Introduction à la Sécurité Prouvable

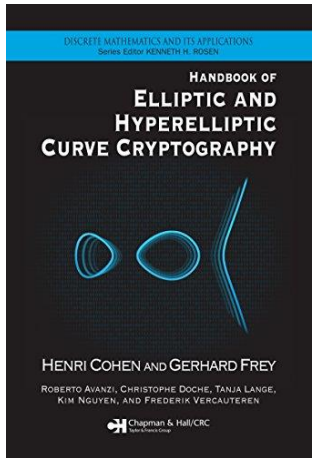
Application à la cryptographie à clé publique

Olivier Sanders (Orange)

Echange Sécurisé

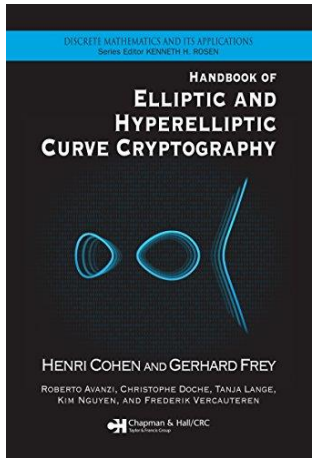
amazon

U.S.



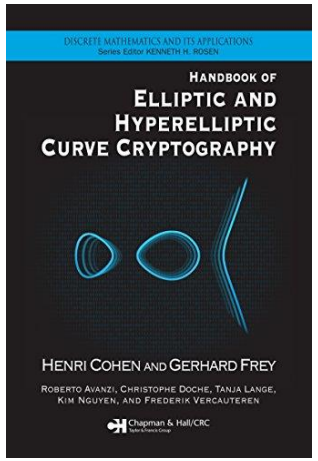
Comment transmettre une information sensible?

Echange Sécurisé



Comment transmettre une information sensible?

Echange Sécurisé



Comment transmettre sk?

Echange Clés Diffie-Hellman

p,q premiers
g d'ordre q mod p

amazon
U.S.

Y = g^y

X = g^x

Secret: x

K = Y^x

Secret: y

K = X^y

Clé partagée: **g^xy**

Chiffrement El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

Clé publique: Y = g^y

X = g^x et Z = M * g^xy

Secret: y

Déchiffrement: 1) K = X^y

2) M = Z * K^(-1)

Chiffrement El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

Clé publique Y = g^y

X = g^x et Z = M * g^xy

amazon U.S.

Secret: y

Est-ce sûr?

Chiffrement El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

Clé publique Y = g^y

X = g^x et Z = M * g^xy

amazon U.S.

Secret: y

Est-ce sûr?

Que signifie « être sûr »?

Définition Sécurité

amazon

U.S.

Clé publique: pk

Enc(pk, données)

Clé secrète: sk

Que signifie « être sûr »?

Définition Sécurité

Clé publique: pk

Enc(pk, données)

Clé secrète: sk

Que signifie « être sûr »?

Difficile de retrouver la clé secrète?

Définition Sécurité

Clé publique: pk

Enc(pk, données)

Clé secrète: sk

Que signifie « être sûr »?

Difficile de retrouver la clé secrète? Difficile de retrouver le message?

Définition Sécurité

Clé publique: pk

Enc(pk, données)

Clé secrète: sk

Que signifie « être sûr »?

Difficile de retrouver la clé secrète?

Difficile de retrouver le message?

Difficile d'obtenir de l'information sur le message?

Définition Sécurité

- Définir la sécurité d'un système nécessite
  - 1) la définition formelle (syntaxe) du système
  - 2) l'objectif d'un attaquant contre le système
  - 3) les moyens d'un attaquant

Chiffrement à clé publique

Un mécanisme de chiffrement à clé publique est défini par 3 algorithmes

- Keygen(1^n): cet algorithme prend en entrée un paramètre de sécurité n et retourne une paire de clés (sk,pk).
- Encrypt(pk,m): cet algorithme prend en entrée une clé publique pk et un message m et retourne un chiffré c.
- Decrypt(sk,c): cet algorithme prend en entrée une clé secrète sk et un chiffré c et retourne un message m.

A l'exception d'un nombre négligeable de clés (sk,pk), ces algorithmes satisfont:

$$\text{Decrypt}(sk,\text{Encrypt}(pk,m)) = m$$

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

Les objectifs et moyens de sécurité sont en général définis par un jeu de sécurité impliquant un adversaire A. Il en existe plusieurs pour une même primitive, représentant la diversité des cas d'usage.

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

Une liberté totale est donnée à l'adversaire sur le choix des messages qu'il essaie de détecter

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

L'adversaire remporte le jeu si b' = b.

On définit l'avantage de A comme |P(b'=b) -1/2|

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

L'adversaire remporte le jeu si b' = b.

L'adversaire doit deviner lequel des deux messages a été chiffré

On définit l'avantage de A comme |P(b'=b) -1/2|

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

L'adversaire remporte le jeu si b' = b.

On définit l'avantage de A comme |P(b'=b) -1/2|

On peut gagner naïvement avec proba ½. Un adversaire n'est intéressant que s'il diffère de cette valeur

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

IND = Indistinguishabitity représente l'objectif de l'adversaire:
Distinguer un chiffré de m_0 d'un chiffré de m_1

L'adversaire remporte le jeu si b' = b.

On définit l'avantage de A comme |P(b'=b) -1/2|

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

CPA = Chosen Plaintext Attacks représente les moyens de l'adversaire:
Capacité de chiffrer n'importe quel message à partir de la clé publique pk

L'adversaire remporte le jeu si b' = b.

On définit l'avantage de A comme |P(b'=b) -1/2|

Modèle de sécurité

1) (sk,pk) <- Keygen(1^n)

2) m choisi aléatoirement parmi l'ensemble M des messages possibles

3) c <- Encrypt(pk, m)

4) m' <- A(pk, c)

Jeu OW-CPA

OW = One-Way représente l'objectif de l'adversaire: Inverser la fonction de chiffrement pour retrouver le message

L'adversaire remporte le jeu si m' = m.

On définit l'avantage de A comme |P(m'=m) -1/M|

Définition Sécurité

- Un système satisfait une propriété de sécurité (*e.g.* IND-CPA) si tout adversaire A fonctionnant en temps polynomial dispose d'un avantage au plus négligeable

- Un modèle de sécurité permet de définir précisément la sécurité d'un système

  → Un système peut satisfaire une propriété (e.g. OW-CPA) et non une autre (e.g. IND-CPA)

- Certaines propriétés de sécurité peuvent être hiérarchisées

  e.g. IND-CPA → OW-CPA (voir TD)

Preuves de Sécurité

- Une fois que la sécurité est définie, il reste à la prouver!
- L'approche de la cryptographie à clé publique est de reposer sur des hypothèses calculatoires, à savoir la difficulté de résoudre certains problèmes mathématiques
- Ces problèmes mathématiques permettent de dimensionner les paramètres du système
- Exemples de problèmes
  - Logarithme discret (DL): étant donnés g et g^x mod p, retrouver x
  - Factorisation: étant donné N = p*q, où p et q premiers, retrouver p
  - CDH : étant donnés g, g^x, g^y, calculer g^xy
  - DDH: étant donnés g, g^x,g^y et g^z, décider si z = xy ou si z est aléatoire.

Preuves de Sécurité: El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

amazon
U.S.

Clé publique Y = g^y

X = g^x et Z = M * g^xy

Secret: y

Comment identifier la bonne hypothèse calculatoire?

Preuves de Sécurité: El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

amazon U.S.

Clé publique Y = g^y

X = g^x et Z = M * g^xy

Secret: y

Exemple: supposons qu'on puisse résoudre le problème DL

1) On retrouve y à partir de Y en résolvant DL

2) On calcule X^y = g^xy et on retrouve m

→ Si on sait résoudre DL, on peut attaquer l'IND-CPA d'El Gamal

Preuves de Sécurité: El Gamal

aléa: x

p,q premiers
g d'ordre q mod p

amazon U.S.

Clé publique Y = g^y

X = g^x et Z = M * g^xy

Secret: y

Exemple: supposons qu'on puisse résoudre le problème DDH

1) Dans le jeu IND-CPA, étant donné c = (g^x, m_b g^xy), on calcule K_0 = c[2]/m_0 et K_1 = c[2]/m_1

2) On utilise le solveur DDH sur (g, g^x, g^y, K_b'), pour b'=0,1. Le solveur retourne 1 dans le cas où b=b'

→ Si on sait résoudre DDH, on peut attaquer l'IND-CPA d'El Gamal

Preuves de Sécurité

- On a montré que si on sait casser DL, alors on sait casser l'IND-CPA de El Gamal
- On a montré que si on sait casser DDH, alors on sait casser l'IND-CPA de El Gamal
- On a besoin d'un résultat dans l'autre sens:
  « si on sait casser l'IND-CPA de El Gamal, alors on sait casser le problème XXX »

→Preuves par réduction

1) on suppose l'existence d'un adversaire A contre l'IND-CPA disposant d'un avantage q
2) on montre qu'on peut utiliser A pour résoudre le problème XXX avec un avantage proche de q

→si XXX est dur alors q ne peut être que négligeable

Preuve que El Gamal repose sur DDH

Soit A un adversaire contre l'IND-CPA d'El Gamal avec avantage q = |P(b'=b) -1/2|. On construit D prenant en entrée (g, g^x,g^y et g^z) et retournant 1 si z=xy (et 0 sinon)

1) On définit pk =g^y

2) (m_0,m_1) <- A(pk)

3) On construit c = (g^x, g^z m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

5) Si b=b', D retourne 1. Sinon D retourne 0.

Preuve que El Gamal repose sur DDH

Soit A un adversaire contre l'IND-CPA d'El Gamal avec avantage q = |P(b'=b) -1/2|. On construit D prenant en entrée (g, g^x,g^y et g^z) et retournant 1 si z=xy (et 0 sinon)

1) On définit pk =g^y

2) (m_0,m_1) <- A(pk)

3) On construit c = (g^x, g^z m_b) pour b aléatoire.

4) b' <- A(pk, c)

5) Si b=b', D retourne 1. Sinon D retourne 0.

Il faut simuler correctement toutes les entrées de A dans le jeu IND-CPA:
Aucune hypothèse n'est faite sur le comportement de A

Preuve que El Gamal repose sur DDH

Soit A un adversaire contre l'IND-CPA d'El Gamal avec avantage q = |P(b'=b) -1/2|. On construit D prenant en entrée (g, g^x,g^y et g^z) et retournant 1 si z=xy (et 0 sinon)

1) On définit pk =g^y

Il n'y a pas besoin de construire les données secrètes pour A (e.g. sk)

2) (m_0,m_1) <- A(pk)

3) On construit c = (g^x, g^z m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

5) Si b=b', D retourne 1. Sinon D retourne 0.

Preuves de Sécurité

Pour que D soit utile, il faut que son comportement diffère en fonction de z

$$|P(\ D(g,g^x,g^y,g^{xy})=1) - P(D(g,g^x,g^y,g^z)=1)|$$
$$= |\ P(\text{A retourne b'=b } |z = xy) - P(\text{A retourne b'=b } |z \text{ aléa})\ |$$

Preuves de Sécurité

Pour que D soit utile, il faut que son comportement diffère en fonction de z

|P(D(g,g^x,g^y,g^xy)=1) – P(D(g,g^x,g^y,g^z)=1)|
= | P(A retourne b'=b |z =xy) - P(A retourne b'=b |z aléa) |

= | 1/2+q

Ici c = (g^x, g^xy m_b), le jeu est parfaitement simulé:
A retourne la bonne réponse avec proba 1/2+q

Preuves de Sécurité

Pour que D soit utile, il faut que son comportement diffère en fonction de z

$$|P(\ D(g,g^x,g^y,g^{xy})=1) - P(D(g,g^x,g^y,g^z)=1)|$$
$$= |\ P(\text{A retourne b'=b } |z = xy) - P(\text{A retourne b'=b } |z \text{ aléa})\ |$$

$$= |\ 1/2+q\ -1/2\ |$$

$$= q$$

Ici $c = (g^x, g^z m_b)$, le message m_b est parfaitement masqué. A ne peut retourner le bon b' qu'avec proba proche de 1/2

Preuves de Sécurité

Pour que D soit utile, il faut que son comportement diffère en fonction de z

$$|P(D(g,g^x,g^y,g^{xy})=1) – P(D(g,g^x,g^y,g^z)=1)|$$
$$= | P(A \text{ retourne } b'=b \mid z =xy) - P(A \text{ retourne } b'=b \mid z \text{ aléa}) |$$

$$= | 1/2+q \ -1/2 |$$

$$= q$$

Le comportement de D diverge significativement en fonction de z si q n'est pas négligeable

→Si DDH est dur q doit être négligeable!

Preuves de Sécurité

Pour que D soit utile, il faut que son comportement diffère en fonction de z

$$|P(D(g,g^x,g^y,g^{xy})=1) – P(D(g,g^x,g^y,g^z)=1)|$$
$$= | P(A \text{ retourne } b'=b \mid z = xy) - P(A \text{ retourne } b'=b \mid z \text{ aléa}) |$$

$$= | 1/2+q \ -1/2 |$$

$$= q$$

Le comportement de D diverge significativement en fonction de z si q n'est pas négligeable

→Si DDH est dur El Gamal satisfait la propriété IND-CPA

Preuves de Sécurité

- Les preuves par réduction sont des outils puissants permettant d'évaluer le niveau de sécurité d'un système et d'en déduire des paramètres concrets

- Elles n'ont de sens que si le mécanisme cryptographique est utilisé conformément au modèle.
  - Exemple de RSA PKCS#1.5

Le chiffrement RSA (Rappel)

- Clé publique :
  - N = pq produit de deux grands nombres premiers
  - e un entier premier avec $\varphi(N) = (p-1)(q-1)$
- Clé privée :
  - La factorisation de N
  - d tel que $e \times d = 1 \bmod \varphi(N)$

Fonctionnement basique (Rappel)

- Chiffrement basique de m:
  - $C = m^e \bmod N$
- Déchiffrement :
  - calcul de racine e-ième modulaire
  - $m = C^d \bmod N$
- En effet, on a :

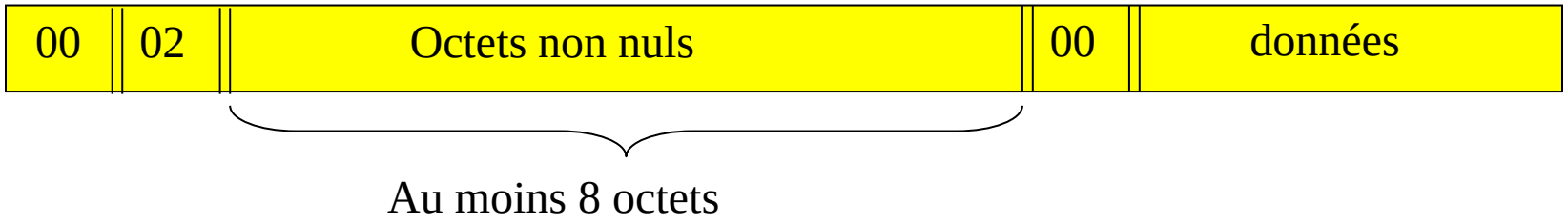
$$C^d = (m^e)^d \bmod N = m^{ed} \bmod N$$
$$= m \bmod N$$

Le chiffrement RSA

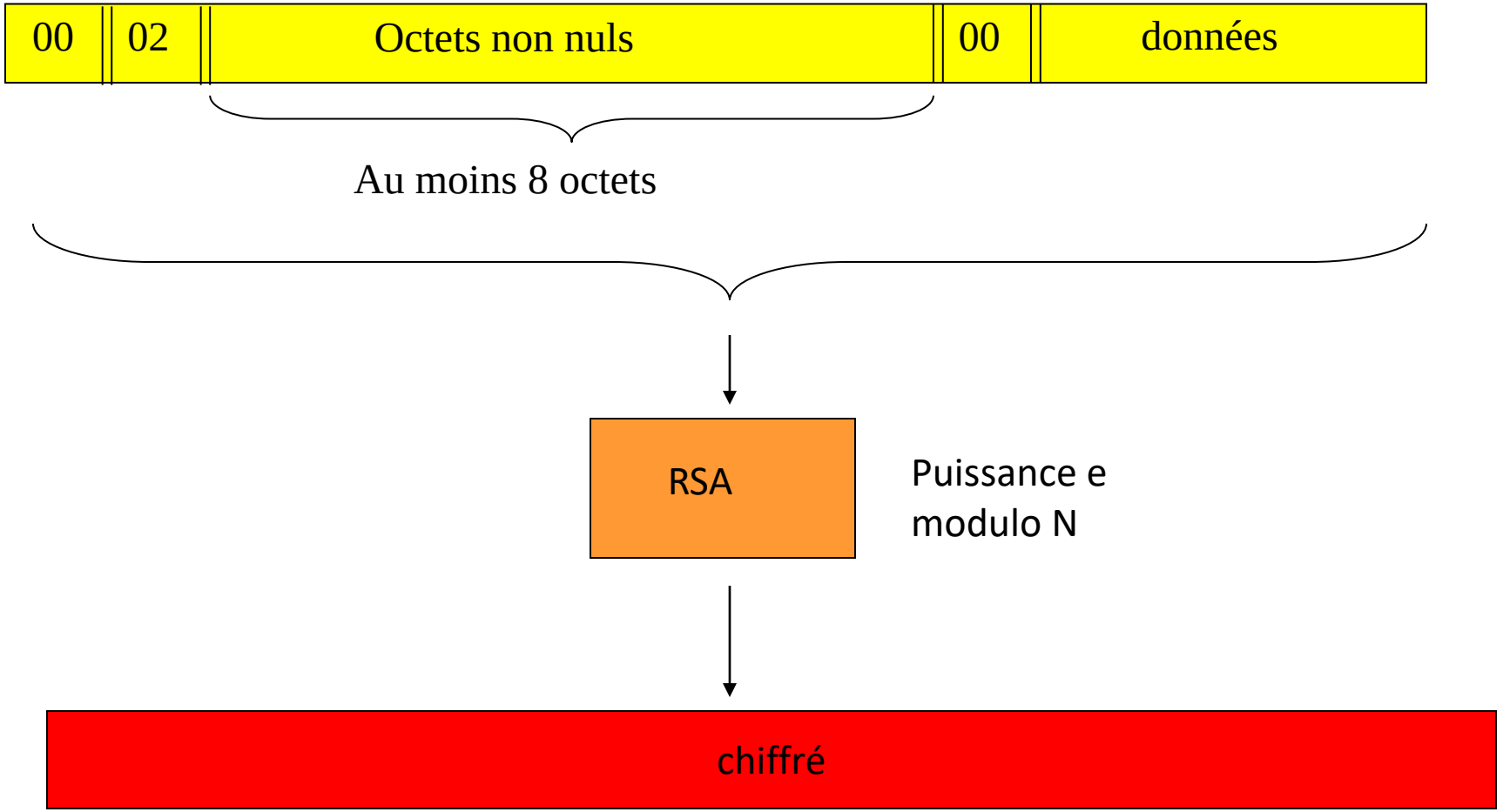
- Pour éviter toutes les attaques élémentaires sur RSA :
  - on utilise un padding, ou encodage, du message avant chiffrement
  - En déchiffrement, on vérifie si le padding est correct avant de renvoyer le clair
- Normes : PKCS #1 versions 1.5 et 2.1 proposées par RSA Laboratories

Description de PKCS #1 v1.5

- Norme de 1993 publiée par RSA laboratories
- Recommandations pour l'utilisation de RSA
- Préconise l'emploi d'un padding



Chiffrement RSA PKCS #1 v1.5



Déchiffrement RSA PKCS #1 v1.5

- Pour déchiffrer un chiffré C:
  - On calcule $M = C^d \bmod N$
  - On vérifie que M sur k octets se compose
    - D'un octet nul
    - D'un octet égal à 0x02
    - D'au moins 8 octets non nuls
    - D'un octet nul parmi ceux qui restent
- Si un des tests échoue, le clair est invalide

PKCS #1 v1.5

- Pas de sécurité contre les attaques à chiffrés choisis : attaque par réaction de Bleichenbacher (Crypto 1998)
  - En pratique (SSL 3), un attaquant peut exploiter le message d'erreur renvoyé par le serveur si le chiffré n'est pas valide
  - Ce serveur joue le rôle d'un oracle qui retourne un bit selon la validité d'un chiffré soumis
  - L'adversaire peut retrouver le clair grâce à un accès à des requêtes à cet oracle

Attaque de Bleichenbacher

Attaquant : cherche à déchiffrer C

Serveur

Chiffré C*

C* : modification astucieuse de C

Déchiffrement de C*

Vérification de conformité

Le clair n'est pas correctement encodé

erreur

Attaque de Bleichenbacher

Attaquant : cherche à déchiffrer C

Serveur

Chiffré C*

C* : modification astucieuse de C

Déchiffrement de C*

Vérification de conformité

OK

Le clair est correctement encodé

Attaque de Bleichenbacher

Attaquant : cherche à déchiffrer C

Serveur

Chiffré C*

$C_i = C \times r^e_i \bmod N$

Déchiffrement de C*

$M_i = (C \times r^e_i)^d \bmod N$
$= M \times r_i^{ed} \bmod N$
$= M \times r_i \bmod N$

OK

M_i est un clair conforme
En particulier, les octets de poids forts sont 0x00 0x02

Attaque de Bleichenbacher

- Si l'attaquant apprend de l'information pour assez de valeurs $M \cdot r_i$, il peut en déduire le clair M
- Précisément, si C_i est valide, alors (k=|N|/8) :

$$\mathbf{2 \cdot 2^{8(k-2)} \leq M \cdot r_i \leq 3 \cdot 2^{8(k-2)}}$$

- Cette équation permet de réduire l'ensemble des valeurs possibles pour le message M
- Pour un module de 1024 bits, quelques millions de requêtes sont nécessaires

Sécurité IND-CCA

1) (sk,pk) <- Keygen(1^n)

2) (m_0,m_1) <- A(pk)

3) c <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A(pk, c)

Jeu IND-CPA

Le jeu IND-CPA ne modélise pas l'accès de A à cet oracle de conformité du chiffrement

Sécurité IND-CCA

1) (sk,pk) <- Keygen(1^n)

2) $(m_0,m_1) \leftarrow A^{O}(pk)$

3) c* <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- $A^{O}(pk, c^*)$

Jeu IND-CCA

Le jeu IND-CCA offre un accès à un oracle de déchiffrement O:
O(c) retourne Decrypt(sk,m)

Sécurité IND-CCA

1) (sk,pk) <- Keygen(1^n)

2) $(m_0,m_1) \leftarrow A^{O}(pk)$

3) c* <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- $A^{O}(pk, c^*)$

Cet oracle est beaucoup plus puissant que celui de PKCS 1.5

Jeu IND-CCA

Le jeu IND-CCA offre un accès à un oracle de déchiffrement O:

O(c) retourne Decrypt(sk,m)

Sécurité IND-CCA

1) (sk,pk) <- Keygen(1^n)

2) $(m_0,m_1) \leftarrow A^{O}(pk)$

3) $c^* \leftarrow$ Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) $b' \leftarrow A^{O}(pk, c^*)$

Jeu IND-CCA

CCA = Chosen Ciphertext Attacks représente les moyens de l'adversaire:
Capacité de déchiffrer n'importe quel chiffré de son choix

Le jeu IND-CCA offre un accès à un oracle de déchiffrement O:

O(c) retourne Decrypt(sk,m)

O ne peut évidemment pas être sollicité sur c*

Preuves de Sécurité

- La sécurité IND-CCA est plus forte que la sécurité IND-CPA car elle octroie plus de capacités à l'attaquant
- Elle est cependant plus compliquée à satisfaire

→ conduit à des constructions plus complexes

- (TD) El Gamal satisfait-il la sécurité IND-CCA?

Il n'y a pas de bonne ou mauvaise propriété dans l'absolu, il faut choisir celle qui est adaptée au cas d'usage !

Le modèle de l'oracle aléatoire

- Plus généralement, il peut être utile de modéliser l'accès à certaines fonctions (souvent une fonction de hachage H) du système par des oracles

- Principe:
  - H ne peut être évalué par l'adversaire A directement
  - A doit demander le résultat de H(x) en transmettant x à l'oracle

- Intérêt:
  - Permet de choisir la valeur de H(x) du moment qu'elle soit bien distribuée
  - Permet de connaître toutes les valeurs x que l'adversaire cherche à évaluer

- Limite:
  - L'oracle aléatoire est une modélisation inexacte de H. Elle est donc contestable théoriquement mais très utile en pratique.

Exemple 1

- El Gamal (rappel).

  Clé secrète: y. Clé publique: g^y.

  Chiffrement(pk,m): soit x un entier aléatoire. On retourne c=(g^x,m * g^xy)

  Déchiffrement(sk,c): On retourne c[2]/c[1]^y

- On a prouvé El Gamal IND-CPA sûr sous l'hypothèse DDH.

- Dans certains groupes (e.g. groupes bilinéaires) DDH est facile.

Exemple 1

- Supposons que DDH soit facile: il existe un algorithme D distinguant (g,g^x,g^y,g^xy) de (g,g^x,g^y,g^z) où z est quelconque.

- El Gamal modifié:

  Clé secrète: y. Clé publique: g^y.

  Chiffrement(pk,m): soit x un entier aléatoire. On retourne c=(g^x,m * H(g^xy)) où H est une fonction de hachage.

  Déchiffrement(sk,c): On retourne c[2]/H(c[1]^y)

- Exercice: Montrer que El Gamal modifié est IND-CPA sous l'hypothèse CDH

Exemple 2

Problème RSA: étant donnés N = pq, e et y, trouver x tel que x^e =y mod N

Clé publique et clé secrète comme dans RSA basique.

Encrypt(pk,m): r choisi aléatoirement, retourne (r^e, H(r) *m) où H est une fonction de hachage modélisée comme un oracle aléatoire

Decrypt(sk,c): calcule k = c[1]^d et retourne c[2]/H(k)

(TD) Prouver que le chiffrement est IND-CPA si le problème RSA est dur

(TD) est-il IND-CCA sous la même hypothèse?