Mécanismes avancés/ Preuves par jeux

Olivier Sanders (Orange)

(Rappel) Chiffrement El Gamal

- Clé Publique: G un groupe d'ordre p. g un générateur de G. $h = g^x$
- Clé Secrète: x
- Encrypt(pk,m) : r aléa mod p. $C = (g^r, h^r * m)$
- Decrypt(sk,c): Soit $C = (c_1,c_2)$. $m = c_2 * c_1^{-x}$

(Rappel) Chiffrement El Gamal

- Clé Publique: G un groupe d'ordre p. g un générateur de G. $h = g^x$
- Clé Secrète: x
- Encrypt(pk,m) : r aléa mod p. $C = (g^r, h^r * m)$
- Decrypt(sk,c): Soit $C = (c_1,c_2)$. $m = c_2 * c_1^{(-x)}$
- El Gamal satisfait la propriété IND-CPA sous l'hypothèse DDH
- El Gamal ne satisfait pas la propriété IND-CCA

Naor-Yung

- Soit E le protocole de chiffrement El-Gamal.

- Keygen: deux exécutions de E.Keygen pour obtenir (sk_1,pk_1) et (sk_2,pk_2)

pk= (pk_1,pk_2) et sk =(sk_1)

- Encrypt(pk,m):

  C_1 = E.Encrypt(pk_1,m) et C_2 = E.Encrypt(pk_2,m).

  Génération d'une preuve P zéro-knowledge non-interactive que C_1 et C_2 chiffrés d'un même message m

  C = (C_1,C_2,P)

- Decrypt(sk,C)

  Vérification de la preuve P

  Si correct, m = Decrypt(sk_1,C_1)

Naor-Yung (Détails)

- On a $pk_1 = h_1 = g^{x_1}$ et $pk_2 = h_2 = g^{x_2}$
- $C_1 = [g^{r_1}, h_1^{r_1} * m]$ et $C_2 = [g^{r_2}, h_2^{r_2} * m]$
- Preuve P:

k_1 et k_2 aléatoires

$$\xrightarrow{K_1 = g^{k_1}\ ;\ K_2 = g^{k_2};\ K_3 = h_1^{k_1}/h_2^{k_2}}$$

$$\xleftarrow{c,\ \text{aléatoire}}$$

$$\xrightarrow{z_1 = k_1 + c * r_1;\ z_2 = k_2 + c * r_2}$$

$g^{z_1} = K_1 * C_1[1]^{c}$

$g^{z_2} = K_2 * C_2[1]^{c}$

$h_1^{z_1}/h_2^{z_2} =$
$K_3 * (C_1[2]/C_2[2])^{c}$

Naor-Yung (Détails)

- On a $pk_1 = h_1 = g^{x_1}$ et $pk_2 = h_2 = g^{x_2}$
- $C_1 = [g^{r_1}, h_1^{r_1} * m]$ et $C_2 = [g^{r_2}, h_2^{r_2} * m]$
- Preuve P:

k_1 et k_2 aléatoires

$$\xrightarrow{K_1 = g^{k_1} \,;\, K_2 = g^{k_2};\, K_3 = h_1^{k_1}/h_2^{k_2}}$$

$$\xleftarrow{c,\ \text{aléatoire}}$$

$$\xrightarrow{z_1 = k_1 + c * r_1;\ z_2 = k_2 + c * r_2}$$

$g^{z_1} = K_1 * C_1[1]^{c}$

$g^{z_2} = K_2 * C_2[1]^{c}$

$h_1^{z_1}/h_2^{z_2} = K_3 * (C_1[2]/C_2[2])^{c}$

On utilise Fiat-Shamir pour rendre la preuve non-interactive

Sécurité IND-CCA (Rappel)

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

2) (m_0,m_1) <- 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)

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

On veut montrer que le chiffrement est IND-CCA si El Gamal est IND-CPA:

- On fait l’hypothèse qu’il existe un adversaire A contre l’IND-CCA
- On montre comment utiliser A pour casser l’IND-CPA d’El Gamal

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

Difficulté 1: Insérer la clé publique pk du jeu IND-CPA ainsi que le challenge c* tout en étant capable de répondre aux requêtes de déchiffrement

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1 sous pk_1

Un chiffré El Gamal C_2 sous pk_2

Une preuve P que tout est bien formé

On génère la paire de clés (sk_1, pk_1) normalement

Difficulté 1: Insérer la clé publique pk du jeu IND-CPA ainsi que le challenge c^* tout en étant capable de répondre aux requêtes de déchiffrement

Solution: On traite différemment chaque chiffré

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

On insère le challenge dans la deuxième clé:
pk_2 = pk

Difficulté 1: Insérer la clé publique pk du jeu IND-CPA ainsi que le challenge c*
tout en étant capable de répondre aux requêtes de déchiffrement

Solution: On traite différemment chaque chiffré

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1 sous pk_1

Un chiffré El Gamal C_2 sous pk_2

Une preuve P que tout est bien formé

Lorsque l'adversaire soumet une requête de déchiffrement sur C = (C_1,C_2,P):
1) Je déchiffre en utilisant sk_1 et je retourne m?

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

Lorsque l'adversaire soumet une requête de déchiffrement sur C = (C_1,C_2,P):
1) ~~Je déchiffre en utilisant sk_1 et je retourne m~~?

Rien ne dit que C est bien formé !
Répondre serait une divergence par rapport au fonctionnement normal

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

Lorsque l'adversaire soumet une requête de déchiffrement sur C = (C_1,C_2,P):

1) Je vérifie que P est valide

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

Lorsque l'adversaire soumet une requête de déchiffrement sur C = (C_1,C_2,P):

1) Je vérifie que P est valide
2) Je déchiffre C_1 en utilisant sk_1

Le déchiffrement peut être parfaitement simulé

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2

Une preuve P que tout est
bien formé

Difficulté 2: Utiliser le chiffré challenge (IND-CCA) pour distinguer le chiffré challenge c* IND-CPA

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

Un chiffré El Gamal C_2
sous pk_2 = pk

Une preuve P que tout est
bien formé

C_2= c*

Difficulté 2: Utiliser le chiffré challenge (IND-CCA) pour distinguer le chiffré challenge c* IND-CPA

Solution (partielle): Utiliser c* pour C_2*

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

?????

Un chiffré El Gamal C_2
sous pk_2 = pk

C_2 = c*

Une preuve P que tout est
bien formé

?????

Mais comment construire de manière cohérente les autres éléments???

Preuves par Jeux

Le principe d'une preuve par jeux est de modifier progressivement le jeu initial G_0 pour arriver au jeu de notre choix G_n

- Chaque jeu intermédiaire G_i est construit de manière à être indistinguable du précédent.
- Ainsi, la probabilité de succès d'un adversaire dans le jeu G_n doit rester proche de celui de ce même adversaire dans le jeu initial
- Permet de limiter le nombre d'éléments à changer à chaque tour.

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1 sous pk_1

?????

Un chiffré El Gamal C_2 sous pk_2 = pk

Une preuve P que tout est bien formé

?????

C_2 = c*

Dans notre cas, on voudrait décorréler C_1 et P de C_2

Sécurité IND-CCA (Rappel)

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

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

3) C* = (C_1*,C_2*,P*) <- Encrypt(pk, m_b) pour un bit b choisi aléatoirement.

4) b' <- A^O(pk, C*)

Jeu IND-CCA –G_0

Preuves par Jeux

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

2) (m_0,m_1) <- $A^O(pk)$

3) Pour un bit b choisi aléatoirement:
   C*_1 <- Enc(pk_1,m_b)
   C*_2 <- Enc(pk_2,m_b)
   P' généré à l'aide du simulateur ZK
   C* = (C*_1,C*_2,P')

4) b' <- A^O(pk, C*)

Jeu G_1

On commence par changer P*

Preuves par Jeux

- Il faut alors prouver que G_0 et G_1 sont indistinguables

- Ici l’argument est simple, distinguer ces jeux revient à distinguer une preuve simulée d’une vraie preuve:

- On a Adv_1(A) > Adv_0(A) - Adv_ZK(A), où Adv_i(A) est l’avantage de A dans le jeu G_i

Concrètement, si le comportement de A diffère selon les jeux, alors A peut casser la propriété de zero-knowledge des preuves

Preuves par Jeux

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

2) (m_0,m_1) <- $A^O(pk)$

3) Pour un bit b choisi aléatoirement:
   C*_1 aléatoire dans G^2
   C*_2 <- Enc(pk_2,m_b)
   P’ généré à l’aide du simulateur ZK
   C* = (C*_1,C*_2,P’)

4) b’ <- A^O(pk, C*)

Jeu G_2

On change ensuite C*_1 pour le remplacer par de l’aléa

Preuves par Jeux

- Il faut alors prouver que G_1 et G_2 sont indistinguables

- On peut prouver ici que ces deux jeux sont indistinguables sous l'hypothèse DDH

- On a Adv_2(A) > Adv_1(A) - Adv_DDH(A),

Concrètement, si le comportement de A diffère selon les jeux, alors A peut résoudre DDH

- Au total:

  Adv_2(A) > Adv_1(A) - Adv_DDH(A) > Adv_0(A) - Adv_DDH(A) – Adv_ZK(A)

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

C*_1 aléatoire

Un chiffré El Gamal C_2
sous pk_2 = pk

C*_2= c*

Une preuve P que tout est
bien formé

P simulé

Difficulté 2: Utiliser le chiffré challenge (IND-CCA) pour distinguer le chiffré challenge c* IND-CPA

Solution : Utiliser c* pour C_2* et générer le reste comme dans le jeu G_2

Idée de la preuve

- Un chiffré Naor-Yung est constitué de

Un chiffré El Gamal C_1
sous pk_1

C*_1 aléatoire

Un chiffré El Gamal C_2
sous pk_2 = pk

C*_2= c*

Une preuve P que tout est
bien formé

P simulé

On peut alors montrer qu’un adversaire contre le jeu G_2 peut casser
l’IND-CPA d’El-Gamal avec même probabilité

Preuves par Jeux

- On avait Adv_2(A) > Adv_1(A) - Adv_DDH(A) > Adv_0(A) - Adv_DDH(A) – Adv_ZK(A)

- Si El Gamal IND-CPA alors Adv_2(A) négligeable:
  - Soit Adv_0(A) négligeable -> La construction est IND-CCA

  - Soit Adv_0(A) non négligeable:

    Adv_DDH(A) et/ou Adv_ZK(A) non négligeable également -> contradiction

La construction est IND-CCA si El Gamal est IND-CPA, DDH est difficile et P est généré à l'aide d'un système de preuves zero-knowledge