Variante de Chiffrement Basé sur l'Identité

Florent LLOMBARD et Lambert ROSIQUE

Présentation Orale

19 mars 2013

Un problème

On considère un problème $\mathcal{P}$, par exemple le problème du log discret.

Un problème

On considère un problème $\mathcal{P}$, par exemple le problème du log discret.

Une hypothèse

On construit une hypothèse à partir de ce problème. **Il est difficile de résoudre le problème $\mathcal{P}$.**

Un problème

On considère un problème $\mathcal{P}$, par exemple le problème du log discret.

Une hypothèse

On construit une hypothèse à partir de ce problème. **Il est difficile de résoudre le problème $\mathcal{P}$.**

Sécurité prouvée

Un schéma de chiffrement est prouvé sûr tant que l'hypothèse n'est pas contredite.
On réduit la sécurité du schéma au problème difficile.

One-time pad (masque jetable)

- $K \in G$ clé secrète, $M \in G$ message clair
  Chiffrement : $C = MK$
  Déchiffrement : $M = CK^{-1}$

One-time pad (masque jetable)

- $K \in G$ clé secrète, $M \in G$ message clair
  Chiffrement : $C = MK$
  Déchiffrement : $M = CK^{-1}$
- Avantage : inconditionnellement sûr, $H(M|C) = H(M)$
  Inconvénients : crypto symétrique, une clé par message

One-time pad (masque jetable)

- $K \in G$ clé secrète, $M \in G$ message clair
  Chiffrement : $C = MK$
  Déchiffrement : $M = CK^{-1}$
- Avantage : inconditionnellement sûr, $H(M|C) = H(M)$
  Inconvénients : crypto symétrique, une clé par message

Un masque calculable

$\mathcal{P}$ est le problème de calculer K. Si $\mathcal{P}$ est à trappe, alors on peut donner cette trappe au déchiffreur pour calculer K et déchiffrer.

Protocole de Diffie Hellman

Protocole

Alice envoie à Bob g^a
Bob envoie à Alice g^b
Secret partagé : g^{ab}
Un attaquant connaît g^a et g^b

Protocole de Diffie Hellman

Protocole

Alice envoie à Bob g^a
Bob envoie à Alice g^b
Secret partagé : g^{ab}
Un attaquant connaît g^a et g^b

Hypothèse DDH (Decisional Diffie Hellman)

Soit $G =< g >$ groupe cyclique. Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer (g^a, g^b, g^z) et (g^a, g^b, g^{ab}) avec un avantage non-négligeable, pour a, b, z aléatoires.

ElGamal

Trappe

(a, g^b) ou (g^a, b) sont deux *trappes* dans le problème de calculer g^{ab}.

ElGamal

Trappe

(a, g^b) ou (g^a, b) sont deux *trappes* dans le problème de calculer g^{ab}.

Chiffrement asymétrique d'ElGamal

s est une valeur secrète, et g^s publique.

- Chiffrement : Soit r aléatoire. (r, g^s) est une trappe pour construire $K = g^{sr}$. $C = (g^r, M \cdot K)$
- Déchiffrement : (s, g^r) est une trappe pour construire $K = g^{rs}$

Fonctions bilinéaires admissibles

Définition

Soient G_1, G_2 deux groupes d'ordre p premier.
On dit que $e : G_1 \times G_1 \rightarrow G_2$ est une fonction bilinéaire admissible si

- e est bilinéaire : $\forall g, h \in G_1, \forall a, b \in \mathbb{Z}_p, e(g^a, h^b) = e(g, h)^{ab}$
- e est non-dégénérée : $(\forall x \in G_1, e(g, x) = 1_{G_2}) \Rightarrow (g = 1_{G_2})$
- e est calculable en temps polynomial

Une nouvelle hypothèse

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \rightarrow G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

Une nouvelle hypothèse

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \rightarrow G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

Diffie-Hellman tripartite

3 utilisateurs qui s'échangent g^a, g^b, g^c, dont chacun connaît un des log dicret a, b, c. Secret partagé : $e(g,g)^{abc}$

Une nouvelle hypothèse

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \to G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

Diffie-Hellman tripartite

3 utilisateurs qui s'échangent g^a, g^b, g^c, dont chacun connaît un des log dicret a, b, c. Secret partagé : $e(g,g)^{abc}$

Trappes évidentes

- (a, g^b, g^c) est une trappe : $e(g^b, g^c)^a = e(g,g)^{abc}$.
- (g^{ab}, g^c) est une trappe : $e(g^{ab}, g^c) = e(g,g)^{abc}$.

Chiffrement basé sur l'identité

Crypto asymétrique : système généré par le déchiffreur (clé publique et clé privée)

Chiffrement basé sur l'identité

Crypto asymétrique : système généré par le déchiffreur (clé publique et clé privée)
Chiffrement basé sur l'identité : système généré par une entité indépendante, qui calcule les clés privées

Une nouvelle entité, le PKG (Private Key Generator), génère le système, et les clés privées.

- **setup** : le PKG génère les paramètres globaux publics du système, et une master-key gardée secrète
- **extract** : à partir de la master-key et d'une clé publique d'un utilisateur, le PKG calcule la clé secrète de l'utilisateur
- **encrypt, decrypt**

Le système de Boneh et Franklin

$H : \{0;1\}^* \rightarrow G_1$

Le système de Boneh et Franklin

$H : \{0;1\}^* \rightarrow G_1$
master-key : s
paramètres publics : e, G_1, G_2, H, g^s

Le système de Boneh et Franklin

$H : \{0;1\}^* \rightarrow G_1$
master-key : s
paramètres publics : e, G_1, G_2, H, g^s
utilisateur $u \in \{0;1\}^*$
clé publique : $I_u = H(u) = g^{t_u}$
clé privée : $d_u = {I_u}^s = g^{st}$

Le système de Boneh et Franklin

$H : \{0;1\}^* \rightarrow G_1$
master-key : s
paramètres publics : e, G_1, G_2, H, g^s
utilisateur $u \in \{0;1\}^*$
clé publique : $I_u = H(u) = g^{t_u}$
clé privée : $d_u = {I_u}^s = g^{st}$

- Chiffrement : on utilise la trappe (r, g^s, g^t) pour construire $K = e(g^s, I_u)^r = e(g,g)^{rst}$: $C = (g^r, MK)$

Le système de Boneh et Franklin

$H : \{0;1\}^* \rightarrow G_1$
master-key : s
paramètres publics : e, G_1, G_2, H, g^s
utilisateur $u \in \{0;1\}^*$
clé publique : $I_u = H(u) = g^{t_u}$
clé privée : $d_u = {I_u}^s = g^{st}$

- Chiffrement : on utilise la trappe (r, g^s, g^t) pour construire $K = e(g^s, I_u)^r = e(g,g)^{rst}$: $C = (g^r, MK)$
- Déchiffrement : on utilise la trappe (g^r, g^{st_u}) pour calculer $K = e(g^r, d_u) = e(g,g)^{rst}$

Lagrange

q polynome de degré $k-1$ entièrement défini par k valeurs qu'il prend. Si on note $\lambda_i(x) = \prod_{0 \leq j \leq k-1, j \neq i} \frac{x_j - x}{x_j - x_i}$

alors $q(x) = \sum_{i=0}^{k-1} q(x_i)\lambda_i(x)$

Lagrange

q polynome de degré $k-1$ entièrement défini par k valeurs qu'il prend. Si on note $\lambda_i(x) = \prod_{0 \leq j \leq k-1, j \neq i} \frac{x_j - x}{x_j - x_i}$

alors $q(x) = \sum_{i=0}^{k-1} q(x_i)\lambda_i(x)$

Partage de secret de Shamir

y secret à partager divisé en n parts tel que
k suffisent pour retrouver y (mais pas moins) :

Lagrange

q polynome de degré $k-1$ entièrement défini par k valeurs qu'il prend. Si on note $\lambda_i(x) = \prod_{0 \leq j \leq k-1, j \neq i} \frac{x_j - x}{x_j - x_i}$

alors $q(x) = \sum_{i=0}^{k-1} q(x_i)\lambda_i(x)$

Partage de secret de Shamir

y secret à partager divisé en n parts tel que
k suffisent pour retrouver y (mais pas moins) :
On prend q polynôme aléatoire de degré $k-1$ tel que $q(0) = y$

Lagrange

q polynome de degré $k-1$ entièrement défini par k valeurs qu'il prend. Si on note $\lambda_i(x) = \prod_{0 \leq j \leq k-1, j \neq i} \frac{x_j - x}{x_j - x_i}$

alors $q(x) = \sum_{i=0}^{k-1} q(x_i)\lambda_i(x)$

Partage de secret de Shamir

y secret à partager divisé en n parts tel que
k suffisent pour retrouver y (mais pas moins) :
On prend q polynôme aléatoire de degré $k-1$ tel que $q(0) = y$

$\Longrightarrow y = q(0) = \sum_{i=0}^{k-1} y_i \lambda_i(0)$ avec $y_i = q(x_i)$ (Lagrange)

Lagrange

q polynome de degré $k-1$ entièrement défini par k valeurs qu'il prend. Si on note $\lambda_i(x) = \prod_{0 \leq j \leq k-1, j \neq i} \frac{x_j - x}{x_j - x_i}$

alors $q(x) = \sum_{i=0}^{k-1} q(x_i)\lambda_i(x)$

Partage de secret de Shamir

y secret à partager divisé en n parts tel que
k suffisent pour retrouver y (mais pas moins) :
On prend q polynôme aléatoire de degré $k-1$ tel que $q(0) = y$
$\Longrightarrow y = q(0) = \sum_{i=0}^{k-1} y_i \lambda_i(0)$ avec $y_i = q(x_i)$ (Lagrange)
$\Longrightarrow$ Pour trouver y, il faut donc les x_i et les y_i.

Hypothèse DDH

Soit $G =< g >$ groupe cyclique. Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer (g^a, g^b, g^z) et (g^a, g^b, g^{ab}) avec un avantage non-négligeable, pour a, b, z aléatoires.

Hypothèse DDH

Soit $G =< g >$ groupe cyclique. Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer (g^a, g^b, g^z) et (g^a, g^b, g^{ab}) avec un avantage non-négligeable, pour a, b, z aléatoires.

Nouvelles trappes

Soit q polynôme aléatoire de degré k tel que $q(0) = a$.
a partagé en n parts (x_i, y_i), avec $y_i = q(x_i)$.

- $\left((x_i, y_i)_{0 \leq i \leq k}, g^b\right)$ est une trappe pour calculer

$$g^{ab} = g^{b \sum_{i=0}^{k-1} y_i \lambda_i} = \prod_{i=0}^{k-1} g^{b y_i \lambda_i}$$

Hypothèse DDH

Soit $G =< g >$ groupe cyclique. Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer (g^a, g^b, g^z) et (g^a, g^b, g^{ab}) avec un avantage non-négligeable, pour a, b, z aléatoires.

Nouvelles trappes

Soit q polynôme aléatoire de degré k tel que $q(0) = a$.
a partagé en n parts (x_i, y_i), avec $y_i = q(x_i)$.

- $\left((x_i, y_i)_{0 \leq i \leq k}, g^b\right)$ est une trappe pour calculer

$$g^{ab} = g^{b \sum_{i=0}^{k-1} y_i \lambda_i} = \prod_{i=0}^{k-1} g^{b y_i \lambda_i}$$

- $\left(\left(x_i, g^{b y_i}\right)_{0 \leq i \leq k}\right)$ est aussi une trappe.

Trappes aussi pour BDH

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \rightarrow G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

Trappes aussi pour BDH

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \to G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

- $\left((x_i, y_i)_{0 \leq i \leq k}, g^b, g^c\right)$ est une trappe pour calculer

$$e(g,g)^{abc} = e(g^b, g^c)^{\sum_{i=0}^{k-1} y_i \lambda_i} = \prod_{i=0}^{k-1} e(g,g)^{bcy_i\lambda_i}$$

Trappes aussi pour BDH

L'hypothèse BDH (decisional Bilinear Diffie Hellman)

Soit $e : G_1 \times G_1 \to G_2$ une fonction bilinéaire admissible.
Il n'existe pas d'algorithme polynomial probabiliste capable de distinguer $(g^a, g^b, g^c, e(g,g)^z)$ et $(g^a, g^b, g^c, e(g,g)^{abc})$ avec un avantage non-négligeable, pour a, b, c, z aléatoires.

- $\left((x_i, y_i)_{0 \leq i \leq k}, g^b, g^c\right)$ est une trappe pour calculer

$$e(g,g)^{abc} = e(g^b, g^c)^{\sum_{i=0}^{k-1} y_i \lambda_i} = \prod_{i=0}^{k-1} e(g,g)^{bcy_i\lambda_i}$$

- $\left(\left(x_i, g^{by_i}\right), g^c{}_{0 \leq i \leq k}\right)$ est aussi une trappe.

Hypothèse BDH Modifiée (MBDH)

Hypothèse MBDH : Aucun adversaire polynomial en temps ne peut distinguer le tuple $(g^a, g^b, g^c, e(g,g)^{\frac{ab}{c}})$ du tuple $(g^a, g^b, g^c, e(g,g)^z)$ avec plus qu'un avantage négligeable.

Hypothèse BDH Modifiée (MBDH)

Hypothèse MBDH : Aucun adversaire polynomial en temps ne peut distinguer le tuple $(g^a, g^b, g^c, e(g,g)^{\frac{ab}{c}})$ du tuple $(g^a, g^b, g^c, e(g,g)^z)$ avec plus qu'un avantage négligeable.

Trappes

- $(g^{ab}, g^{\frac{1}{c}})$ est une trappe : $e(g^{ab}, g^{\frac{1}{c}}) = e(g,g)^{\frac{ab}{c}}$.
- $(g^a, g^{\frac{b}{c}})$ est une trappe : $e(g^a, g^{\frac{b}{c}}) = e(g,g)^{\frac{ab}{c}}$.

Algorithme

Setup

- *Univers* $\mathcal{U}$

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, \ldots t_{|\mathcal{U}|}$ *au hasard*

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g, g)^y$

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g,g)^y$

Clé Maître : $\{t_1, ..., t_{|\mathcal{U}|}\}, y$

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g,g)^y$

Clé Maître : $\{t_1, ..., t_{|\mathcal{U}|}\}, y$

Key Generation

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g,g)^y$

Clé Maître : $\{t_1, ..., t_{|\mathcal{U}|}\}, y$

Key Generation

- q *polynôme de degré* $k-1$ *tel que* $q(0) = y$

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g,g)^y$

Clé Maître : $\{t_1, ..., t_{|\mathcal{U}|}\}, y$

Key Generation

- q *polynôme de degré* $k-1$ *tel que* $q(0) = y$
- *Composantes :* $D_i = g^{\frac{q(i)}{t_i}}$ *pour* $i \in \omega$

Algorithme

Setup

- *Univers* $\mathcal{U}$
- $t_1, ...t_{|\mathcal{U}|}$ *au hasard*
- *On définit* $\forall i, T_i = g^{t_i}$ *et* $Y = e(g,g)^y$

Clé Maître : $\{t_1, ..., t_{|\mathcal{U}|}\}, y$

Key Generation

- q *polynôme de degré* $k-1$ *tel que* $q(0) = y$
- *Composantes :* $D_i = g^{\frac{q(i)}{t_i}}$ *pour* $i \in \omega$

Clé Privée : $\{D_i, i \in \omega\}$

Encryption

Clé publique ω', message M

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré :* $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré :* $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$

Decryption

Si $|\omega' \cap \omega| \geq k$

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré : $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$*

Decryption

Si $|\omega' \cap \omega| \geq k$

- *S sous-ensemble de taille k au hasard*

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré : $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$*

Decryption

Si $|\omega' \cap \omega| \geq k$

- *S sous-ensemble de taille k au hasard*
- *Calculer $Y^r = \prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}}$*

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré : $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$*

Decryption

Si $|\omega' \cap \omega| \geq k$

- *S sous-ensemble de taille k au hasard*
- *Calculer $Y^r = \prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}}$*
- *En déduire $M = C'/Y^r$*

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré : $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$*

Decryption

Si $|\omega' \cap \omega| \geq k$

- *S sous-ensemble de taille k au hasard*
- *Calculer $Y^r = \prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}}$*
- *En déduire $M = C'/Y^r$*

En effet, $\prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}} = \prod_{i \in S} e(g^{\frac{q(i)}{t_i}}, g^{rt_i})^{\lambda_{i,s}}$

Encryption

Clé publique ω', message M

- *$r \in \mathbb{Z}_p$ au hasard*
- *Chiffré : $C = (\omega', C' = MY^r, \{C_i = T_i^r\}_{i \in \omega'})$*

Decryption

Si $|\omega' \cap \omega| \geq k$

- *S sous-ensemble de taille k au hasard*
- *Calculer $Y^r = \prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}}$*
- *En déduire $M = C'/Y^r$*

En effet, $$\begin{aligned}\prod_{i \in S} e(D_i, C_i)^{\lambda_{i,s}} &= \prod_{i \in S} e(g^{\frac{q(i)}{t_i}}, g^{rt_i})^{\lambda_{i,s}} \\ &= e(g,g)^{r \sum_{i \in S} q(i)\lambda_{i,s}}\end{aligned}$$

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics
- Il peut demander des clés privées d'identités γ_j éloignées de ω

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics
- Il peut demander des clés privées d'identités γ_j éloignées de ω
- Il envoie M_0, M_1, l'un au hasard est chiffré par le challenger

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics
- Il peut demander des clés privées d'identités γ_j éloignées de ω
- Il envoie M_0, M_1, l'un au hasard est chiffré par le challenger
- Il peut redemander des clés privées

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics
- Il peut demander des clés privées d'identités γ_j éloignées de ω
- Il envoie M_0, M_1, l'un au hasard est chiffré par le challenger
- Il peut redemander des clés privées
- Il doit deviner de quel message provient le chiffré

Selective-ID Model

Fuzzy Selective-ID Model (FSID)

Sécurité : Tout adversaire $\mathcal{A}$ polynomial en temps a, au mieux, un avantage négligeable dans le *modèle* suivant :

- Identité ω choisie par l'adversaire
- Il reçoit les paramètres publics
- Il peut demander des clés privées d'identités γ_j éloignées de ω
- Il envoie M_0, M_1, l'un au hasard est chiffré par le challenger
- Il peut redemander des clés privées
- Il doit deviner de quel message provient le chiffré

$\mathrm{Adv}(\mathcal{A}) = \Pr[\text{"}\mathcal{A} \text{ devine"}] - \frac{1}{2}$

Preuve de sécurité

IDEE : **Réduction** entre FSID Model et Decisional MBDH

Preuve de sécurité

IDEE : **Réduction** entre FSID Model et Decisional MBDH

- On se donne un adversaire avec un *avantage* α pour FSID Model

Preuve de sécurité

IDEE : **Réduction** entre FSID Model et Decisional MBDH

- On se donne un adversaire avec un *avantage* α pour FSID Model $\Longleftrightarrow$ Il sait différentier deux messages avec le chiffré (dans ce modèle)

Preuve de sécurité

IDEE : **Réduction** entre FSID Model et Decisional MBDH

- On se donne un adversaire avec un *avantage* α pour FSID Model $\Longleftrightarrow$ Il sait différentier deux messages avec le chiffré (dans ce modèle)

- Si le chiffré est *aléatoire* (sans rapport avec les messages), il répond au hasard

Preuve de sécurité

IDEE : **Réduction** entre FSID Model et Decisional MBDH

- On se donne un adversaire avec un *avantage* α pour FSID Model $\Longleftrightarrow$ Il sait différentier deux messages avec le chiffré (dans ce modèle)

- Si le chiffré est *aléatoire* (sans rapport avec les messages), il répond au hasard

- On construit un simulateur qui a un avantage $\frac{\alpha}{2}$ pour MBDH en *exploitant la réponse* de l'adversaire

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*
- *Si* $\mathcal{A}$ *demande* γ *on doit construire les* D_i *:*
  - $\Gamma = \gamma \cap \omega$, Γ' *entre* Γ *et* γ *de taille* $k-1$, $S = \Gamma' \cup \{0\}$

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*
- *Si* $\mathcal{A}$ *demande* γ *on doit construire les* D_i *:*
  - $\Gamma = \gamma \cap \omega$, Γ' *entre* Γ *et* γ *de taille* $k-1$, $S = \Gamma' \cup \{0\}$
  - *Si* $i \in \Gamma$, $D_i = g^{r_i}$ *(r_i au hasard) ie* $q(i) = c\beta_i s_i$

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*
- *Si* $\mathcal{A}$ *demande* γ *on doit construire les* D_i *:*
  - $\Gamma = \gamma \cap \omega$, Γ' *entre* Γ *et* γ *de taille* $k-1$, $S = \Gamma' \cup \{0\}$
  - *Si* $i \in \Gamma$, $D_i = g^{r_i}$ *(r_i au hasard) ie* $q(i) = c\beta_i s_i$
  - *Si* $i \in \Gamma' - \Gamma$, $D_i = g^{\frac{\lambda_i}{\omega_i}}$ *(λ_i au hasard) ie* $q(i) = \lambda_i$

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 1$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^z)$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*
- *Si* $\mathcal{A}$ *demande* γ *on doit construire les* D_i *:*
  - $\Gamma = \gamma \cap \omega$, Γ' *entre* Γ *et* γ *de taille* $k-1$, $S = \Gamma' \cup \{0\}$
  - *Si* $i \in \Gamma$, $D_i = g^{r_i}$ *(r_i au hasard) ie* $q(i) = c\beta_i s_i$
  - *Si* $i \in \Gamma' - \Gamma$, $D_i = g^{\frac{\lambda_i}{\omega_i}}$ *(λ_i au hasard) ie* $q(i) = \lambda_i$
  - *Sinon, on a assez de valeurs de q pour extrapoler*

  $\Longrightarrow$ D_i *est définie pour tout* $i \Rightarrow$ *clé privée de* γ

Preuve

- *(Challenger)* $\epsilon_0 \overset{\$}{\in} \{0,1\}$
  Si $\epsilon_0 = 0$ *on pose :*
  $(A, B, C, Z) = (g^a, g^b, g^c, e(g,g)^{\frac{ab}{c}})$
- *(Simulateur)* $Y = e(g,g)^a$ *et*
  $\forall i \in \omega \ T_i = g^{c\beta_i}$ *(β_i au hasard)*
  et sinon $T_i = g^{\omega_i}$ *(ω_i au hasard)*
- *Si* $\mathcal{A}$ *demande* γ *on doit construire les* D_i *:*
  - $\Gamma = \gamma \cap \omega$, Γ' *entre* Γ *et* γ *de taille* $k-1$, $S = \Gamma' \cup \{0\}$
  - *Si* $i \in \Gamma$, $D_i = g^{r_i}$ *(r_i au hasard) ie* $q(i) = c\beta_i s_i$
  - *Si* $i \in \Gamma' - \Gamma$, $D_i = g^{\frac{\lambda_i}{\omega_i}}$ *(λ_i au hasard) ie* $q(i) = \lambda_i$
  - *Sinon, on a assez de valeurs de q pour extrapoler*

  $\Longrightarrow$ D_i *est définit pour tout* $i \Rightarrow$ *clé privée de* γ

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$
  Si $\epsilon_0 = 1$, $Z = g^z \Longrightarrow Z$ *est complètement aléatoire*

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$
  Si $\epsilon_0 = 1$, $Z = g^z \Longrightarrow Z$ *est complètement aléatoire*
  Si $\epsilon_0 = 0$, $Z = e(g,g)^{\frac{ab}{c}} \Longrightarrow$ *si* $r = \frac{b}{c}$, $C' = M_{\epsilon_1} Y^r$ *et*
  $C_i = T_i^r$ *donc* $\mathcal{A}$ *a un avantage*

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$
  Si $\epsilon_0 = 1$, $Z = g^z \Longrightarrow Z$ *est complètement aléatoire*
  Si $\epsilon_0 = 0$, $Z = e(g,g)^{\frac{ab}{c}} \Longrightarrow$ *si* $r = \frac{b}{c}$, $C' = M_{\epsilon_1} Y^r$ *et* $C_i = T_i^r$ *donc* $\mathcal{A}$ *a un avantage*
- $\mathcal{A}$ *peut redemander des clés d'identités*

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$
  Si $\epsilon_0 = 1$, $Z = g^z \Longrightarrow Z$ *est complètement aléatoire*
  Si $\epsilon_0 = 0$, $Z = e(g,g)^{\frac{ab}{c}} \Longrightarrow$ *si* $r = \frac{b}{c}$, $C' = M_{\epsilon_1} Y^r$ *et*
  $C_i = T_i^r$ *donc* $\mathcal{A}$ *a un avantage*
- $\mathcal{A}$ *peut redemander des clés d'identités*
- $\mathcal{A}$ *devine* ϵ_1'. *Le simulateur renvoie* $\epsilon_0' = 0$ *si* $\mathcal{A}$ *a raison et* 1 *sinon*
  En effet, si $\epsilon_0 = 1$ $\mathcal{A}$ *a répondu au hasard complet (1 chance sur 2) donc le simulateur a aussi une chance sur 2.*
  Sinon, $\mathcal{A}$ *avait un avantage donc il fera plus souvent juste, donc le simulateur aussi.*

Preuve

- $\mathcal{A}$ *envoie* M_0, M_1 *au simulateur*
- $\epsilon_1 \overset{\$}{\in} \{0,1\} \longrightarrow M_{\epsilon_1}$ *chiffré :*
  $C = (\omega, C' = M_{\epsilon_1} Z, \{C_i = B^{\beta_i}\}_{i \in \omega})$
  Si $\epsilon_0 = 1$, $Z = g^z \Longrightarrow Z$ *est complètement aléatoire*
  Si $\epsilon_0 = 0$, $Z = e(g,g)^{\frac{ab}{c}} \Longrightarrow$ *si* $r = \frac{b}{c}$, $C' = M_{\epsilon_1} Y^r$ *et*
  $C_i = T_i^r$ *donc* $\mathcal{A}$ *a un avantage*
- $\mathcal{A}$ *peut redemander des clés d'identités*
- $\mathcal{A}$ *devine* ϵ_1'. *Le simulateur renvoie* $\epsilon_0' = 0$ *si* $\mathcal{A}$ *a raison et* 1 *sinon*
  En effet, si $\epsilon_0 = 1$ $\mathcal{A}$ *a répondu au hasard complet (1 chance sur 2) donc le simulateur a aussi une chance sur 2.*
  Sinon, $\mathcal{A}$ *avait un avantage donc il fera plus souvent juste, donc le simulateur aussi. Son avantage est bien* $\frac{\alpha}{2}$

Conclusion

Merci de votre attention