Quelques mots sur la cryptographie

Exposé au Séminaire Étudiant du LSP

Djalil Chafaï
http://www.lsp.ups-tlse.fr/Chafai/

Premier avril 1999 (si si...)

Comme nous sommes le 1er avril, il faut bien que je vous fasse une blague : je parlerai très peu de maths, encore moins de probabilités et statistique... À mon grand regret.

Avertissement

Je ne connais que certains aspects restreints de la cryptographie et je ne suis en aucune manière un spécialiste du domaine. Aussi, ne croyez surtout pas que ce qui est dit ici est représentatif de la cryptographie. La mathématisation de cette discipline fait appel à des notions poussées en théorie des codes, en algorithmique, en théorie des nombres et en probabilités et statistique qui m'échappent complètement malheureusement. ;-(snif. Cela dit, si certains veulent bien éclairer ma lanterne, je suis preneur. Voilà, j'espère ne pas dire de bêtises dans la suite en tout cas...

Introduction et vocabulaire

La cryptographie est l'art du cryptage. Crypter un message, on dit aussi chiffrer un message, consiste à le transformer selon un procédé recélant un certain secret en un message inintelligible appelé cryptogramme. La connaissance du procédé inverse de décryptage ou de déchiffrement permet alors de retrouver le message d'origine à partir du cryptogramme. La cryptologie quant à elle est la discipline qui étudie les algorithmes de cryptage.

Les messages cryptés sont la plupart du temps échangés entre un emetteur et un récepteur (pas forcément humains d'ailleurs) communiquant à travers un certain canal de communication (courrier, réseaux téléinformatiques divers etc).

L'algorithme de cryptage qu'utilisent l'émetteur et le récepteur est souvent connu, cela permet une plus grande souplesse. Le secret réside alors le plus souvent en l'utilisation d'une clé secrète qui apparaît comme un paramètre spécial de l'algorithme.

Le cryptage ne sert pas seulement à rendre secret un message, il permet également d'authentifier ou signer un message, crypté ou non. Ce qui s'avère très utile dans une société de plus en plus informatisée comme la notre.

On parle aussi de code au lieu de méthode de cryptage. Casser un code consiste à retrouver le message d'origine à partir du cryptogramme. La méthode « brutale » consiste à explorer toutes les possibilités, éventuellement en procédant à des élagages astucieux. Bien évidement, l'intérêt d'un code réside en la difficulté de son inversion en terme de temps de calcul. La cryptanalyse est l'art du cassage des codes cryptographiques. La cryptanalyse « texte en clair choisi » consiste, lorsque l'on connaît l'algorithme de cryptage utilisé, à appliquer l'algorithme sur un texte en clair pour tenter de retrouver le cryptogramme.

Et si je vous racontais des histoires...

Comme le dit Andrew Tanenbaum, la cryptographie a une histoire longue et haute en couleurs. Quatre populations d'êtres humains ont utilisé et contribué au développement de la cryptographie :

les militaires, les diplomates, les auteurs de journaux intimes et les amants.

La méthode de cryptage la plus simple, celle à laquelle nous avons tous pensé un jour, consiste à remplacer chaque caractère d'un texte par son successeur dans l'alphabet. C'est la méthode César, l'empereur romain l'aurait utilisé dit-on. Par extension, on parle de code César quand l'algorithme de chiffrement consiste à permuter les symboles utilisés dans le message d'origine. Un cas particulier et le ROT13, il consiste en une translation de 13 caractères vers la droite dans l'alphabet. Il est très utilisé sur l'Internet pour masquer un texte potentiellement offensant. Il a la particularité d'être involutif : appliqué deux fois, il redonne le message d'origine (je rappelle aux étourdis que l'alphabet latin comporte 26=13+13 lettres !-).

La meilleure méthode de chiffrement est sans doute celle dont personne n'a entendu parlé, celle que vous inventez dans votre coin et que seul votre correspondant connaît. Mais il faut qu'elle soit suffisement complexe pour ne pas être dévoilée par des tests usuels comme les tests de comparaison de fréquences d'apparition de symboles à celle des langues courantes qui permettent par exemple de casser le code César assez facilement.

Je me rappelle encore avec nostalgie de la méthode de cryptage à grille utilisée par Mathias Sandorf dans le fameux roman de Jules Vernes. Vous devez vous aussi avoir moult souvenirs de polards palpitants où de sournois espions rivalisent d'ingéniosité pour communiquer secrètement avec leur semblables.

Un grand maître cryptologue s'appelait Alan Turing. Il dirigea l'équipe de scientifiques qui cassèrent le système de cryptage utilisé par les Nazis pendant la seconde guerre mondiale, la fameuse machine Enigma. Turing fut sans doute avec Babbage et Von Neumann, l'un des pères fondateurs de l'informatique.

Entropie d'un texte, code de Huffman

L'entropie est une manière de mesurer la qualité d'une méthode de cryptage. Elle mesure la densité d'information, le bruit ajouté, la non redondance ou encore l'anti signal-bruit. Une entropie basse signifie une faible densité d'information. Une entropie haute en revanche est le signe d'une haute densité d'information. Un bon système de cryptage augmente donc l'entropie. L'entropie d'un texte usuel est basse car l'« information d'un caractère » est basse : il faut beaucoup de lettres pour coder un texte, il y a beaucoup de redondances en quelque sorte. Comme nous sommes censés faire des mathématiques, voici une formule donnant l'entropie d'un système S constitué de n sous systèmes différents $s_1, \ldots, s_n$:

$$\mathbf{Ent}(S) = -\sum_{i=1}^{n} \frac{|s_i|}{|S|} \log \frac{|s_i|}{|S|}.$$

Si par exemple S est un texte constitué de lettres de l'alphabet, les sous systèmes que l'on peut considérer sont l'ensemble des « a » du texte, l'ensemble des « b » du texte etc. On peut aussi prendre comme sous systèmes les couples de caractères. Comme vous le voyez, le choix des sous systèmes influe sur l'entropie. On peut montrer facilement que le code César ne change pas l'entropie d'un texte, en d'autres termes, il n'y ajoute aucun bruit. Il s'agit donc d'une très mauvaise méthode de cryptographie.

Voici une autre formule plus couramment utilisée en informatique. Elle donne l'entropie d'un système S (un texte) constitué de n sous systèmes $s_1, \ldots, s_n$ (ensemble des « a » du texte, ensemble des « b » etc) dont les éléments (lettres) ont une probabilité d'apparition $p_1, \ldots, p_n$

$$\mathbf{Ent}(S) = \sum_{i=1}^{n} p_i \log_2 \frac{1}{p_i}.$$

Cette quantité représente en fait le nombre de bits nécessaires au codage d'une lettre du système S. Cette entropie est appelée entropie d'ordre 0 car elle est calculée sur les caractères, on pourrait définir une entropie d'ordre 1 sur les couples de caractères etc.

Code de Huffman

Il s'agit d'une méthode de codage performante d'un texte, qui permet de le compresser, c'est à dire de le coder de façon plus succincte. Elle est basée sur la fréquence d'apparition des caractères du texte, on parle alors de code statistique. Je parle de texte mais en fait on peut coder n'importe quelle série de symboles à vrai dire. L'idée est simple, elle consiste à dire qu'il suffit d'attribuer des codes de longueur différente à chaque caractère, les codes les plus longs sont alors affectés aux caractères les moins fréquents. Le seul problème est qu'il faut pouvoir relire le texte en sachant ou s'arrête chaque code. Je vous renvoie aux ouvrages de référence pour la mise en œuvre pratique et la construction des fameux arbres de Huffman. Le code de Huffman est important car il maximise l'entropie en maximisant l'« information moyenne des caractères », il permet en quelque sorte de déterminer le code optimal, au sens de l'entropie, la taille du système est réduite. Le monde des codes et de la compression de données est vaste et riche, il pourrait faire l'objet d'au moins un exposé à lui tout seul. Le code de Huffman n'est pas une méthode de cryptographie mais plutôt une technique de compression conservative assez utilisée dans la pratique.

Signatures numériques

En informatique, on a souvent besoin de savoir si un texte, ou plus généralement un fichier, a été altéré suite à un transfert (par exemple à travers un réseau). La comparaison avec la version originale n'est pas très intéressante car elle nécessite précisément la possession de cette dernière. Une idée simple consiste à associer à un texte un nombre de taille fixe appelé signature numérique du texte. Cette signature est calculée à partir du texte et possède une propriété particulière : elle change si on modifie un seul caractère du texte. De plus, sa taille ne dépend pas de la taille du texte. Les matheux que vous êtes diront: « beuh... c'est pas une bijection ». Oui, mais cela permet de repérer les petites modifications, qui forment l'essentiel des cas pratiques.

Bien évidement, tout dépend de l'algorithme utilisé pour calculer la signature. Il faut qu'il soit rapide et efficace. Le plus rapide consiste sans doute à calculer une somme de contrôle. Des algorithmes plus sophistiqués existent, comme le CRC (Cyclic Redundancy Check), le SHA, le MD2 ou le MD4. MD signifie Message Digest. Le MD5 inventé par Rivest est très couramment utilisé de nos jours. Il fournit des signatures sur 128 bits.

Cryptographie symétrique à clé unique (ou secrète)

L'idée consiste à utiliser un algorithme de chiffrement paramétré par une clé. Une clé n'est rien d'autre qu'une suite finie et assez courte de symboles. Pour chiffrer, il faut connaître la clé, idem pour pouvoir déchiffrer. La méthode repose alors sur le fait qu'il est très difficile, en terme de temps de calcul, de déchiffrer lorsque l'on ne connais pas la clé utilisée. Donc pour utiliser cette méthode, il faut se mettre d'accord sur un algorithme et surtout sur une clé que seuls l'expéditeur et le destinataire des messages connaissent. On parle d'algorithme symétrique à clé unique car la même clé sert à la fois pour le cryptage et le décryptage. On parle aussi d'algorithme à clé privée ou secrète car la clé doit rester secrète. Beaucoup d'algorithmes de ce type furent mis au point. Le code César en est sans doute l'exemple le plus trivial. La clé n'est alors rien d'autre que la permutation utilisée.

DES

Le Data Encryption Standard a été adopté par le gouvernement américain en 1977 comme méthode de cryptage standard. Il dérive d'un algorithme mis au point par IBM. La forme utilisée actuellement diffère un peu de l'originale. En réalité, plusieurs versions légèrement différentes coexistent. Il est basé sur une clé de 56 bits et un chiffrement par block de 64 bits. L'algorithme comporte plusieurs étapes de transformation qui utilisent différentes fonctions de la clé. DES est un algorithme très rapide, mais cassable par des machines pas si chères que ça. Pour déchiffrer, il suffit d'exécuter les étapes dans l'ordre inverse avec la même clé. On peut montrer que le DES augmente l'entropie

lorsque celle-ci est basse. Pour casser le DES, certains ont pensé à la loterie chinoise : équiper chaque radio et téléviseur d'une puce DES capable de réaliser par matériel 1 million de chiffrements par seconde. Je vous laisse imaginer la suite... Le triple DES, qui consiste en une triple application de DES avec deux clés différentes permet de le rendre plus robuste tout en restant compatible. Pour explorer tout l'espace des clés de 112 bits, avec un milliard de puces exécutant un milliard d'opérations par seconde, il faudrait environs 100 millions d'années...

IDEA

IDEA a été mis au point en Suisse en 1992. Il est basé sur une clé de 128 bits. Sa structure est semblable à celle du DES. Il est très robuste grâce à la largeur de la clé utilisée. Il n'existe pas à l'heure actuelle de machine capable de casser IDEA. Certaines implémentations matérielles permettent une vitesse de chiffrement/déchiffrement considérable (en connaissant la clé bien entendu).

Critique

D'autres algorithmes existent bien sûr, comme le SKIPJACK inventé en 1993 et basé sur une clé de 80 bits. Il y a aussi BLOWFISH, CRAB, FEAL, SAFER etc. La principale critique que l'on peut porter à ces algorithmes est que la sécurité du système repose entièrement sur le secret de la clé. Ce qui suppose l'existence préalable d'un canal sécurisé pour la communication de la clé entre l'émetteur et le récepteur. Ce problème est résolu pas la cryptographie asymétrique à clés publique & privée.

Cryptographie asymétrique à clés publique & privée

Diffie et Hellman, deux chercheurs à Stanford, proposèrent en 1976 une nouvelle façon de voir les choses. Imaginons que nous disposions de deux algorithmes de chiffrement $F_c(\cdot)$ et $F_d(\cdot)$ inverses l'un de l'autre associés a des clés c et d respectivement. On a donc $F_c \circ F_d = F_d \circ F_c = Id$. On suppose de plus que ces deux algorithmes sont très difficilement cassables par une méthode du type « texte en clair choisi » et qu'il est très difficile de déterminer l'une des clés en ne connaissant que l'autre.

Imaginons maintenant qu'Alice veuille envoyer un message secret M à Benoît, qu'elle n'a jamais rencontré physiquement. Les méthodes symétriques ne lui sont d'aucune utilité car elle ne peut pas communiquer une clé de façon sûre à Benoît. Et bien d'après Diffie et Hellman, il suffit que Benoît dispose des deux clés c_B et d_B associées aux deux algorithmes $F_{c_B}(\cdot)$ et $F_{d_B}(\cdot)$. Benoît rend publique la clé c_B, on dira que c_B est sa clé publique. Alice en prend connaissance et code le message M grâce à $F_{c_B}(\cdot)$. Elle envoie ensuite à Benoît le résultat $F_{c_B}(M)$. Personne d'autre que Benoît ne peut déchiffrer ce cryptogramme, pas même Alice. Benoît, à la réception du message crypté $F_{c_B}(M)$, le décrypte grâce à la fonction $F_{d_B}(\cdot)$ associée à la clé d qu'il est seul à connaître. On dit que d est sa clé privée. $F_{d_B}(F_{c_B}(M)) = M$.

Cette méthode est séduisante. Elle permet de plus la signature et l'authentification de l'expéditeur. En effet, il suffit qu'Alice possède elle aussi un coupe de clés publique et privée (c_A, d_A). Elle rend publique sa clé publique c_A pour que Benoît puisse en prendre connaissance. Ensuite, elle joint $F_{d_A}(M)$ à M et envoie le tout $F_{c_B}(F_{d_A}(M) \cup M)$ à Benoît. Benoît obtient $F_{d_A}(M) \cup M$ grâce à sa propre clé privée d_B, ce qui lui permet de prendre connaissance de M comme précédemment. Mais cette fois-ci, il peut vérifier que ce message provient bien d'Alice en utilisant la clé publique d'Alice c_A en comparant M à $F_{c_A}(F_{d_A}(M))$. Alice est la seule personne capable de joindre $F_{d_A}(M)$ au message car elle est la seule à connaître sa propre clé privée d_A.

Le talon d'Achille de ce stratagème réside dans le fait qu'un espion peut très bien s'interposer entre Alice et Benoît et permuter les clés publiques. Mais cela peut être plus ou moins surmonté par la mise en place de serveurs de clefs authentifiés. De toute manière, la méthode est très pratique, reste à trouver de tels algorithmes... Nous allons voir que certaines fonctions relativement simples mais très difficiles à inverser en terme de temps de calcul (on parle de fonctions trappes) permettent d'y parvenir.

Le sac à dos (Knapsack)

Le premier algorithme de ce type fût inventé par Merkle et Hellman en 1978. Imaginons un grand nombre d'objets de poids tous différents. Le chiffrement consiste alors à choisir secrètement les objets que l'on place dans le sac à dos. Le poids total du sac à dos ainsi que la liste des poids de tous les objets est public. Le contenu du sac à dos reste quant à lui secret. Ce système a été cassé par un certain Adi Shamir qui montra comment retrouver les objets contenus dans le sac à dos en un temps raisonnable. Comme nous allons le voir, Shamir ne s'est pas contenté de ça...

RSA

La méthode RSA utilise un chiffrement dit exponentiel. Elle due à un groupe de chercheurs du Massashusets Institute of Technology, Rivest, Shamir et Adelman, qui l'on découverte en 1978.

Comme vous êtes matheux, vous avez droit à la mathématique correspondante. Le niveau requis en algèbre est celui du DEUG A, étonnant non ? Allons-y. On choisit deux nombres premier distincts très grands p et q , typiquement de plusieurs centaines de chiffres. On pose $n = pq$. L'indicateur d'Euler $\varphi(n)$ de n vaut alors $(p-1)(q-1)$. D'après le petit théorème de Fermat-Euler, pour tout entier m premier avec n, on a $m^{(p-1)(q-1)} \equiv 1[n]$. Soit à présent un entier e compris entre 1 et $n-1$ premier avec $(p-1)(q-1)$. Il existe alors un entier f compris entre 1 et $n-1$ tel que $ef \equiv 1[(p-1)(q-1)]$. En d'autres termes, il existe un entier naturel k tel que $k(p-1)(q-1)+1 = ef$. Pour tout entier m compris entre 1 et $n-1$ et premier avec n on a alors

$$m^{ef} = m^{1+k(p-1)(q-1)} = m(m^{(p-1)(q-1)})^k \equiv m[n].$$

Les deux fonctions $(\cdot)^e \equiv [n]$ et $(\cdot)^f \equiv [n]$ sont donc inverses l'une de l'autre. Il est maintenant facile de voir comment utiliser ce résultat pour le chiffrement. Les entiers p et q sont choisis secrètement et ne doivent surtout pas être divulgués. On peut alors prendre pour clé publique (n, e) et pour clé privée (n, f) ou réciproquement, cela n'a aucune importance. Le chiffrement par la clé publique se fera par la fonction $(\cdot)^e \equiv [n]$ et le chiffrement par la clé privée se fera par $(\cdot)^f \equiv [n]$. Je rappelle que ces deux fonctions sont inverses l'une de l'autre. Elles sont entièrement déterminées par la donnée du triplet (n, e, f). Pour employer les notations du paragraphe sur Diffie et Hellman, on a $c = (n, e)$, $d = (n, f)$, $F_c(\cdot) = (\cdot)^e \equiv [n]$ et $F_d(\cdot) = (\cdot)^f \equiv [n]$.

Pour crypter un message, il suffit de le tronçonner en petits morceaux chiffrables par notre méthode. La factorisation des entiers étant très difficile, il sera très coûteux en terme de temps de calcul de retrouver les nombres premiers p et q et donc f en ne connaissant que la clé publique $(n, e) = (pq, e)$. De plus, il est très difficile d'inverser la fonction $F_c(\cdot)$ avec une méthode du type « texte en clair choisi ».

En réalité, on peut montrer que la meilleure façon de casser RSA serait de factoriser n plutôt que de faire de la cryptanalyse texte en clair choisi. La complexité (asymptotique) du meilleur algorithme de factorisation connu est alors sous-exponentielle : $\mathcal{O}(\exp(c\sqrt[3]{\log n}\,{}^{2/3}\sqrt[3]{\log\log n}))$. Ici, la taille des données est de l'ordre de $\log_2 n$. L'informatique théorique n'a toujours pas résolu le problème de classification de la complexité des algorithmes.

Mise en œuvre dans PGP

Le Sénat américain décida en 1991 de faire passer une loi (The Anti-Crime Bill) qui autorisait le gouvernement à déchiffrer toute communication chiffrée. Outré, Philip Zimmerman, un programmeur cryptophile se mit à écrire un programme de cryptage basé sur RSA, IDEA et MD5 pour la signature. Ainsi naquit Pretty Good Privacy. Le programme fût exporté clandestinement.

L'algorithme RSA est assez lent en terme de temps de calcul. PGP l'utilise uniquement pour crypter une clé aléatoire IDEA avec laquelle le message est effectivement crypté. MD5 quant à lui est utilisé pour l'authentification. C'est la signature MD5 qui est cryptée grâce à RSA.

RSA fait l'objet d'un brevet, d'où des complications supplémentaires. Zimmerman trouva quand même un compromis en utilisant RSAREF, une implémentation de RSA gratuite pour une utilisation non commerciale. De plus, la loi américaine interdit l'exportation des systèmes de cryptographie forts, ce qui a créé à Zimmerman beaucoup d'ennuis. Le code source de PGP est disponible mais n'est pas aussi libre qu'il pourrait l'être. Ce qui motiva le lancement du projet GNU Privacy Guard (GPG), un remplaçant totalement libre de PGP.

PGP est spécialement conçu pour être utilisé pour le courrier électronique et les fichiers textes. Il possède un système pratique de gestion de trousseaux de clés publiques et privées (key rings). Le programme est disponible sur un grand nombre de plateformes (MS-DOS, les Windows, les Unix, Macintosh, Amiga, Atari etc). Ses versions les plus récentes sont très conviviales. De nombreux programmes sont désormais capables de faire appel à ses services, comme par exemple Pine et Mutt sous Unix.

Épilogue

La cryptographie est en pleine explosion de nos jours. L'informatisation croissante de notre société la rend utile au plus grand nombre. Le commerce électronique par exemple a besoin des méthodes de cryptage pour l'authentification des transactions. Plus généralement, la cryptographie permet la protection de la vie privée et l'authentification des échanges sur les réseaux. Cependant, un lourd passé militaire a érigé des murs législatifs quant à son utilisation par les civils. Certains pays comme les États Unis, la France, l'Iran et quelques autres interdisent plus ou moins son utilisation. Mais croyez-vous que les mafiozis attendent que la loi leur permette de crypter des messages ? D'autant plus que certaines méthodes permettent par exemple de cacher un message dans une image sans que cela ressemble à un cryptogramme ! Ce sont sans doute toutes ces réflexions qui ont amené Lionel Jospin, notre actuel premier ministre, à annoncer la prochaine libéralisation en France de la cryptographie à clé de moins de 128 bits. Autant vous dire que les militaires et les policiers ne sont pas contents.

Pour les informaticophiles, je signale l'existence de plusieurs programmes faisant appel à des algorithmes cryptographiques pour des besoins d'authentification et de sécurité : ssh, qui remplace avantageusement telnet et ftp; ssl, qui permet un HTTP sécurisé, il existe aussi des systèmes de fichiers cryptés pour les mordus. Même les téléphones peuvent tirer partie de la cryptographie pour échapper aux écoutes...

Aux dernières nouvelles, Sarah Flannery, une jeune prodige irlandaise de seize ans aurait découvert il y a quelques temps une nouvelle méthode de cryptographie asymétrique, aussi puissante que RSA mais beaucoup plus rapide à mettre en œuvre. Elle aurait dit « c'est bien de partager ses idées avec la communauté scientifique » et « le brevet entrave ce processus ». Mais il semble qu'elle ne soit plus sûre de rien « mon travail doit être publié dans une revue scientifique, et testé avant de savoir s'il vaut de l'argent. »

Références

Ce que j'ai exposé repose sur des lectures diverses que j'ai faites ces dernières années, aussi bien sur la toile que dans les livres et magazines. J'ai aussi beaucoup appris et compris en utilisant des programmes faisant appel à de la cryptographie ou encore en assistant à des exposés, comme par exemple celui que Franz-Albert Van Den Busches a donné au Club des Utilisateurs de Linux Toulousains il y a bien longtemps maintenant. Aussi, j'aurai beaucoup de mal à présenter une liste de références digne de ce nom. En voici tout de même quelques unes.

• *Andrew Tanenbaum, Réseaux. Troisième édition, 1997. Prentice Hall. ISBN 2-7296-0643-2.*

• *Bruce Schneier, Cryptographie Appliquée. Seconde Edition, Internationnal Thomson Publishing France. ISBN 2-84180-036-9.*

• *S.Garfinkel, PGP - Pretty Good Privacy. Edition O'Reilly. ISBN 2-84177-012-5.*

• *S. Roman, Coding and information theory. 1992. Springer.*

• *Douglas Stinson, Cryptographie Théorie et pratique. Internationnal Thomson Publishing France.*

ISBN 2-84180-013-X.
Pour le petit passage sur la complexité, je me suis un peu servi des notes de Louis Goubin de Marseille (mars 1998) que m'a gentiment prêté Vincent. J'ai été influencé pour certains passages par les pages WWW sur la cryptographie de David Chatenay, qui n'existent plus maintenant malheureusement. Ce qui pose le problème de la mémoire de l'Internet, mais c'est une autre histoire... J'en ai une copie chez moi pour ceux qui seraient intéressés.