La Cryptographie Asymétrique et les Preuves de Sécurité

David Pointcheval
Chargé de recherche CNRS
Département d'informatique
École normale supérieure

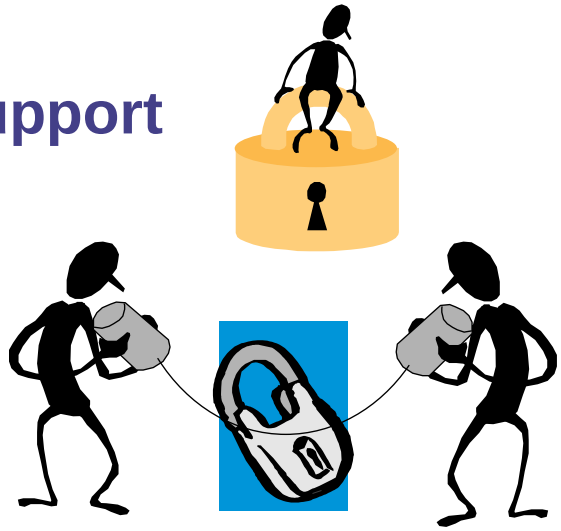
Sommaire

Introduction

1. Introduction
2. Les hypothèses algorithmiques
3. Le chiffrement asymétrique
4. Les preuves de sécurité
5. La signature
6. Conclusion

Confidentialité

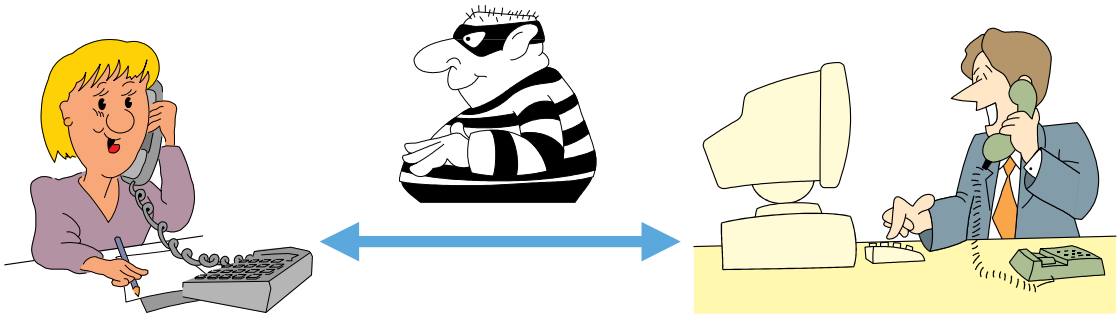
- Archiver sur un **support**
- Émettre sur un **canal**



de façon qu'un tiers ne puisse en prendre connaissance

Authentification (1)

Prouver de façon **interactive** son identité à un interlocuteur

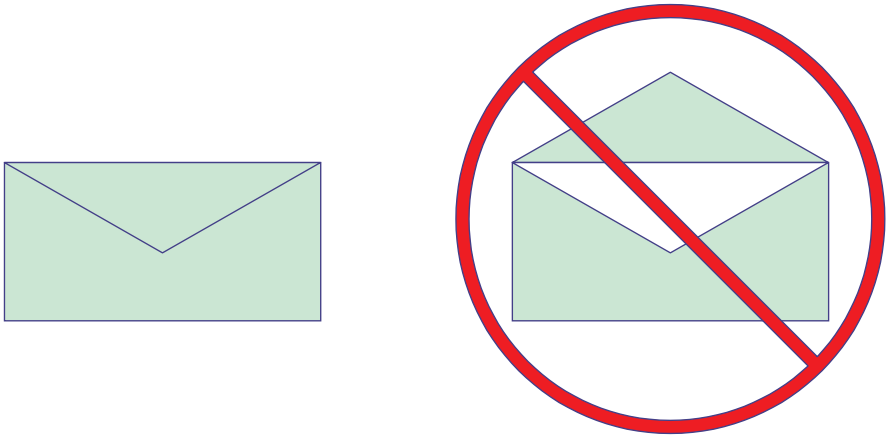


Authentification (2)

- Attacher, à un message, une preuve **non-interactive** de son origine
- Si cette preuve peut convaincre un tiers
  $\Rightarrow$ **signature**

Intégrité

Garantir qu'un message, un document, un fichier, n'a pas subi de modification (aussi bien accidentelle qu'intentionnelle)



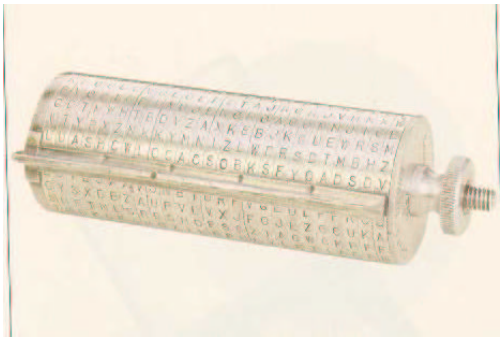
Les 3 âges de la cryptologie

- L'âge artisanal : jusqu'en 1918
- L'âge technique : de 1919 à 1975
- L'âge paradoxal : de 1976 à nos jours

Âge artisanal

Substitutions et permutations :

cadrans cylindres



Algorithmes secrets

Scytale Lacédémonienne :

Enrouler un ruban autour d'un bâton, écrire puis dérouler

⇒ permutation des caractères

Chiffrement de César :

Décalage constant des lettres du message (à l'origine +3)

MESSAGE ⇨ PHVVDJH

⇒ substitution des caractères

Chiffrement par décalage

Chaque lettre (codée dans $[0,25]$)
subit un décalage constant K

$$C_K(x) = x + K \bmod 26$$

$$D_K(y) = y - K \bmod 26$$

Cryptanalyse : recherche exhaustive

elle consiste à essayer toutes les clés K possibles (soit seulement 26)

Chiffrement par substitution

Chaque lettre (codée dans $[0,25]$)
est substituée par une autre

$$C_\pi(x) = \pi(x)$$

$$D_\pi(y) = \pi^{-1}(y)$$

où π est une permutation de $[0,25]$

Cryptanalyse : recherche exhaustive ?

Sur un alphabet de 26 lettres : 26! Possibilités

Soit plus de $4.\ 10^{26}$, ou 2^{88}

Cas particuliers

Chiffrement par décalage :

$$\pi(x) = x + K \bmod 26$$

Chiffrement affine :

$$\pi(x) = a\,x + b \bmod 26$$

Chiffrement par permutation

Cette fois-ci on garde les mêmes lettres, mais on change l'ordre (ex: scytale)

$C_K(x_1,\ldots,x_n) = (x_{\pi(1)}, x_{\pi(2)}, \ldots, x_{\pi(n)})$

$D_K(y_1,\ldots,y_n) = (y_{\pi^{-1}(1)}, y_{\pi^{-1}(2)}, \ldots, y_{\pi^{-1}(n)})$

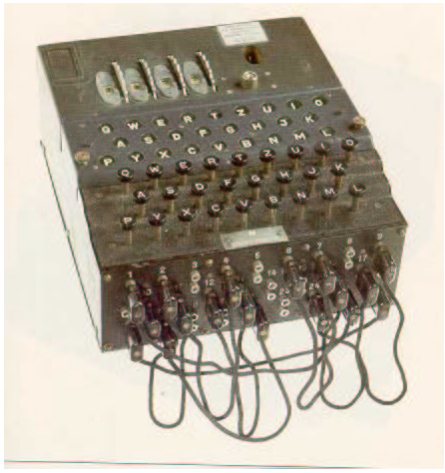
où π est une permutation de $[1,n]$

C'est un cas particulier de Hill : matrice de permutation K : $Y = KX$

Âge technique

Machines chiffrantes

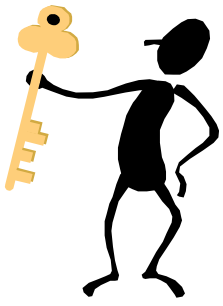
Automatisation
des permutations
et substitutions



Enigma

Âge paradoxal

- Cryptographie à clé secrète
- Cryptographie à clé publique



Fonctions à sens-unique
(éventuellement à trappe)

$\Rightarrow$ Preuves de sécurité

Principes de Kerckhoffs

En 1883, Kerckhoffs énonce plusieurs principes dont :

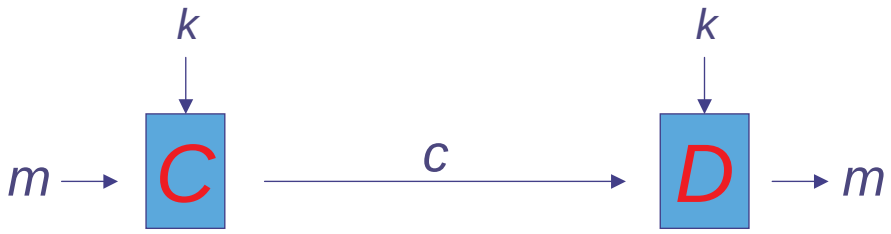
« la sécurité d'un système ne doit pas être fondée sur son caractère secret »

En effet, un jour ou l'autre il y a des fuites au sujet du système, ainsi

« seule une donnée de petite taille (clé) doit assurer la sécurité »

Chiffrement symétrique

Algorithme de chiffrement, *C*
Algorithme de déchiffrement, *D*



Sécurité : impossible de retrouver *m* à partir de *c* sans *k*

Chiffrement de Vigenère

Version ***poly-alphabétique***
du chiffrement par décalage :
décalage des lettres
en fonction de leur position

$C_K(x_1,\ldots,x_n) = (x_1 + K_1, \ldots, x_n + K_n)$

$D_K(y_1,\ldots,y_n) = (y_1 - K_1, \ldots, y_n - K_n)$

où $K = (K_1, \ldots, K_n)$

```
  M E S S A G E
+ C L E C L E C
= O P W U L K G
```


Sécurité

Recherche exhaustive :
si la clé est de longueur n,
il y a 26^n possibilités

$$n = 5 \Rightarrow 2^{23} \quad n = 10 \Rightarrow 2^{47} \quad n = 15 \Rightarrow 2^{70}$$

$\rightarrow$ Recherche exhaustive
vite hors de portée

Analyses statistiques

Probabilité d'occurrence de chaque lettre/digramme/trigramme

$\Rightarrow$ on casse aisément un chiffrement par substitution

Entropie : conservée par substitution ou permutation

$\Rightarrow$ on retrouve la longueur de la clé

Attaque de Kasiski : elle casse Vigenère en temps linéaire !

Sécurité parfaite ?

Chiffrement de Vernam :

Combiner le message à chiffrer avec une suite de bits aléatoires

$$c = m \oplus k \qquad m = c \oplus k$$

Sécurité inconditionnelle
à condition que la « clé » k soit aussi longue que le message m (Shannon)

En pratique …?

- César/Vigenère/… pas sûrs
- Vernam pas pratique

Sécurité inconditionnelle pas pratique :
mais Shannon a aussi montré que
la combinaison de substitutions
et de permutations
apportait une sécurité convenable

$\Rightarrow$ systèmes produits

Chiffrement symétrique

- clé unique *k*
  - chiffrement
  - déchiffrement
- conception heuristique :
  - permutations (diffusion)
  - substitutions (confusion)
- orienté implémentation matérielle

$\rightarrow$ débit rapide

Chiffrement symétrique

Par blocs *(block cipher)* :

décompose le message en blocs

- **DES** (IBM-NBS)
  - normal - clé de 56 bits, blocs de 64 bits
  - renforcé (**triple-DES) -** clé de 112/168 bits
- **IDEA** (Massey-Lai)
  - clé de 128 bits, blocs de 64 bits
- **AES** (RIJNDAEL : Daemen-Rijmen)
  - clés de 128, 196 ou 256 bits, blocs de 128 bits

Chiffrement symétrique

Par flot *(stream cipher)* :

chiffre un flux de données

- **Vernam** (générateur pseudo-aléatoire) :

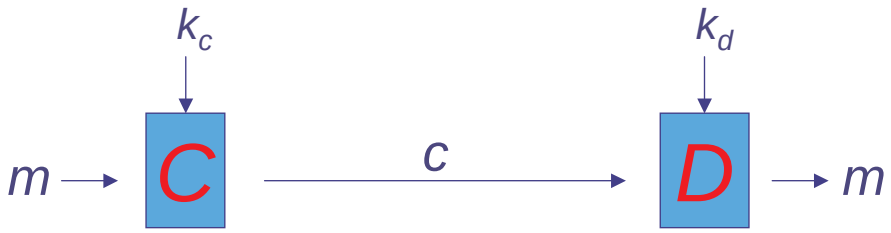
$$G(k) = k_1 k_2 \ldots \text{ et } m = m_1 m_2 \ldots$$
$$c = c_1 c_2 \ldots \text{ où } c_i = m_i \oplus k_i$$

- **RC4** (Rivest)

  chiffre octet par octet

Chiffrement asymétrique

Algorithme de chiffrement, C
Algorithme de déchiffrement, D



Sécurité : impossible de retrouver m à partir de c sans k_d

Chiffrement asymétrique

- Lois de Kerckhoffs poussées à bout
- Deux clés :
  - chiffrement k_c → publique
  - déchiffrement k_d → privée
- Fondements mathématiques :
  - fonctions à sens-uniques
  - fonctions à trappe
- Analyse de sécurité

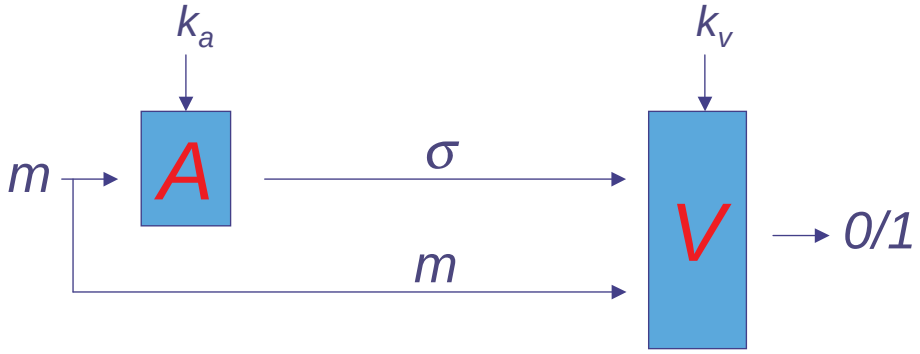
  → sécurité calculatoire

Chiffrement asymétrique

- **RSA** (Rivest-Shamir-Adleman 1978)
  racines modulaires (factorisation)
- **Merkle-Hellman** (1978)
  problème du « sac à dos » (tous cassés…)
- **Mc Eliece** (1978)
  codes correcteurs d'erreurs
- **El Gamal** (1985)
  logarithme discret
- **HFE** (Patarin 1996)
  polynômes multivariables

Authentification

- Algorithme d'authentification, *A*
- Algorithme de vérification, *V*



Sécurité: impossible de produire un σ valide sans k_a

Authentification « artisanale »

En raison du caractère secret des procédés de chiffrement

– clé secrète commune

– voire algorithme secret commun

→ Identité de l'expéditeur garantie

Mais l'expéditeur et le destinataire connaissent la convention secrète

→ Pas de non-répudiation

Signature numérique

- Deux clés :
  - signature k_a → privée
  - vérification k_v → publique
- Fondements mathématiques :
  - fonctions à sens-uniques
- Analyse de sécurité

→ sécurité calculatoire

Signature numérique

– **RSA** (Rivest-Shamir-Adleman 1978)

Extraction de Racines Modulaires

– **El Gamal** (1985)

Logarithme Discret

Variantes : Schnorr (1989), DSA (1994)

– Combinatoires (Problèmes NP-complets) :

PKP (Shamir 1989), SD (Stern 1993),
CLE (Stern 1994), PPP (Pointcheval 1995)

Identification : contrôle d'accès

- Authentification interactive :

Alice (client) veut prouver
à Bob (serveur) son identité

Je m'appelle
Alice

Prouve-le moi !

Preuves Zero-Knowledge

Protocoles « zero-knowledge » :

- preuve de connaissance d'un secret
- ***sans transfert d'information*** sur ce secret, seule la conviction de connaissance

Intégrité

La signature numérique garantit l'intégrité du message reçu

N'y aurait-il pas plus simple pour garantir l'intégrité de son propre disque dur ?

- Conserver une copie en lieu sûr !…
- Conserver un « condensé » en lieu sûr

→ ***fonctions de hachage cryptographiques***

Fonctions de hachage

Créer une empreinte/un condensé $h = \mathrm{H}(x)$

- de petite taille (moins de 256 bits ?)
- garantissant qu'une altération
  - accidentelle, mais également intentionnelle
  - de la part d'un tiers, ou même du créateur

du disque modifiera la valeur du condensé en résultant (détection de la différence)

Exemples

- MD-2, MD-4 (Rivest, 1990, 128 bits)
- MD-5 (Rivest, 1991, 128 bits) :
  très utilisée dans le monde UNIX
- SHA (NIST, 1992, 160 bits) :
  « Standard Américain » en 1993
  Remplacé, « pour faiblesse technique »
- SHA-1 (NIST, 1994, 160 bits)
- SHA-256/384/512 (NIST, 2000)
  condensés sur 256, 384 ou 512 bits

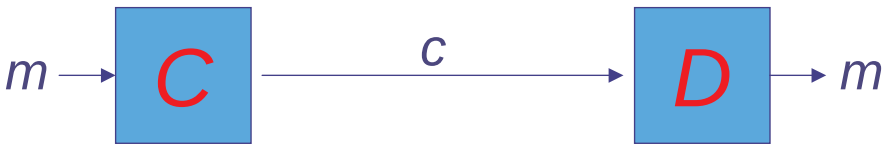
Les hypothèses algorithmiques

1. Introduction
2. Les hypothèses algorithmiques
3. Le chiffrement asymétrique
4. Les preuves de sécurité
5. La signature
6. Conclusion

Chiffrement asymétrique

Algorithme de chiffrement, C

$m \rightarrow c$: transformation aisée



Algorithme de déchiffrement, D

$c \rightarrow m$: transformation difficile
à moins de connaître une trappe

Outils nécessaires

- Fonctions à sens-unique :

  étant donnée une relation $y = f(x)$

  $x \longrightarrow y$ aisé, mais $y \longrightarrow x$ difficile

- Fonctions à sens-unique à trappe :

  étant donnée une relation $y = f(x)$

  $x \longrightarrow y$ aisé,

  $y \longrightarrow x$ difficile

  $y \longrightarrow x$ aisé avec z (« trappe »)

Où trouver des candidats ?
les mathématiques

Un premier exemple : la factorisation

$p, q \longrightarrow n = p.q$ facile $k=|n|=log_2(n)$

en $O(k^2)$ opérations élémentaires

en revanche

$n = p.q \longrightarrow p, q$ difficile

en $O\left(e^{1.92(\ln n)^{1/3}(\ln \ln n)^{2/3}}\right)$

Record actuel : nombre n produit de deux facteurs premiers de 78 chiffres

Et la trappe ?

- Le produit/factorisation
  est une fonction à sens-unique

  mais où est la trappe ?
- Certains calculs dans $\mathbf{Z}/n\mathbf{Z}$ sont
  - faciles si on connaît les facteurs de n
  - difficiles sans les facteurs de n

Théorie des nombres

L'anneau quotient :

$$\mathbf{Z}_n = \mathbf{Z}/n\mathbf{Z} = \{0,1,\dots,n-1\}$$

Théorème de Bezout

Soient a et b deux entiers

$$(\exists u,v)(au+bv=1) \Leftrightarrow \text{pgcd}(a,b)=1$$

$\mathbf{Z}_n$ corps $\Leftrightarrow$ n premier

Algorithme d'Euclide

- Trouver l'inverse d'un élément x modulo n revient à trouver u et v tels que

$$ux + vn = 1$$

- L'algorithme d'Euclide calcule le PGCD de deux entiers
- Une version « étendue » calcule les coefficients u et v.

Ces deux algorithmes ont une complexité ***linéaire*** en la taille des arguments.

Théorème des restes chinois

Quand $n=pq$ est composé ?

$$f : \mathbf{Z}_n \to \mathbf{Z}_p \times \mathbf{Z}_q \qquad g : \mathbf{Z}_p \times \mathbf{Z}_q \to \mathbf{Z}_n$$

$$x \mapsto (x \bmod p, x \bmod q) \qquad (a,b) \mapsto auq + bvp$$

f isomorphisme d'anneaux

$$u = q^{-1} \bmod p \qquad v = p^{-1} \bmod q$$

$$\mathbf{Z}_n \cong \mathbf{Z}_p \times \mathbf{Z}_q \qquad \mathbf{Z}_n^* \cong \mathbf{Z}_p^* \times \mathbf{Z}_q^*$$

Fonction d'Euler

$$(\forall n)\ \ \varphi(n) = \mathrm{card}(\mathbf{Z}_n^*)$$

p est premier :

$$\varphi(p) = \mathrm{card}(\mathbf{Z}_p^*) = p - 1$$

$n=pq$ est composé :

$$\varphi(n) = \mathrm{card}(\mathbf{Z}_n^*) = \mathrm{card}(\mathbf{Z}_p^* \times \mathbf{Z}_q^*)$$
$$= \varphi(p)\varphi(q) = (p-1)(q-1)$$

Fonction d'Euler

$$n = \prod p_i^{\,v_i} \Rightarrow \varphi(n) = n \times \prod \left(1 - \frac{1}{p_i}\right)$$

connaissance
(d'un multiple) de $\varphi(n)$

⇕

connaissance de
la factorisation de n

Théorème d'Euler

Pour tout groupe G, de cardinal c : $(\forall x \in G)(x^c = 1)$

En particulier, pour tout entier n

$$(\forall x \in \mathbf{Z}_n^*)(x^{\varphi(n)} = 1 \bmod n)$$

Petit théorème de Fermat : pour tout entier premier p

$$(\forall x < p)(x^{p-1} = 1 \bmod p)$$

La fonction RSA

- Soit un module $n=pq$
- Soit un exposant e premier avec $\varphi(n)$
- Soit $d=e^{-1} \bmod \varphi(n)$

$$e\,d + u\,\varphi(n) = 1$$

$$f : \mathbf{Z}_n^* \to \mathbf{Z}_n^* \qquad m \mapsto m^e \bmod n$$

$$g : \mathbf{Z}_n^* \to \mathbf{Z}_n^* \qquad c \mapsto c^d \bmod n$$

Hypothèse RSA

multiple de $\varphi(n) \Leftrightarrow$ factorisation de n

- calculer des racines e-ièmes :
  $\varphi(n)$ suffisant
- nécessaire ?
  - En pratique : oui (calcul de d)
  - En théorie : non
- Hypothèse RSA :

$$RSA(n,e) \leq FACT(n)$$

$$RSA(n,e) \approx FACT(n)$$

Estimations de complexité

Record de factorisation : 512 bits (155 dig.)
coût estimé : 2^{13} Mips-Years (2^{58} op.)

Module	Mips-Year	Opérations	En 2015
512	13	58	48
1024	35	80	70
2048	66	111	101
4096	104	149	139
8192	156	201	191

Le logarithme discret

- $(\mathbf{G},\times)$ groupe d'ordre q
- $g \in \mathbf{G}$ et $y \in \langle g \rangle$

$$\mathrm{Log}_g(y) = \min\{x > 0 \mid y = g^x\}$$

- Pour $\mathbf{G} \subseteq \mathbf{Z}_p^*$, le calcul du logarithme discret est en

$$O\left(e^{1.92 (\ln\ p)^{1/3} (\ln\ \ln\ p)^{2/3}} \right)$$

Le théorème de Lagrange

- $(\mathbf{G},\times)$ groupe d'ordre q
- pour tout sous-groupe $\mathbf{H} \subseteq \mathbf{G}$

$$\mathrm{Card}\ \mathbf{H} \mid \mathrm{Card}\ \mathbf{G}$$

- Pour $\mathbf{G} \subseteq \mathbf{Z}_p^*$,

$$q = \mathrm{Card} \langle g \rangle \mid p\text{-}1$$

Le problème Diffie-Hellman

$\mathbf{G} \subseteq \mathbf{Z}_p^*$ groupe cyclique d'ordre q
g générateur : $\mathbf{G} = \langle g \rangle$

- Étant donnés $A=g^a$ et $B=g^b$
- Calculer DH$(A,B) = C=g^{ab}$

Clairement DH ≤ LD

($a = \mathrm{Log}_g A$ et $C=B^a$)

Et les trappes ?

- Le problème du logarithme discret est difficile, mais pas moyen de le rendre facile !
- Et le problème Diffie-Hellman ?

Étant donnés $A=g^a$ et $B=g^b$
Calculer DH$(A,B) = C=g^{ab}$

Difficile, mais si on connaît a : $C=B^a$

La fonction Diffie-Hellman

- Soit un groupe cyclique **G** = <g>
- Soit un exposant x (privé)
- Soit $y = g^x$ (public)

$$f : \mathbf{G} \rightarrow \mathbf{G}$$
$$g^r \mapsto y^r$$

$$f : \mathbf{G} \rightarrow \mathbf{G}$$
$$R \mapsto R^x$$

Hypothèse Diffie-Hellman

- Évaluation de f :
  r ou x suffisent
- nécessaire ?
  - En pratique : oui
  - En théorie : non
- Hypothèse Diffie-Hellman :

$$DH(g) \approx LD(g)$$

Estimations de complexité

Records de logarithme discret

- sur les courbes elliptiques : GF(2^{109})
- dans $\mathbf{Z}_p^*$: 120 chiffres

  plus difficile que la factorisation

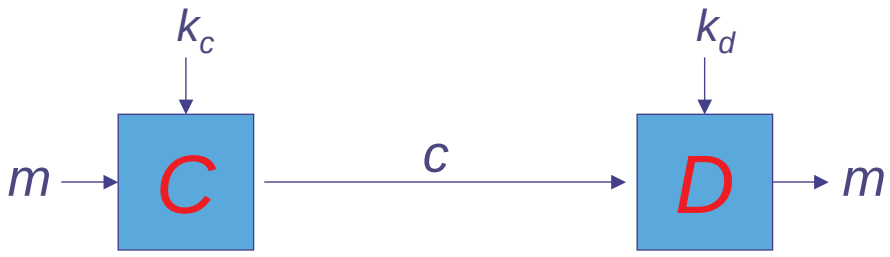
  $\Rightarrow$ estimations pour la factorisation donnent une borne inférieure

Le chiffrement asymétrique

Le Chiffrement Asymétrique

Algorithme de chiffrement, C

k_c : clé de chiffrement (publique)



Algorithme de déchiffrement, D

k_d : clé de déchiffrement (privée)

Chiffrement RSA (1978)

- $n=pq$: module **public**
- e : exposant **public**
- $d=e^{-1} \bmod \varphi(n)$: exposant **secret**

$$\mathrm{C}(m) = m^e \bmod n \rightarrow c$$
$$\mathrm{D}(c) = c^d = \left(m^e\right)^d = m \bmod n$$

Propriétés de RSA

n, e **publics**, puis d **secret**

Efficacité

$$C(m) = m^e \bmod n$$ $3|e|/2$ mult mod n

$$D(c) = c^d \bmod n = g(f(c)^d)$$ $3|n|/8$ mult mod n

Sécurité : Le calcul de racines e-ièmes semble nécessiter d

$\Rightarrow e\,d - 1$ est un multiple de $\varphi(n)$

$\Rightarrow$ factorisation de n

Chiffrement de Rabin (1978)

- $n=pq$: module **public**
- B : élément **public**
- p,q : clé **privée**

public $C(m) = m(m+B) \bmod n \rightarrow c$

secret $D(c) \in \left\{ \sqrt{c + \frac{B^2}{4}} - \frac{B}{2} \bmod n \right\}$

Propriétés de Rabin

n, B **publics**, puis p, q **secrets**

Efficacité
- Chiffrement : 1 mult $\bmod n$
- Déchiffrement : $3|n|/8$ mult $\bmod n$

Sécurité : Le calcul de racines carrées
$\Rightarrow$ factorisation de n

Échange de clé Diffie-Hellman (1976)

$q \mid p$-1 et g : éléments **communs**

Alice choisit x et publie $X=g^x$
Bob choisit y et publie $Y=g^y$

Secret commun $K = X^y = Y^x = g^{xy}$

Chiffrement El Gamal (1985)

- g : générateur **commun**
- x : clé **privée**
- $y=g^x$: clé **publique**

public $\mathrm{C}(m) = (g^a, y^a m) \rightarrow (c,d)$

secret $\mathrm{D}(c,d) = d / c^x$

Inversion de El Gamal

g : générateur **commun**

x : clé **privée** $y=g^x$: clé **publique**

Soit $\mathcal{A}$ un algorithme qui inverse le chiffrement El Gamal : $\mathcal{A}(y,c,d) \rightarrow m$

Soient $A=g^a$ et $B=g^b$ puis d aléatoire

$\mathcal{A}(A,B,d) \rightarrow m$ alors DH$(A,B)=d/m$

Inversion EG = DH

Les preuves de sécurité

1. Introduction
2. Les hypothèses algorithmiques
3. Le chiffrement asymétrique
4. Les preuves de sécurité
5. La signature
6. Conclusion

Protocole prouvé sûr

Pour prouver la sécurité d'un protocole cryptographique, on doit

- préciser les notions de sécurité à garantir
- préciser les hypothèses calculatoires
- décrire un protocole
- présenter une réduction :
  un attaquant permet de contredire les hypothèses

Notions de sécurité

En fonction des besoins, on définit

- les objectifs de l'adversaire
- les moyens,
  soit les informations mises à sa disposition.

Hypothèses calculatoires

Les hypothèses sont

- Une fonction est à sens-unique
- Une fonction est à sens-unique à trappe
- Tel problème est insoluble
  en un temps t donné

Protocole cryptographique

On décrit un protocole
ex. basé sur l'hypothèse RSA

Il est impossible d'extraire des racines modulaires, pour des modules de 1024 bits, en moins de 2^{80} opérations

- $n = pq$, e exposant premier avec $\varphi(n)$
- (n, e) : clé **publique**
- $d = e^{-1} \bmod \varphi(n)$: clé **privée**

$$\mathbf{E}(m) = m^e \bmod n \qquad \mathbf{D}(c) = c^d \bmod n$$

Sécurité par réductions

Réduction d'un problème difficile **P**
à une attaque *Atk* :

- Si l'attaquant A parvient à son but
  alors A peut être utilisé pour résoudre **P**

P insoluble $\Rightarrow$ schéma incassable

Efficacité de la réduction :

- Théorie de la complexité : polynomiale
  $\Rightarrow$ sécurité asymptotique
- Sécurité exacte : réduction précise

Sécurité pratique

Soit un attaquant A qui parvient à son but en temps t, et une réduction qui permet de résoudre **P**, avec A, en temps $t'=f(t)$

- Théorie de la complexité : f polynomiale
- Sécurité exacte : f explicite
- Sécurité pratique : f petite (linéaire)

Ex: $t' = 2t \Rightarrow$ schéma incassable en moins de 2^{79} opérations

Divers modèles de sécurité

Les réductions efficaces sont rares
$\Rightarrow$ on fait des hypothèses idéales :

- Fonctions de hachage aléatoires
  random oracle model
- Chiffrement par blocs parfait
  ideal cipher model
- Attaque indépendante du groupe
  generic model

Chiffrement

- Sécurité parfaite :

  le chiffré et les données publiques
  ne révèlent aucune information
  sur le message clair,
  sauf éventuellement la taille

  Au sens de la théorie de l'information
  $\Rightarrow$ impossible

Buts de l'adversaire

- Non-inversibilité (OW - One-Wayness)

  Sans la clé privée, il est calculatoirement impossible de retrouver le message clair

- Sécurité sémantique

  (IND - Indistinguishability):

  Aucun adversaire polynomial ne peut apprendre d'information sur le message clair à partir du chiffré et des données publiques (sauf éventuellement la taille)

Attaques possibles

- à clairs choisis
  (CPA - Chosen-Plaintext Attacks)

  Dans l'environnement à clé publique, l'adversaire peut chiffrer tout message de son choix, grâce à la clé publique
- à chiffrés choisis
  (CCA - Chosen-Ciphertext Attacks)

  L'adversaire à accès à un oracle de déchiffrement, qui déchiffre tout chiffré de son choix
  (sauf le challenge)

Notions de sécurité

- OW-CPA: *(sécurité minimale)*
- CC Attaques
  parfois faciles à mettre en œuvre
  $\Rightarrow$ les rendre inutiles
- Espace des messages clairs
  souvent limité : oui/non -- achat/vente
  $\Rightarrow$ IND nécessaire

IND-CCA niveau de sécurité souhaitable

Principaux niveaux de sécurité

- OW-CPA: *(le plus faible)*

$$\Pr_{m,r}\left[A(c) = m \middle| c = \mathbf{E}(m;r)\right]$$

= Succ négligeable

- IND-CCA: *(le plus fort* - BDPR C '98*)*

$$2\Pr_{r,b}\left[A_2^{\mathbf{D}}(m_0,m_1,c,s) = b \middle| \begin{array}{c}(m_0,m_1,s) \leftarrow A_1^{\mathbf{D}}(k_e)\\ c \leftarrow \mathbf{E}(m_b,r)\end{array}\right] - 1$$

= Adv négligeable

Ex. I : Chiffrement RSA

- $n = pq$, produit de deux grands premiers
- e, exposant premier avec $\varphi(n) = (p\text{-}1)(q\text{-}1)$
- n, e : clé **publique**
- $d = e^{-1} \bmod \varphi(n)$: clé **privée**

$$\mathbf{E}(m) = m^e \bmod n \qquad \mathbf{D}(c) = c^d \bmod n$$

OW-CPA = problème RSA

Ex. II : Chiffrement El Gamal

- **G** = ($<g>$, $\times$) groupe d'ordre q
- x : clé **privée**
- $y=g^x$: clé **publique**

$$\mathbf{E}(m) = (g^a, y^a m) \rightarrow (c,d) \qquad \mathbf{D}(c,d) = d / c^x$$

OW-CPA = problème DH

El Gamal : OW-CPA

$$\mathbf{E}(m) = (g^a, y^a m) \rightarrow (c,d) \qquad \mathbf{D}(c,d) = d / c^x$$

G = ($<g>$, $\times$) et (A,B)
en entrée de B

$$\varepsilon = \Pr_{m,a}\left[\mathcal{A}(c') = m \,|\, c' = \mathbf{E}(m;a)\right]$$

- $y \leftarrow A$ et $c \leftarrow B$
- Valeur aléatoire d : A$(c,d) \rightarrow m$
- Retourne d/m

Si m est correct, DH$(A,B)=d/m$
Succ^{DH} (B) = $\text{Succ}^{\text{OW-CPA}}$(A) = ε

El Gamal : IND-CPA

$$\varepsilon = 2 \Pr_{a,b}\left[\mathcal{A}_2(m_0, m_1, c', s) = b \;\middle|\; \begin{array}{c}(m_0, m_1, s) \leftarrow \mathcal{A}_1(k_e) \\ c' \leftarrow \mathbf{E}(m_b, a)\end{array} \right] - 1$$

$\mathbf{G} = (\langle g \rangle, \times)$ et (A, B, C) en entrée de B

- $y \leftarrow A$ et $c \leftarrow B$: $\mathsf{A}_1(y) \rightarrow (m_0, m_1)$
- $b \in \{0,1\}$ et $d \leftarrow C\ m_b$: $\mathsf{A}_2(c,d) \rightarrow b'$
- retourne $(b=b')$

Supposons que $m_0, m_1 \in \mathbf{G}$

- Si C=DH(A,B), $\Pr[b=b'] = \mathrm{Succ}(\mathsf{A}) = (\varepsilon+1)/2$
- Si $C \neq$DH(A,B), $\Pr[b=b'] = 1/2$

Les problèmes Diffie-Hellman

$\mathbf{G} \subseteq \mathbf{Z}_p^*$ groupe cyclique d'ordre q
g générateur : $\mathbf{G} = \langle g \rangle$

Calculatoire :

- Étant donnés $A=g^a$ et $B=g^b$
- Calculer DH$(A,B) = C=g^{ab}$

Décisionnel :

- Étant donnés $A=g^a$, $B=g^b$ et $C=g^c$
- Décider si DH$(A,B) = C$ (ou $c=ab \bmod q$)

Attaques à chiffrés choisis

Pour résister aux attaques à chiffrés choisis, l'oracle de déchiffrement ne doit apporter aucune information utile à l'adversaire

- preuve non-interactive zero-knowledge de la connaissance du clair
- plaintext-awareness (dans le model de l'oracle aléatoire)

Schémas IND-CCA

- Rackoff-Simon (1991), définition de la notion CCA + exemple (pas pratique)
- Cramer-Shoup (1998), premier exemple pratique : IND-CCA = DDH

Mais pas très efficace

$\Rightarrow$ modèle de l'oracle aléatoire

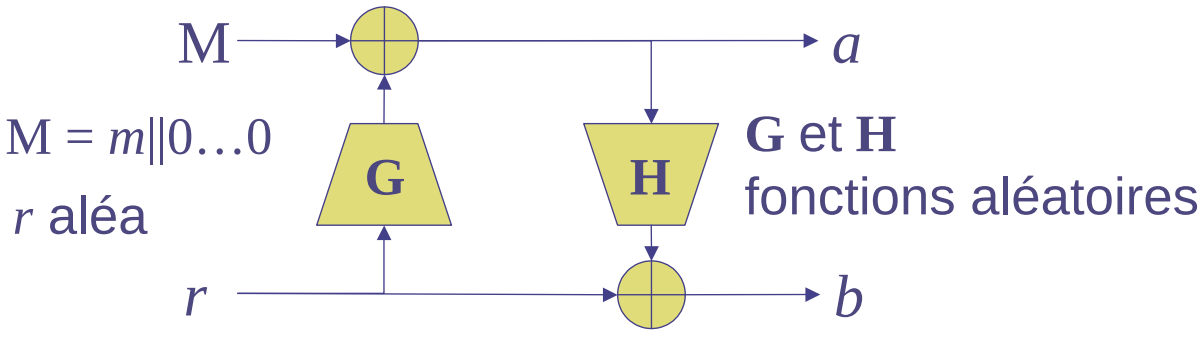
Conversions génériques

- Toute fonction injective à sens-unique à trappe = chiffrement OW-CPA
- Mais OW-CPA insuffisant
- Comment atteindre IND-CCA ?

  $\Rightarrow$ conversions génériques
  de OW-CPA en IND-CCA

$(\mathcal{E},\mathcal{D})$ est supposé un chiffrement faible et on fabrique un chiffrement fort (**E**,**D**)

OAEP (Bellare-Rogaway EC '94)



E(m): Calculer a,b puis retourner $\mathcal{E}(a||b)$
D(c): Calculer $a||b = \mathcal{D}(c)$
inverser OAEP, et retourner m
(*si la redondance est satisfaite*)

OAEP (suite)

Il fournit une conversion « optimale »
de toute ***permutation*** *à sens-unique sur un domaine partiel, à trappe*
en un chiffrement IND-CCA

Efficacité : optimale (+ 2 hachés)

Taille : optimale

Application : RSA (la seule permutation !)

RSA-OAEP sécurité (FOPS '01)

$$\mathbf{E}(\mathrm{M}) = \left(a = \mathrm{M} \oplus \mathrm{G}(r) \,\|\, b = r \oplus \mathrm{H}(a)\right)^e \bmod n \rightarrow c$$

Deviner 1 bit de M $\Leftrightarrow$ deviner r $\Leftrightarrow$ deviner a
$\Leftrightarrow$ deviner (a,b) $\Leftrightarrow$ inverser RSA

$$\mathbf{D}(c) = c^d \bmod n \rightarrow (a, b)$$
$r = \mathrm{H}(a) \oplus b$ et $\mathrm{M} = a \oplus \mathrm{G}(r)$
si $\mathrm{M} = m\|0\ldots0$ alors $m = x$ sinon « refus »

Chiffré valide $\Leftrightarrow$ $\mathrm{H}(a)$ et $\mathrm{G}(r)$ $\Leftrightarrow$ clair connu
Plaintext-awareness

RSA-OAEP - norme

OAEP : premier padding
avec « preuve » de sécurité

- Mais preuve originale de BR ‘94 incomplète
- Preuve complète récente FOPS ‘01

Heureusement car retenu par
RSA PKCS, SET, IETF, IEEE, ISO, …

Cependant, mauvaise réduction : quadratique
$\Rightarrow$ sécurité avec 1024 bits en 2^{40} !

Autres conversions

OAEP ne convient que pour des permutations : peu de candidats !

- Fujisaki-Okamoto (PKC ‘99):
  IND-CPA $\longrightarrow$ IND-CCA
- Fujisaki-Okamoto (Crypto ‘99)
- Pointcheval (PKC ‘00):
  OW-CPA $\longrightarrow$ IND-CCA

mais déchiffrement non-optimal

Nouvelle conversion: REACT
(Okamoto-Pointcheval RSA '01)

E$(m ; r\|\|s)=$ $a = \mathcal{E}(r ; s)$ avec $r \in \mathcal{M}$, $s \in \mathcal{R}$

$b = k \oplus m$ où $k = \mathrm{G}(r)$

$c = \mathrm{H}(m,r,a,b)$

D(a,b,c): Calculer $r = \mathcal{D}(a)$ et $k = \mathrm{G}(r)$

extraire $m = k \oplus b$

si $c = \mathrm{H}(m,r,a,b)$ et $r \in \mathcal{M}$

alors retourner m sinon « refus »

Conversion de tout chiffrement ***OW-PCA*** en chiffrement IND-CCA

Nouvelle attaque : PCA

Plaintext Checking Attack : l'adversaire

- peut obtenir le chiffré de tout message de son choix (en le chiffrant lui-même)
- a de plus accès à un PC-oracle qui, sur la paire (m,c) répond si c chiffre m, ou non

Remarque: IND-PCA impossible

$\Rightarrow$ mais seul OW-PCA nous intéresse

Résultat de sécurité

$$G : \mathcal{M} \rightarrow \{0,1\}^{\ell_G} \qquad H : \{0,1\}^* \rightarrow \{0,1\}^{\ell_H}$$

Si un adversaire A contre IND-CCA
obtient un avantage Adv^{A}
après q_G, q_H et $q_{\mathbf{D}}$ questions
à G, H et **D** resp.
alors on peut casser la OW-PCA
de $(\mathcal{E},\mathcal{D})$ avec probabilité

$$\frac{\mathrm{Adv}^{A}}{2} - \frac{q_{\mathbf{D}}}{2^{\ell_H}}$$

Preuve : IND-CPA

Soit (a,b,c) tel que $a = \mathcal{E}(r\,;\,s)$,
$k = G(r)$, $b = k \oplus m_d$, $c = H(m_d,r,a,b)$

Pour deviner le bit d, un adversaire doit avoir demandé

- r à G pour obtenir k (et vérifier b)
- (m_0,r,a,b) ou (m_1,r,a,b) à H (et vérifier c)

à cause du caractère imprédictible de G et H

Preuve IND-CPA (suite)

Probabilité pour que r ($= \mathcal{D}(a)$) ait été demandé à G ou H supérieure à $\mathrm{Adv}^{\mathrm{A}}/2$

On trouve le bon, grâce au PC-oracle, dans la liste des questions posées à G et H

$\Rightarrow q_{\mathrm{G}} + q_{\mathrm{H}}$ questions au PC-oracle

Plaintext-awareness

(a,b,c) chiffré valide $\Rightarrow$ on a

- demandé $H(m,r,a,b)$ pour obtenir un c valide
- deviné c, mais avec probabilité $1/2^{\ell_H}$

Plaintext-extractor

Lors d'une question (a,b,c) posée par l'attaquant à l'oracle de déchiffrement, on regarde dans la liste des questions (m,r,a,b) posées à H telles que

$$\mathrm{H}(m,r,a,b) = c$$

puis on vérifie si

- $r = \mathcal{D}(a)$ (grâce au PC-oracle)
- $b = k \oplus m$ pour $k = \mathrm{G}(r)$

Extraction correcte, sauf avec probabilité $1 - 1/2^{\ell_H}$

IND-CCA

Après $q_{\mathbf{D}}$ questions à l'oracle de déchiffrement

- toutes les simulations de déchiffrement ont réussi, avec probabilité

$$(1 - 1/2^{\ell_H})^{q_{\mathbf{D}}} \geq 1 - q_{\mathbf{D}}/2^{\ell_H}$$

- r a été demandé à G ou à H (et extrait grâce au PC-oracle) avec probabilité supérieure à

$$\frac{\mathrm{Adv}^{A}}{2} - \frac{q_{\mathbf{D}}}{2^{\ell_H}}$$

Les problèmes Diffie-Hellman

- Calculatoire
  - Étant donnés $A=g^a$ et $B=g^b$
  - Calculer DH(A,B) = $C=g^{ab}$
- Décisionnel
  - Étant donnés A, B et C
  - Décider si C = DH(A,B)
- Gap

  Résoudre le problème calculatoire, avec un accès à un oracle de décision

Le Gap-Diffie-Hellman
(Okamoto-Pointcheval PKC '01)

Le CDH est considéré difficile
(hypothèse Diffie-Hellman)

GDH facile $\Rightarrow$ DDH = CDH

DDH facile $\Rightarrow$ GDH = CDH

CDH est considéré
beaucoup plus difficile que DDH
$\Rightarrow$ GDH difficile

REACT – El Gamal

- **G** un groupe, et g d'ordre q
- G et H : fonctions de hachage
- **E**, **D**: chiffrement symétrique

E(m): $a \leftarrow_R \mathbf{Z}_q$, $R \leftarrow_R \mathbf{G}$
$A \leftarrow g^a$, $A' \leftarrow R\, y^a$
$k \leftarrow \mathrm{G}(R)$, $B \leftarrow \mathbf{E}_k(m)$,
$C \leftarrow \mathrm{H}(R, m, A, A', B)$

x : clé **privée**
$y=g^x$: clé **publique**

$\longrightarrow (A, A', B, C)$

D(A, A', B, C): $R \leftarrow A'/A^x$,
$k \leftarrow \mathrm{G}(R)$, $m \leftarrow \mathbf{D}_k(B)$,
vérifier $C = \mathrm{H}(R, m, A, A', B)$

Sécurité pratique

Soit A un attaquant en temps t,
qui pose q_{G}, q_{H} et $q_{\mathbf{D}}$ questions à G, H
et l'oracle de déchiffrement

Alors

$$\mathrm{Adv}^{\mathrm{A}}(t) \leq 2\, \mathrm{Succ}^{\mathrm{GDH}}(t') + 2\, \mathrm{Adv}^{\mathrm{E}}(t') + 2\, q_{\mathbf{D}}/2^{\ell_H}$$

avec

$$t' \leq t + (q_{\mathrm{G}} + q_{\mathrm{H}})\, T_{\mathrm{DDH}}$$

La signature

1. Introduction
2. Les hypothèses algorithmiques
3. Le chiffrement asymétrique
4. Les preuves de sécurité
5. La signature
6. Conclusion

Authentification

Deux scénarios classiques nécessitent de l'authentification :

- interactif : on prouve son identité à un interlocuteur
- non-interactif : on attache, à un message, une preuve de son origine

Signature

Si cette preuve peut convaincre un tiers :
non-répudiation

$\Rightarrow$ **signature**

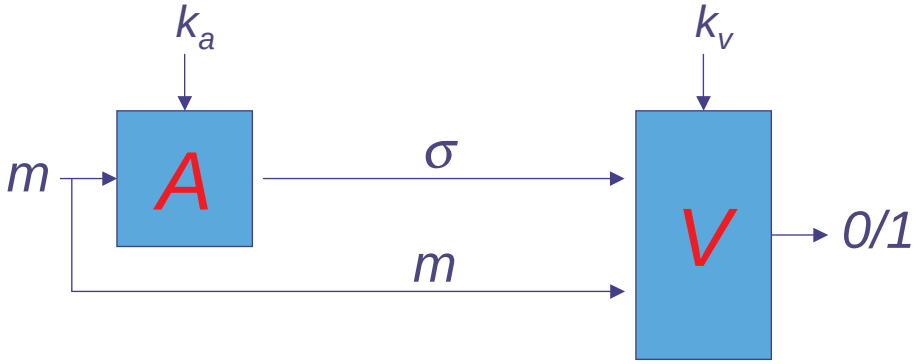
Schéma de signature

- Algorithme de signature :

  (A, k_a) : secret

- Algorithme de vérification :

  (V, k_v) : public



Sécurité

- Une signature produite avec k_a est toujours valide (acceptée) :

$$\forall m, \text{si } \sigma = A(k_a, m) \text{ alors } V(\sigma, m, k_v) = 1$$

- Falsification universelle :
  être capable d'authentifier tout message sans k_a
- Falsification existentielle :
  être capable d'authentifier un message sans k_a

Attaques

Attaque sans message :
l'attaquant ne connaît que la clé publique
(même pas de signatures
$\Rightarrow$ très faible)

Attaque à messages connus :
l'attaquant connaît de plus
des couples message-signature

Attaque à messages choisis adaptative :
l'attaquant a accès à un oracle de signature

Signature sûre

Un schéma de signature est dit SÛR
s'il garantit l'impossibilité de
falsifications existentielles
même selon des attaques
à messages choisis adaptatives

Une telle signature garantit :

- l'identité de l'expéditeur
- la non-répudiation:
  l'expéditeur ne pourra pas renier
  avoir émis cette signature

Permutations à sens-unique à trappe

Soient C une permutation à sens-unique
et D son inverse (avec trappe)

ex : $C(x) = x^e \bmod n$ et $D(y)=y^d \bmod n$

- C public et D secret

$$D(C(x)) = C(D(x)) = x$$

- Soit m un message :

$A(m) = \sigma = D(m)$ secret
$V(\sigma,m) = (C(\sigma) = m)$ public

Signature RSA

- $n=pq$: module **public**
- e : exposant **public**
- $d=e^{-1} \bmod \varphi(n)$: exposant **secret**

$$A(m,d)=m^d \bmod n \rightarrow \sigma$$

$$V(\sigma,m,e)=\left(m \stackrel{?}{=} \sigma^e \bmod n \right)$$

Sécurité

C public et D secret

$$D(C(x)) = C(D(x)) = x$$

- Calculer σ pour tout m :
  calcul de $D(m)$ pour tout m
  $\Rightarrow$ falsification universelle = D
- Falsification existentielle :
  Soit σ quelconque et $m = C(\sigma)$
  $V(\sigma,m) = (C(\sigma) = m) = 1$
  (m,σ) paire valide sans D !

Full-Domain Hash

C public et D secret

$$D(C(x)) = C(D(x)) = x$$

h : fonction de hachage « aléatoire »

$$\{0,1\}^* \rightarrow \mathrm{Im}(C)$$

$$A(m) = D(h(m)) \rightarrow \sigma$$

$$V(\sigma, m) = \left(h(m) \stackrel{?}{=} C(\sigma) \right)$$

Intérêt de FDH-RSA

- Efficacité : même si le message à signer est long,
  - un seul bloc à signer
  - $|\sigma| = |n| = 1024$ bits : overhead fixe
- Sécurité : si h aléatoire,
  une falsification existentielle
  est équivalente au problème RSA

  il est aisé de trouver $(h(m), \sigma)$,
  mais ensuite il faut remonter à m :
  inverser h

Hypothèses de complexité

- Chiffrement asymétrique :
  besoin de fonctions à sens-unique
  avec une trappe
  peu de candidats : RSA, Diffie-Hellman
- Signature numérique :
  une fonction à sens-unique suffit
  pas besoin de trappe
  $\Rightarrow$ tout problème difficile :
  problèmes NP-complets, RSA, LD

Signature El Gamal (1985)

p et g : éléments **communs**

x : clé **privée** $y=g^x$: clé **publique**

Signature de m :
choisir $k \in \mathbf{Z}_{p-1}$ inversible
calculer $r=g^k \bmod p$
et $s=(m-xr)\times k^{-1} \bmod p-1$

$\sigma = (r,s)$

Vérification de (m,σ) :
$(g^{xr}\, g^{k(m-xr)k^{-1}} =)$ $y^r\, r^s = g^m \bmod p$

Falsification existentielle

Équation de vérification :

$$y^r r^s = g^m \bmod p$$

Posons $r = y^a g^b \bmod p$

Alors $y^r r^s = y^r y^{as} g^{bs} = g^m \bmod p$

$r+as = 0 \bmod p\text{-}1$

$bs = m \bmod p\text{-}1$

Soit $s = -r/a \bmod p\text{-}1$ et $m = bs \bmod p\text{-}1$

Signature Schnorr (1989)

$q \mid p\text{-}1$ et g : éléments **communs**

x : clé **privée** $y=g^x$: clé **publique**

Signature de m :

choisir $k \in \mathbf{Z}_q$ et calculer $r=g^k \bmod p$

ainsi que $e=h(m,r)$

et $s = k\text{-}xe \bmod q$

$\sigma = (e,s)$

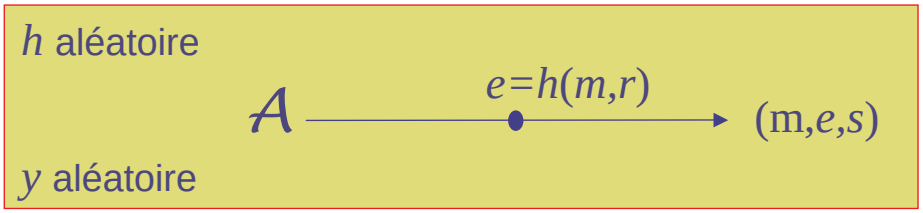
Vérification de (m,σ) :

$u = g^s y^e \ (= g^{k-xe} g^{xe} \bmod p)$ tester $e=h(m,u)$?

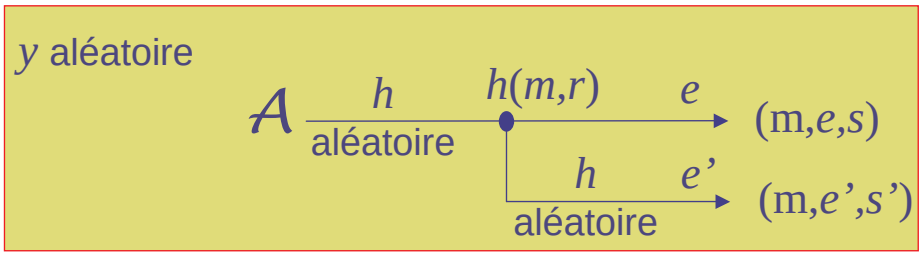
Preuve de sécurité (PS '96)

Falsification existentielle
selon une attaque à messages
choisis adaptative
dans le modèle de l'oracle aléatoire
= résolution du logarithme discret
(extraction de la clé privée)

Lemme de bifurcation



prob. ε



prob. ε

prob. ε/q_h

Après 2 exécutions, avec probabilité ε^2/q_h :

$$g^s y^e = r = g^{s'} y^{e'} \Rightarrow g^{s-s'} = y^{e'-e}$$

Posons $\alpha = (s-s')/(e'-e) \bmod q \Rightarrow y=g^\alpha$

Digital Signature Algorithm
Norme américaine (1994)

Variante de Schnorr : (p,q,g) et (x,y)

Signature de m :

choisir $k \in \mathbf{Z}_q$ inversible

et calculer $r=(g^k \bmod p) \bmod q$

ainsi que $e=\mathrm{SHA}(m)$ et $s = (e+xr)\times k^{-1} \bmod q$

$\sigma = (r,s)$

Vérification de (m,σ) :

$u = \mathrm{SHA}(m)\, s^{-1} \bmod q$ et $v = r\, s^{-1} \bmod q$

tester si $r = (g^u\, y^v \bmod p) \bmod q$?

Comparaison

- El Gamal : Pas sûr et coûteux
- Schnorr :

  Avantages

  – efficace : réduction modulo q
  suppression de l'inversion

  – signature courte : 240 bits

  – sûr : $e = h(m,r)$

  Inconvénients

  – breveté … par Schnorr

Comparaison

- DSA :

  Avantages

  – pas breveté (norme)

  Inconvénients

  – ~~efficace :~~ rajout d'inversions

  – ~~sûr~~ : $e=h(m)$ (et non $h(m,r)$ qui suffirait)
- ECDSA : DSA sur courbes elliptiques
  – preuve de sécurité (sous hypothèse forte)

Sans théorie des nombres

Suppressions des calculs coûteux :

- PKP : Permuted Kernel Problem
  (Shamir -1989)
- SD : Syndrome Decoding
  (Stern - 1993)
- CLE : Constraint Linear Equations
  (Stern - 1994)
- PPP : Permuted Perceptrons Problem
  (Pointcheval - 1995)

PKP (Shamir 1989)

Problème des Noyaux Permutés :

- Soit p un petit premier ($p = 251$)
  Soit $\mathbf{A}$ une matrice $m \times n$ dans $\mathbf{Z}_p$
  Soit $\mathbf{V}$ un vecteur de taille n dans $\mathbf{Z}_p$
- Existe-t-il une permutation π sur les coordonnées de $\mathbf{V}$ qui mette $\mathbf{V}_\pi$ dans le noyau de $\mathbf{A}$?

$$(\exists ? \pi)(\mathbf{V}_\pi \in \mathrm{Ker}\,\mathbf{A})$$

PKP : complexité

- PKP est un problème **NP**-complet
- Difficile à résoudre en pratique :

  $p = 251$, $m = 16$ et $n = 32$

  suffisent à le rendre insoluble
- Shamir a proposé une preuve de connaissance de π, Zero-Knowledge
  - 3 passes : 2 chances sur 3 de frauder
  - 5 passes : ~1 chance sur 2 de frauder

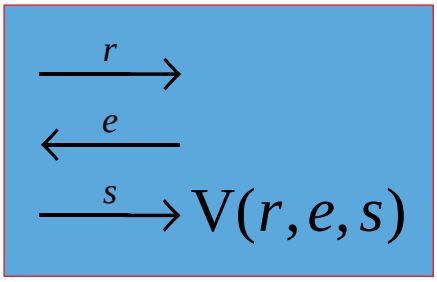
Signature

Toute preuve interactive « Zero-Knowledge » (vérifieur honnête) peut être transformée en schéma de signature sûr

(Heuristique : Fiat-Shamir 1986)

(Preuve : Pointcheval-Stern 1996)

Preuve interactive



Signature de m

Calcul de r
Calcul de $e = \mathrm{H}(m,r)$
Calcul de $s \rightarrow (r, s)$
Vérification : $\mathrm{V}(r,e,s)$

Non-répudiation

- Un schéma de signature est dit ***sûr*** si ***aucun*** attaquant ne peut produire de ***falsification existentielle***.

  $\Rightarrow$ tout couple (m,σ) valide a nécessairement été produit par l'utilisateur possédant la clé publique

Nul ne peut renier un message signé

Conclusion

1. Introduction
2. Les hypothèses algorithmiques
3. Le chiffrement asymétrique
4. Les preuves de sécurité
5. La signature
6. Conclusion

Conclusion

La sécurité prouvée se fait en 3 étapes

1. notions formelles de sécurité
2. hypothèses algorithmiques précises
3. réduction
   d'une mise en défaut d'une hypothèse
   en le cassage d'une notion de sécurité

Plus la réduction est efficace, plus la sécurité du système est proche de l'hypothèse algorithmique

$\Rightarrow$ Sécurité pratique