

Table des matières

Introduction	3
1 Étude sur la cryptographie à clef publique	4
1.1 Introduction	4
1.2 Définition de la cryptographie	4
1.3 Repères historiques	5
1.3.1 Objectifs de la cryptographie	8
1.4 Cryptographie conventionnelle	9
1.4.1 Gestion de clé et chiffrement conventionnel	10
1.5 Cryptographie à clé publique	10
1.5.1 Historique	10
2 Étude sur la notion de complexité	13
2.1 Introduction	13
2.2 Notions générales	13
2.2.1 La complexité	14
2.3 Complexité des algorithmes	15
2.3.1 Classification des algorithmes	15
2.3.2 L'importance de complexité algorithmique	16
2.4 Efficacité des algorithmes	18

3	Étude sur les fonctions à sens unique	19
3.1	Signatures numériques	19
3.2	La cryptographie sans secret	20
3.2.1	Fonction à sens unique ("one way function")	20
3.2.2	Fonctions à brèche secrète	21
3.2.3	Fonctions de hachage	21
3.3	Signature et fonctions de hachage	23
3.4	Quelques applications	24
3.4.1	Le principe de Diffie et Hellman	24
3.4.2	Algorithme de chiffrement d'El Gamal	25
3.4.3	L'algorithme à clé publique RSA	26
	Bibliographie	32

Introduction

Dans les années 1970, la cryptographie n'est plus seulement le domaine des militaires. Les banques, pour la sécurité de leurs transactions, sont devenues de grandes consommatrices de messages codés. Les chiffrements disponibles alors, comme le célèbre *DES*, sont sûres, eu égard aux possibilités d'attaque contemporaines. Le problème essentiel est alors la distribution des clés, ce secret que l'expéditeur et le destinataire doivent partager pour pouvoir respectivement chiffrer et déchiffrer. Les armées ont recours aux valises diplomatiques pour ces échanges, mais ceci n'est pas accessible aux civils...

En 1976, *Whitfield Diffie* et *Martin Hellman* propose une nouvelle façon de chiffrer, qui contourne cet écueil. Commençons par expliquer leur procédé de façon imagée. Un ami doit vous faire parvenir un message très important par la poste, mais vous n'avez pas confiance en votre facteur que vous soupçonnez d'ouvrir vos lettres. Comment être sûr de recevoir ce message sans qu'il soit lu ? Vous commencez par envoyer à votre ami un cadenas sans sa clé, mais en position ouverte. Celui-ci glisse alors le message dans une boîte qu'il ferme à l'aide du cadenas, puis il vous envoie cette boîte. Le facteur ne peut pas ouvrir cette boîte, puisque vous qui possédez la clé pouvez le faire.

Le but de ce mémoire est d'étudier les fonctions à sens unique en cryptographie.

Le mémoire est subdivisé en trois chapitres :

Le premier chapitre de ce mémoire est consacré à la présentation de concepts cryptologiques. On y couvre la cryptographie conventionnelle, la cryptographie à clef publique et leur objectif.

Le second chapitre porte sur la notion de complexité. On y présente des notions importantes, on donne dans ce chapitre les différents types de complexité et leurs importances.

Dans le dernier chapitre de ce mémoire, on envisage la définition de ces fonctions, et aussi nous étudions dans ce chapitre deux applications basées sur les fonctions à sens unique.

Chapitre 1

Étude sur la cryptographie à clef publique

« Lots of people working in cryptography have no deep concern with real application issues. They are trying to discover things clever enough to write papers about » *Whitfield Diffie* [9].

1.1 Introduction

Dans ce chapitre, j'introduis de façon très générale les bases de la cryptographie, en particulier la cryptographie à clé publique. Les définitions énoncées constituent la base pour explorer dans les différents chapitres.

1.2 Définition de la cryptographie

La cryptographie, du grec *kryptos* qui signifie caché, et *graphien* écriture, est l'art de transformer un message pour tenter de le rendre illisible par toute autre personne que son destinataire. Alors que la cryptographie consiste à sécuriser les données, la cryptanalyse est l'étude des informations cryptées, afin d'en découvrir le secret. La cryptanalyse clas-

sique implique une combinaison intéressante de raisonnement analytique, d'application d'outils mathématiques, de recherche de modèle, de détermination. Ces cryptanalystes sont également appelés des pirates [11].

1.3 Repères historiques

La plus part des ouvrages distinguent trois périodes dans l'histoire de la cryptologie.

L'âge artisanal part des origines : *Jules César* utilisait, semble-t-il un mécanisme de confidentialité, où chaque lettre d'un message était remplacée par celle située trois positions plus loin dans l'alphabet. La méthode se généralise en opérant une permutation quelconque de l'alphabet, et prend le nom de substitution. Une autre méthode, dite de transposition, change l'ordre des lettres. De façon générale, jusqu'au début du *vingtième* siècle, la cryptographie était affaire de substitutions et de transpositions. On opérait d'ailleurs fréquemment des substitutions non seulement sur des lettres mais sur des mots, en s'aidant d'une sorte de dictionnaire à double entrée, nommé code ou répertoire. La cryptanalyse, quant à elle, utilisait des méthodes statistiques simples, fondées principalement sur la fréquence des lettres ou des suites de deux lettres dans un texte.

L'âge technique : garde les substitutions et les permutations mais à l'aide de machines mécaniques ou électromécaniques. Les plus célèbres sont la Hagelin et l'Enigma utilisée par l'armée allemande durant *la seconde* guerre mondiale. La complexité des méthodes rendues ainsi accessibles étant plus grande, la cryptanalyse devient plus conceptuelle et a aussi recours à des machines. Pour venir à bout de l'Enigma, les Alliés réunissent à *Bletchley Park* un groupe de scientifiques, dont *Alan Turing*, inventeur des machines qui portent son nom si chères à l'informatique théorique. *Turing* parvient à réaliser une spectaculaire cryptanalyse en la réduisant à une recherche de cas suffisamment restreinte pour être menée par une machine spécialement construite à cet effet.

L'âge paradoxal couvre les *trente* dernières années. Il voit l'introduction de mécanismes donnant des réponses positives à des questions a priori hors d'atteinte telles que :

- Comment assurer un service de confidentialité sans avoir au préalable établi une convention secrète commune ?
- Comment assurer un service d'authenticité - basé sur la possession d'un secret - sans révéler la moindre information sur le secret ?

La période récente est également marquée par le développement d'une importante communauté de recherche, représentée par l'association internationale pour la recherche cryptologique (*IACR*). Cette communauté a largement transformé l'image de la cryptologie : elle a apporté une plus grande rigueur à la cryptographie en essayant de produire autant que possible, des preuves partielles de sécurité, de type mathématique. Elle a également donné un statut nouveau à la cryptanalyse.



cryptage et décryptage

Comment fonctionne la cryptographie ?

Un algorithme cryptographique est une fonction mathématique utilisée dans le processus de chiffrement et de déchiffrement. Un algorithme cryptographique fonctionne en combinaison avec une clé (un mot, un nombre, ou une phrase) Pour chiffrer le texte clair. Le même texte clair se chiffre en un texte chiffré différent si l'on utilise des clés différentes. La sécurité des données chiffrées est entièrement dépendante de deux choses : la force de l'algorithme cryptographique et le secret de la clé.

Clés

Une clé est la valeur utilisée dans un algorithme cryptographique, afin de générer un texte chiffré. Les clés sont en réalité des nombres extrêmement importants.

La taille d'une clé se mesure en bits et le nombre correspondant à une clé de 1024 bits est gigantesque. Dans la cryptographie de clé publique, plus la clé est grande, plus la sécurité du texte chiffré est élevée. Cependant, la taille de la clé publique et de la clé secrète de cryptographie conventionnelle sont complètement indépendantes. Une clé conventionnelle de 80 bits est aussi puissante qu'une clé publique de 1024 bits. De même une clé conventionnelle de 128 bits équivaut à une clé publique de 3000 bits. Encore une fois, plus la clé grande, plus elle est sécurisée, mais les algorithmes utilisés pour chaque type de cryptographie sont très différents. Autant essayer de comparer une pomme avec une orange. Même si les clés publiques et privées sont liées par une relation mathématique, il est très difficile de deviner la clé privée uniquement à partir de la clé publique.

Remarque 1.1.

Les termes "cryptage" et "crypter" sont des anglicismes, dérivés de l'anglais to encrypt, souvent employés incorrectement à la place de chiffrement et chiffrer. En toute rigueur, ces termes n'existent pas dans la langue française. Si le "cryptage" existait, il pourrait être défini comme l'inverse du décryptage, c'est -à-dire comme l'action consis-

tant à obtenir un texte chiffré à partir d'un texte en clair sans connaître la clef.

1.3.1 Objectifs de la cryptographie

Les objectifs fondamentaux de la cryptographie sont la confidentialité, l'intégrité et l'authenticité

Confidentialité : La confidentialité permet de protéger le contenu des informations sauvegardées ou transmises sur un réseau. Seules les personnes autorisées doivent pouvoir accéder aux informations ainsi protégées. Les protections utilisées peuvent être physiques ou mathématiques. Le chiffrement de l'information permet de résoudre le problème de la confidentialité : une personne souhaitant transmettre un message lui applique au préalable une fonction dite de chiffrement, et transmet le résultat au destinataire. Ce dernier retrouve le message original en utilisant une fonction de déchiffrement.

Intégrité : Le problème de l'intégrité est le contrôle du contenu : on veut pouvoir détecter toute modification, accidentelle ou intentionnelle, des données sauvegardées ou transmises. On résout ce problème à l'aide des fonctions de hachage.

Authenticité : La notion d'authenticité s'applique à la fois aux personnes, on parle dans ce cas d'identification, et aux documents, ce qui correspond à l'authentification. Pour s'identifier, un individu prouve qu'il connaît une information secrète. Une notion particulièrement élégante est celle de preuve interactive à divulgation nulle de connaissance, dans laquelle un individu (le prouveur) convainc une autorité (le vérifieur) qu'il possède une information secrète, sans révéler aucune information sur la donnée secrète elle-même. Ainsi, il n'est pas possible qu'un espion écoutant la ligne se fasse ensuite passer pour l'individu, l'espion n'ayant appris aucune information sur la donnée secrète.

L'authentification d'un document permet de prouver son caractère authentique, c'est à dire son lien avec une entité précise. On garantit ainsi l'origine de l'information contenue

dans le document. Les techniques utilisées sont les codes d'authentification de message (notés de façon usuelle *MAC* pour *Message Authentication Code*) en clé secrète, et la signature électronique en clé publique. Ces techniques assurent implicitement l'intégrité du document.

1.4 Cryptographie conventionnelle

Définition 1.1. La cryptographie traditionnelle, aussi appelée à clé secrète ou à clé symétrique est l'étude des méthodes permettant de transmettre des données de manière confidentielle. Afin de protéger un message, on lui applique une transformation qui le rend incompréhensible; c'est ce qu'on appelle le chiffrement, qui, à partir d'un texte clair, donne un texte chiffré ou cryptogramme. Dans la cryptographie conventionnelle une [seule et même] clé est utilisée à la fois pour le chiffrement et le déchiffrement. *Le data Encryption Standar (DES)* est un exemple de cryptosystème conventionnel qui est largement employée par le Gouvernement fédéral américain.

Exemple 1.2. (*Le chiffrement de César*)

Un exemple extrêmement simple de chiffrement conventionnel est un chiffre à substitution. Un chiffre remplace un morceau d'information par un autre. Le plus souvent, cela est effectué en décalant des lettres de l'alphabet. Dans le chiffre de *César* l'algorithme en un décalage de l'alphabet, et la clé est le nombre de caractères à décaler. Par exemple si nous codons le mot "secret" en utilisant une valeur de 3 pour la *clé de César*, nous décalons l'alphabet de telle sorte que la troisième lettre en descendant (*D*) commence l'alphabet. Donc en commençant avec *A B C D E F G H I G K L M N O P Q R S T U V W X Y Z*. Et en décalant le tout de 3, vous obtenez *D E F G H I J K L M N O P Q R S T U V W X Y Z A B C*, où $D = A$, $E = B$, $F = C$, et ainsi de suite. En utilisant ce schéma, le texte clair, "secret" se chiffre comme "vhfuhw". Pour permettre à quelqu'un d'autre de lire le texte chiffré, vous lui dites que la clé est 3.

Évidemment, c'est de la cryptographie excessivement faible au regard des nombres actuelles, mais bon. Cela marchait pour César, et cela illustre aussi comment fonctionne la cryptographie conventionnelle.

1.4.1 Gestion de clé et chiffrement conventionnel

Le chiffrement conventionnel a des avantages. Il est très rapide. Il est particulièrement utile pour chiffrer des données qui ne vont aller nulle part. Cependant, le chiffrement conventionnel seul en tant que moyen de transmission de données sécurisée peut être assez onéreux simplement en raison de la difficulté de la distribution sécurisée de la clé.

Pour qu'un expéditeur et un destinataire communiquent de façon sûre en utilisant un chiffrement conventionnel, ils doivent se mettre d'accord sur une clé et la garder secrète entre eux. S'ils sont dans des lieux géographiques différents, ils doivent faire confiance à un message, Bat phone, ou à un autre moyen de communication sûr pour empêcher la divulgation de la clé secrète pendant la transmission. Le problème continu avec le chiffrement conventionnel est la distribution de la clé. Comment donnez-vous la clé au destinataire sans que personne ne puisse l'intercepter ?.

1.5 Cryptographie à clé publique

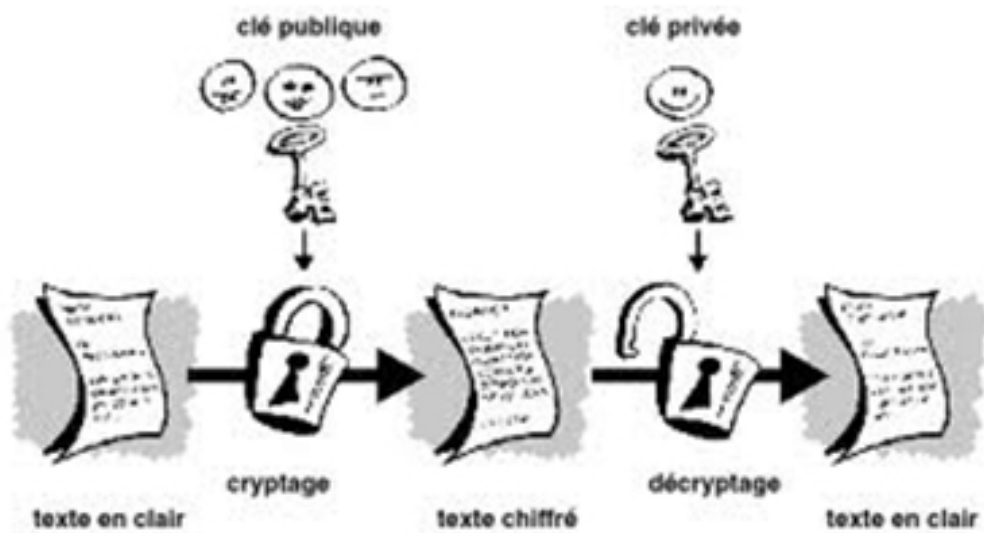
1.5.1 Historique

Le concept de cryptographie à clef publique-autre nom de la cryptographie asymétrique-est dû à *Whitfield Diffie* et à *Martin Hellman*. Il fut présenté pour la première fois à la National Computer Conférence en 1976, puis publié quelques mois plus tard dans *New Directions in Cryptography*. On considère que *Ralph Merkle* a découvert indépendamment la cryptographie à clef publique, même si ses articles furent publiés plus tard. En réalité, *James Ellis*, qui travaillait au service du chiffre britannique (*GCHQ*, *Gouverne-*

ment *Communications Headquarters*), avait eu cette idée peu avant. En 1973, *C.C. Cocks* décrit (pour le même service du chiffre) ce qu'on a appelé l'algorithme *RSA*. Enfin en 1974, *M. J. Williamson* invente un protocole d'échange de clé très proche de celui de *Diffie* et de *Hellman*. Ces découvertes n'ont été rendues publiques qu'en 1997 par le *GCHQ*.

Le but de la cryptographie à clef publique est résoudre le problème de distribution des clefs posé par la cryptographie à clef secrète. De nombreux algorithmes permettant de réaliser un cryptosystème à clef publique. Ils sont le plus souvent basés sur des problèmes mathématiques difficiles à résoudre, donc leur sécurité est conditionnée par ces problèmes, sur lesquels on a maintenant une vaste expertise. Mais, si quelqu'un trouve un jour le moyen de simplifier la résolution d'un de ces problèmes, l'algorithme correspondant s'écroulera. Avec les algorithmes asymétriques, les clefs de chiffrements et de déchiffrement sont distinctes et ne peuvent se déduire l'une de l'autre. On peut donc rendre l'une des deux publique tandis que l'autre reste privée. C'est pour quoi on parle de chiffrement à clef publique. Si la clef publique sert au chiffrement, tout le monde peut chiffrer un message, que seul le propriétaire d'utiliser la clef privée pour chiffrer. Dans ce cas, n'importe qui pourra déchiffrer, mais seul le possesseur de la clef privée peut chiffrer. Cela permet donc la signature de messages.

Le principal avantage de la cryptographie à clé publique est qu'elle permet à des gens qui n'ont pas d'accord de sécurité préalable d'échanger des messages de manière sûr. La nécessité pour l'expéditeur et le destinataire de partager des clés secrètes via un canal sûr est éliminée ; toutes les communications impliquent uniquement des clés publiques et aucune clé privée n'est jamais transmise ou partagée.



la cryptographie à clef publique

Symétrique versus Asymétrique

★ Symétrique (clé secrète)

- Alice et Bernard ont chacun une clé de la boîte aux lettres.
- Alice utilise sa clé pour déposer sa lettre dans la boîte. Bernard utilise sa clé pour récupérer la lettre.
- Alice et Bernard sont les seuls à pouvoir ouvrir la boîte aux lettres.

★Asymétrique (clé publique)

- Alice trouve l'adresse de Bernard dans un annuaire public, elle envoie sa lettre à Bernard, qui utilise sa clé secrète pour la lire.
- Tout le monde peut envoyer un message à Bernard, lui seul peut les lire.

Chapitre 2

Étude sur la notion de complexité

2.1 Introduction

La nécessité de résoudre des problèmes concrets conduit, de mettre au point modèle mathématique ; à leur tour, posent la question de l'élaboration de procédures de calcul qui peut être très complexe, mais que, en tout cas aboutir à la solution du problème étudié dans nombre fini d'étapes. Une telle procédure est appelée un algorithme, à partir du nom du mathématicien de langue arabe et astronome *al-Khawarizmi*, qui a vécu dans le 9^{ème} siècle. C'est un bon moment de poser la question principale : quel est le coût d'un algorithme, en termes d'effort et le temps nécessaire à l'exécution de ses étapes du début à la fin de l'opération nécessaire ? [7].

2.2 Notions générales

Algorithme

Ensemble des règles opératoires dont l'application permet de résoudre un problème énoncé au moyen d'un nombre fini d'opérations.

Le mot "fini" doit être ici analysé : quel serait l'intérêt d'un algorithme nécessitant un

temps infini avant de fournir la solution du problème posé ? Démontrer qu'un algorithme est "fini" n'est cependant satisfaisant que pour le théoricien : le praticien devra estimer le temps que demandera l'algorithme proposé. Il peut arriver que l'algorithme soit fini, mais que le temps de calcul soit de quelques siècles, ce qui le rend impraticable. L'estimation de ce temps est l'objet de la théorie de la complexité [3].

2.2.1 La complexité

La complexité d'un algorithme est le nombre d'opérations élémentaires (affectations, comparaisons, opérations arithmétiques) effectuées par un algorithme. Ce nombre s'exprime en fonction de la taille n des données. En informatique, le mot complexité recouvre en fait deux réalités :

★La complexité des algorithmes

C'est l'étude de l'efficacité comparée des algorithmes. On mesure ainsi le temps et aussi l'espace nécessaire à un algorithme pour résoudre un problème. Cela peut se faire de façon expérimentale ou formelle.

★La complexité des problèmes

La complexité des algorithmes a abouti à une classification des problèmes en fonction des performances des meilleurs algorithmes connus qui les résolvent. Techniquement, les ordinateurs progressent de jour en jour. Cela ne change rien à la classification précédente.

2.3 Complexité des algorithmes

2.3.1 Classification des algorithmes

Nous ne sommes généralement pas intéressés par une détermination précise de la complexité d'un algorithme, mais dans une estimation plausible de dessus. C'est, en fait, nous avons besoin afin d'évaluer le temps nécessaire pour l'exécution de l'algorithme. Pour exprimer cette estimation d'une manière efficace, c'est-à-dire sans des détails inutiles, il est courant d'utiliser la notation O , dit communément *Big – Oh*, introduite par *Paul Bachmann* en 1894. On dit que la complexité de l'algorithme est $O(f(n))$ où f est d'habitude une combinaison de polynômes, logarithmes ou exponentielles. Ceci reprend la notation mathématique classique, et signifie que le nombre d'opérations effectuées est borné par $cf(n)$, où c est une constante, lorsque n tend vers l'infini. Les algorithmes usuels peuvent être classés en un certain nombre de grandes classes de complexité.

(1) Les algorithmes sous-linéaires, dont la complexité est en général en $O(\log(n))$, C'est le cas de la recherche d'un élément dans un ensemble ordonné fini de cardinal n .

(2) Les algorithmes linéaires en complexité $O(n)$ ou en $O(n \log(n))$ sont considérés comme rapides, comme l'évaluation de la valeur d'une expression composée de n symboles.

(3) Plus lents sont les algorithmes de complexité située entre $O(n^2)$ et $O(n^3)$, c'est le cas de la multiplication des matrices et du parcours dans les graphes.

(4) Au delà, les algorithmes polynomiaux en $O(n^k)$ pour $k > 3$ sont considérés comme lents, sans parler des algorithmes exponentiels (dont la complexité est supérieure à tout polynôme en n) que l'on s'accorde à dire impraticables dès que la taille des données est supérieure à quelques dizaines d'unités.

La recherche de l'algorithme ayant la plus faible complexité, pour résoudre un problème donné, fait partie du travail régulier de l'informaticien.

2.3.2 L'importance de complexité algorithmique

L'importance de la notion de complexité algorithmique est évident. Pour ainsi, il nous permet de distinguer entre :

- * Les algorithmes que peut être exécuté dans un délai raisonnable à la mise à disposition de calcul.
- * Les algorithmes qui ne peuvent être éventuellement exécutées dans un délai raisonnable à l'aide des dispositifs de calcul disponibles, par exemple à la main, mais que peut être exécutée dans un délai raisonnable en utilisant des instruments les plus avancés.
- * Les algorithmes que théoriquement possible, car dans la pratique de leur exécution nécessite un temps soit inconditionnellement trop long ou le dépassement du temps à notre disposition, indépendamment des ressources de calcul disponibles. L'unité qui sera utilisée pour mesurer la complexité d'un algorithme est l'opération de bits.

Estimation asymptotique

Définition 2.1. Soit f et g deux fonctions à valeurs positives, défini sur une S sous ensemble de IR contenant IN . Nous dirons que f est dominée par g s'il existe des constantes k et c dans IR tel que $f(x) \leq k \cdot g(x)$ pour tous les $x \in S$, avec $x > c$.

Nous désignerons par $O(g)$ l'ensemble des fonctions dominé par g . Si $f \in O(g)$, nous disons que la fonction f est dans la classe $O(g)$ et que nous ont donné une estimation O pour la fonction f au moyen de la fonction g de référence. Remarquer que, si $f \in O(g)$ et $g \in O(h)$, alors $f \in O(h)$. Si $f \in O(g)$ et $g \in O(f)$, nous dirons que f et g ont la même commande.

- On définit symétriquement la minoration de f par g : $f = \Omega(g)$.
- On dit que f est du même ordre de grandeur que g et on écrit $f = \theta(g)$ quand $f = O(g)$ et $g = O(f)$.

Voyons quelques exemples avec $S = N$.

Exemple 2.1.

Soit $f = n^4$ et $g = n^4 + 8n^2 + 5n + 10$. Alors $f \in O(g)$, avec $k = 1$ et $c = 0$. En outre, $g \in O(f)$ aussi, avec $k = 24$ et $c = 0$.

En effet, nous avons ce que $8n^4 + 5n + 10 > 0$ pour tout $n \in N$, et ainsi de $f(n) = n^4 < 1 \times (n^4 + 8n^2 + 5n + 10) = 1 \times g(n)$, $\forall n \in N$.

En outre, il est facile de vérifier que, pour tout $n \in N$, $g(n) = n^4 + 8n^2 + 5n + 10 < n^4 + 8n^4 + 5n^4 + 10n^4 = 24n^4$; où $g(n) < 24 \times f(n)$ pour tout n .

Exemple 2.2.

Soit $f(n) = n^2 \log n$. Alors $f \in O(n^3)$.

Ordres de grandeurs

Du plus petit au plus grand : $O(1), O(\log(n)), O(n^x), O(n), O(n \log(n)), O(n^c), O(c^n), O(n!)$ avec $0 < x < 1 < c$. La notation $f = O(g)$ est scabreuse car elle ne dénote pas une égalité mais plutôt l'appartenance de f à la classe des fonctions en $O(g)$.

$O(1)$	constante
$O(\log(n))$	logarithmique
$O(n)$	linéaire
$O(n \log(n))$	$n \log(n)$
$O(n^2)$	quadratique
$O(n^3)$	cubique
$O(2^n)$	exponentiel

2.4 Efficacité des algorithmes

La conception et l'analyse d'algorithmes informatiques "efficaces" constitue une problématique centrale des sciences du traitement de l'informatique.

Par exemple, pour déterminer si un mot donné X se trouve dans un dictionnaire contenant n mots, on pourrait effectuer une recherche séquentielle. Cette recherche compare X avec le premier mot de dictionnaire, puis le deuxième et ainsi de suite jusqu'à trouver X ou, dans le cas le plus défavorable, jusqu'à ne rien trouver. Le cas le plus défavorable nécessite n comparaisons. Par ailleurs, une recherche par dichotomie compare X avec le "mot de milieu" du dictionnaire et utilise l'ordre lexicographique des mots pour décider de répéter la recherche dans la première ou la seconde moitié du dictionnaire. On répète l'opération sur un dictionnaire dont la "taille" est divisée par deux, et ainsi de suite. Dans le cas le plus défavorable (lorsque le mot n'est pas dans le dictionnaire), cette méthode nécessite $1 + \log_2 n$ comparaisons. Comme l'illustre le tableau, une recherche par dichotomie est nettement plus efficace qu'une recherche séquentielle [13].

n	$1 + \log n$
8	4
64	7
$2^{18} \approx 250000$	19

Conclusion : Un algorithme de complexité exponentielle est en pratique inutilisable. Mais les algorithmes de complexité linéaire, en $n \times \log(n)$ ou en n^2 sont considérés comme "rapides".

Chapitre 3

Étude sur les fonctions à sens unique

3.1 Signatures numériques

L'un des principaux avantages de la cryptographie à clé publique est qu'elle offre une méthode d'utilisation des signatures numériques. Celles-ci permettent au destinataire de vérifier leur authenticité, leur origine, mais également de s'assurer qu'elles sont intactes. Ainsi, les signatures numériques de clé publique garantissent l'authentification et l'intégrité des données. Elles fournissent également une fonctionnalité de non répudiation, afin d'éviter que l'expéditeur ne prétende qu'il n'a pas envoyé les informations. Ces fonctions jouent un rôle tout aussi important pour la cryptographie que la confidentialité, sinon plus. Une signature numérique a la même utilité qu'une signature manuscrite. Cependant une signature manuscrite peut être facilement imitée, alors qu'une signature numérique est pratiquement infalsifiable. De plus, elle atteste du contenu des informations, ainsi que de l'identification du signataire.

3.2 La cryptographie sans secret

3.2.1 Fonction à sens unique ("one way function")

Le système décrit précédemment comporte certains problèmes. Il est lent et produit un volume important de données (au moins le double de la taille des informations d'origine). L'ajout d'une fonction à sens unique dans le processus permet d'améliorer ce système.

Définition 3.1. Soit $n \in \mathbb{N}$ grand. On pose $A_n = \{1, \dots, n\}$ et B_n deux ensembles finis indexés par n et $f_n : A_n \rightarrow B_n$. 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 n'importe quel $x \in A_n$.

f_n **facile à calculer** : un algorithme "rapide" pour calculer $f_n(x)$, nécessitant une quantité polynomiale 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 $x \in A_n$, $C_n(x) \leq M \times (\log n)^\alpha$.

(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 polynomiale) 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) [10].

Utilisation

Les fonctions à sens unique sont utiles pour garder sous forme inaccessible des mots de passe. Par contre pour la confidentialité elles sont peu utiles car une fois m chiffré on ne sait pas déchiffrer m . Sous certaines conditions, elle est facile à inverser si on garde secret certaines informations.

3.2.2 Fonctions à brèche secrète

Une fonction à brèche secrète, comme les fonctions à sens unique, est facile à calculer mais son inverse est difficile à calculer en l'absence d'une information secrète. En général on parle de familles de fonctions à brèche secrète où un intervenant peut choisir au hasard une "brèche secrète" et construire une fonction à brèche secrète correspondant à cette information, lui permettant ainsi d'inverser cette fonction tout en sachant que calculer l'inverse sera difficile pour qui que ce soit d'autre.

Exemple 3.2.

Utilisation de *RSA*

$$n = p \times q.$$

$$f(M) = M^e \text{ mod } n.$$

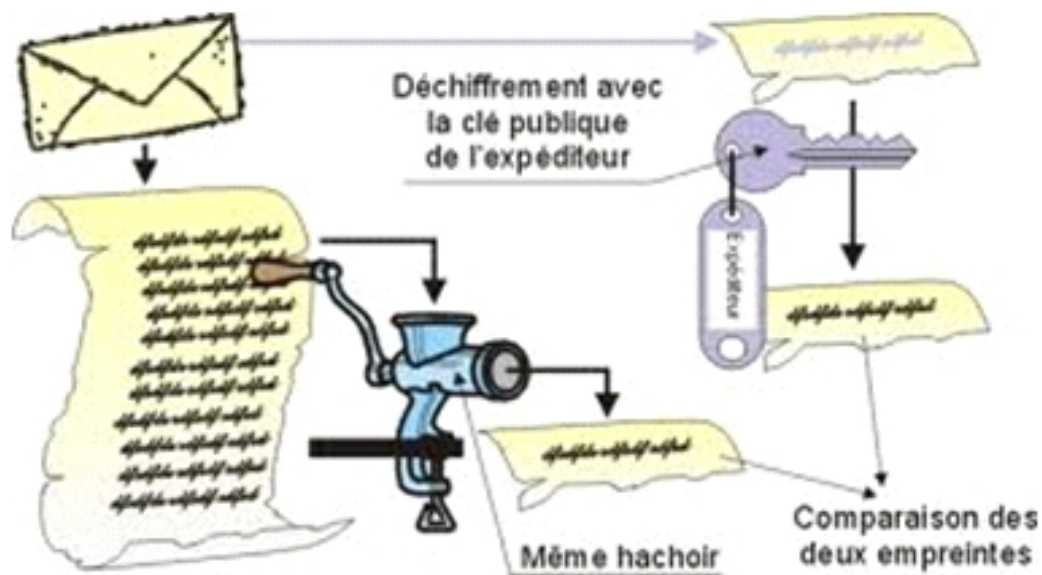
Le secret est constitué par la connaissance de p, q et e .

3.2.3 Fonctions de hachage

Est une fonction qui transforme une la chaîne binaire de longueur quelconque en une chaîne binaire de longueur fixée (généralement 128 ou 160 bits), la chaîne résultante est appelée *empreinte* (digest en anglais) ou condensé de la chaîne initiale, une fonction de hachage doit avoir certaines propriétés particulières :

- (1) Pour un x donné, il devrait être facile de calculer $h(x)$.
 - (2) Pour un y donné, il devrait être difficile de trouver une image inverse x telle que $h(x) = y$, c'est-à-dire h est à sens unique.
 - (3) Pour un x donné, il devrait être difficile de trouver un autre x' tel que $h(x) = h(x')$.
- Si c'est le cas, on dit que h est faiblement résistante aux collisions [12].

chiffrementdel'empreinteaveclaclépublique



Exemple 3.3.

On veut protéger un programme important de toute modification (le programme qui calcule le chiffrement). On mémorise le hachage du programme dans un endroit sûr (par exemple, une smart card). Avant d'utiliser le programme, on compare toujours le hachage du programme à la valeur mémorisée dans un endroit sûr.

Notion de fonction de hachage à sens unique sans clé

C'est une fonction de hachage à sens unique qui peut être calculée sans connaissance d'un secret (par n'importe qui).

Exemple 3. 4.

type : *MD4*, *MD5*, *SHA* etc

Notion de fonction de hachage à sens unique avec clé

C'est une fonction de hachage à sens unique qui ne peut être calculée que par une entité détentrice de la clé. Nombreux exemples de fonctions de hachage à sens unique avec clé déduites de méthodes de cryptographie.

Générateur d'aléa

Un générateur d'aléa est en quelque sorte le contraire d'une fonction de hachage. Alors qu'une fonction de hachage renvoie des condensés de taille fixe à partir d'entrées de taille arbitrairement grande, un générateur d'aléa est une fonction à sens unique qui prend en entrée des données de taille fixe et renvoie une suite infinie de bits.

L'entrée correspond à une petite quantité secrète, la graine, qui peut par exemple être un mot de passe, ou être produite par des mesures physiques. La sortie quant à elle est utilisée comme une source de bits aléatoires pour bon nombre de mécanismes cryptographiques, citons la génération de clefs cryptographiques ou le chiffrement asymétrique. D'un point de vue cryptographique, on souhaite que la sortie d'un générateur d'aléa soit indistinguable de bits parfaitement aléatoires.

3.3 Signature et fonctions de hachage

Pour signer des messages, on a recours, avant d'appliquer des algorithmes, à des fonctions de hachage cryptographiques. Deux raisons motivent l'utilisation de ces fonctions : la première, on l'a vu, est la lenteur des systèmes à clef publique : les messages à signer étant relativement longs ; on leur applique donc d'abord une fonction de hachage, et on signe le hache du message, et non le message lui-même.

La deuxième raison, liée à la sécurité, est d'empêcher certains types d'attaques sur les schémas de signature. Par exemple dans le cas de *RSA*, la signature du produit de deux messages est le produit des signatures de chacun d'eux.

Pour pouvoir être utilisée dans des applications cryptographiques, une fonction de hachage doit cependant satisfaire la contrainte suivante : il doit être impossible en pratique de trouver une collision, c'est-à-dire deux messages qui aient le même hache. Pour que la recherche de collisions nécessite au moins 2^{64} essais, le haché doit avoir une longueur d'au moins 128 bits.

La plupart des fonctions de hachage utilisées actuellement sont des améliorations de la fonction *MD4*. Cette dernière était fréquemment utilisée avant sa cryptanalyse en 1996. Parmi les principales fonctions de hachage, on peut citer la fonction *MD5* (*Message Digest 5*). Cette fonction produit un condensé de 128 bits. Certaines faiblesses dans sa construction ont été décelées récemment mais elles ne mettent pas directement en cause sa sécurité. Toutefois, certains préfèrent éviter son utilisation. On peut donc lui préférer le standard américain *SHA-1* (Secure Hash Algorithm), utilisé dans le schéma de signature *DSA* ou la fonction *RIPEMD-160*, qui a été conçue dans le cadre d'un projet européen. Ces deux fonctions ont également l'avantage de produire des condensés de 160 bits.

3.4 Quelques applications

3.4.1 Le principe de Diffie et Hellman

Inventé en 1976 par *Witfield Diffie* et *Martain Hellman*, ce protocole permet à deux tiers de générer un secret partagé sans avoir aucune information préalable l'un sur l'autre. Il est basé sur la cryptographie à clé publique, car il fait intervenir des valeurs publiques et des valeurs privées. Sa sécurité dépend de la difficulté de calculer des logarithmes discrets sur un corps fini. On se place dans un corps fini $k = \mathbb{F}_q$ où q est un entier grand premier ou puissance d'un nombre premier. On en détermine un noté g à des générateurs du groupe multiplicatif k^* . Le couple (\mathbb{F}_q, g) est public.

Voici le protocole que doivent suivre Alice et Bernard pour se confectionner une clé secrète.

- Alice choisit un entier a vérifiant $1 < a < q - 1$, qu'elle garde secret, et calcule sa valeur publique, $A = g^a$. Bernard fait de même et génère $1 < b < q - 1$ et $B = g^b$.
- Alice envoie A à Bernard ; Bernard envoie B à Alice.
- Alice calcule $k_{ab} = B^a$; Bernard calcule $k_{ab} = A^b$. $k_{ab} = k_{ba} = g^{ab}$ est le secret

partagé par Alice et Bernard.

Une personne qui écoute la communication connaît g , q , $A = g^a$ et $B = g^b$, ce qui ne permet pas de calculer g^{ab} . Il lui faudrait pour cela calculer le logarithme de A ou B pour retrouver a ou b [5].

Vulnérabilité du protocole de Diffie -Hillman "Nécessité d'une authentification"

Le protocole de *Diffie -Hillman* est vulnérable à l'attaque suivante, appelée attaque par interposition.

Francis choisit un entier c tel que $1 < c < q - 1$ et calcule g^c . Il intercepte la clé publique g^a envoyée par Alice à Bernard et lui substitue la clé g^c qu'il envoie à Bernard, lequel croit recevoir la clé publique d'Alice.

Lorsque Bernard envoie à Alice sa clé publique g^b , Francis l'intercepte et envoie g^c à Alice. La clé secrète d'Alice devient donc $k_{ab} = g^{ac}$ et celle de Bernard $k_{ba} = g^{bc}$, Francis les connaît puisque qu'il suffit d'élever chacune des clés publique d'Alice et Bernard à la puissance c .

Francis intercepte des leur tous les message d'Alice à Bernard, est capable de les chiffrer avec la clé k_{ab} , de les modifier puis de les envoyer à Bernard en les chiffrant avec la clé k_{ba} .

Il procède de même avec les messages envoyés par Bernard à Alice, qu'il déchiffre avec la clé k_{ba} et envoie à Alice en les déchiffrant à l'aide de la clé. La vulnérabilité du protocole de Diffie -Hillman réside dans le fait qu'il ne permet pas d'authentifier les participants.

3.4.2 Algorithme de chiffrement d'El Gamal

C'est un cryptosystème basé sur le problème du logarithme discret .

Soit \mathbb{F}_q un corps à q éléments, et soit g un élément primitif de \mathbb{F}_q . On suppose que

le problème du logarithme discret de base g dans \mathbb{F}_q est difficile. Le couple (\mathbb{F}_q, g) est publique.

Si Bernard veut permettre à un tiers de lui envoyer des messages secrets, il choisit un entier a tel que $1 < a < q - 1$, qui sera sa clé secrète. Il calcule $\alpha = g^a$, qu'il publie, et qui sera sa clé publique.

Pour envoyer à Bernard le message m appartenant à \mathbb{F}_q , Alice choisit aléatoirement un entier x tel que $1 < x < q - 1$, et lui transmet le couple $(\beta, \gamma) = (g^x, m \times \alpha^x) = (g^x, m \times g^{ax})$ qui est le message crypté.

Bernard, grâce à sa clé secrète a , peut calculer l'inverse δ de $g^{ax} \Rightarrow \gamma\delta = g^{ax}g^{-ax}m = m$. Pour authentifier le message il utilise la signature qui est très difficile à imiter et en plus liée au contenu du message m .

Exemple 3.5.

Soit $k = \mathbb{F}_3/\langle p \rangle$ tel que $p = x^3 + x + 1$ est un polynôme irréductible dans \mathbb{F}_3 . Alors k est un corps [11]

Alice rend public le corps k , l'élément $g = \alpha$. Sa clé publique $\alpha^a = 2 + \alpha + 2\alpha^2$.

Bernard veut crypter le message $2 + \alpha^2$ en utilisant $x = 9$. Il transmet à Alice $(\alpha^9, m(\alpha^a)^9)$. Il calcule $g^{a \times 9} = (2 + 2\alpha + 2\alpha^2)^9 = 2 + \alpha^9 + 2\alpha^{18}$. On sait que $\alpha^3 = \alpha - 1$ et $\alpha^{13} = -1 = 2$. Alors $g^{ax} = 2 + (\alpha - 1)^3 + \alpha^5 = 2 + 2\alpha + 2\alpha^2$ donc le message crypté est $(2 + \alpha^2)(2 + 2\alpha + 2\alpha^2) = 2 + \alpha + 2\alpha^2$.

Si Alice transmet 2α à Bernard, on a $\alpha^5 = 2 + \alpha + 2\alpha^2$ donc $a = 5$ et $c = \alpha^{19}$ d'inverse $\alpha^7 = 2 + 2\alpha + \alpha^2$ alors le message de Bernard était donc $2\alpha(2 + 2\alpha + \alpha^2) = 1 + \alpha^2$.

3.4.3 L'algorithme à clé publique RSA

Imaginé par les trois chercheurs américains *Ronald Rivest*, *Adi Shamir* et *Leonard Adleman* en 1977, Ils s'intéressèrent à la recherche de fonctions à sens unique pour trouver une réponse au problème de *Diffie* de chiffrement asymétrique. Après beaucoup d'essais c'est en avril 1977 que *Rivest* eut la bonne idée ; ses deux collègues ne découvrirent cette

fois-ci aucune faille dans la méthode proposée. *Martin* Gardner (né en 1914) publia la méthode dans le numéro d'août de *Scientific American*, sous le titre A new kind of cipher that would take millions of years to break avec un problème qui fut, en fait, résolu 17 ans plus tard, grâce au développement des ordinateurs, d'une part, et des méthodes de factorisation d'entiers en produits de facteurs premiers, d'autre part. Le *RSA* est basé sur la théorie des nombres premiers, et sa robustesse tient du fait qu'il n'existe aucun algorithme de décomposition d'un nombre en facteurs premiers. Alors qu'il est facile de multiplier deux nombres premiers, il est très difficile de décomposer deux entiers si l'on en connaît le produit.

Voici le déroulement de l'algorithme :

- On choisit au hasard deux nombres premiers p et q suffisamment grands pour que le nombre $n = pq$ ne soit pas factorisable avec les méthodes et les moyens de calcul actuels.
- On choisit ensuite au hasard un entier e premier avec $\varphi(n) = (p-1)(q-1)$ et on calcule son inverse d modulo $\varphi(n)$; on est le seul à pouvoir faire ce calcul parce qu'on connaît p et q .

Les applications $D, E : \mathbb{Z}/n\mathbb{Z} \longrightarrow \mathbb{Z}/n\mathbb{Z}$ définies par $E(x) = x^e \bmod n$ et $D(x) = x^d \bmod n$ sont des bijections réciproques car $ed = 1$ D'où $D(E(x)) = x^{ed} = x \bmod n$. Les lettres E, e et D, d ont été choisies pour évoquer les opérations d'encodage, et de décodage.

Pour transmettre un message M , on le découpe en blocs codés par des entiers $< n$. Pour chiffrer un bloc x , on calcule $y = x^e \bmod n$. Pour le déchiffrer, on calcule $y^d \bmod n$. L'émetteur a besoin de connaître e et n , le récepteur d et n .

Exemples d'utilisation du RSA

Exemple 3.6.

On applique le chiffrement *RSA* sur cet exemple (trivial) qui vous permettra de vérifier le fonctionnement de ce système même si vous n'avez pas de calculatrice à la main.

Prenons le couple $p = 3$ et $q = 5$ sont deux nombres premiers. Nous calculons alors $n = p \times q = 15$ et $\varphi(n) = 8$. Dans ce cas, $e = 3$ (e est premier avec 8). On peut choisir $d = 3$ puisque $e \times d = 3 \times 3 = 9 = 8 \times 1 + 1$.

On essayer de coder $A = 2$. Nous avons $A^e = 2^3 = 8$, donc A^e est congruent à 8 modulo 15. Le nombre codé est donc $B = 8$.

Pour décoder, nous prenons le reste de la division de B^d par n . $B^d = 8^3 = 512$. En faisant la division euclidienne de 512 par 15, nous obtenons : $B^d = 512 = 34 \times 15 + 2$. Le reste est donc 2, c'est-à-dire A , le nombre que nous avons codé au départ.

Exemple 3.7.

On choisit d'utiliser deux nombres premiers $p = 17$ et $q = 11$ et le nombre $e = 7$ qui est premier avec $\varphi(n) = 160$. Nous calculons à partir de ce nombre e sa clé privée qu'il ne divulguera jamais. Pour ce faire, on doit chercher un nombre d tel que si il existe un entier $M : ed = M(p-1)(q-1) + 1$. Pour $M = 1$, on observe que $M(p-1)(q-1) + 1 = 161 = 7 \times 23$. La clé privée sera donc $d = 23$.

L'auteur rend aussi publique une correspondance entre lettres et chiffres afin de coder des messages textuels, et choisit la convention suivante :

$a = 01$	$j = 10$	$s = 19$	$. = 28$	$. = 28$
$b = 02$	$k = 11$	$t = 20$	$? = 29$	$8 = 38$
$c = 03$	$l = 12$	$u = 21$	$0 = 30$	$9 = 39$
$d = 04$	$m = 13$	$v = 22$	$1 = 31$	$! = 40$
$e = 05$	$n = 14$	$w = 23$	$2 = 32$	$\iota = 41$
$f = 06$	$o = 15$	$x = 24$	$3 = 33$...
$g = 07$	$p = 16$	$y = 25$	$4 = 34$	
$h = 08$	$q = 17$	$z = 26$	$5 = 35$	
$i = 09$	$r = 18$	$, = 27$	$6 = 36$	

L'espace entre deux mots sera codé par 00. Bob envoie la clé ($n = 187, e = 7$) à Alice, appelée clé publique et garde pour lui seul, le nombre $d = 23$, aussi appelé clé privée. Si Bob, décide de communiquer un message à Alice : « les maths, c'est génial ! ». Bob va donc coder ce message avec la clé publique (n, e). Pour commencer, il convertit les lettres en chiffres, en obtient ainsi le nombre : 1205190013012008190003410519200007051409011240

Il regroupe ce nombre en blocs de chiffres strictement inférieurs à $n = 187$:

120 51 90 013 012 008 19 000 34 105 19 20 000 70 51 40 90 112 40

Chacun de ces nombres inférieurs à n sera codé en utilisant la clé privée. Il s'agit donc pour Bob d'élever chaque nombre à la puissance 7 et d'en prendre le reste de la division par 187. Il note chaque mot codé en ajoutant des zéros à gauche afin que chaque nombre soit composé de trois chiffres et il obtient le code :

120017095106177134145000034096145147000060017116095073116 .C'est ce message codé qu'il envoie à Alice.

Toute personne recevant ce message est incapable de retrouver le message original envoyé par Bob, s'il ne connaît pas la clé secrète. Néanmoins, Alice possède la clé secrète $d = 23$. Pour décoder le message, elle lui suffit de découper le message en blocs de nombres à trois chiffres, et prendre chaque nombre de cette décomposition, l'élever à la puissance d puis de diviser le résultat par n et de noter le reste obtenu. Alice va ainsi obtenir le message non codé. Elle ne lui reste plus qu'à prendre son tableau de correspondance alphabétique pour retrouver le message original « les maths, c'est génial ! ».

Bien entendu, codage et décodage se font sur ordinateur, en raison de la longueur des opérations. Mais si le reste de la division d'un nombre par un autre peut se calculer informatiquement de façon très efficace en utilisant l'algorithme d'Euclide, et que le temps de calcul est linéaire en fonction du nombre de chiffres. Il n'existe pas de méthode connue pour le diviser en temps polynomial.

Supposons donc maintenant qu'un ennemi de l'auteur aimerait bien connaître le message envoyé par Bob. Il connaît la clé publique ($n = 187, e = 7$), il ne peut cependant pas décrypter le message car il lui faudrait avoir la clé secrète $d = 23$. Pour l'obtenir, il doit

décomposer n en facteurs premiers, ce qui est certes facile avec 187, mais impossible actuellement - à moins d'y consacrer des siècles et des milliers d'ordinateurs ou de recourir à la divination - avec un nombre de 150 à 200 chiffres.

Exemple 3.9. "*Racine cubique modulo 1000*"

On prend un nombre x de trois chiffres, puis on calcule le cube de x . On ne conserve que les trois derniers chiffres = reste de la division par 1000, c'est y .

Si on essaye de calculer $x^3 = y \bmod 1000$ c'est une fonction à sens unique car il est facile de trouver y à partir de x , mais inversement difficile sauf si on utilise des méthodes qui peuvent être éventuellement exécutés dans un délai raisonnable à l'aide des dispositifs de calcul.

*** Application**

Supposons que les trois derniers chiffres de x^3 sont 631. Alors $y = 631, x^3 = 631 \bmod 1000$ lorsqu'il n'existe pas des algorithmes de complexité linéaire si pour sa on a besoin d'utiliser des méthodes qui simplifier la résolution. Ainsi on résout l'équation par la solution brutale qui permet de calculer la valeur de x . Après beaucoup de temps et plusieurs vérifications on trouve $x = 35631$.

Exemple 3.10. "*Racine 7^{ème} modulo 1000*"

La fonction $x^7 = y \bmod 1000$ est fonction à sens unique : étant donné x , il est facile de trouver y , mais la racine 7^{ème} modulo 1000 est difficile de trouver même si on utilise les techniques actuelles i.e n'existe pas un algorithme efficace qui donne exactement la racine 7^{ème} modulo 1000. En effet, il existe des algorithmes pour trouver la valeur de x , mais en pratique si on connait une information précédente sur la fonction lui-même.

*** Application**

Pour trouver la valeur de x dans l'équation $x^7 = 871 \bmod 1000$, en pratique est impossible, mais l'existence de cette hypothèse que : 7 est la clé publique (car on calcule x^7) et 43 une clé secrète. On peut trouver x . Cela signifie que si on calcule la puissance

43 de 871, on trouve x :

$$\begin{aligned} 871^{43} = x \bmod 1000 &\Rightarrow (x^7)^{43} \equiv x \bmod 1000 \\ &\equiv 111 \bmod 1000. \text{ Donc } x = 111. \end{aligned}$$

Bibliographie

- [1] A. CANTEAUT AND F. L.-DIT-VÉHEL. *La cryptologie moderne*, 75739 Paris Cedex 15, 1 – 13.
- [2] F. ARNAULT. *Théorie des Nombres & Cryptographie*. Cours de D. E. A. 7 mai 2002.
- [3] G. DUBERTRET. *INITIATION À LA CRYPTOGRAPHIE*, 2^{ème} édition, Paris.
- [4] G. FLORIN. *LES TECHNIQUES DE CRYPTOGRAPHIE*. CNAM. Cedric.
- [5] G. LABOURET. *Introduction à la cryptographie*, 1999 – 2001. Hervé Schauer Consultants.
- [6] J. PIERRE ESCOFIER. *TOUTE L'ALGÈBRE DE LA LICENCE*. Cours et exercices corrigés. 2^{ème} édition. Avril 2006. Belgique.
- [7] M. W. BALDONI AND C. CILIBERTO AND G. M. PIACENTINI CATTANEO. *Elementary Number Theory. Cryptography and Codes*. 2009 Springer. Italy.
- [8] M. WALDSCHMIDT. *Cryptographie : une introduction élémentaire*. Lundi 26 octobre 2009.
- [9] P. Q. NGUYÊN. *Théorie et Pratique de la Cryptanalyse à Clef Publique*. Diplôme d'Habilitation à Diriger des Recherches . Paris. 23 novembre 2007.
- [10] P. ROUCHON. *Cryptographie et fonctions à sens unique*. Mines ParisTech . 2010.
- [11] P. WASSEF. *ARITHMÉTIQUE. APPLICATION AUX CODES CORRECTEURS ET À LA CRYPTOGRAPHIE*. Août 2008. France.

- [12] R. AMADIO. *Sécurité*. Cours de Master .Paris.2011 – 2012.
- [13] R. HAGGARTY. *Mathématiques discrètes appliquées à l'informatique*.France.Juin 2005
- [14] S. LAPLANTE. *Le principe des boîtes noires : en cryptologie application et limites*. Mémoire de l'obtention du grade de Maître ès sciences (M.Sc) en informatique. Juillet 1992.
- [15] S. IOVLEFF. *Mathématiques Pour l'Informatique I : Arithmétique et Cryptographie*. 15 novembre 2004.