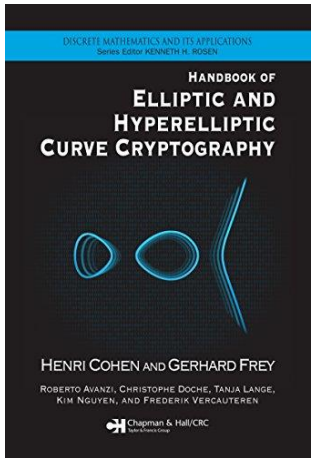
Introduction à la Sécurité Prouvable

Application à la cryptographie à clé publique

Olivier Sanders (Orange)

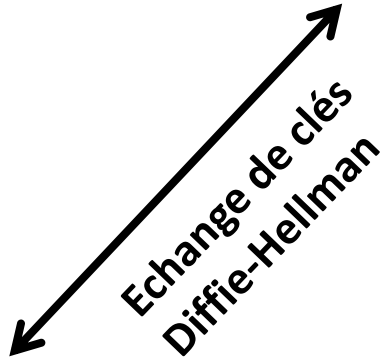
Echange Sécurisé

Echange de clés Diffie-Hellman



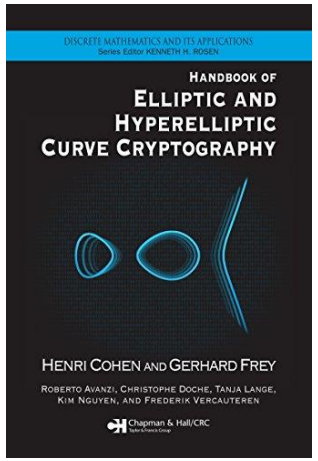
Alice sait échanger une clé…

Echange Sécurisé



amazon

U.S.



Mais avec qui???

Signature Numérique

- Mécanisme introduit en 1976
  - Une clé secrète permet de signer les messages
  - Une clé publique permet à tout le monde de vérifier la validité des signatures

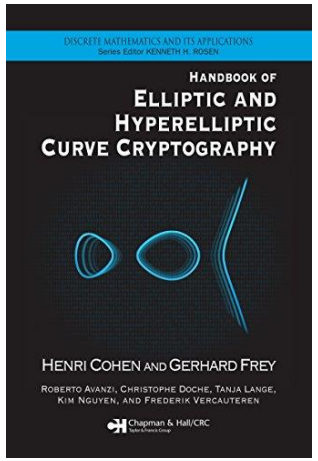
Signature Numérique

amazon

U.S.

Clé secrète: sk
Clé publique pk

Echange de clés
Diffie-Hellman
Signature

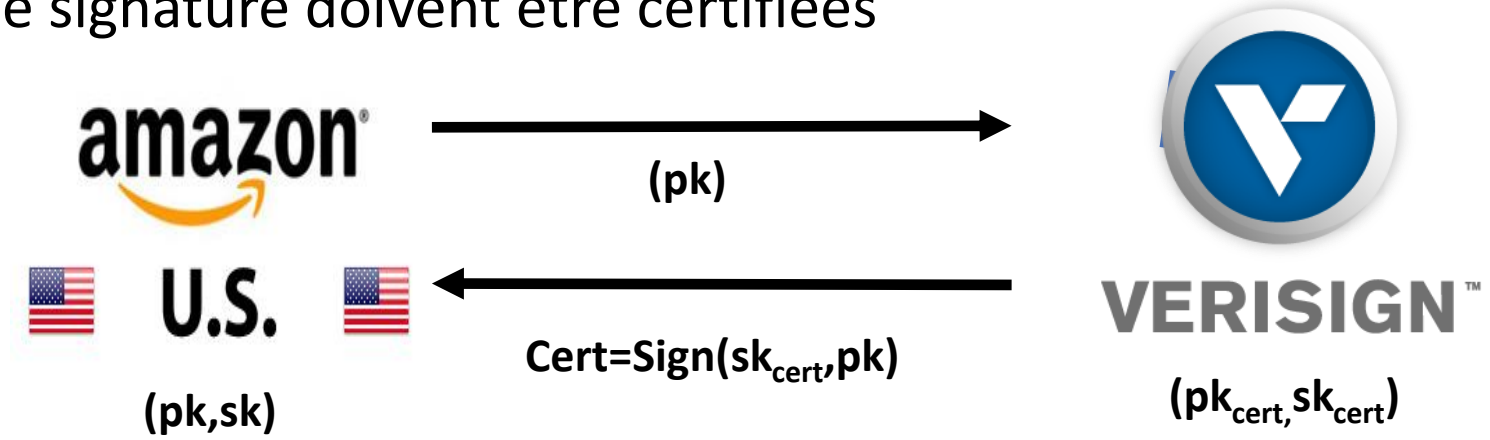


Amazon va signer différents éléments (dont les données de l'échange de clés)

→ Alice sait qu'elle échange une clé avec Amazon

Certification des clés

- Les clés de signature doivent être certifiées



- En pratique: recours à des autorités de certification

Signature numérique

Un mécanisme de signature numérique 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).

- Sign(sk,m): cet algorithme prend en entrée une clé secrète sk et un message m et retourne une signature S.

- Verify(pk,S,m): cet algorithme prend en entrée une clé publique pk, une signature S et un message m et retourne 1 (valide) ou 0 (invalide).

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

$$\text{Verify}(pk,\text{Sign}(sk,m),m) = 1$$

Sécurité

- Exemple de la signature numérique

Que signifie casser la sécurité?

S= Sign(je dois 100 $ à Bob)

- Objectifs:
  - Retrouver la clé secrète?

Définition de la sécurité

- Exemple de la signature numérique

Que signifie casser la sécurité?

S= Sign(je dois 1000 $ à Bob)

- Objectifs:
  - ~~Retrouver la clé secrète?~~
  - Produire une signature sur un nouveau message? (EUF)

Définition de la sécurité

- Exemple de la signature numérique

Que signifie casser la sécurité?

S’= Sign(je dois 100 $ à Bob)

- Objectifs:
  - ~~Retrouver la clé secrète?~~
  - Produire une nouvelle signature sur le même message? (SUF)

Modèle de sécurité

Osign(m_i) retourne Sign(sk, m_i)

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

Jeu EUF-CMA

Modèle de sécurité

Osign(m_i) retourne Sign(sk, m_i)

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

Osign permet à l'adversaire d'obtenir des signatures sur les messages de son choix

Jeu EUF-CMA

CMA = Chosen Message Attacks

Modèle de sécurité

Osign(m_i) retourne Sign(sk, m_i)

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

En contrepartie, il ne gagne que s'il produit une signature sur un message jamais demandé à Osign

Jeu EUF-CMA

EUF= Existential UnForgeability

Modèle de sécurité

Osign(m_i) retourne Sign(sk, m_i)

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et S n'a jamais été retourné par Osign
   Retourner 1

4) Sinon retourner 0

Les conditions de succès sont affaiblies: l'adversaire gagne même si m a été requis à Osign, du moment que la signature diffère

Jeu SUF-CMA

SUF= Strong UnForgeability

Une Première tentative

- Clé publique: N = pq , e un entier premier avec $\varphi(N) = (p-1)(q-1)$
- Clé secrète: d tel que $e*d = 1 \bmod \varphi(N)$
- Sign(sk,m): Retourne $S=m^d$
- Verify(pk,S,m): Retourne 1 si $S^e = m$ et 0 sinon

Correct car $S^e = (m^d)^e = m$

Une Première tentative

- Clé publique: N = pq , e un entier premier avec $\varphi(N) = (p-1)(q-1)$
- Clé secrète: d tel que $e*d = 1 \bmod \varphi(N)$
- Sign(sk,m): Retourne $S=m^d$
- Verify(pk,S,m): Retourne 1 si $S^e = m$ et 0 sinon

Correct car $S^e = (m^d)^e=m$

Est-ce (EUF-CMA) sûr?

Une Première tentative

- Clé publique: N = pq , e un entier premier avec $\varphi(N) = (p-1)(q-1)$
- Clé secrète: d tel que $e*d = 1 \bmod \varphi(N)$
- Sign(sk,m): Retourne $S=m^d$
- Verify(pk,S,m): Retourne 1 si $S^e = m$ et 0 sinon

Correct car $S^e = (m^d)^e = m$

~~Est-ce (EUF-CMA) sûr?~~ NON

Tout S mod N est une signature valide sur $m = S^e$

L’approche Hash-and-Sign

- Clé publique: N = pq , e un entier premier avec $\varphi(N) = (p-1)(q-1)$, H une fonction de hachage à valeur dans (Z/NZ)*
- Clé secrète: d tel que $e*d = 1 \bmod \varphi(N)$
- Sign(sk,m): Retourne $S=H(m)^d$
- Verify(pk,S,m): Retourne 1 si $S^e = H(m)$ et 0 sinon

Correct car $S^e = (H(m)^d)^e = H(m)$

Est-ce (EUF-CMA) sûr?

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

On veut utiliser un adversaire contre l'EUF-CMA pour résoudre ce problème

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

Jeu EUF-CMA

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

Pour résoudre le problème, il faut insérer les challenges dans le jeu. Ici, on va donc définir pk = N,e

Jeu EUF-CMA

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

Osign(m_i)

?????
Retourne S tel que $S^e = H(m\_i)$

Mais comment répondre aux requêtes Osign sans sk ?

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

Osign(m_i)

?????

Retourne S tel que $S^e = H(m\_i)$

H(m_i)

Si m_i jamais demandé:

u_i <- Z/ZNZ

Retourne u_i^e

Sinon retourne le même résultat que la requête initiale sur m_i

On utilise les capacités de l'oracle aléatoire

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

Osign(m_i)

?????

Retourne S tel que $S^e = H(m\_i)$

H(m_i)

Si m_i jamais demandé:

u_i <- Z/ZNZ

Retourne u_i^e

Sinon retourne le même résultat que la requête initiale sur m_i

Permet de définir tout H(m_i) comme un certain u_i^e pour un u_i connu

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

Osign(m_i)
On consulte les requêtes H(m_i)
On pose S = u_i
Retourne S tel que $S^e = H(m_i)$

Utilise l'oracle aléatoire pour construire une signature valide

H(m_i)
Si m_i jamais demandé:
    u_i <- Z/ZNZ
    Retourne u_i^e
Sinon retourne le même résultat que la requête initiale sur m_i

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

Osign(m_i)
On consulte les requêtes H(m_i)
On pose S = u_i
Retourne S tel que $S^e = H(m_i)$

Utilise l'oracle aléatoire pour construire une signature valide

H(m_i)
Si m_i jamais demandé:
    u_i <- Z/ZNZ
    Retourne u_i^e
Sinon retourne le même résultat que la requête initiale sur m_i

La réponse u_i est générée aléatoirement -> la simulation est parfaite

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

Mais comment insérer y dans le jeu pour avoir x?

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

On va utiliser les capacités de A:
Si H(m) = y, alors l'adversaire nous donnera la réponse:
$S^e = H(m)$ -> x = S

Mais comment insérer y dans le jeu pour avoir x?

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

H(m_i)
Si i = j, retourner y
Sinon, si m_i jamais demandé:
u_i <- Z/ZNZ
Retourne u_i^e
Sinon retourne le même résultat que la requête initiale sur m_i

L'oracle aléatoire fonctionne normalement pour toutes les requêtes…

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

… sauf une (notée j) choisie au hasard!
On insère y dans celle-ci

H(m_i)
Si i = j, retourner y
Sinon, si m_i jamais demandé:
u_i <- Z/ZNZ
Retourne u_i^e
Sinon retourne le même résultat
que la requête initiale sur m_i

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

… sauf une (notée j) choisie au hasard!
On insère y dans celle-ci

H(m_i)
Si i = j, retourner y
Sinon, si m_i jamais demandé:
u_i <- Z/ZNZ
Retourne u_i^e
Sinon retourne le même résultat
que la requête initiale sur m_i

Si l'adversaire demande m lors de la requête j, on a gagné !

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

… sauf une (notée j) choisie au hasard!
On insère y dans celle-ci

H(m_i)
Si i = j, retourner y
Sinon, si m_i jamais demandé:
u_i <- Z/ZNZ
Retourne u_i^e
Sinon retourne le même résultat
que la requête initiale sur m_i

Sinon, on perd !

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

On va utiliser les capacités de A:
Si H(m) = y, alors l'adversaire nous donnera la réponse:
$S^e = H(m)$ -> x = S

Si q est le nombre de requêtes à H(m), A peut servir à résoudre le problème RSA avec proba 1/q

Preuve EUF-CMA

Problème RSA: Etant donnés N,e et y mod N retrouver x tel que $x^e = y$

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

2) (m,S) <- A^{Osign}(pk)

3) Si Verify(pk,S,m) = 1, et m n'a jamais été demandé à Osign
   Retourner 1

4) Sinon retourner 0

On va utiliser les capacités de A:
Si H(m) = y, alors l'adversaire nous donnera la réponse:
$S^e = H(m)$ -> x = S

Le schéma de signature est EUF-CMA dans le modèle de l'oracle aléatoire si le problème RSA est dur

Une autre construction: BLS

- Groupes Bilinéaires: 3 groupes G_1, G_2 et G_T d'ordre p et une application e appelée couplage tels que:
  - Bilinéarité: $e(g^a, h^b) = e(g,h)^{ab}$ pour tout g dans G_1, h dans G_2, et a,b mod p
  - Non-dégénérescence: $e(g,h) = 1 \rightarrow g = 1$ ou $h = 1$
- Construits à partir de courbes elliptiques
- Très utilisés en cryptographie

Signature BLS

- Clé publique: h dans G_2, h^x , H une fonction de hachage à valeur dans G_1
- Clé secrète: x
- Sign(sk,m): Retourne $S=H(m)^x$
- Verify(pk,S,m): Retourne 1 si $e(S,h) = e(H(m), h^x)$ et 0 sinon

Correct car $e(S,h) = e(H(m)^x , h) = e(H(m),h)^x = e(H(m), h^x)$

Repose sur la bilinéarité du couplage

Signature BLS

- Clé publique: h dans G_2, h^x , H une fonction de hachage à valeur dans G_1
- Clé secrète: x
- Sign(sk,m): Retourne $S=H(m)^x$
- Verify(pk,S,m): Retourne 1 si $e(S,h) = e(H(m), h^x)$ et 0 sinon

Problème CDH (couplage) : Etant donnés g, g^a, g^b dans G_1 et h, h^a dans G_2, calculer g^{ab}

Exercice: Montrer que les signatures BLS sont EUF-CMA dans le ROM si le problème CDH est difficile

Sécurité weak CMA

1) $(m_1, \ldots, m_q)$ <- A

A peut demander des signatures sur les messages de son choix mais avant de voir la clé publique

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

3) S_i <- Sign(sk, m_i)

4) (m,S) <- A (pk, $S_1, \ldots, S_q$)

5) Si Verify(pk,S,m) = 1, et m diffère de tous les m_i
Retourner 1

6) Sinon retourner 0

Jeu wEUF-CMA

Sécurité weak CMA

1) $(m_1, \ldots, m_q) \leftarrow A$

A peut demander des signatures sur les messages de son choix mais avant de voir la clé publique

2) $(sk,pk) \leftarrow Keygen(1^n)$

3) $S_i \leftarrow Sign(sk, m_i)$

4) $(m,S) \leftarrow A\,(pk, S_1, \ldots, S_q)$

5) Si $Verify(pk,S,m) = 1$, et m diffère de tous les m_i
Retourner 1

6) Sinon retourner 0

Jeu wEUF-CMA

Plus faible que EUF-CMA mais des conversions génériques existent pour atteindre ce niveau

Signature Boneh-Boyen

- Clé publique: g dans G_1, h dans G_2, h^x
- Clé secrète: x
- Sign(sk,m): Retourne S=g^(1/ (x+m))
- Verify(pk,S,m): Retourne 1 si e(S, $h^x \times h^m$) = e(g, h) et 0 sinon

Correct car e(S, $h^x \times h^m$) = e(g^(1/ (x+m)) , $h^{(x+m)}$) = e(g, h)

Problème SDH : Etant donnés u,u^a, u^{a^2},…,u^{a^q} dans G_1 et h,h^a dans G_2, calculer (w,u^(1/ (a+w)) pour un certains w mod p

Exercice: Montrer que les signatures BB sont wEUF-CMA dans le modèle standard (sans ROM) si SDH est difficile