Cryptographie et fonctions à sens unique

Pierre Rouchon

Centre Automatique et Systèmes
Mines ParisTech
pierre.rouchon@mines-paristech.fr

Octobre 2012

Plan

Cryptographie moderne

La cryptographie moderne fondée sur les fonctions à sens unique commence en 1976 où Diffie et Hellman proposent une solution à un problème considéré alors comme insoluble :

- Alice et Bob souhaitent échanger de façon confidentielle des informations.
- Alice et Bob ne disposent pour communiquer que d'une ligne de transmission écoutée en permanence par le méchant Oscar ;
- Oscar ne peut pas se faire passer pour Alice (resp. Bob) auprès de Bob (resp. Alice).

Un exposé détaillé : l'excellent cours de Gilles Zémor de l'ENST.

Fonction à sens unique

Soit $n \in \mathbb{N}$ grand. On pose $A_n = \{1, \ldots, n\}$ et B_n deux ensembles finis indexés par n et $f_n : A_n \mapsto B_n$. f_n est à sens unique, si et seulement si, pour n devenant très grand

1. Il est facile de calculer $f_n(x)$ pour importe quel $x \in A_n$.
   f_n facile à calculer : un algorithme "rapide" pour calculer $f_n(x)$, nécessitant une quantité polynômiale de calculs en fonction du nombre de bits nécessaires pour coder $x \in A_n$, à savoir $\log_2(n)$. Si on note $C_n(x)$ le nombre d'opérations élémentaires nécessaires au calcul de $f_n(x)$, cela veut dire qu'il existe $M, \alpha > 0$ tels que pour tout $n \in \mathbb{N}$, et pour tout $x \in A_n$, $C_n(x) \leq M(\log n)^\alpha$.
2. Il est difficile pour $y \in f_n(A_n)$ de trouver un x tel que $f_n(x) = y$.
   f_n difficile à inverser : on ne connaît pas d'algorithme rapide (.i.e. de complexité polynômiale) qui permette de résoudre $f_n(x) = y$. Pour n grand, il est illusoire pour trouver un tel x d'utiliser la méthode "brutale" en testant tous les x de A_n via le calcul "rapide" de f_n (quantité exponentielle de calculs)

Stockage des mots de passe d'un compte utilisateur

Au lieu de stocker les mots de passe en clair sur un fichier même protégé du compte "système", il suffit de stocker dans un fichier l'image des mots de passe via la fonction f_n. Lorsque l'on se connecte avec son mot de passe, disons π, la machine calcule $f_n(\pi)$ et vérifie s'il est bien dans la liste qu'elle a sur son disque dur. Nous voyons qu'il est difficile même connaissant n, f_n et la liste des images par f_n des mots de passe, de remonter à ceux-ci.

Exo: Imaginer une extension qui garantit la sécurité de l'accès même si le mot de passe π est ébruité lors de la connexion (utiliser une suite du type $\pi_k = f_n(\pi_{k-1})$, $k = 1, \ldots, m$ avec m grand et supérieur au nombre total de connexion au cours de la vie d'un compte utilisateur).
Exo: Alice et Bob communiquent via internet. Ils disposent d'une même fonction à sens unique f. Alice a une pièce de monnaie. Imaginer un protocole pour qu'ils puissent jouer à pile ou face sans possibilité de fraude.

Pile ou face sur internet

Alice lance la pièce, Bob parie et Alice lui dit s'il a gagné. Problème : il ne faut pas qu'Alice puisse tricher.
Une solution reposant sur l'engagement d'Alice (commitment) : une fonction a sens unique injective connue de Bob et d'Alice, $f : A \mapsto B$, une partition $A = A_p \cup A_f$ en deux parties de même taille A_f pour face et A_p pour pile.
Le protocole (Blum) est le suivant (ordre chronologique) :

1. Alice lance la pièce : si c'est face (resp. pile) elle choisit aléatoirement $x \in A_f$ (resp. $x \in A_p$), calcule $y = f(x)$ et envoie y à Bob. Alice s'engage en communiquant le résultat y sans dévoiler x.
2. Bob déclare son choix à Alice, pile ou face.
3. Alice lui dit s'il a gagné ou perdu en communiquant à Bob x ; Bob se persuade qu'Alice n'a pas triché en vérifiant que $f(x)$ est bien égal à y.

Exponentielle modulaire

Soit p un nombre premier, a priori grand.

- L'ensemble $\mathbb{Z}_p^* = (\mathbb{Z}/p\mathbb{Z})\backslash\{0\}$, les entiers définis modulo p et différents de zéro, forme un groupe cyclique pour la multiplication : il est engendré par les puissances de certains de ses éléments, dits éléments primitifs.
- Soit $\alpha \in \mathbb{Z}^*$ primitif. Alors les gens s'accordent pour dire que la fonction f définit par

$$\begin{array}{ccc} \mathbb{Z}_p^* & \to & \mathbb{Z}_p^* \\ x & \mapsto & \alpha^x \end{array}$$

  est à sens unique. Pour le calcul de α^x on identifie $x \in \mathbb{Z}_p^*$ à l'ensemble des entiers entre 1 et $p-1$, et l'on calcul α^x modulo p.

Exo:

1. Monter que $f(p-1) = 1$ quelque soit α primitif modulo p.
2. Donner un algorithme rapide de calcul de l'exponentielle.

Logarithme discret

- Tous les algorithmes connus pour inverser cette fonction (le logarithme discret) nécessitent un temps de calcul non polynômial en $\log(p)$ et sont impraticables dès que p est un nombre de quelques centaines de bits.
- Pourquoi donc prendre un nombre premier p et un élément primitif modulo-p. Tout d'abord, p premier implique que tout entier non nul plus petit que p est inversible. Ensuite α primitif implique que l'application f est bijective : l'exponentielle modulaire f est une permutation difficile à inverser d'un ensemble à p éléments.

Exo: Soit p premier et $\alpha \in \mathbb{Z}_p^*$ primitif modulo p. On note par $\log_\alpha(y)$ l'unique x dans $\mathbb{Z}_p^*$ tel que $\alpha^x = y \mod (p)$.

1. Montrer que $\forall \beta, \gamma \in \mathbb{Z}_p^*$ et $\forall s \in \mathbb{N}$, on a $\log_\alpha(\beta^s) = s \log_\alpha(\beta) \mod (p)$ et $\log_\alpha(\beta\gamma) = \log_\alpha(\beta) + \log_\alpha(\gamma) \mod (p)$.
2. Montrer que, si l'on dispose d'un algorithme rapide pour calculer $\log_\alpha$, alors on dispose aussi d'un algorithme rapide pour calculer $\log_\beta$.

Le protocole de Diffie-Hellman

Pour communiquer de façon confidentielle, Alice et Bob vont se mettre d'accord sur un nombre secret S qui leur servira de clé à un système classique de chiffrement. En utilisant uniquement le canal public, ils doivent s'échanger la clé secrète S.

1. Alice et Bob échangent publiquement un nombre premier p et un élément primitif α modulo-p.
2. Alice choisit secrètement a et Bob b. Alice et Bob calculent chacun une exponentielle (facile), $A = \alpha^a \mod (p)$ pour Alice et $B = \alpha^b \mod (p)$ pour Bob.
3. Alice et Bob échangent publiquement A et B.
4. Alice calcule $S = B^a \mod (p)$ et Bob calcule $S = A^b \mod (p)$.

La clé secrète est alors $S = \alpha^{ab} \mod (p)$

Di-symétrie dans le chiffrement et le déchiffrement avec premier p et α primitif modulo-p.
Le destinataire Bob dispose de la clé publique (p, α, P) et la clé secrète s avec laquelle il a calculé $P = \alpha^s \mod (p)$;
Alice souhaite envoyer le message M à Bob. Alice choisit alors un nombre k au hasard et calcule les deux exponentielles suivantes :

$$A = \alpha^k \mod (p), \quad B = MP^k \mod (p).$$

Alice envoie alors à Bob A et B : (A, B) forme le message chiffré.
Pour le décodage, Bob connaissant s, calcule $A^s \mod (p)$ qui n'est autre que $P^k \mod (p)$ et obtient le message en clair via $M = B/A^s \mod (p)$.

Digital Signature Standard

DSS : Digital Signature Standard qui date de 1994 : authentification de message M envoyé en clair avec une signature publique (p, α, P).

1. Alice dispose d'un nombre premier p et de α primitif modulo-p. Elle choisie un nombre s au hasard, une fois pour toute et calcule $P = \alpha^s \mod (p)$. Elle communique de façon officielle sa signature par le triplet rendu public (p, α, P).
2. Connaissant la signature d'Alice (p, α, P), Bob reçoit un jour un message en clair M et il veut être bien sûr que c'est Alice qui lui a envoyé ce message. Pour cela Alice adjoint à M deux nombres $S = (u, v)$ qui authentifient le message et en forment une signature très difficile à imiter et en plus liée au contenu du message M.

Digital Signature Standard (suite)

- (u, v) sont construits de la façon suivante : Alice choisit au hasard un nombre k premier avec $p-1$. Elle calcule ensuite $u = \alpha^k \mod (p)$ et alors v est l'unique solution de $M = us + kv \mod (p-1)$ (k est inversible modulo-$(p-1)$).
- Ainsi Bob recevant M avec la signature $S = (u, v)$ connaissant (p, α, α^s) vérifie par une simple exponentiation que seule Alice peut avoir envoyé ce message. En effet, le calcul de α^M donne

$$\alpha^M = \alpha^{us+kv+r(p-1)} = (\alpha^s)^u (\alpha^k)^v$$

où r est un certain entier et utilisant le petit théorème de Fermat qui assure que $\alpha^{p-1} = 1 \mod (p)$ dès que p est premier et α entre 1 et $p-1$. Comme $(\alpha^s)^u = P^u$ et $(\alpha^k)^v = u^v$, Bob peut calculer P^u et u^v et s'assurer que leur produit donne bien α^M modulo-p.

Digital Signature Standard (fin)

- Pour signer le message M sans connaître s on est face à un problème difficile : trouver u et v vérifiant

$$\alpha^M = P^u u^v \mod (p)$$

  problème qui nécessite a priori le calcul d'un logarithme.
- En conclusion, tout le monde sait authentifier le message d'Alice mais seule Alice peut authentifier ses messages. Pour cela Alice utilise astucieusement l'exponentielle modulaire et le petit théorème de Fermat.

Remarque : en général, le message M que signe Alice n'est pas le message en clair directement (il faudrait pour le coder un nombre M beaucoup trop grand). On utilise alors une fonction de hachage qui fait correspondre au message en clair (de grande taille) un nombre M de quelques milliers de bits. Ce nombre M, "intriqué" au message complet, est alors signé par DSS.

Exo: Expliciter le protocole à base d'exponentielles modulaires qui permet de jouer à pile ou face sur internet tout en garantissant le fait qu'aucun des deux joueurs ne pourra tricher sans que l'autre ne le sache.

- RSA inventé par Rivest, Shamir et Adleman en 1977 : Ce système di-symétrique s'appuie sur la difficulté de factoriser un grand nombre n (plusieurs milliers de bits) qui est le produit de deux grands nombres premiers p et q : clé publique n et e inversible modulo $\varphi(n) = (p-1)(q-1)$, clé secrète (p, q).
- Chiffrement d'un message en clair défini par un entier $M \mod (n)$ se fait par la transformation suivante :

$$M \mapsto M^e \mod (n).$$

- Déchiffrement : Bob reçoit via un canal public $A = M^e$. Pour déchiffrer, il lui suffit d'élever A à la puissance d où d est l'inverse de e modulo $\varphi(n) = (p-1)(q-1)$. On sait en utilisant le théorème d'Euler-Fermat et le théorème chinois que $M^{k\varphi(n)+1} = M \mod (n)$, donc

$$A^d = (M^e)^d = M^{ed} = M \mod (n)$$

car $ed = 1 + r\varphi(n)$ pour un certain r.

RSA (fin)

- Remarquons maintenant que le grand méchant Oscar doit trouver M vérifiant

$$A = M^e \mod (n)$$

en ne connaissant que A, e et n. On ne sait pas comment faire un tel calcul en temps polynômial sans connaître $\varphi(n)$. Comme, $\varphi(n) = (p-1)(q-1)$ et $n = pq$ cela revient à connaître p et q.

- On ne sait pas à l'heure actuelle démontrer qu'il n'existe pas d'algorithme polynômial qui donne, quand c'est possible, un diviseur non trivial d'un nombre n. Le premier qui donne la démonstration de cette non existence empoche 1 Million de $ car il aura alors montré la grande conjecture de la théorie de la complexité : **P** $\neq$ **NP**.

RSA en mode signature

- En mode signature, il suffit de permuter le rôle de e et d. Ainsi, Alice choisit p et q deux grands nombres premiers et communique sa signature officielle sous la forme de $(n = pq, e)$ où e est inversible par rapport à $\varphi(n)$.
- Alice garde bien-sûr sa clé secrète d qui est l'inverse de e modulo $\varphi(n)$. Pour signer son message Alice envoie à Bob M et sa signature M^d.
- Alors Bob authentifie le message comme provenant d'Alice en vérifiant que $(M^d)^e = M \mod (n)$ qui signifie implicitement que l'expéditeur connaît un inverse de e modulo $\varphi(n)$ et donc la factorisation de n. Ce ne peut donc être qu'Alice.

Exo: Imaginer un système de monnaie électronique anonyme utilisant la fonction puissance de RSA (solution : voir la suite et aussi cours de Zémor).

Exercice : la monnaie électronique

Imaginer un système de monnaie électronique faisant intervenir trois acteurs, la banque qui émet la monnaie, l'utilisateur de la monnaie émise par la banque, le vendeur qui est payé avec la monnaie émise par la banque. Le système doit satisfaire au minimum les deux contraintes suivantes :

- Assurer un paiement anonyme.
- Impossibilité de fausse monnaie.

On supposera que,

- pour l'émission de la monnaie, l'utilisateur contacte la banque où il a un compte approvisionné.
- lors du paiement le vendeur contacte la banque qui a émis la monnaie que lui donne l'utilisateur en échange du bien qu'il achète.

Indication : utiliser la fonction puissance du RSA conjointement à une fonction à sens unique f.

La monnaie électronique (solution de l'exercice)

La banque possède $n = pq$ et e inversible modulo $\varphi(n)$, d'inverse d. Elle garde secret p, q et d. Elle ne communique que n et e. On dispose aussi d'une fonction à sens unique f de $\{1, \ldots, n-1\}$ dans lui même. Sont publics : n, e et la fonction f.

- L'utilisateur U souhaite une unité de monnaie. Il choisit aléatoirement r et x modulo n, calcule $Y = r^e f(x) \mod (n)$ et envoie uniquement Y à la banque.
- La banque calcule $Z = Y^d \mod (n)$, prélève le compte de U et communique Z à U. Celui ci divise alors Z par r pour en déduire $X = f(x)^d \mod (n)$.

L'unité de monnaie électronique est alors (x, X). Si U souhaite payer anonymement avec (x, X), le vendeur vérifie bien que $f(x) = X^e \mod (n)$ et contacte la banque pour savoir si le billet (x, X) n'a pas déjà été utilisé. La banque est incapable de remonter à l'utilisateur U (r joue le rôle de masque vis à vis de la banque).

Exo: A quoi sert la fonction à sens unique f ?

Identification et preuve sans transfert de connaissance

Le prouveur P détient le secret $s \in \{1, \ldots, p-1\}$ et est identifié par $I = \alpha^s \mod (p)$ avec p premier et α primitif modulo p. Il s'agit de convaincre le vérificateur V que P est bien celui qui connaît le logarithme de I.

- P choisit $r \mod (p-1)$ aléatoire, calcule $t = \alpha^r \mod (p)$ et envoie t à V.
- V choisit un bit aléatoire $\epsilon = 0$ ou 1 et l'envoie à P. Si $\epsilon = 0$, P renvoie $x = r \mod (p-1)$ à V ; si $\epsilon = 1$ P renvoie $x = r + s \mod (p-1)$ à V.
- V vérifie que $\alpha^x = I^\epsilon t \mod (p)$.

Après k échanges de ce type V est convaincu, avec la probabilité de $1 - 2^{-k}$ que P est bien de détenteur du logarithme discret de I. Maintenant, l'information qu'a obtenue V n'est qu'une liste d'entiers aléatoires x et leurs exponentielles $\alpha^x \mod (p)$, liste que V aurait pu construire sans passer par P.

Exo: Monnaie électronique En déduire une extension de la monnaie électronique précédente qui évite au vendeur de contacter la banque au moment de la transaction et qui interdit aussi à l'utilisateur d'utiliser plusieurs fois le même couple (x, X) sans révéler du même coup son identité.

Monnaie électronique sans contact vendeur/banque (solution de l'exo)

On reprend ce qui précède avec comme données publiques fournies par la banque émettrice : un entier RSA n, un exposant RSA e et une fonction à sens unique f. On décompose $x = (u, v)$ en deux parties (u et v sont deux chaînes de bits de même longueur formant l'écriture binaire de x). On considère en plus une autre fonction à sens unique g telle que u et v sont obtenus via g par

$$u = g(a, b), \quad v = g(a + U, c)$$

où a, b, c sont choisis au hasard (choix qui remplace le choix de x) et où U identifie celui qui possède la monnaie, l'utilisateur.
En fait une unité de monnaie électronique est caractérisée par un k-uple $(x_i, X_i)_{1 \leq i \leq k}$ où $\sqrt[e]{f(x_i)} = X_i$, $x_i = (u_i, v_i)$, $u_i = g(a_i, b_i)$, $v_i = g(a_i + U, c_i)$ avec k assez grand (quelques dizaines). L'utilisateur connaît pour chaque i, a_i, b_i, c_i. L'utilisateur paie le vendeur via le protocole suivant : le vendeur choisit aléatoirement $(\epsilon_i)_{1 \leq i \leq k}$ dans $\{0, 1\}$ et communique son choix à l'utilisateur. Pour chaque i, l'utilisateur communique au vendeur

$$(z_i^1, z_i^2, z_i^3, z_i^4) = \begin{cases} (a_i, b_i, v_i, X_i) & \text{si } \epsilon_i = 0\,; \\ (a_i + U, c_i, u_i, X_i) & \text{si } \epsilon_i = 1\,; \end{cases}$$

qui en déduit x_i et vérifie que $f(x_i) = X_i^e$ sans communiquer avec la banque.

Monnaie électronique sans contact vendeur/banque (suite)

- Si l'utilisateur utilise deux fois la même monnaie électronique $(x_i, X_i)_{1 \leq i \leq k}$ auprès de deux vendeurs différents, il dévoile du même coup son identité U avec une probabilité de $1 - 1/2^k$.
- Du dépôt à la banque de la même monnaie électronique par deux vendeurs différents, les $(z_i^1, z_i^2, z_i^3, z_i^4, \epsilon_i)$ pour le premier vendeur et les $(\bar{z}_i^1, \bar{z}_i^2, \bar{z}_i^3, \bar{z}_i^4, \bar{\epsilon}_i)$ pour le second, on n'a qu'une chance sur $1/2^k$ pour que $\forall i$, $\epsilon_i = \bar{\epsilon}_i$. Avec une probabilité de $1 - 1/2^k$ il existe un r tel que $\epsilon_r \neq \bar{\epsilon}_r$. Il est très facile à la banque de remonter à l'utilisateur U avec la différence de z_r^1 et $\bar{z}_r^1$.

Monnaie électronique sans contact vendeur/banque (fin)

La banque doit cependant vérifier que lors de la fabrication des (x_i, X_i), l'utilisateur a bien mis son identifiant U et pas celui de son meilleur ennemi. Pour cela on modifie le protocole d'émission entre la Banque et l'utilisateur U.

- L'utilisateur U choisit aléatoirement (r_i, a_i, b_i, c_i), calcule $x_i = (u_i, v_i)$ avec $u_i = g(a_i, b_i)$, $v_i = g(a_i + U, c_i)$ et transmet à la banque $2k$ entiers $Y_i = r_i^e f(x_i) \mod (n)$.
- La banque calcule les $Z_i = \sqrt[e]{Y_i} = Y_i^d \mod (n)$, prélève le compte de U et communique les Z_i à U. Celui ci divise alors Z_i par r_i pour en déduire $X_i = \sqrt[e]{f(x_i)} \mod (n)$.
- La banque vérifie que l'utilisateur n'a pas triché. Elle tire au hasard k entiers entre 1 et $2k$, demande à U pour chacun de ces k entiers i, (r_i, a_i, b_i, c_i), et vérifie que les Y_i déjà communiqués par U sont bien de la forme $r_i^e f((g(a_i, b_i), g(a_i + U, c_i))$.

A la fin de ce protocole, les k entiers restant permettent de former la monnaie électronique anonyme que U ne peut utiliser plusieurs fois sans être tôt ou tard démasqué par la banque (avec une probabilité de $1 - 1/2^k$).