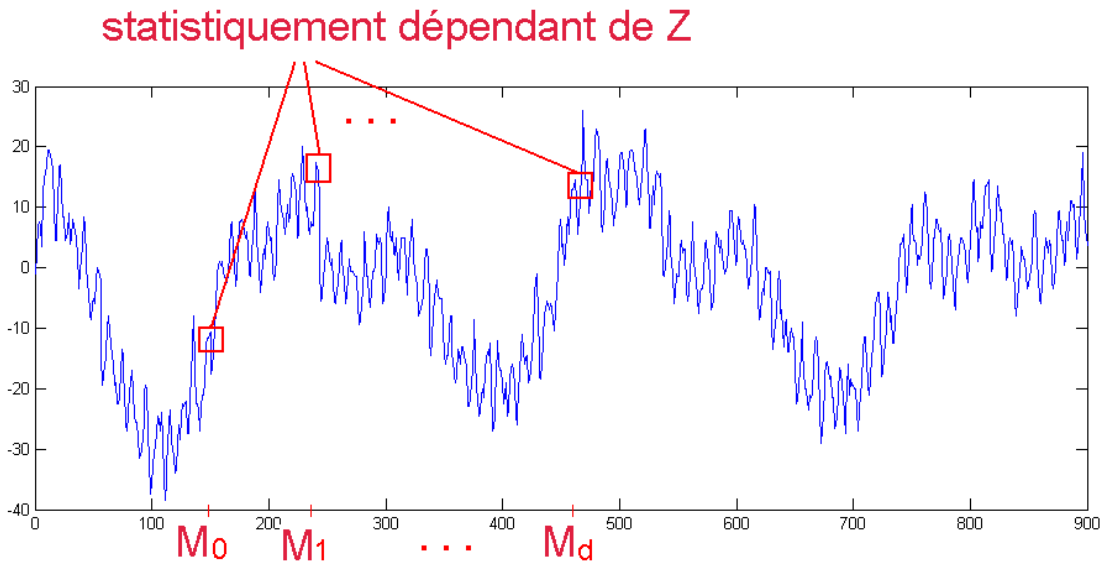
Attaques d'ordre supérieur

- Masquage d'ordre 1 : $M_0 = Z \oplus M_1$
- Attaque d'ordre 2 :



Attaques d'ordre supérieur

- Masquage d'ordre d : $M_0 = Z \oplus M_1 \oplus \cdots \oplus M_d$
- Attaque d'ordre $d + 1$:



Attaques d'ordre supérieur

- DPA d'ordre supérieur [CJRR99,Mes00]
  - combine les $d+1$ fuites par une fonction : $\mathcal{C}(L_0, \ldots, L_d)$
  - calcule la corrélation :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), \mathcal{C}(L_0, \ldots, L_d)\right]$$

Attaques d'ordre supérieur

- DPA d'ordre supérieur [CJRR99,Mes00]
  - combine les $d+1$ fuites par une fonction : $\mathcal{C}(L_0, \ldots, L_d)$
  - calcule la corrélation :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), \mathcal{C}(L_0, \ldots, L_d)\right]$$

- Analyse publiée à IEEE-TC [PRB09] étendue à CHES 2009 [RPD09]
  - combinaison par produit normalisé :
    $\mathcal{C}(L_0, \ldots, L_d) = (L_0 - \mathrm{E}(L_0)) \times \cdots \times (L_d - \mathrm{E}(L_d))$

Attaques d'ordre supérieur

- DPA d'ordre supérieur [CJRR99,Mes00]
  - combine les $d+1$ fuites par une fonction : $\mathcal{C}(L_0, \ldots, L_d)$
  - calcule la corrélation :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), \mathcal{C}(L_0, \ldots, L_d)\right]$$

- Analyse publiée à IEEE-TC [PRB09] étendue à CHES 2009 [RPD09]
  - combinaison par produit normalisé :
    $\mathcal{C}(L_0, \ldots, L_d) = (L_0 - \mathrm{E}(L_0)) \times \cdots \times (L_d - \mathrm{E}(L_d))$
- MIA d'ordre supérieur [PR09] :
  - calcule l'information mutuelle :

$$\mathrm{I}\big(\hat{\varphi} \circ f(X, k), (L_0, \ldots, L_d)\big)$$

Attaques d'ordre supérieur

- DPA d'ordre supérieur [CJRR99,Mes00]
  - combine les $d+1$ fuites par une fonction : $\mathcal{C}(L_0, \ldots, L_d)$
  - calcule la corrélation :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), \mathcal{C}(L_0, \ldots, L_d)\right]$$

- Analyse publiée à IEEE-TC [PRB09] étendue à CHES 2009 [RPD09]
  - combinaison par produit normalisé :
    $\mathcal{C}(L_0, \ldots, L_d) = (L_0 - \mathrm{E}(L_0)) \times \cdots \times (L_d - \mathrm{E}(L_d))$
- MIA d'ordre supérieur [PR09] :
  - calcule l'information mutuelle :

$$\mathrm{I}\big(\hat{\varphi} \circ f(X, k), (L_0, \ldots, L_d)\big)$$

  - évite la perte d'information intrinsèque à la combinaison

Attaques d'ordre supérieur

- DPA d'ordre supérieur [CJRR99,Mes00]
  - combine les $d+1$ fuites par une fonction : $\mathcal{C}(L_0, \ldots, L_d)$
  - calcule la corrélation :

$$d_k = \rho\left[\hat{\varphi} \circ f(X, k), \mathcal{C}(L_0, \ldots, L_d)\right]$$

- Analyse publiée à IEEE-TC [PRB09] étendue à CHES 2009 [RPD09]
  - combinaison par produit normalisé :
    $\mathcal{C}(L_0, \ldots, L_d) = (L_0 - \mathrm{E}(L_0)) \times \cdots \times (L_d - \mathrm{E}(L_d))$
- MIA d'ordre supérieur [PR09] :
  - calcule l'information mutuelle :

$$\mathrm{I}\big(\hat{\varphi} \circ f(X, k), (L_0, \ldots, L_d)\big)$$

  - évite la perte d'information intrinsèque à la combinaison
  - devient meilleur que la DPA d'ordre 2 quand le bruit augmente

Sécurité du masquage

- La complexité d'une attaque augmente **exponentiellement** avec l'ordre de l'attaque

Sécurité du masquage

- La complexité d'une attaque augmente **exponentiellement** avec l'ordre de l'attaque
- [CJRR99] nombre de mesures nécessaires :

$$N > o(1) \cdot \sigma^d$$

  - où σ est l'écart type du bruit

Sécurité du masquage

- La complexité d'une attaque augmente **exponentiellement** avec l'ordre de l'attaque
- [CJRR99] nombre de mesures nécessaires :

$$N > o(1) \cdot \sigma^d$$

  - où σ est l'écart type du bruit
- le masquage (avec addition de bruit) est une contre-mesure efficace

Plan

1 Introduction

2 Attaques par canaux auxiliaires
- Modèle
- Méthodes classiques
- Masquage et attaques d'ordre supérieur

3 Schémas de masquage
- Schémas de premier ordre
- **Cryptanalyse d'un schéma d'ordre supérieur**
- Schémas d'ordre supérieur

4 Attaques par fautes
- **Sécurisation du RSA par double chaîne d'additions**
- Autres travaux

5 Conclusion

Schémas de masquage pour chiffrements par blocs

- Réseaux de transformations linéaires et de boîtes-S non-linéaires (*e.g.* DES, AES)

Schémas de masquage pour chiffrements par blocs

- Réseaux de transformations linéaires et de boîtes-S non-linéaires (*e.g.* DES, AES)
- Les différentes transformations doivent satisfaire :

Complétude

La variable masquée M_0 et les masques M_i doivent vérifier :

$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

Schémas de masquage pour chiffrements par blocs

- Réseaux de transformations linéaires et de boîtes-S non-linéaires (*e.g.* DES, AES)
- Les différentes transformations doivent satisfaire :

Complétude

La variable masquée M_0 et les masques M_i doivent vérifier :

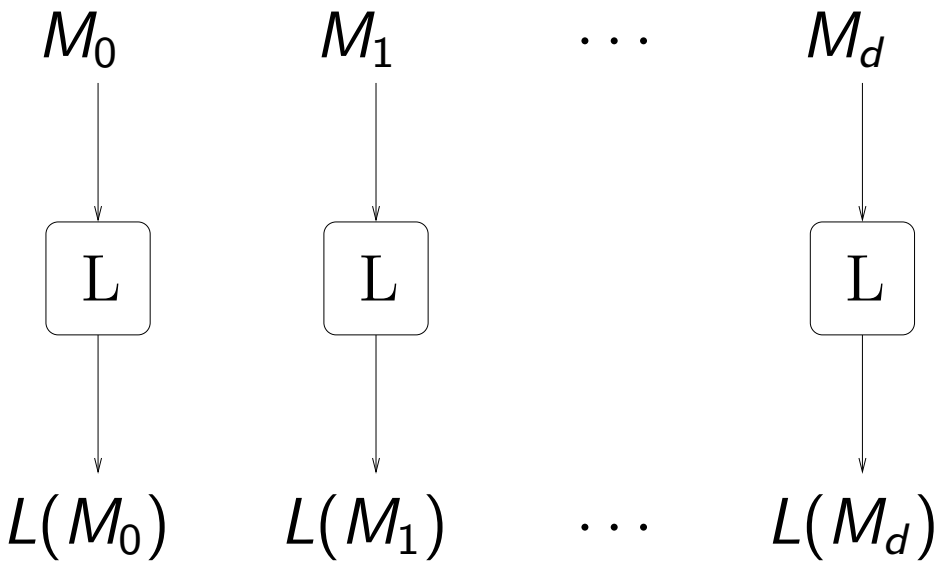
$$M_0 \oplus \cdots \oplus M_d = Z \ .$$

Sécurité

La variable masquée M_0 et les masques M_i doivent être manipulés séparément.

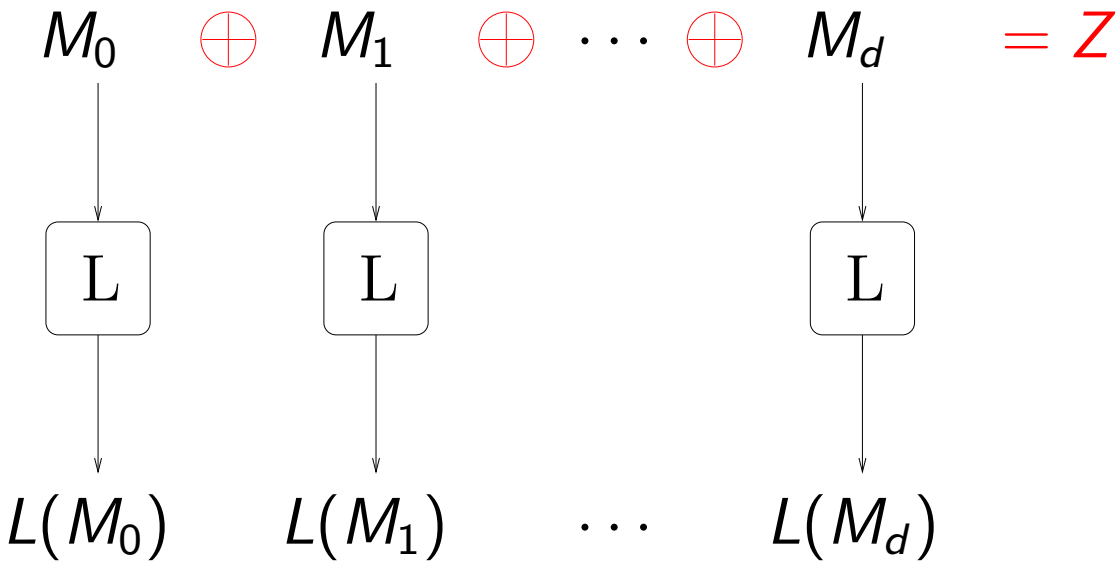
Schémas de masquage pour chiffrements par blocs

- Propagation à travers une **transformation linéaire**



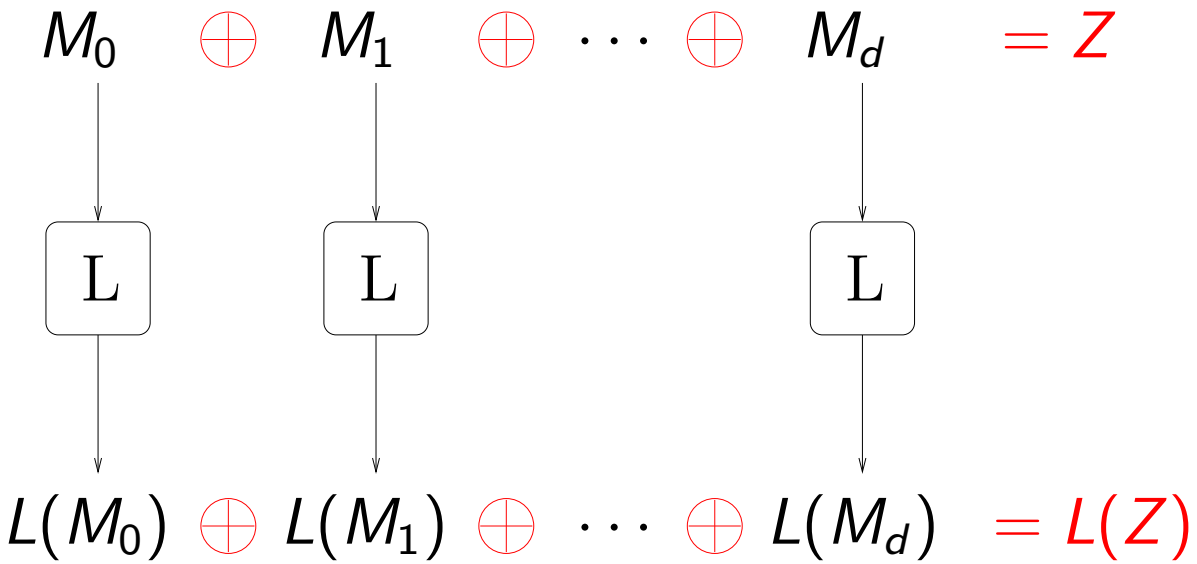
Schémas de masquage pour chiffrements par blocs

- Propagation à travers une **transformation linéaire**



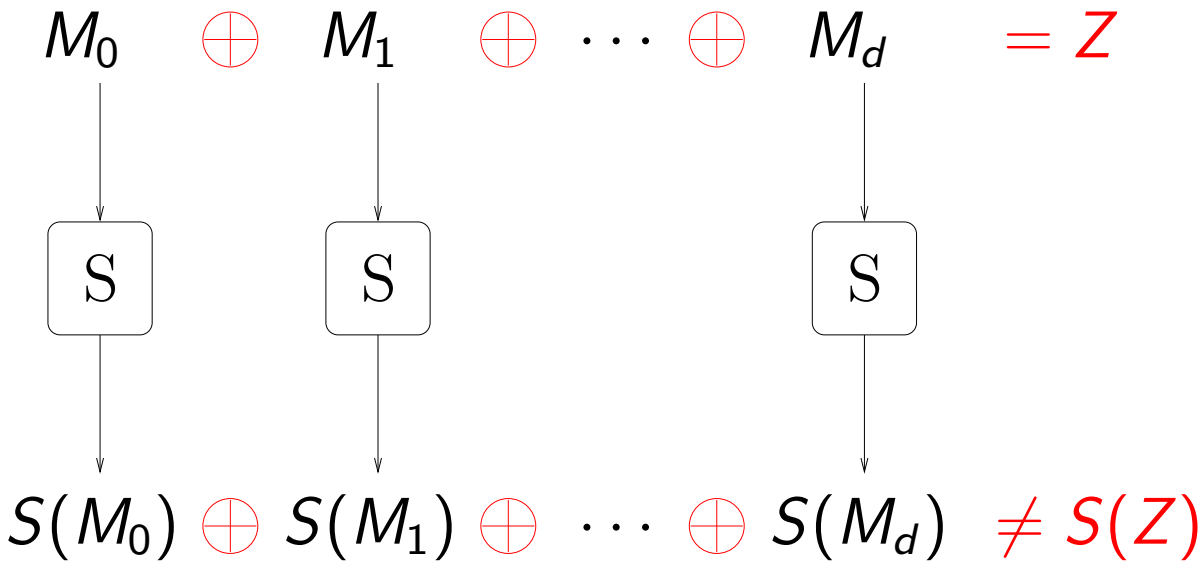
Schémas de masquage pour chiffrements par blocs

- Propagation à travers une **transformation linéaire**



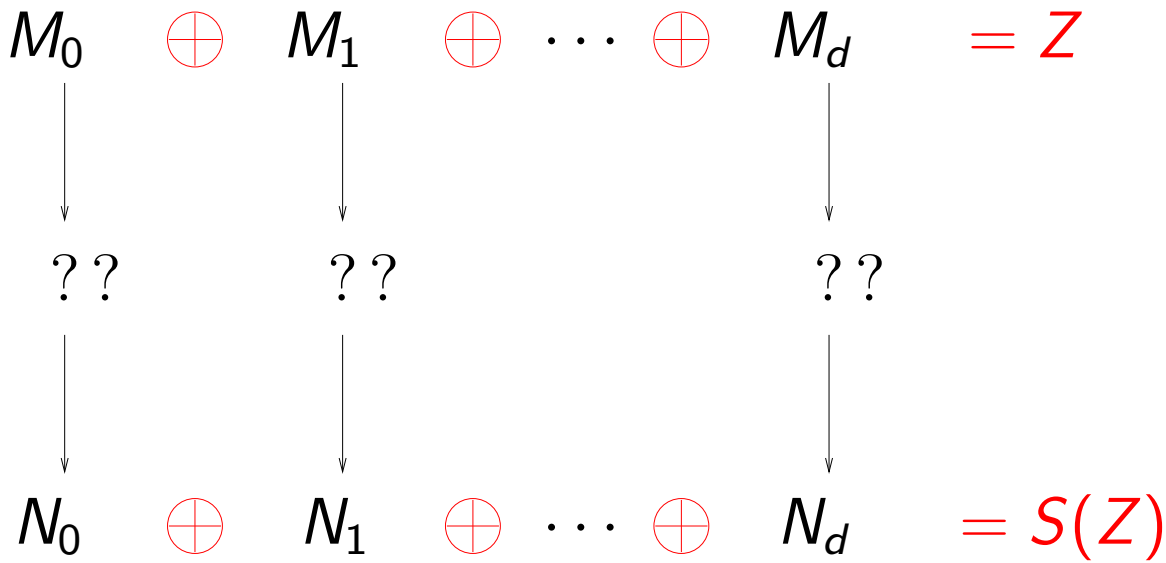
Schémas de masquage pour chiffrements par blocs

- Propagation à travers une **boîte-S non linéaire**



Schémas de masquage pour chiffrements par blocs

- Propagation à travers une **boîte-S non linéaire**

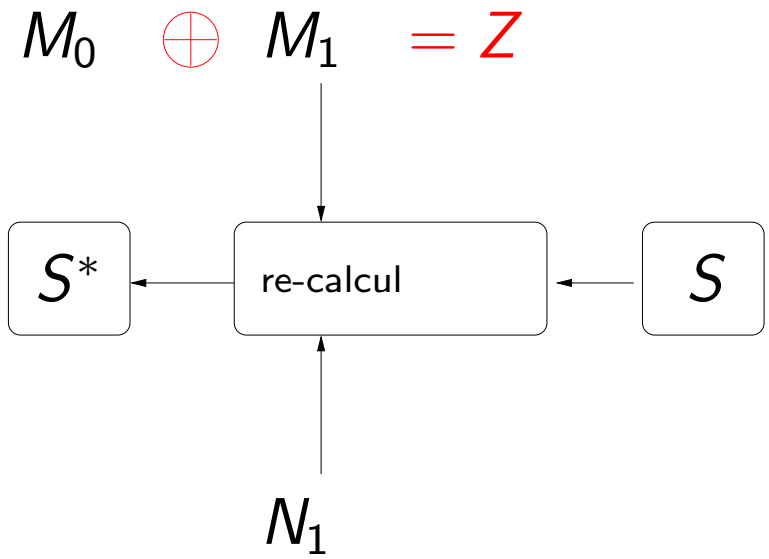


Problème

Comment calculer $(N_0, \ldots, N_d)$ à partir de $(M_0, \ldots, M_d)$ de manière sécurisée ?

Schémas de premier ordre

- Méthode par re-calcul de table

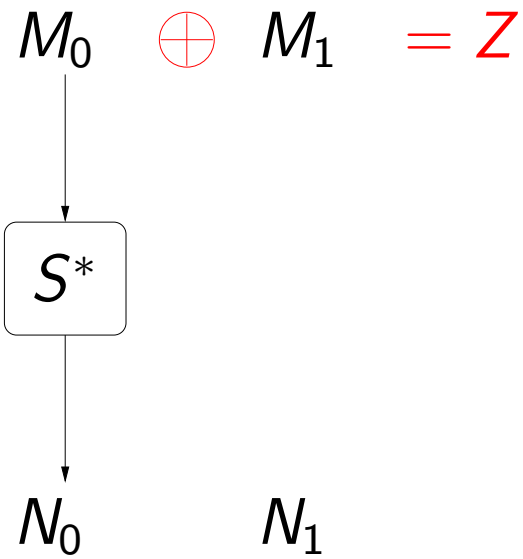


re-calcul de table

Pour tout x : $S^*(x) \leftarrow S(x \oplus M_1) \oplus N_1$

Schémas de premier ordre

- Méthode par re-calcul de table



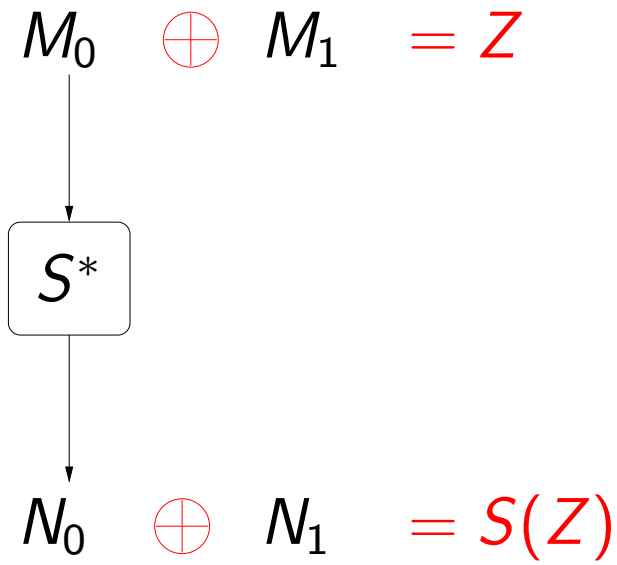
re-calcul de table

Pour tout x : $S^*(x) \leftarrow S(x \oplus M_1) \oplus N_1$

- $N_0 \leftarrow S^*(M_0)$

Schémas de premier ordre

- Méthode par re-calcul de table



re-calcul de table

Pour tout x : $S^*(x) \leftarrow S(x \oplus M_1) \oplus N_1$

- $N_0 \leftarrow S^*(M_0) = S(M_0 \oplus M_1) \oplus N_1 = S(Z) \oplus N_1$

Calcul à la volée

- Calcul à la volée sans stockage de tables en RAM
- Solution proposée à CHES 2006 se basant sur la *transformée de Fourier*

Calcul à la volée

- Calcul à la volée sans stockage de tables en RAM
- Solution proposée à CHES 2006 se basant sur la *transformée de Fourier*
- Attaque et une réparation de cette solution à CHES 2008 [CGPR08]

Calcul à la volée

- Calcul à la volée sans stockage de tables en RAM
- Solution proposée à CHES 2006 se basant sur la *transformée de Fourier*
- Attaque et une réparation de cette solution à CHES 2008 [CGPR08]
- Méthode alternative plus efficace à WISA 2007 [PR07]

Cryptanalyse d'un schéma d'ordre supérieur

- Publié par Schramm et Paar à CT-RSA 2006 [SP06]

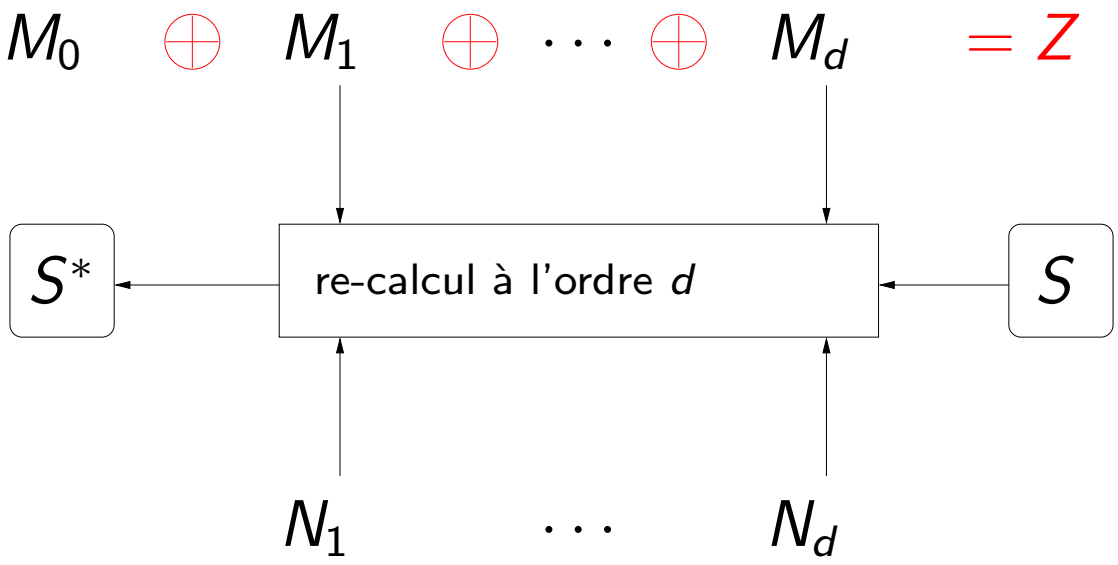
Cryptanalyse d'un schéma d'ordre supérieur

- Publié par Schramm et Paar à CT-RSA 2006 [SP06]
- Schéma de masquage d'ordre d choisi
- Généralisation de la méthode par re-calcul de table

Cryptanalyse d'un schéma d'ordre supérieur

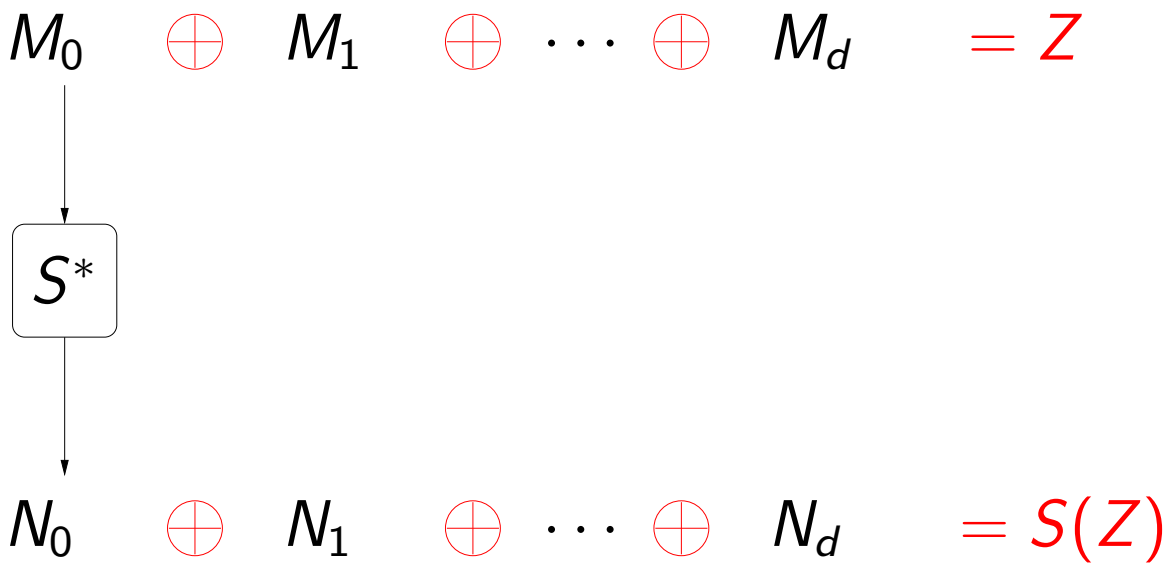
- Publié par Schramm et Paar à CT-RSA 2006 [SP06]
- Schéma de masquage d'ordre d choisi
- Généralisation de la méthode par re-calcul de table
- Cryptanalyse publiée à CHES 2007 [CPR07]

Re-calcul de table à l'ordre d



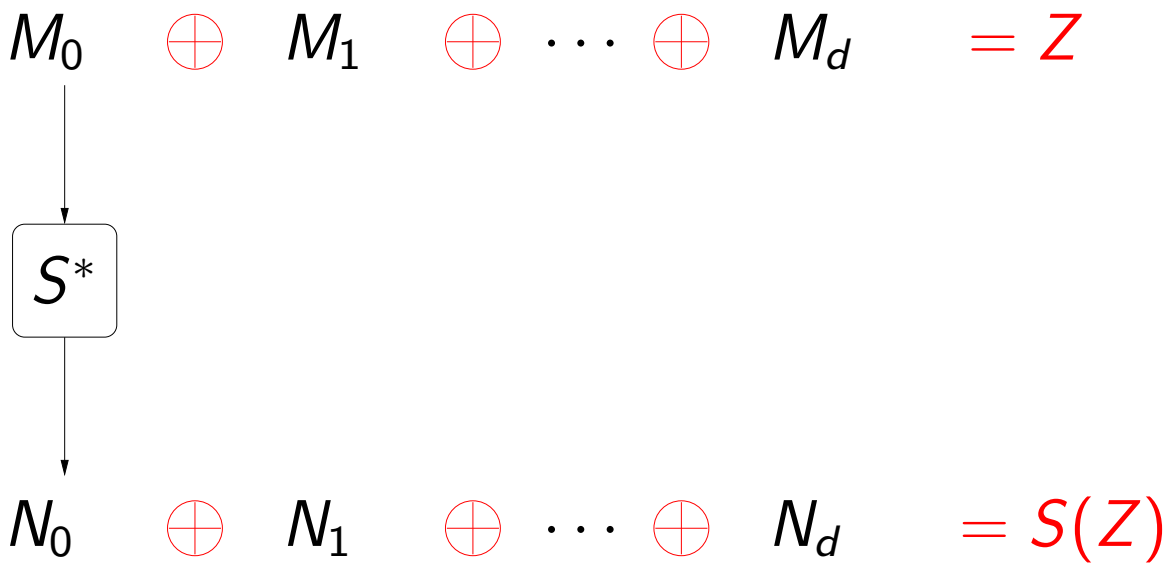
$$S^*(\cdot) = S\left(\cdot \oplus M_1 \oplus \cdots \oplus M_d\right) \oplus N_1 \oplus \cdots \oplus N_d$$

Re-calcul de table à l'ordre d



$$S^*(\cdot) = S\left(\cdot \oplus M_1 \oplus \cdots \oplus M_d\right) \oplus N_1 \oplus \cdots \oplus N_d$$

Re-calcul de table à l'ordre d



$$S^*(\cdot) = S\left(\cdot \oplus M_1 \oplus \cdots \oplus M_d\right) \oplus N_1 \oplus \cdots \oplus N_d$$

Problème

Comment calculer S^* à partir de $(S, (M_i)_i, (N_i)_i)$ de manière sécurisée ?

Schéma générique

- Effectue d re-calculs de table successifs
  - $S_1(x) = S(x \oplus M_1) \oplus N_1$

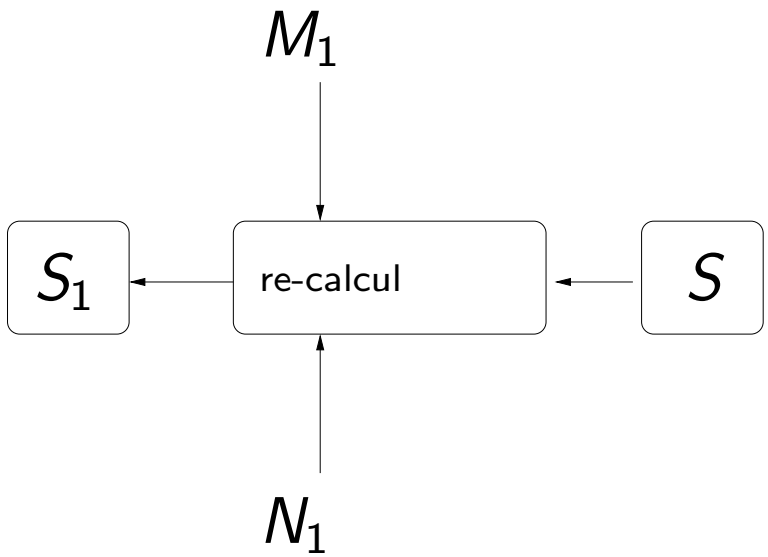


Schéma générique

- Effectue d re-calculs de table successifs
  - $S_1(x) = S(x \oplus M_1) \oplus N_1$
  - $S_2(x) = S(x \oplus M_1 \oplus M_2) \oplus N_1 \oplus N_2$

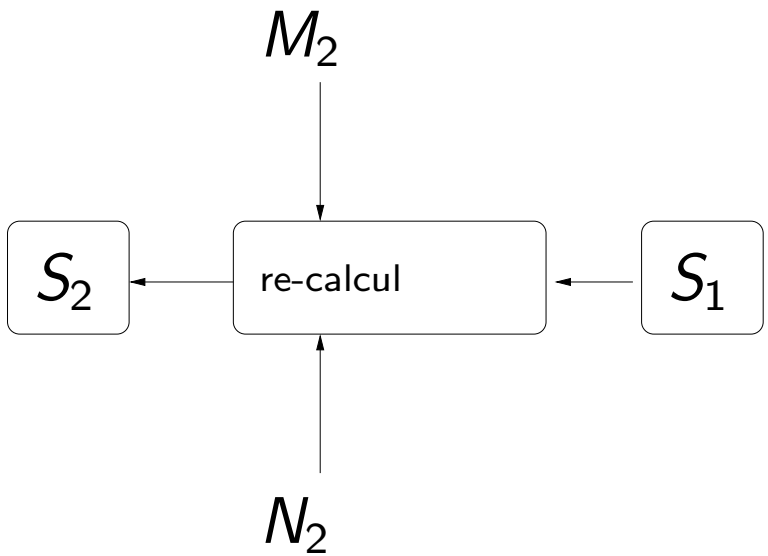
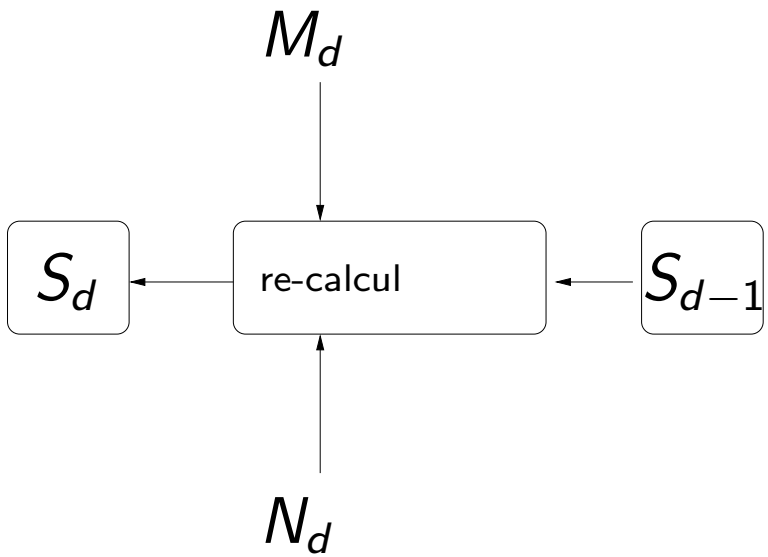


Schéma générique

- Effectue d re-calculs de table successifs
  - $S_1(x) = S(x \oplus M_1) \oplus N_1$
  - $S_2(x) = S(x \oplus M_1 \oplus M_2) \oplus N_1 \oplus N_2$
  - ...
  - $S_d(\cdot) = S(\cdot \oplus M_1 \oplus \cdots \oplus M_d) \oplus N_1 \oplus \cdots \oplus N_d$



Faille d'ordre 3

- La variable masquée M_0 vérifie :
  1) $M_0 = Z \oplus \bigoplus_i M_i$
- Pendant le calcul de la table S_d :
  2) $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  3) $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$

Faille d'ordre 3

- La variable masquée M_0 vérifie :
  1) $M_0 = Z \oplus \bigoplus_i M_i$
- Pendant le calcul de la table S_d :
  2) $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  3) $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- La distribution de $(M_0, S_d(0), S_d(1))$ dépend de Z
  - faille à l'ordre 3 !
  - attaque d'ordre 3 possible !

Faille d'ordre 3

- À partir de :
  - $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  - $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- On a : $S_d(0) \oplus S_d(1) = S(\bigoplus_i M_i) \oplus S(\bigoplus_i M_i \oplus 1)$
  - dépend de $\bigoplus_i M_i$

Faille d'ordre 3

- À partir de :
  - $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  - $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- On a : $S_d(0) \oplus S_d(1) = S(\bigoplus_i M_i) \oplus S(\bigoplus_i M_i \oplus 1)$
  - dépend de $\bigoplus_i M_i$
- Avec $M_0 = Z \oplus \bigoplus_i M_i$
  - dépend de Z

Faille d'ordre 3

- À partir de :
  - $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  - $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- On a : $S_d(0) \oplus S_d(1) = S(\bigoplus_i M_i) \oplus S(\bigoplus_i M_i \oplus 1)$
  - dépend de $\bigoplus_i M_i$
- Avec $M_0 = Z \oplus \bigoplus_i M_i$
  - dépend de Z
- Le triplet $(M_0, S_d(0), S_d(1))$ dépend de Z

Faille d'ordre 3

- À partir de :
  - $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  - $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- On a : $S_d(0) \oplus S_d(1) = S(\bigoplus_i M_i) \oplus S(\bigoplus_i M_i \oplus 1)$
  - dépend de $\bigoplus_i M_i$
- Avec $M_0 = Z \oplus \bigoplus_i M_i$
  - dépend de Z
- Le triplet $(M_0, S_d(0), S_d(1))$ dépend de Z
  - valable également pour tout $(M_0, S_d(a), S_d(b))$ avec $a \neq b$

Faille d'ordre 3

- À partir de :
  - $S_d(0) = S(0 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
  - $S_d(1) = S(1 \oplus \bigoplus_i M_i) \oplus \bigoplus_i N_i$
- On a : $S_d(0) \oplus S_d(1) = S(\bigoplus_i M_i) \oplus S(\bigoplus_i M_i \oplus 1)$
  - dépend de $\bigoplus_i M_i$
- Avec $M_0 = Z \oplus \bigoplus_i M_i$
  - dépend de Z
- Le triplet $(M_0, S_d(0), S_d(1))$ dépend de Z
  - valable également pour tout $(M_0, S_d(a), S_d(b))$ avec $a \neq b$

Conclusion

L'approche consistant à effectuer d re-calculs de table n'est pas valable pour parer les attaques d'ordre d.

Schéma amélioré

- Amélioration proposée [SP06] :
  - d re-calculs de table pour le premier calcul de boîte-S
  - 1 **seul re-calcul** pour chaque calcul de boîte-S
- Comment ?
  - chaque boîte-S masquée est dérivée de la précédente
- Attaques d'ordre 3 sur le schéma amélioré !

Simulations d'attaques

Implémentation	**Attaque**	**Nb. mesures**
schéma générique	DPA d'ordre 3	6.10^6
schéma générique	attaque profilée d'ordre 3	2.10^3
schéma amélioré	DPA d'ordre 3	10^5
schéma amélioré	attaque profilée d'ordre 3	10^3

TAB.: Nombre de mesures pour un taux de succès de 50%.

- Attaques pratiques dans un modèle classique

Schémas d'ordre supérieur

- Méthodes alternatives à l'ordre 2 à FSE 2008 [RDP08]
  - sécurité formellement prouvée
  - différents compromis temps-mémoire
  - améliorations donnant les meilleures performances publiées

Schémas d'ordre supérieur

- Méthodes alternatives à l'ordre 2 à FSE 2008 [RDP08]
  - sécurité formellement prouvée
  - différents compromis temps-mémoire
  - améliorations donnant les meilleures performances publiées
- Approche différente à CHES 2009 [RPD09]
  - combinaison de masquage d'ordre supérieur et de randomisation de l'ordre des opérations
  - la sécurité n'est pas prouvée à un certain ordre
  - assure un degré de resistance choisi contre les attaques DPA

Plan

1. Introduction
2. Attaques par canaux auxiliaires
   - Modèle
   - Méthodes classiques
   - Masquage et attaques d'ordre supérieur
3. Schémas de masquage
   - Schémas de premier ordre
   - **Cryptanalyse d'un schéma d'ordre supérieur**
   - Schémas d'ordre supérieur
4. Attaques par fautes
   - **Sécurisation du RSA par double chaîne d'additions**
   - Autres travaux
5. Conclusion

Attaques par fautes

Attaques par fautes

- Exploitation des résultats erronés pour en déduire la clé secrète [BDL97]

Attaques par fautes

- Exploitation des résultats erronés pour en déduire la clé secrète [BDL97]
- Techniques d'injection de fautes :
  - *glitch* d'alimentation ou d'horloge
  - injection lumineuse (*e.g.* flash, laser)

Sécurisation des implémentations cryptographiques

- Schémas de détection d'erreurs

Sécurisation des implémentations cryptographiques

- Schémas de détection d'erreurs
- Doublement du calcul

Sécurisation des implémentations cryptographiques

- Schémas de détection d'erreurs
- Doublement du calcul
- Doublement du temps d'exécution (ou bien du hardware)
  - envisageable pour les algorithmes symétriques
  - prohibitif pour le RSA

Sécurisation des implémentations cryptographiques

- Schémas de détection d'erreurs
- Doublement du calcul
- Doublement du temps d'exécution (ou bien du hardware)
  - envisageable pour les algorithmes symétriques
  - prohibitif pour le RSA
- Nouvelle contre-mesure publiée à CT-RSA 2009 [Riv09a]

RSA

Cryptosystème RSA

- Deux nombres premiers p, q, un module public $N = pq$
- Un exposant public e, un exposant privé d, t.q. :

$$ed = 1 \mod \phi(N)$$

RSA

Cryptosystème RSA

- Deux nombres premiers p, q, un module public $N = pq$
- Un exposant public e, un exposant privé d, t.q. :

$$ed = 1 \mod \phi(N)$$

- Clé publique $PK = (N, e)$, clé privée $SK = (N, d)$

RSA

Cryptosystème RSA

- Deux nombres premiers p, q, un module public $N = pq$
- Un exposant public e, un exposant privé d, t.q. :

$$ed = 1 \mod \phi(N)$$

- Clé publique $PK = (N, e)$, clé privée $SK = (N, d)$

Chiffrement

$$c = m^e \mod N$$

Déchiffrement

$$m = c^d \mod N$$

Signature

$$s = m^d \mod N$$

Vérification

$$m \stackrel{?}{=} s^e \mod N$$

RSA et théorème des restes chinois (CRT)

RSA-CRT

- 2 exponentiations, 1 *recombinaison* :

$$\begin{aligned} s_p &= m^{d_p} \bmod p \\ s_q &= m^{d_q} \bmod q \\ s &= \mathrm{CRT}(s_p, s_q) \end{aligned}$$

- $d_p = d \bmod (p-1)$, $d_q = d \bmod (q-1)$
- $\mathrm{CRT}(s_p, s_q) = s_p + p\left((s_q - s_p)(p^{-1} \bmod q) \bmod q\right)$
- Largement utilisé car 4 fois plus rapide

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} s_p &= m^d \bmod p \\ s_q &= m^d \bmod q \\ s &= \mathrm{CRT}(s_p, s_q) \end{aligned}$$

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} \widetilde{s_p} &\neq m^d \bmod p \\ s_q &= m^d \bmod q \\ \widetilde{s} &= \mathrm{CRT}(\widetilde{s_p}, s_q) \end{aligned}$$

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} \widetilde{s_p} &\neq m^d \bmod p \\ s_q &= m^d \bmod q \\ \widetilde{s} &= \mathrm{CRT}(\widetilde{s_p}, s_q) \end{aligned}$$

$$\widetilde{s} \not\equiv m^d \bmod p \;\Big|\; \widetilde{s} \equiv m^d \bmod q$$

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} \widetilde{s_p} &\neq m^d \bmod p \\ s_q &= m^d \bmod q \\ \widetilde{s} &= \mathrm{CRT}(\widetilde{s_p}, s_q) \end{aligned}$$

$$\begin{array}{c|c} \widetilde{s} \not\equiv m^d \bmod p & \widetilde{s} \equiv m^d \bmod q \\ p \nmid (\widetilde{s} - s) & q \mid (\widetilde{s} - s) \end{array}$$

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} \widetilde{s_p} &\neq m^d \bmod p \\ s_q &= m^d \bmod q \\ \widetilde{s} &= \mathrm{CRT}(\widetilde{s_p}, s_q) \end{aligned}$$

$$\begin{array}{c|c} \widetilde{s} \not\equiv m^d \bmod p & \widetilde{s} \equiv m^d \bmod q \\ p \nmid (\widetilde{s} - s) & q \mid (\widetilde{s} - s) \end{array}$$

$$\Rightarrow \mathrm{pgcd}(\widetilde{s} - s, N) = q$$

Attaques par faute sur le RSA

- Attaque de Bellcore [BDL97] :

$$\begin{aligned} \widetilde{s_p} &\neq m^d \bmod p \\ s_q &= m^d \bmod q \\ \widetilde{s} &= \mathrm{CRT}(\widetilde{s_p}, s_q) \end{aligned}$$

$$\begin{array}{c|c} \widetilde{s} \not\equiv m^d \bmod p & \widetilde{s} \equiv m^d \bmod q \\ p \nmid (\widetilde{s} - s) & q \mid (\widetilde{s} - s) \end{array}$$

$$\Rightarrow \mathrm{pgcd}(\widetilde{s} - s, N) = q$$

- D'autres attaques existent sur le RSA standard

Problème

Problème

Effectuer un calcul RSA qui détecte les erreurs.

Problème

Problème

Effectuer un calcul RSA qui détecte les erreurs.

Solutions

- doubler le calcul
  - double le temps d'exécution (ou bien le hardware)

Problème

Problème

Effectuer un calcul RSA qui détecte les erreurs.

Solutions

- doubler le calcul
  - double le temps d'exécution (ou bien le hardware)
- vérifier la signature calculée : $s^e \bmod N \stackrel{?}{=} m$
  - e n'est pas toujours disponible
  - e peut être grand $\Rightarrow$ double le temps d'exécution

Problème

Problème

Effectuer un calcul RSA qui détecte les erreurs.

Solutions

- doubler le calcul
  - double le temps d'exécution (ou bien le hardware)
- vérifier la signature calculée : $s^e \bmod N \stackrel{?}{=} m$
  - e n'est pas toujours disponible
  - e peut être grand $\Rightarrow$ double le temps d'exécution

Problème

Effectuer un calcul RSA qui détecte les erreurs de manière efficace et sans se servir de e.

Solutions existantes

Solutions

- **Extension du module** : redondance dans les opérations modulaires
  - Contre-mesure de Shamir [Sha98,BOS03,CJ05]
    - $s^* = m^d \bmod N \cdot t$
    - $s^* \stackrel{?}{\equiv} m^d \bmod t$
  - Contre-mesure de Vigilant [Vig08]

Solutions existantes

Solutions

- **Extension du module** : redondance dans les opérations modulaires
  - Contre-mesure de Shamir [Sha98,BOS03,CJ05]
    - $s^* = m^d \bmod N \cdot t$
    - $s^* \stackrel{?}{\equiv} m^d \bmod t$
  - Contre-mesure de Vigilant [Vig08]
- **Exponentiation sécurisée** : redondance incluse dans l'algorithme d'exponentiation
  - Contre-mesure de Giraud [Gir06]
    - $(m^{d-1}, m^d) \leftarrow \text{EchelleMontgomery}(m, d)$
    - $m^{d-1} \cdot m \stackrel{?}{\equiv} m^d \bmod N$
  - Contre-mesure de Boscher *et al.* [BNP07]

Principe général

- Effectuer une *double exponentiation* pour calculer :

$$(s, c) \leftarrow \left(m^d \bmod N,\ m^{\phi(N)-d} \bmod N \right)$$

Principe général

- Effectuer une *double exponentiation* pour calculer :

$$(s, c) \leftarrow \left(m^d \bmod N,\ m^{\phi(N)-d} \bmod N\right)$$

- Vérifier la cohérence du calcul :

$$s \cdot c \stackrel{?}{\equiv} 1 \bmod N\ .$$

Principe général

- Effectuer une *double exponentiation* pour calculer :

$$(s, c) \leftarrow \left(m^d \bmod N,\ m^{\phi(N)-d} \bmod N\right)$$

- Vérifier la cohérence du calcul :

$$s \cdot c \stackrel{?}{\equiv} 1 \bmod N\ .$$

- Si aucune erreur ne se produit :

$$s \cdot c \equiv m^d \cdot m^{\phi(N)-d} \equiv m^{\phi(N)} \equiv 1 \bmod N\ .$$

Principe général

- Effectuer une *double exponentiation* pour calculer :

$$(s, c) \leftarrow \left(m^d \bmod N,\ m^{\phi(N)-d} \bmod N\right)$$

- Vérifier la cohérence du calcul :

$$s \cdot c \stackrel{?}{\equiv} 1 \bmod N\ .$$

- Si une erreur se produit :

$$s \cdot c \not\equiv m^d \cdot m^{\phi(N)-d} \equiv m^{\phi(N)} \equiv 1 \bmod N\ .$$

Principe général

- Effectuer une *double exponentiation* pour calculer :

$$(s, c) \leftarrow \left(m^d \bmod N,\ m^{\phi(N)-d} \bmod N\right)$$

- Vérifier la cohérence du calcul :

$$s \cdot c \stackrel{?}{\equiv} 1 \bmod N\ .$$

- Si une erreur se produit :

$$s \cdot c \not\equiv m^d \cdot m^{\phi(N)-d} \equiv m^{\phi(N)} \equiv 1 \bmod N\ .$$

Problème

Concevoir un algorithme de double exponentiation.

Chaîne d'additions

Definition (Chaîne d'additions)

Une *chaîne d'additions* pour b est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$ et $x_n = b$
- pour tout k il existe $i, j < k$ t.q. $x_k = x_i + x_j$

Chaîne d'additions

Definition (Chaîne d'additions)

Une *chaîne d'additions* pour b est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$ et $x_n = b$
- pour tout k il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer m^b : suite $m_k = m^{x_k}$
  - $m_0 \leftarrow m$

Chaîne d'additions

Definition (Chaîne d'additions)

Une *chaîne d'additions* pour b est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$ et $x_n = b$
- pour tout k il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer m^b : suite $m_k = m^{x_k}$
  - $m_0 \leftarrow m$
  - pour $k = 1 \ldots n$:

$$m_k \leftarrow m_i \cdot m_j$$

Chaîne d'additions

Definition (Chaîne d'additions)

Une *chaîne d'additions* pour b est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$ et $x_n = b$
- pour tout k il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer m^b : suite $m_k = m^{x_k}$
  - $m_0 \leftarrow m$
  - pour $k = 1 \ldots n$:

$$m_k \leftarrow m_i \cdot m_j = m^{x_i} \cdot m^{x_j} = m^{x_i + x_j} = m^{x_k}$$

Chaîne d'additions

Definition (Chaîne d'additions)

Une *chaîne d'additions* pour b est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$ et $x_n = b$
- pour tout k il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer m^b : suite $m_k = m^{x_k}$
  - $m_0 \leftarrow m$
  - pour $k = 1 \ldots n$:

$$m_k \leftarrow m_i \cdot m_j = m^{x_i} \cdot m^{x_j} = m^{x_i + x_j} = m^{x_k}$$

  - Finalement : $m_n = m^b$

Double chaîne d'additions

Definition (Double chaîne d'additions)

Une *double chaîne d'additions* pour (a, b) est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$, $x_n = b$ et $a \in \{x_i\}$
- pour tout k, il existe $i, j < k$ t.q. $x_k = x_i + x_j$

Double chaîne d'additions

Definition (Double chaîne d'additions)

Une *double chaîne d'additions* pour (a, b) est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$, $x_n = b$ et $a \in \{x_i\}$
- pour tout k, il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer :

$$(m^a, m^b)$$

Double chaîne d'additions

Definition (Double chaîne d'additions)

Une *double chaîne d'additions* pour (a, b) est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$, $x_n = b$ et $a \in \{x_i\}$
- pour tout k, il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer :

$$(m^a, m^b)$$

- un algorithme de double exponentiation

Double chaîne d'additions

Definition (Double chaîne d'additions)

Une *double chaîne d'additions* pour (a, b) est une suite $x_0, x_1, \cdots, x_n$ telle que :

- $x_0 = 1$, $x_n = b$ et $a \in \{x_i\}$
- pour tout k, il existe $i, j < k$ t.q. $x_k = x_i + x_j$

- Méthode pour calculer :

$$(m^a, m^b)$$

- un algorithme de double exponentiation

Problème

Méthode de construction de doubles chaînes d'additions.

Construction d'une double chaîne d'additions

But

Construire une double chaîne d'additions pour un couple (a, b) :

- aussi courte que possible
  - Nb. de multiplications dans l'exponentiation = nb. d'additions dans la chaîne
- appropriée pour une implémentation restreinte en mémoire
  - Nb. de registres = nb. de x_i à stocker

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Exemple

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + b_i, b_i) & \text{if } \omega_i = 1 \\ \end{cases}$$

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Exemple

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + b_i, b_i) & \text{if } \omega_i = 1 \\ (2a_i, b_i) & \text{if } \omega_i = 2 \end{cases}$$

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Exemple

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + b_i, b_i) & \text{if } \omega_i = 1 \\ (2a_i, b_i) & \text{if } \omega_i = 2 \\ (a_i + 1, b_i) & \text{if } \omega_i = 3 \end{cases}$$

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Exemple

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + b_i, b_i) & \text{if } \omega_i = 1 \\ (2a_i, b_i) & \text{if } \omega_i = 2 \\ (a_i + 1, b_i) & \text{if } \omega_i = 3 \\ (a_i, a_i + b_i) & \text{if } \omega_i = 4 \end{cases}$$

Construction d'une double chaîne d'additions

- On garde 3 résultats intermédiaires : a_i, b_i, et 1
  - *i.e.* m^{a_i}, m^{b_i}, et m pour l'exponentiation
- On veut une suite $(a_i, b_i)_{i \leq n}$
  - t.q. $(a_0, b_0) = (0, 1)$ et $(a_n, b_n) = (a, b)$
  - (a_{i+1}, b_{i+1}) est dérivé de (a_i, b_i) par une addition
- On a besoin d'une représentation $\omega = (\omega_1, \ldots, \omega_n)$ où ω_i indique l'addition à effectuer

Exemple

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i + b_i, b_i) & \text{if } \omega_i = 1 \\ (2a_i, b_i) & \text{if } \omega_i = 2 \\ (a_i + 1, b_i) & \text{if } \omega_i = 3 \\ (a_i, a_i + b_i) & \text{if } \omega_i = 4 \\ \ldots \end{cases}$$

Construction d'une double chaîne d'additions

- Construction de la chaîne ω :
  - On part du couple (a, b)
  - On construit la suite inverse $(\alpha_i, \beta_i)_i$ s.t.
    - $(\alpha_0, \beta_0) = (a, b)$
    - $(\alpha_n, \beta_n) = (0, 1)$

Construction d'une double chaîne d'additions

- Construction de la chaîne ω :
  - On part du couple (a, b)
  - On construit la suite inverse $(\alpha_i, \beta_i)_i$ s.t.
    - $(\alpha_0, \beta_0) = (a, b)$
    - $(\alpha_n, \beta_n) = (0, 1)$
  - $(\alpha_{i+1}, \beta_{i+1})$ est calculé à partir de α_i, β_i et 1
    - par soustraction ou division par 2

Construction d'une double chaîne d'additions

- Construction de la chaîne ω :
  - On part du couple (a, b)
  - On construit la suite inverse $(\alpha_i, \beta_i)_i$ s.t.
    - $(\alpha_0, \beta_0) = (a, b)$
    - $(\alpha_n, \beta_n) = (0, 1)$
  - $(\alpha_{i+1}, \beta_{i+1})$ est calculé à partir de α_i, β_i et 1
    - par soustraction ou division par 2
- On suppose $a \leq b$ et on conserve $\alpha_i \leq \beta_i$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si} \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ \\ \\ \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si} \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si} \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} \quad \omega \leftarrow (01||\omega) \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} & \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} & \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 & \omega \leftarrow (1||\omega) \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} & \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} & \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 & \omega \leftarrow (1||\omega) \end{cases}$$

Calcul de la suite $(a_i, b_i)_i$

$$(a_{i+1}, b_{i+1}) = \begin{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} \quad \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 \quad \omega \leftarrow (1||\omega) \end{cases}$$

Calcul de la suite $(a_i, b_i)_i$

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i, 2b_i) & \text{si } \omega_i = 00 \\ \\ \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} \quad \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 \quad \omega \leftarrow (1||\omega) \end{cases}$$

Calcul de la suite $(a_i, b_i)_i$

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i, 2b_i) & \text{si } \omega_i = 00 \\ (a_i, 2b_i + 1) & \text{si } \omega_i = 01 \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} \quad \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 \quad \omega \leftarrow (1||\omega) \end{cases}$$

Calcul de la suite $(a_i, b_i)_i$

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i, 2b_i) & \text{si } \omega_i = 00 \\ (a_i, 2b_i + 1) & \text{si } \omega_i = 01 \\ (b_i, a_i + b_i) & \text{si } \omega_i = 1 \end{cases}$$

Heuristique de construction

Calcul de la chaîne ω

$$(\alpha_{i+1}, \beta_{i+1}) = \begin{cases} (\alpha_i, \beta_i/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 0 \end{cases} \quad \omega \leftarrow (00||\omega) \\ (\alpha_i, (\beta_i - 1)/2) & \text{si } \begin{cases} \alpha_i \leq \beta_i/2 \\ \beta_i \bmod 2 = 1 \end{cases} \quad \omega \leftarrow (01||\omega) \\ (\beta_i - \alpha_i, \alpha_i) & \text{si } \alpha_i > \beta_i/2 \quad \omega \leftarrow (1||\omega) \end{cases}$$

Calcul de la suite $(a_i, b_i)_i$

$$(a_{i+1}, b_{i+1}) = \begin{cases} (a_i, 2b_i) & \text{si } \omega_i = 00 \\ (a_i, 2b_i + 1) & \text{si } \omega_i = 01 \\ (b_i, a_i + b_i) & \text{si } \omega_i = 1 \end{cases}$$

$\Longrightarrow$ algorithme de double exponentiation

Résumé

- On a une double chaîne d'additions

Résumé

- On a une double chaîne d'additions
  - donc une double exponentiation

Résumé

- On a une double chaîne d'additions
  - donc une double exponentiation
  - donc une exponentiation sécurisée

$$(s, c) \leftarrow \text{DoubleExp}\big(m, (d, \phi(N) - d), N\big)$$
if $(s \cdot c \bmod N = 1)$ **return** s
else return "error"

Résumé

- On a une double chaîne d'additions
  - donc une double exponentiation
  - donc une exponentiation sécurisée

$$(s, c) \leftarrow \text{DoubleExp}\big(m, (d, \phi(N) - d), N\big)$$
if $(s \cdot c \bmod N = 1)$ **return** s
else return "error"

- Généralisation au RSA-CRT

Comparaison avec le solutions précédentes

- Contre-mesures par extension du module :
  - $(+)$ marche avec tout algorithme d'exponentiation
  - $(-)$ module plus grand $\Longrightarrow$ multiplications modulaires plus lentes

Comparaison avec le solutions précédentes

- Contre-mesures par extension du module :
  - (+) marche avec tout algorithme d'exponentiation
  - (−) module plus grand $\Longrightarrow$ multiplications modulaires plus lentes
- Exponentiations sécurisées précédentes
  - (+) pas de calculs additionnels (ni pré-calculs)
  - (−) plus de multiplications modulaires (21%)

Comparaison avec le solutions précédentes

- Contre-mesures par extension du module :
  - $(+)$ marche avec tout algorithme d'exponentiation
  - $(-)$ module plus grand $\Longrightarrow$ multiplications modulaires plus lentes
- Exponentiations sécurisées précédentes
  - $(+)$ pas de calculs additionnels (ni pré-calculs)
  - $(-)$ plus de multiplications modulaires (21%)
- Estimation théorique des performances :
  - notre contre-mesure est plus avantageuse que les exponentiations sécurisées précédentes
  - de l'ordre des contre-mesures par extension de module

Autres travaux

- Sécurisaion des chiffrements par blocs :
  - doubler certains tours uniquement
  - combien de tours doivent être doublés ?
- Attaque sur les tours du milieu du DES publiée à CHES 2009 [Riv09b]

Autres travaux

- Sécurisaion des chiffrements par blocs :
  - doubler certains tours uniquement
  - combien de tours doivent être doublés ?
- Attaque sur les tours du milieu du DES publiée à CHES 2009 [Riv09b]
- Implémentation des contrôles de cohérence

Autres travaux

- Sécurisaion des chiffrements par blocs :
  - doubler certains tours uniquement
  - combien de tours doivent être doublés ?
- Attaque sur les tours du milieu du DES publiée à CHES 2009 [Riv09b]
- Implémentation des contrôles de cohérence
  - peuvent être sautés par une injection de faute
    $\Longrightarrow$ attaques par double faute [KQ07]
  - contre-mesure simple et efficace proposée à WISTP 2009 [DGRS09]

Plan

1. Introduction
2. Attaques par canaux auxiliaires
   - Modèle
   - Méthodes classiques
   - Masquage et attaques d'ordre supérieur
3. Schémas de masquage
   - Schémas de premier ordre
   - **Cryptanalyse d'un schéma d'ordre supérieur**
   - Schémas d'ordre supérieur
4. Attaques par fautes
   - **Sécurisation du RSA par double chaîne d'additions**
   - Autres travaux
5. Conclusion

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage
- Nouvelles attaques par fautes et contre-mesures

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage
- Nouvelles attaques par fautes et contre-mesures

Perspectives

- Etude de la MIA et des attaques profilées d'ordre supérieur

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage
- Nouvelles attaques par fautes et contre-mesures

Perspectives

- Etude de la MIA et des attaques profilées d'ordre supérieur
- Design de schémas de masquage d'ordre supérieur

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage
- Nouvelles attaques par fautes et contre-mesures

Perspectives

- Etude de la MIA et des attaques profilées d'ordre supérieur
- Design de schémas de masquage d'ordre supérieur
- Recherche de techniques de randomisation avec une meilleure résistance que le masquage Booléen

Conclusion

Contributions

- Analyse théorique et amélioration des attaques par canaux auxiliaires
- Design et cryptanalyse de schémas de masquage
- Nouvelles attaques par fautes et contre-mesures

Perspectives

- Etude de la MIA et des attaques profilées d'ordre supérieur
- Design de schémas de masquage d'ordre supérieur
- Recherche de techniques de randomisation avec une meilleure résistance que le masquage Booléen
- Recherche d'algorithmes d'exponentiation sécurisés plus rapides

Publications

CGPR08 J.-S. Coron, C. Giraud, E. Prouff, and M. Rivain. *Attack and Improvement of a Secure S-Box Calculation Based on the Fourier Transform*. In CHES 2008.

CPR07 J.-S. Coron, E. Prouff, and M. Rivain. *Side Channel Cryptanalysis of a Higher Order Masking Scheme*. In CHES 2007.

DGRS09 E. Dottax, C. Giraud, M. Rivain, and Y. Sierra. *On Second-Order Fault Analysis Resistance for CRT-RSA Implementations*. In WISTP 2009.

PR07 E. Prouff and M. Rivain. *A Generic Method for Secure SBox Implementation*. In WISA 2007.

PR09 E. Prouff and M. Rivain. *Theoretical and Practical Aspects of Mutual Information Based Side Channel Analysis*. In ANCS 2009.

Publications

PRB09 E. Prouff, M. Rivain, and R. Bévan. *Statistical Analysis of Second Order Differential Power Analysis*. IEEE Trans. Comput., 58(6) :799–811, 2009.

Riv08 M. Rivain. *On the Exact Success Rate of Side Channel Analysis in the Gaussian Model*. In SAC 2008.

Riv09a M. Rivain. *Securing RSA against Fault Analysis by Double Addition Chain Exponentiation*. In CT-RSA 2009.

Riv09b M. Rivain. *Differential Fault Analysis on DES Middle Rounds*. In CHES 2009.

RDP08 M. Rivain, E. Dottax, and E. Prouff. *Block Ciphers Implementations Provably Secure Against Second Order Side Channel Analysis*. In FSE 2008.

RPD09 M. Rivain, E. Prouff, and J. Doget. *Higher-Order Masking and Shuffling for Software Implementations of Block Ciphers*. In CHES 2009.

Bibliographie

Aum07 S. Aumônier. *Generalized Correlation Power Analysis*. Ecrypt Workshop Tools For Cryptanalysis 2007.

BDL97 D. Boneh, R. DeMillo, and R. Lipton. *On the Importance of Checking Cryptographic Protocols for Faults*. In EUROCRYPT '97,

BOS03 J. Blömer, M. Otto, and J.-P. Seifert. *A New RSA-CRT Algorithm Secure against Bellcore Attacks*. In CCS'03.

BNP07 A. Boscher, R. Naciri, and E. Prouff. *CRT RSA Algorithm Protected against Fault Attacks*. In WISTP 2007.

BCO04 E. Brier, C. Clavier, and F. Olivier. *Correlation Power Analysis with a Leakage Model*. In CHES 2004

CJRR99 S. Chari, C. Jutla, J. Rao, and P. Rohatgi. *Towards Sound Approaches to Counteract Power-Analysis Attacks*. In *CRYPTO '99*.

Bibliographie

CJ05b M. Ciet and M. Joye. *Practical Fault Countermeasures for Chinese Remaindering Based RSA*. In FDTC'05.

GBTP08 B. Gierlichs, L. Batina, P. Tuyls and B. Preneel. *Mutual Information Analysis*. In CHES 2008.

Gir06 C. Giraud. *An RSA Implementation Resistant to Fault Attacks and to Simple Power Analysis*. IEEE Transactions on Computers, 55(9) :1116–1120, 2006.

KQ07 C. H. Kim and J.-J. Quisquater. *Fault Attacks for CRT Based RSA : New Attacks, New Results, and New Countermeasures*. In WISTP 2007.

KJJ99 P. Kocher, J. Jaffe, and B. Jun. *Differential Power Analysis*. In CRYPTO '99.

Mes00 T. Messerges. *Using Second-order Power Analysis to Attack DPA Resistant Software*. In CHES 2000.

Bibliographie

PY06 R. Phan and S.-M. Yen. *Amplifying Side-Channel Attacks with Techniques from Block Cipher Cryptanalysis*. In CARDIS 2006

SP06 K. Schramm and C. Paar. Higher Order Masking of the AES. In *CT-RSA 2006*.

Sha98 A. Shamir. *Improved Method and Apparatus for Protecting Public Key Schemes from Timing and Fault Attacks*. Publication number : WO9852319, Nov. 1998.

SMY09 F.-X. Standaert, T. Malkin, and M. Yung. A Unified Framework for the Analysis of Side-Channel Key Recovery Attacks. In EUROCRYPT 2009.

Vig08 D. Vigilant. *RSA with CRT : A New Cost-Effective Solution to Thwart Fault Attacks*. In CHES 2008.