HAI709I - Fondements cryptographiques pour la sécurité

Cours 11 - Signatures numériques

Rocco Mora

8 Décembre, 2025

Université de Montpellier – Faculté des Sciences
M1 informatique, parcours Algo, IASD, Imagine

Signatures numériques pour intégrité/authenticité

Signatures numériques : Analogues aux MAC, mais à clé publique

Scénario : Une société de logiciels publie des mises à jour. Si la mise à jour est authentifiée, les clients peuvent vérifier que la source est fiable.

Avantages supplémentaires :

+ La distribution et la gestion des clés sont simplifiées, lorsque il y a plusieurs destinataires
+ Les signatures numériques sont vérifiables publiquement : des tiers peuvent vérifier qu'un message est légitime
+ Les signatures numériques offrent non-répudiation : l'expéditeur qui signe un message ne peut pas le nier par la suite $\rightarrow$ utile pour les applications juridiques

Cependant : les MAC sont plus courts et plus efficaces à générer/vérifier

Schéma de signature numérique [🇬🇧 Digital signature scheme]

Un schéma de signature numérique est composé de trois algorithmes en temps polynomial $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ tels que :

- L'**algorithme de génération de clé** probabiliste Gen prend en entrée un paramètre de sécurité 1^n et produit une paire de clés $(\mathtt{pk}, \mathtt{sk}) \leftarrow \mathsf{Gen}(1^n)$ appelées respectivement clés publique et privée d'une longueur d'au moins n, et n peut être déterminé à partir de $\mathtt{pk}, \mathtt{sk}$.
- L'**algorithme de signature** [🇬🇧 signing algorithm] Sign prend en entrée une clé privée $\mathtt{sk}$ et un message m provenant d'un espace de messages qui peut dépendre de $\mathtt{pk}$. Il produit une signature $\sigma \leftarrow \mathsf{Sign}_{\mathtt{sk}}(m)$.
- L'**algorithme de vérification** [🇬🇧 verification algorithm] déterministe Vrfy prend en entrée une clé publique $\mathtt{pk}$, un message m et une signature σ. Il produit un bit $b := \mathsf{Vrfy}_{\mathtt{pk}}(m, \sigma)$, où $b = 1$ signifie que la signature est valide et $b = 0$ invalide.

Pour chaque message légal m, il est requis que $\boxed{\mathsf{Vrfy}_{\mathtt{pk}}(m, \mathsf{Sign}_{\mathtt{sk}}(m)) = 1}$, sauf avec une probabilité négligeable par rapport au caractère aléatoire de Gen.

L'expérience de signature $\texttt{Sig-forge}_{\mathcal{A},\Pi}(n)$

1. Une paire de clés $(\texttt{pk}, \texttt{sk})$ est générée à partir de $\text{Gen}(1^n)$.
2. $\mathcal{A}$ reçoit $\texttt{pk}$ et l'accès à l'oracle $\text{Sign}_{\texttt{sk}}$. L'adversaire produit (m, σ). Soit $\mathcal{Q}$ l'ensemble de toutes les requêtes que $\mathcal{A}$ a adressées à l'oracle.
3. Le résultat de l'expérience est 1 ($\mathcal{A}$ réussit) si et seulement si $\text{Vrfy}_{\texttt{pk}}(m, \sigma) = 1 \wedge m \notin \mathcal{Q}$.

Infalsifiabilité existentielle sous une attaque adaptative par message choisi

Un schéma de signature $\Pi = (\text{Gen}, \text{Sign}, \text{Vrfy})$ est existentiellement infalsifiable sous une attaque adaptative par message choisi, ou sûr, si pour tout adversaire PPT $\mathcal{A}$, il existe une fonction négligeable $\texttt{negl}$ telle que

$$\Pr(\texttt{Sig-forge}_{\mathcal{A},\Pi}(n) = 1) \leq \texttt{negl}(n).$$

- Si, pour chaque $(\texttt{pk}, \texttt{sk}) \leftarrow \text{Gen}(1^n)$, $m \in \{0,1\}^{\ell(n)}$ pour une certaine fonction $\ell(n)$, alors on parle d'un schéma de signature pour les messages de longueur $\ell(n)$.
- Comme pour les MAC, on peut définir des signatures numériques fortement sûres.

Paradigme 1 : Hash-and-Sign

Paradigme Hash-and-Sign

Soit $\Pi = (\mathsf{Gen}, \mathsf{Sign}, \mathsf{Vrfy})$ un schéma de signature pour des messages de longueur $\ell(n)$ et soit $\mathcal{H} = (\mathsf{Gen}_H, H)$ une fonction de hachage avec une longueur de sortie $\ell(n)$. Construisons le schéma de signature suivant $\Pi' = (\mathsf{Gen}', \mathsf{Sign}', \mathsf{Vrfy}')$:

- Gen' : en entrée 1^n, exécutez $\mathsf{Gen}(1^n)$ pour obtenir $(\mathtt{pk}, \mathtt{sk})$ et exécutez $\mathsf{Gen}_H(1^n)$ pour obtenir s, la clé publique est $(\mathtt{pk}, s)$ et la clé privée est $(\mathtt{sk}, s)$.
- Sign' : en entrée une clé privée $(\mathtt{sk}, s)$ et un message $m \in \{0,1\}^*$, produit $\sigma \leftarrow \mathsf{Sign}_{\mathtt{sk}}(H^s(m))$.
- Vrfy' : en entrée une clé publique (pk, s), un message $m \in \{0,1\}^*$ et une signature σ, produit 1 si et seulement si $\mathsf{Vrfy}_{\mathtt{pk}}(H^s(m), \sigma) = 1$.

Théorème

Si Π est un schéma de signature sûr pour les messages de longueur ℓ et $\mathcal{H}$ est résistante aux collisions, alors le paradigme Hash-and-Sign fournit un schéma de signature sûr pour les messages de longueur arbitraire.

Schéma de signature RSA simple

Soit GenRSA comme d'habitude. Définissons le schéma de signature suivant :

- Gen : (N, e, d) est obtenu à partir de $\mathrm{GenRSA}(1^n)$. La clé publique est (N, e) et la clé privée est (N, d).
- Sign : en entrée, une clé privée $\mathtt{sk} = (N, d)$ et un message $m \in \mathbb{Z}_N^*$, calculer la signature
$$\sigma := m^d \mod N.$$
- Vrfy : en entrée une clé publique $\mathtt{pk} = (N, e)$, un message $m \in \{0,1\}^*$ et une signature $\sigma \in \mathbb{Z}_N^*$, renvoie 1 si et seulement si
$$m = \sigma^e \mod N.$$

Propriété de correction : $\sigma^e = (m^d)^e = m^{ed} \mod \phi(N) = m^1 = m \mod N.$

Attaques contre la signature RSA simple

Même sans résoudre le problème RSA, un attaquant peut falsifier des signatures :

- **Attaque 1 :** Étant donné une clé publique $\mathrm{pk} = (N, e)$, $\mathcal{A}$ choisit un $\sigma \in \mathbb{Z}_N^*$ uniforme, calcule $m := \sigma^e \mod N$ et produit la paire (m, σ)
  - \+ l'attaque ne nécessite aucune paire de messages/signatures légitimes
  - − aucun contrôle sur le message falsifié m
- **Attaque 2 :** Étant donné une clé publique $\mathrm{pk} = (N, e)$, $\mathcal{A}$ choisit un message $m \in \mathbb{Z}_N^*$ qu'il souhaite falsifier. Ensuite, $\mathcal{A}$ choisit arbitrairement $m_1, m_2 \in \mathbb{Z}_N^*$ tels que $m = m_1 \cdot m_2 \mod N$ et obtient les signatures respectives σ_1, σ_2 de l'oracle. Ensuite, $\mathcal{A}$ produit la signature $\sigma := \sigma_1 \cdot \sigma_2 \mod N$ pour le message m. En effet

$$\sigma^e = (\sigma_1 \cdot \sigma_2)^e = (m_1^d \cdot m_2^d)^e = m_1^{ed} \cdot m_2^{ed} = m_1 \cdot m_2 = m \mod N.$$

  - − l'attaque nécessite 2 paires message/signature
  - \+ le message falsifié est arbitraire

Comment prévenir ces attaques ?

RSA-FDH

Schéma de signature RSA-FDH (full domain hash)

Soit GenRSA comme d'habitude. Définissons le schéma de signature suivant :

- Gen : (N, e, d) est obtenu à partir de GenRSA(1^n). La clé publique est (N, e) et la clé privée est (N, d).
- Sign : en entrée une clé privée $\mathtt{sk} = (N, d)$ et un message $m \in \{0,1\}^*$, calculer
$$\sigma := H(m)^d \mod N.$$
- Vrfy : en entrée une clé publique (N, e), un message m et une signature σ, renvoie 1 si et seulement si $H(m) = \sigma^e \mod N$.

Soit $H : \{0,1\}^* \to \mathbb{Z}_N^*$ une fonction telle que :

- H est difficile à inverser
- il est difficile de trouver m, m_1, m_2 tels que $H(m) = H(m_1) \cdot H(m_2)$
- H est résistante aux collisions

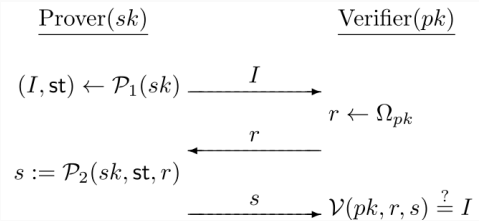
Problème RSA dur + H oracle aléatoire $\Rightarrow$ RSA-FDH sûr

Paradigme 2 : la transformation de Fiat-Shamir

Protocole d'identification :[🇬🇧 Identification scheme] protocole interactif permettant de s'authentifier auprès d'un autre (par ex., un utilisateur se connectant à un site web)

Dans ce cours : protocoles d'identification particuliers à 3 tours $\Pi = (\mathsf{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$:

- protocole entre 2 parties : prouveur (clé secrète) et vérifieur (clé publique)
- Algorithmes $\mathcal{P}_1, \mathcal{P}_2, \mathcal{V}$, I message initial, $\mathtt{st}$ état
- Ω_{pk} ensemble de défis [🇬🇧 set of challenges] défini par la clé publique du prouveur
- tout ce qui est envoyé forme une transcription [🇬🇧 transcript] $\mathtt{tr} = (I, r, s)$



Condition technique : chaque I a une probabilité négligeable d'être généré par $\mathcal{P}_1(\mathtt{sk})$

Propriété de correction : si le prouveur légitime exécute correctement le protocole, le vérificateur doit toujours accepter

Sécurité d'un protocole d'identification

L'expérience d'identification $\texttt{Ident}_{\mathcal{A},\Pi}(n)$

1. $(\texttt{pk}, \texttt{sk})$ est obtenu à partir de $\text{Gen}(1^n)$.
2. $\mathcal{A}$ reçoit $\texttt{pk}$ et l'accès à un oracle $\texttt{tr}_{sk}$.
3. $\mathcal{A}$ génère un message I. Un défi uniforme $r \in \Omega_{\texttt{pk}}$ est choisi et donné à $\mathcal{A}$.
4. $\mathcal{A}$ a toujours accès à $\texttt{tr}_{sk}$ et produit un certain s.
5. Le résultat de l'expérience est 1 si et seulement si $\mathcal{V}(\texttt{pk}, r, s) = I$.

Sécurité contre une attaque passive

Un protocole d'identification $\Pi = (\text{Gen}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ est sûr (contre une attaque passive), si pour tout PPT $\mathcal{A}$, il existe une fonction négligeable $\texttt{negl}$ telle que

$$\Pr(\texttt{Ident}_{\mathcal{A},\Pi}(n) = 1) \leq \texttt{negl}(n).$$

D'un protocole d'identification à une signature numérique

Idée : le signataire agit en tant que prouveur, simulant le protocole

La transformation de Fiat-Shamir

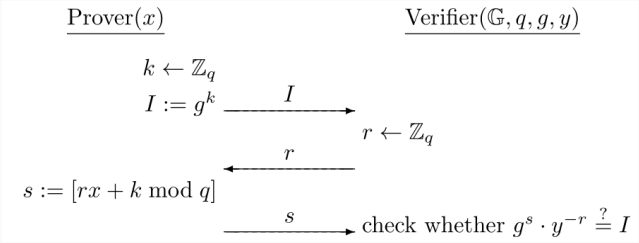
Soit $(\mathrm{Gen}_{id}, \mathcal{P}_1, \mathcal{P}_2, \mathcal{V})$ un protocole d'identification. Voici un schéma de signature :

- Gen : la paire de clés $(\mathtt{pk}, \mathtt{sk})$ est obtenue à partir de Gen_{id}. La clé publique $\mathtt{pk}$ spécifie l'ensemble $\Omega_{\mathtt{pk}}$ et soit $H : \{0,1\}^* \to \Omega_{\mathtt{pk}}$ une fonction.
- Sign : en entrée, une clé privée $\mathtt{sk}$ et un message $m \in \{0,1\}^*$, calculer $(I, \mathtt{st}) \leftarrow \mathcal{P}_1(\mathtt{sk}), r := H(I, m)$ et $s := \mathcal{P}_2(\mathtt{sk}, \mathtt{st}, r)$.
  Afficher la signature (r, s).
- Vrfy : en entrée une clé publique $\mathtt{pk}$, un message m et une signature (r, s), calculer $I := \mathcal{V}(\mathtt{pk}, r, s)$ et afficher 1 si et seulement si $H(I, m) = r$.

Schéma d'identification sûr + H oracle aléatoire $\Rightarrow$ schéma de signature sûr

Protocole d'identification/schéma de signature de Schnorr

Protocole d'identification de Schnorr : basé sur la difficulté du logarithme discret



Théorème

Si le logarithme discret est difficile par rapport à $\mathbb{G}$, alors le schéma d'identification de Schnorr est sûr.

Schéma de signature de Schnorr = Prot. d'ident. de Schnorr + transf. de Fiat-Shamir

Exercice : Écrivez le schéma de signature en détail

Algorithme de signature numérique (DSA) et **Algorithme de signature numérique à courbe elliptique (ECDSA)** : toujours basés sur le modèle "schéma d'identification $\rightarrow$ non interactif", mais $\neq$ Fiat-Shamir.

DSA/ECDSA (description générale)

Soit $\mathcal{G}$ un algorithme de génération de groupe.

- Gen : $(\mathbb{G}, q, g)$ est obtenu à partir de $\mathcal{G}(1^n)$. Choisir $x \in \mathbb{Z}_q$ uniforme et définir $y := g^x$. La clé publique est $(\mathbb{G}, q, g, y)$ et la clé privée est x. Deux fonctions $H : \{0,1\}^* \rightarrow \mathbb{Z}_q$ et $F : \mathbb{G} \rightarrow \mathbb{Z}_q$ sont également spécifiées.
- Sign : en entrée, une clé privée x et un message $m \in \{0,1\}^*$, choisir un nombre uniforme $k \in \mathbb{Z}_q^*$, définir $r := F(g^k)$ et calculer $s := k^{-1} \cdot (H(m) + xr) \mod q$. Répéter jusqu'à ce que $r, s \neq 0 \mod q$ et afficher la signature (r, s).
- Vrfy : à partir de la clé publique $(\mathbb{G}, q, g, y)$, d'un message m et d'une signature (r, s), avec $r, s \neq 0 \mod q$, afficher 1 si et seulement si $r = F(g^{H(m) \cdot s^{-1}} y^{r \cdot s^{-1}})$.

- basé sur le logarithme discret dans différents groupes
  - DSA : $\mathbb{G}$ sous-groupe d'ordre q de $\mathbb{Z}_p^*$ et $F(I) = I \mod q$
  - ECDSA : $\mathbb{G}$ sous-groupe d'ordre q d'un groupe de courbes elliptiques
- normalisé par le NIST et aucune attaque connue
- cependant, k ne doit pas être utilisé pour 2 signatures différentes

→ Piratage de la clé privée principale de la Sony Playstation (PS3) en 2010 :

Si k est utilisé deux fois, alors $r_1 = r_2 = r$. Soient $(r, s_1), (r, s_2)$ les signatures de m_1, m_2. Alors

$$\begin{aligned} s_1 &= k^{-1} \cdot (H(m_1) + xr) \mod q, \\ s_2 &= k^{-1} \cdot (H(m_2) + xr) \mod q \\ \Rightarrow s_1 - s_2 &= k^{-1} \cdot (H(m_1) - H(m_2)) \mod q \\ \Rightarrow k &= (s_1 - s_2)^{-1} \cdot (H(m_1) - H(m_2)) \mod q \\ \Rightarrow x &= (s \cdot k - H(m)) \cdot r^{-1} \mod q, \end{aligned}$$

c'est-à-dire que la clé secrète nécessaire pour signer m est connue.

Conclusion : la cryptographie moderne concerne-t-elle la vie quotidienne ?

Protocole TLS (Transport Layer Security)

- utilisé par les navigateurs web pour se connecter en toute sécurité à un site web à l'aide du protocole `https`
- s'appuie sur le protocole SSL (Secure Sockets Layer) désormais obsolète
- plusieurs versions : TLS 1.0 (1999), 1.1 (2006), 1.2 (2008), 1.3 (2018)

Le protocole se compose de deux parties

1) Handshake protocol : échange de clés authentifié pour établir les clés partagées

- **Échange de clés :** Diffie-Hellman avec courbes elliptiques
- **Authentification :** RSA-PSS
- **Hachage de signature :** SHA-256
- **Dérivation de clé :** HMAC-SHA-256

2) Record protocol : utilise les clés partagées pour chiffrer/authentifier les communications entre les parties

- **Chiffrement + authentification :** AES-GCM, ChaCha20-Poly1305, AES-CCM
- **Intégrité :** Poly1305, GCM