Cours 9. Signatures

HAI709I – Cryptographie

Bruno Grenet

Université de Montpellier – Faculté des Sciences

Introduction

Objectif : authenticité d'un message, dans le cadre de la cryptographie à clef publique

- L'expéditeur *signe* un message m avec une clef privée $\rightarrow \sigma$
- N'importe qui, avec la clef publique de l'expéditeur, peut *vérifier* la signature σ

Comparaison avec les MAC

- MAC : il faut la clef k pour vérifier $\rightarrow$ pas *publiquement* vérifiable
- Une signature est *transférable* :
  - si B reçoit (m, σ), il peut transférer à C pour la convaincre de l'authenticité de m
  - système d'infrastructure à clefs publiques $\rightarrow$ *chaîne de confiance*
- *Non-répudiation* : le signataire ne peut prétendre n'avoir jamais signé

Exemple : pass sanitaire

- Vaccination $\rightarrow$ signature (QR code) avec la clef privée du gouvernement
- Vérification $\rightarrow$ tout le monde peut vérifier, avec la clef publique du gouvernement

Protocole de signature numérique

Définition

Un **protocole de signature** est un triplet d'algorithmes polynomiaux (Gen, Sign, Vrfy) où

- Gen prend en entrée 1^n et renvoie un couple de clefs (pk, sk)
- Sign prend en entrée sk et un message m et renvoie une *signature* σ
- Vrfy prend en entrée pk, m et σ et renvoie 1 si l'~~étiquette~~ signature est *valide*, et 0 sinon.

Un protocole de signature est à **longueur fixée $\ell(n)$** s'il n'accepte que des messages de longueur $\ell(n)$.

Correction

(Gen, Sign, Vrfy) est *correct* si pour tout $(pk, sk) \leftarrow \mathrm{Gen}(1^n)$ et $\sigma \leftarrow \mathrm{Sign}_{sk}(m)$,

$$\Pr\left[\mathrm{Vrfy}_{pk}(m, \sigma) = 0\right] \leq \mathrm{negl}(n)$$

Remarque

- Définition *quasi* identique aux MAC : Mac $\rightarrow$ Sign et « étiquette » $\rightarrow$ « signature »

Sécurité d'un protocole de signature

Expérience de signature

Entrée : paramètre de sécurité 1^n

1. Protocole : $(pk, sk) \leftarrow \mathrm{Gen}(1^n)$
2. Attaquant : peut demander la signature des messages $m_1, \ldots, m_t$ de son choix
3. Attaquant : produit un couple (m, σ), où $m \notin \{m_1, \ldots, m_t\}$

Succès de l'attaquant si $\mathrm{Vrfy}_{pk}(m, \sigma) = 1$

Définition

$(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ est existentiellement infalsifiable par une attaque adaptative à clairs choisis (ou sûr) si pour tout APP, $\Pr\left[\mathrm{Vrfy}_{pk}(m, \sigma) = 1\right] \leq \mathrm{negl}(n)$.

Remarque

- *Quasiment* copié-collé de la définition pour les MAC !

Hash-and-sign

Principes

- Motivation : les signatures sont moins efficaces que les MAC
- Idée : signer le haché du message, plutôt que le message lui-même

Construction

- Ingrédients :
  - $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$: protocole de signature pour les messages de longueur $\ell(n)$
  - (Gen_H, H) : fonction de hachage avec sortie de taille $\ell(n)$
- $\mathsf{Gen}'(1^n)$:
  - $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$
  - $s \leftarrow \mathsf{Gen}_H(1^n)$
  - Clef publique (pk, s) et privée (sk, s)
- $\mathsf{Sign}'_{sk,s}(m) \rightarrow \mathsf{Sign}_{sk}(H^s(m))$
- $\mathsf{Vrfy}'_{pk,s}(m, \sigma) \rightarrow \mathsf{Vrfy}_{pk}(H^s(m), \sigma)$

Sécurité de *Hash-and-sign*

Théorème

Si $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ est sûr et (Gen_H, H) est résistante aux collisions, alors $(\mathrm{Gen}', \mathrm{Sign}', \mathrm{Vrfy}')$ est sûr pour les messages de longueur quelconques

A_0 : attaquant contre $(\mathrm{Gen}', \mathrm{Sign}', \mathrm{Vrfy}')$

↳ Demande des signatures $(m_1, \tau_1), \dots, (m_t, \tau_t)$ puis produit (m, τ)

Cas 1 $\exists\, i$ tq $H^s(m) = H^s(m_i)$ → A_0 a trouvé une collision

Cas 2 $\forall\, i$ $H^s(m) \neq H^s(m_i)$ ⇝ $h = H^s(m)$ et $h_i = H^s(m_i)$ pour tout i.

→ A_0 a produit (h, τ) valide en connaissant les (h_i, τ_i).

$$\mathrm{Proba}[\text{succès } A_0] \leq \Pr[\text{succès cas 1}] + \Pr[\text{succès cas 2}]$$
$$\leq \mathrm{negl}(n) + \mathrm{negl}(n) \leq \mathrm{negl}(n)$$

Certificats et infrastructure à clefs publiques (PKI)

Comment être sûr du propriétaire d'une clef publique ?

Certificat

- $\text{cert}_{B \to C} = \text{Sign}_{sk_B}(\text{« La clef publique de } C \text{ est } pk_C \text{ »})$
- C envoie $(pk_C, \text{cert}_{B \to C})$; A vérifie $\text{cert}_{B \to C}$ avec pk_B
  - si A a *confiance* en $B \to A$ accepte que pk_C est la clef publique de C
  - mais A doit avoir confiance en B

Infrastructure à clefs publique

- Réseau de confiance : si A a confiance en B et B en C, A peut avoir confiance en C
- Certificats d'autorité : une ou plusieurs autorités, en qui tout le monde a confiance

Invalidation de certificat

- Date d'expiration → « La clef publique de C est pk_C ; date de fin de validité : … »
- Révocation : un *numéro de série* par certificat → *certificat de révocation*

Signcryption

Comment combiner signature et chiffrement à clef publique ?

« Chiffrer-puis-authentifier » (solution pour les MAC)

- Expéditeur envoie le chiffré c de m et la signature σ de c
  - c est chiffré avec une clef *publique* pk_D du destinataire, qui déchiffre avec sk_D
  - σ est calculée avec la clef *privée* sk_E de l'expéditeur, et vérifiée avec pk_E
- Attaque :
  - Voyant c, l'attaquant calcule σ_A avec sa clef privée
  - L'attaquant peut faire croire qu'il est l'expéditeur

Solution

- Au choix, « chiffrer-puis-authentifier » ou « authentifier-puis-chiffrer »
- Signer le message ou chiffré **et son identité**
  - $c \leftarrow \mathrm{Enc}_{pk_D}(m)$; $\sigma \leftarrow \mathrm{Sign}_{sk_E}(c, \text{« Je suis } E \text{ »})$
  - $\sigma \leftarrow \mathrm{Sign}_{sk_E}(m)$; $c \leftarrow \mathrm{Enc}_{pk_D}(m, \sigma, \text{« Je suis } E \text{ »})$
- Sûr si chiffrement et signatures sont (suffisamment) sûrs

1. Définitions et sécurité

2. Signature RSA

3. Identification et signature de Schnorr

Version simple

Construction

- $\mathrm{Gen}(1^n)$: *identique au chiffrement RSA*
  - $p, q \leftarrow$ nombres premiers de n bits
  - $N \leftarrow p \times q$ et $\varphi(N) = (p-1) \times (q-1)$
  - $e \leftarrow$ entier aléatoire premier avec $\varphi(N)$ et $d = e^{-1} \bmod \varphi(N)$
  - Renvoyer $pk = (N, e)$ et $sk = (N, d)$
- $\mathrm{Sign}_{sk}(m) : m^d \bmod N$ $\qquad m \in (\mathbb{Z}/N\mathbb{Z})^\times$
- $\mathrm{Vrfy}_{pk}(m, \sigma) : m \stackrel{?}{=} \sigma^e \bmod N$

Correction

- $\mathrm{Vrfy}_{pk}(m, \mathrm{Sign}_{sk}(m)) : m \stackrel{?}{=} (m^d)^e \bmod N$

Attaques

1. Attaquant choisit σ puis calcule $m = \sigma^e \bmod N$
2. Attaquant récupère (m_1, σ_1) et (m_2, σ_2) et calcule $m = m_1 \cdot m_2$ et $\sigma = \sigma_1 \cdot \sigma_2$

RSA-FDH

Construction

- $\mathsf{Gen}(1^n)$: *presque comme RSA*
  - $pk = (N, e)$ et $sk = (N, d)$
  - Choix d'une fonction de hachage $H : \{0,1\}^* \to (\mathbb{Z}/N\mathbb{Z})^\times$
- $\mathsf{Sign}_{sk}(m) : H(m)^d \bmod N$ $\qquad m \in \{0,1\}^*$
- $\mathsf{Vrfy}_{pk}(m, \sigma) : H(m) \stackrel{?}{=} \sigma^e \bmod N$

Quelles spécifications pour H pour éviter les attaques ?

- Attaque 1 : $\sigma \to h = \sigma^e \to H(m) = h$ — *difficile à inverser*
- Attaque 2 : $m_1, m_2 \to H(m) = H(m_1) \cdot H(m_2) \bmod N$ — *« non-multiplicative »*
- Autre 1 : si $H(m_1) = H(m_2)$, $\sigma_1 = \sigma_2$ — *résistante aux collisions*
- Autre 2 : il faut que l'image de H soit tout $(\mathbb{Z}/N\mathbb{Z})^\times$ — *domaine complet*

Mauvaise et bonne nouvelles

- On ne *sait pas* construire H satisfaisante — *pas de preuve de sécurité*
- Si $H : \{0,1\}^* \to (\mathbb{Z}/N\mathbb{Z}))^\times$ est aléatoire, on sait prouver la sécurité

Preuve de RSA-FDH

Théorème

Si l'hypothèse RSA est vérifiée, et si H est aléatoire, alors RSA-FDH est sûr

- $\mathcal{A}$ attaquant contre RSA-FDH $\Rightarrow$ $\mathcal{A}'$ contre l'hyp. RSA.
- $\mathcal{A}'$ connaît (N, e) et $c \in \mathbb{Z}/N\mathbb{Z}^\times$ uniforme, et doit trouver m tq $m^e = c \bmod N$

$\rightarrow$ $\mathcal{A}$ fait les requêtes $m_1, \dots, m_t$ à H et renvoie (m, σ) avec

$m \in \{m_1, \dots, m_t\}$

Algo de $\mathcal{A}'$

1. Choisir j unif. dans $\{1, \dots, t\}$ (parce que $m = m_j$)
2. Si $\mathcal{A}$ demande $H(m_i)$, $i \neq j$: $\mathcal{A}'$ tire σ_i, calcule $h_i = \sigma_i^e$ et renvoie h_i
3. Si $\mathcal{A}$ demande $H(m_j)$, $\mathcal{A}'$ répond c.

Preuve de RSA-FDH

Théorème

Si l'hypothèse RSA est vérifiée, et si H est aléatoire, alors RSA-FDH est sûr

Analyse

- Cas 1 : le pari échoue $\rightsquigarrow$ A demande la signature de m_j.
  $\rightarrow$ l'attaque de A' échoue
- Cas 2 : pari réussi $\rightsquigarrow$ A réussit l'attaque et renvoie (m_j, σ)
  tq $H(m_j) = \sigma^e = c$ $\rightsquigarrow$ A' renvoie σ

$$Pr[\text{succès } A'] = \frac{1}{t} Pr[\text{succès } A] > \frac{negl(n)}{poly(n)} > negl(n)$$

(t : $poly(n)$)

1. Définitions et sécurité

2. Signature RSA

3. Identification et signature de Schnorr

Principe général

Protocole d'identification : prouver son identité à un interlocuteur

- Contexte :
  - Un *prouveur* possède sa clef secrète *sk*
  - Un *vérificateur* connaît la clef publique du prouveur *pk*
- Objectifs :
  - Le prouveur veut convaincre le vérificateur qu'il connaît la clef secrète *sk* associée à *pk*
  - Le prouveur ne veut *rien* dévoiler sur *sk* au vérificateur

Construction de Fiat-Shamir

- Étant donné un protocole d'identification, on peut déduire un protocole de signature

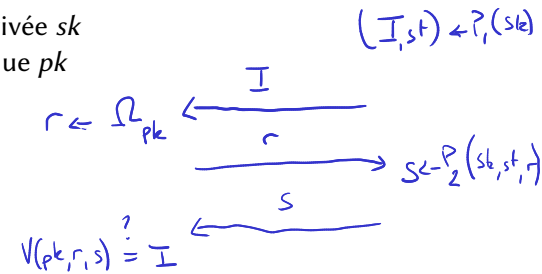
Protocoles de Schnorr

- Protocole d'identification
- Protocole de signature *via* la construction de Fiat-Shamir
- Exemple : DSA & ECDSA sont des variantes du protocole de Schnorr

Protocole d'identification

Description

- Prouveur : deux algorithmes P_1, P_2, la clef privée sk
- Vérificateur : un algorithme V, la clef publique pk
- Protocole :
  1. Prouveur : $(I, \text{st}) \leftarrow P_1(sk)$ et envoie I
  2. Vérificateur: $r \leftarrow_R \Omega_{pk}$ et envoie r
  3. Prouveur : $s \leftarrow P_2(sk, \text{st}, r)$ et envoie s
  4. Vérificateur : teste si $V(pk, r, s) = I$



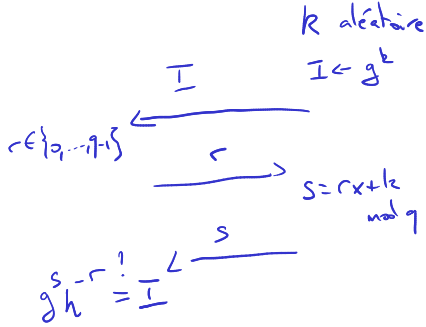
Sécurité

- Expérience d'identification :
  - Attaquant voit passer des échanges associés à sk *plusieurs applications du protocole*
  - Attaquant essaie de se faire passer pour le prouveur
- Protocole sûr : un APP a une probabilité $\leq \text{negl}(n)$ de convaincre un vérificateur

Protocole d'identification de Schnorr

Définition du protocole

- Public : groupe G d'ordre q, générateur g
- Clefs : x (privée) et $h = g^x$ (publique)
- Protocole :
  1. Prouveur : $k \leftarrow_R \{0, \ldots, q-1\}$; $I \leftarrow g^k$; Envoie I
  2. Vérificateur : $r \leftarrow_R \{0, \ldots, q-1\}$; Envoie r
  3. Prouveur : $s \leftarrow (r \cdot x + k) \bmod q$; Envoie s
  4. Vérificateur : teste si $g^s \cdot h^{-r} = I$



Sécurité

$g^s h^{-r} = g^{rx+k} g^{-rx} = g^k$

Si le logarithme discret est difficile dans G, le protocole de Schnorr est sûr

- A_0 attaquant contre Schnorr $\rightsquigarrow$ A' qui étant donné h sait calculer x tq $g^x = h$
- Deux attaques avec le même k et $r_1 \neq r_2$ $\rightarrow$ A_0 renvoie s_1 et s_2

$\rightarrow$ A' renvoie $(s_1 - s_2)(r_1 - r_2)^{-1} \bmod q$ car $g^{s_1} h^{-r_1} = g^{s_2} h^{-r_2} \Rightarrow g^{s_1 - s_2} = h^{r_1 - r_2}$

$\Rightarrow g^{(s_1 - s_2)(r_1 - r_2)^{-1}} = h$

Construction de Fiat-Shamir

Construction

- Ingrédients : $(\mathrm{Gen}_{id}, P_1, P_2, V) \to$ protocole d'identification
- $\mathrm{Gen}(1^n)$:
  - $(pk, sk) \leftarrow \mathrm{Gen}_{id}(1^n)$
  - Choix d'une fonction de hachage $H : \{0,1\}^* \to \Omega_{pk}$
- $\mathrm{Sign}_{sk}(m)$: *simulation du protocole d'identification*
  1. $(I, \mathrm{st}) \leftarrow P_1(sk)$
  2. $r \leftarrow H(I \| m)$
  3. $s \leftarrow P_2(sk, \mathrm{st}, r)$
  4. Renvoyer la signature (r, s)
- $\mathrm{Vrfy}_{pk}(m, r, s)$:
  1. $I \leftarrow V(pk, r, s)$
  2. Teste si $H(I, m) = r$

Théorème (admis)

Si $(\mathrm{Gen}_{id}, P_1, P_2, V)$ est sûr et *H est aléatoire*, $(\mathrm{Gen}, \mathrm{Sign}, \mathrm{Vrfy})$ est sûr

Protocole de signature de Schnorr

Description du protocole

- $\mathrm{Gen}(1^n)$:
  1. Choix d'un groupe cyclique G d'ordre $q \simeq 2^n$ et de générateur g
  2. $x \leftarrow_R \{0, \ldots, q-1\}$; $y \leftarrow g^x$
  3. Renvoyer $pk = (G, q, g, y)$ et $sk = x$
- $\mathrm{Sign}_{sk}(m)$: $m \in \{0,1\}^*$
  1. $k \leftarrow_R \{0, \ldots, q-1\}$
  2. $I \leftarrow g^k$; $r \leftarrow H(I \| m)$; $s \leftarrow rx + k \bmod q$
  3. Renvoyer la signature (r, s)
- $\mathrm{Vrfy}_{pk}(m, r, s)$:
  1. $I \leftarrow g^s \cdot y^{-r}$
  2. Teste si $H(I, m) = r$

Théorème

Si le logarithme discret est difficile dans G et H *est aléatoire*, le protocole de signature de Schnorr est sûr

Conclusion

Protocole de signature

- Objectifs :
  - assurer l'authenticité d'un message — *identité de l'expéditeur*
  - assurer la non-répudiation — *engagement de l'expéditeur*
- Version asymétrique (et plus puissante !) des MAC

Constructions

- Basée sur les problèmes que le chiffrement asymétrique (RSA, log. discret, …)
- Coût de calcul et communication élevés → *Hash-and-sign*
- Liens avec les preuves d'identité

L'authentification sans chiffrement peut-être utile…
… le chiffrement sans authentification ne sert à rien !

Conclusion générale du cours

Cryptographie symétrique

- Confidentialité → chiffrement
  - Chiffrement par blocs / par flot
  - AES, LFSR
- Authenticité → MAC
  - Fonctions de hachage

Cryptographie asymétrique

- Confidentialité → chiffrement
  - Échange de clefs, chiffrement, KEM
  - Diffie-Hellman, RSA, ElGamal
- Authenticité → signatures
  - Protocole d'identification

En commun

- *Objectifs de sécurité*
  - EAV : chiffré seulement
  - CPA : clair choisi
  - CCA : chiffré choisi
- *Hypothèses* :
  - Symétrique : fonction / permutation aléatoire, etc.
  - Asymétrique : RSA, CDH, DDH
- *Preuves de sécurité* : hypothèse $\Rightarrow$ objectif

Pour aller plus loin

Autres objectifs de sécurité

- Intégrité des données, non-répudiation
- Indistinguabilité $\rightarrow$ non-malléabilité, …
- Attaques quantiques, attaques par canaux cachés, …

Autres protocoles

- Vote et monnaie électroniques
- Partage de secret, calcul multipartite
- Preuves à divulgation nulle de connaissance
- Chiffrement homomorphe, calculs secrets délégués

Autres primitives (asymétriques)

- Courbe (hyper-)elliptiques, isogénies de courbes
- Réseaux euclidiens, cryptographie basée sur les codes
- Systèmes polynomiaux