Université de Rennes 1

CRYPTO

CRYPTOLOGIE

Auteur
Sylvain Duquesne

Notes de cours
Victor Lecerf
(Merci Alexis)

2021–2022

Table des matières

1 Histoire de la cryptographie

1.1 Introduction

Le dictionnaire *Larousse* définit la cryptographie comme l'"Ensemble des principes, méthodes et techniques dont l'application assure le chiffrement et le déchiffrement des données, afin d'en préserver la confidentialité et l'authenticité". Historiquement, la cryptographie démarre en Égypte et en Chine avec la stéganographie, avec la nécessité d'envoyer des messages pendant les guerres. En Égypte, on rasait les esclaves pour écrire des messages sur leur tête, l'esclave était donc incapable de donner de lui même le message. Les chinois eux écrivaient des messages qu'ils enfermaient dans des boules de cire, qu'ils avalaient. Plus tard, les grecques inventent la scytale (ruban de papier sur lequel on écrivait un message qui était crypté lorsque le messager déroulait le ruban, il suffisait alors d'enrouler le ruban pour lire le message (chiffrement par permutation)).

Enfin, la méthode la plus connue est celle attribuée à César qui consistait à effectuer un décale de lettres dans l'alphabet (chiffrement par substitution). Mathématiquement, on interprète le chiffrement de César comme l'application de chiffrement $c : \mathbb{Z}/26\mathbb{Z} \to \mathbb{Z}/26\mathbb{Z}$, $x \mapsto x + k$, où $k \in \mathbb{Z}/26\mathbb{Z}$. Il existe plusieurs types d'attaques (*i.e* des méthodes pour retrouver le message secret),

— recherche exhaustive : essayer les vingt-six clés jusqu'à obtenir un message intelligible ;

— analyse de fréquence : déterminer la lettre du message la plus fréquente (la connaissance de la lettre la plus fréquente du latin [1] permet de retrouver la clé) ;

— attaque à clair connu : on suppose connu un couple message clair/message chiffré.

On attend aujourd'hui d'un système de cryptographie de résister à une attaque à clair connu puisqu'un attaquant a très souvent accès à des messages ultérieurs décryptés.

1.2 Chiffrement affine

On considère l'alphabet $\mathbb{Z}/m\mathbb{Z}$ et une clé (a, b) avec a premier à m. L'application de chiffrement est

$$c : \left| \begin{array}{ccc} \mathbb{Z}/m\mathbb{Z} & \longrightarrow & \mathbb{Z}/m\mathbb{Z} \\ x & \longmapsto & ax + b \end{array} \right.$$

et l'application de déchiffrement est

$$d : \left| \begin{array}{ccc} \mathbb{Z}/m\mathbb{Z} & \longrightarrow & \mathbb{Z}/m\mathbb{Z} \\ x & \longmapsto & a'(x - b) \end{array} \right.$$

où a' est l'inverse de a dans $\mathbb{Z}/m\mathbb{Z}$.

Cryptanalyse. On suit les méthodes proposées.

— Recherche exhaustive : cela revient à étudier $m\varphi(m)$ clés possibles.

— Analyse de fréquence simple ou attaque à clair connu : cela revient à étudier un système de congruences. Par exemple, si $m = 26$ et R est la lettre la plus fréquente, alors on peut supposer que E est envoyé sur R. On suppose aussi connu que S est envoyé sur H. Alors, $c(4) = 17$ et $c(18) = 7$. On étudie alors le système

$$\left\{ \begin{array}{rcl} 4a + b & \equiv & 17 \mod 26 \\ 18a + b & \equiv & 7 \mod 26 \end{array} \right.$$

qui a pour solution $(a, b) \equiv (3, 5) \mod 26$.

— Analyse de fréquences complètes : facile et généralisable à tous les cryptosystèmes monoalphabétiques (*i.e* une permutation de $\mathbb{Z}/m\mathbb{Z}$).

1.3 Chiffrement de Vigenère

Dans le chiffrement de Vigenère, on considère une clé $k \in (\mathbb{Z}/m\mathbb{Z})^n$ et la clé de chiffrement

$$c : \left| \begin{array}{ccc} (\mathbb{Z}/m\mathbb{Z})^n & \longrightarrow & (\mathbb{Z}/m\mathbb{Z})^n \\ x & \longmapsto & x + k. \end{array} \right.$$

Dans ce cas, une recherche exhaustive de cryptanalyse demande l'étude de m^n clés possibles. L'attaque à clair connu reste possible avec un système de congruences en n indéterminées. Pour autant, l'analyse de fréquences devient totalement non triviale puisqu'une lettre n'est pas toujours chiffrée de la même manière.

Test de Kasiski. Le principe est le suivant : la distance entre deux groupes identiques de lettres dans le chiffré est probablement un multiple de la taille de la clé. Le teste consiste à calculer le pgcd des distances entre des répétitions. L'inconvénient est qu'une répétition due au hasard fausse le test et doit être éliminée. Autrement, on n'obtiendrait qu'un multiple de la longueur de la clé.

1. En français, le "e" est la lettre la plus fréquente à hauteur de $12,10\%$.

Test de Friedman. Soient I l'indice de coïncidence (*i.e* la probabilité que deux lettres soient identiques), N le nombre de lettres dans l'alphabet, n la taille du chiffré, et i l'indice de coïncidence du texte chiffré. La taille de la clé vaut alors

$$\frac{\left(I - \frac{1}{N}\right) n}{(n-1)i - \frac{n}{N} + I}.$$

1.4 Chiffrement affine par bloc

Cette méthode de chiffrement est une généralisation du chiffrement affine et du chiffrement de Vigenère. La clé est une matrice $A \in \mathcal{M}_n(\mathbb{Z}/m\mathbb{Z})$, et un vecteur $b \in (\mathbb{Z}/m\mathbb{Z})$ tels que $\det A$ est premier avec m. L'application de chiffrement est

$$c : \left| \begin{array}{rcl} (\mathbb{Z}/m\mathbb{Z})^n & \longrightarrow & (\mathbb{Z}/m\mathbb{Z})^n \\ x & \longmapsto & Ax + b \end{array} \right.$$

et l'application de déchiffrement est

$$d : \left| \begin{array}{rcl} (\mathbb{Z}/m\mathbb{Z})^n & \longrightarrow & (\mathbb{Z}/m\mathbb{Z})^n \\ x & \longmapsto & A'(x - b) \end{array} \right.$$

où A' est telle que $AA' = I_n$ dans $\mathcal{M}_n(\mathbb{Z}/m\mathbb{Z})$. Avec un tel système de chiffrement, la méthode exhaustive demande d'étudier environ $O(m^{n^2+n})$ possibilités. Une attaque à clair connu est toujours possible avec $n+1$ couples clair/chiffrés.

1.5 Les machines à rotor, Enigma

2 Introduction à la cryptographie

2.1 Terminologie

La conception d'un système de chiffrement s'appelle la cryptographie, son analyse en vue de l'attaquer s'appelle la cryptanalyse. La cryptologie est la partie des mathématiques qui regroupe la cryptographie et la cryptanalyse.

La problématique étudiée par cette discipline est le suivant : un émetteur et un récepteur désirent s'envoyer un message sur un canal de transmission public. Le but pour nous est de décrire et analyser des procédés (ainsi que leurs implémentations concrètes) permettant de transformer le message original, que l'on désignera par *message clair*[2], en un message équivalent, le *message chiffré*[3] dont le contenu et la signification initial sera dissimulé. Ce procédé sera appelé *chiffrement* (ou cryptage), et son inverse le *déchiffrement*. Cela permet d'assurer la confidentialité du message.

Définition 2.1 (système de chiffrement)**.** *On appelle système de chiffrement (ou cryptosystème) est un quintuplet $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ où*

- *$\mathcal{P}$ est l'espace des messages clairs ;*
- *$\mathcal{C}$ est l'espace des messages chiffrés ;*
- *$\mathcal{K}$ est l'espace des clés ;*
- *$\mathcal{E} = \{E_k : \mathcal{P} \longrightarrow \mathcal{C} \,|\, k \in \mathcal{K}\}$ est l'ensemble des fonctions de chiffrement ;*
- *$\mathcal{D} = \{D_k : \mathcal{C} \longrightarrow \mathcal{P} \,|\, k \in \mathcal{K}\}$ est l'ensemble des fonctions de déchiffrement.*

tel qu'à toute clé e de $\mathcal{K}$ est associée une clé d de $\mathcal{K}$ telle que $D_d(E_e(p))$ pour tout $p \in \mathcal{P}$.

2. En anglais, *plain text.*
3. En anglais, *cypher text.*

2.2 Chiffrement de Vernam (1917)

On étudie ici le chiffrement de Vernam, aussi appelé *masque jetable*[4]. Soit $n \in \mathbb{N}^*$, $\mathcal{P} = \mathcal{C} = \mathcal{K} = \{0,1\}^n$ et $\mathcal{E} = \mathcal{D} = \{\oplus\}$, où $\oplus$ est l'addition sur $\mathbb{Z}/2\mathbb{Z}$ (informatiquement interprété comme l'opération logique XOR). On munit $\mathcal{K}$ de la loi uniforme. Le chiffrement de Vernam est en fait un chiffrement de Vigenère sur l'alphabet $\{0,1\}$ (tout message en alphabet latin est d'abord convertit en binaire). La différence est que le chiffrement de Vernam impose trois conditions : la clé est aussi longue que le message clair, est totalement aléatoire (uniformément choisie), et chaque clé ne n'est utilisé au plus qu'une fois (d'où la métaphore du masque jetable).

Ce chiffrement a plusieurs avantages : il est extrêmement simple, et est parfaitement sûr (Shannon l'a démontré en 1949). C'est en fait le seul chiffrement parfait connu. Pour autant, il possède plusieurs inconvénients. En effet, il n'empêche pas les attaques actives, il nécessite l'échange préalable d'une clé aussi longue que le message, et la clé doit être tirée "parfaitement aléatoirement".

2.3 ?

On distingue deux types d'attaques : les attaques passives (l'attaquant se contente d'observer, ce qui menace la confidentialité), et les attaques actives (l'attaquant peut interférer dans la communication, ce qui menace l'intégrité et l'authenticité du message).

On décrit la fragilité et la sécurité des algorithmes de plusieurs manières.

— Un *total break* est l'éventualité où l'on est capable de trouver la clé.

— Une *déduction globale* est le cas de figure où un algorithme alternatif permet le déchiffrement.

— Une *déduction locale* consiste à trouver le message clair à partir du message chiffré.

— Enfin, une *déduction partielle* consiste à obtenir une information partielle sur la clé ou le message clair.

En cryptographie, on cherche tout particulièrement à éviter la possibilité d'une attaque par déduction partielle, puisqu'elle peut donner lieux à des menaces ou des pressions dans les affaires.

Quels sont les moyens de l'attaquant ?

— *Attaque à chiffré connu (cypthertext-only attack).* On connaît les messages chiffrés C_i sachant que $C_i = E_k(M_i)$ pour $i \in [\![1, N]\!]$, où M_i sont les messages clairs. Étant ces informations, un $N+1$-ième message crypté, C_{N+1}, on cherche le message clair M_{N+1} (en connaissant C_{N+1}).

— *Attaque à clair connu (known-plaintext attack).* On connaît des pairs (M_i, C_i) avec $C_i = E_K(M_i)$. On cherche soit M_{N+1} sachant C_{N+1}, soit K. Il est primordial qu'un système de cryptage de résister aux attaques à clair connu. En effet, on possède généralement beaucoup de clairs connus étant donné un système de cryptage (par exemple les entêtes de mails).

— *Attaque à texte clair choisi (chosen-plaintext (CPA)).* On connaît des pairs (M_i, C_i) pour $i \in [\![1, N]\!]$ avec M_i choisi et $C_i = E_K(M_i)$. On cherche soit M_{N+1} sachant C_{N+1}, soit K. Les pairs sont choisis dans le sens où on peut prédire le type du message M_i. Par exemple lors d'un dialogue, sachant que C_N est une question, on peut prévoir le type de réponse en C_{N+1}.

— *Attaque à texte chiffré choisi (chosen-cypthertext (CCA)).* On connaît des pairs (M_i, C_i) pour $i \in [\![1, N]\!]$ avec C_i choisi et $M_i = D_K(C_i)$. On cherche soit M_{N+1} sachant C_{N+1}, soit K. Les pairs sont choisis dans le sens où on peut prédire le type du message M_i. Par exemple lors d'un dialogue, sachant que C_N est une question, on peut prévoir le type de réponse en C_{N+1}.

4. En anglais, *one time pad*

— Pour ces deux dernières attaques, il existe des variantes adaptives (CPA2 et CCA2), *i.e* que l'attaquant choisit les messages un par un en fonction d'observations qu'il peut faire sur les messages (clairs ou cryptés) choisis.

— On peut également supposer que le but de l'attaquant est simplement de pouvoir distinguer deux messages (IND-CCA2).

Différents types d'attaques.

— *Force brute.* Essayer toutes les combinaisons et possibilités de déchiffrement. Par exemple, si la clé est de 128 bits, au plus 2^{128} seront nécessaires pour trouver la clé. On estime que cette méthode est infaisable à partir de 2^{80} opérations [5]. On parle aussi d'attaque par dictionnaire si toutes les clés sont consultables et que l'on peut toutes les essayer.

— *Rejeu.* Le rejeu consiste à réenvoyer un message. Par exemple, un transfert d'argent est fait en ligne. Un message est envoyé pour effectuer le transfert. L'attaquant intercepte le message et peut faire renvoyer 10 fois de suite le même message et décuplé l'argent récupéré lors du transfert.

— *Compromis temps-mémoire.* C'est une variante de l'attaque par force brute. Le problème de l'attaque par force brute est que le temps pour essayer toutes les possibilités peut être très long. On peut décider de restreindre la taille de mémoire utilisée par les clés pour gagner du temps. Par exemple, on peut étudier dans un dictionnaire uniquement les clés en quatre et six caractères de longueurs.

— *Différentielle.* Cette attaque consiste à observer les différentes entre deux messages.

— *Linéaire.* Si un clair connu donne lieu à un système linéaire avec le chiffré et la clé (comme avec la méthode affine lorsque l'on connaît les fréquences des lettres), on peut résoudre ce système et trouver la clé.

— *Algébrique.* Cette attaque a le même principe que le linéaire, à l'exception qu'on s'autorise des relations algébriques (quadratiques, cubiques, etc...)

— *Biais statistiques.* On peut obtenir des informations sur la clé en remarquant un manque d'aléatoire dans un message. Il est quasiment impossible de concevoir une méthode de cryptage ne possédant aucun biais statistiques. Cependant, il n'est pas non plus aisé pour un attaquant de trouver un biais statistique et de l'exploiter.

— *Paradoxe des anniversaires.* Lorsque les clés sont tirées aléatoirement, il est toujours possible qu'une clé soit réutilisée.

— *Attaque par canaux cachés.* Analyser un rayonnement électromagnétique, une consommation électrique, des bruits d'un ordinateur pour être capable de trouver les calculs effectués par un processeur et donc la clé qu'il utilise. Il existe aussi des attaque par canaux cachés actives, où l'on perturbe un processeur lors de ses calculs cryptiques pour le forcer à commettre des erreurs. Par exemple, si un générateur aléatoire de clé est perturbé à la fin de la génération d'une clé par un laser, on peut connaître les derniers bits du message. Il est cependant très complexe d'exploiter ce genre d'attaque active.

Définition 2.2. *Pour une méthode de chiffrement, on dit que la méthode possède un niveau de sécurité en n bits s'il n'existe pas d'attaque connue demandant moins* 2^n *bits de calculs.*

5. Il est "raisonnable" aujourd'hui d'effectuer 2^{65} opérations avec trois-mille supercalculateurs. On estime que la NSA est probablement capable d'effectuer 2^{80} opérations dans un temps "raisonnable". La facture en électricité de la NSA est de vingt millions de dollars, et c'est pas pour faire tourner des ampoules.

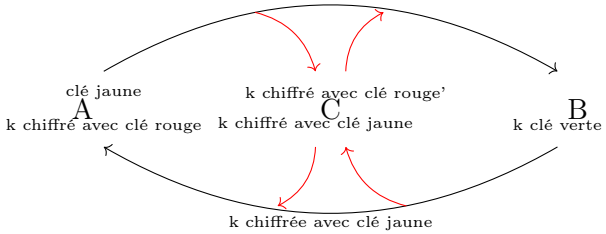

FIGURE 1 – Principe de l'attaque *man in the middle*.

L'attaque *man in the middle*. Le but de l'attaquant est de se faire passer pour l'un (voire les deux) des correspondants, en utilisant par exemple l'ARP spoofing, le spoofing de DNS, ou le déni de service.

3 Chiffrement par blocs

3.1 Principe des chiffrements par blocs

Définition 3.1. *Soit A un alphabet. Un cryptosystème par blocs est un chiffrement par blocs si $\mathcal{P} = \mathcal{C} = A^n$, où n est appelée longueur du blocs.*

Exemple. Chiffrement de VIGENÈRE.

Proposition 3.1. *Les fonctions de chiffrement par blocs sont des permutations.*

En pratique, l'espace des clés n'est qu'un sous-ensemble de $\mathfrak{S}(A^n)$ facile à représenter et à évaluer. Les modes opératoires permettent de déterminer la façon dont on chiffre la suite des blocs d'un message long.

Le mode ECB (*Electronic Code Book*). Le principe est le suivant : on découpe le message en blocs m_i de taille n et on chiffre indépendamment chaque m_i avec la clé k. Par exemple avec $n = 3$ et le message "Je te dois 1000 euros", on a la découpe $c_1c_2c_3c_4c_5c_6c_7$ avec c_1 : "Je ", c_2 : "te ", c_3 : "doi", c_4 : "s 1", c_5 : "000", c_6 : " eu", et c_7 : "ros". On a en fait $c_i = E_k(m_i)$. Ce mode de chiffrement est en fait trop faible dû à l'indépendance des blocs et des chiffrés précédents.

Le mode CBC (*cypher-bloc chaining*). Le principe est le suivant : avant d'être chiffré, un bloc est "Xoré" avec le chiffré du bloc précédent. Cela nécessite un vecteur d'initialisation. Cela nécessite un vecteur d'initialisation (IV) pour chiffrer et déchiffrer. On a $c_j = E_k(c_{j-1} \oplus m_j)$. La propagation d'erreur un inconvénient majeur de ce système.

Le mode CFB (*cypher feedback*). C'est "l'inverse" du mode CBC : on utilise la clé de chiffrement sur le chiffré précédent, et on le xore avec le message.

Le mode OFB (*output feedback*). On choisit z_0 une valeur d'initialisation, puis on pose $z_{i+1} = E_k(z_i)$ pour tout $i \in [\![1, N-1]\!]$, où N est le nombre de blocs. Ensuite, on chiffre en xorant m_i avec z_i : $c_i = m_i \oplus z_i$.

Le mode CTR (*counter*). On a $c_i = m_i \oplus E_k(T+i)$.

Il en existe d'autres comme par exemple le CTS ou EAX. Ces modes de chiffrement n'assurent pas l'intégrité car il faut les coupler avec des MAC (*Message Authentification Code*). Une propagation d'erreur peut compromettre l'intégrité d'un message. Il faut donc rajouter des MAC et des codes correcteurs ou utiliser des modes spécifiques (XCBC, IACBC, IAPM, OCB, CWC, CCM, GCM).

Le mode GCL (Galois *counter mode*). On utilise pour cette méthode des corps finis. C'est une combinaison du mode CTR et de l'utilisation du corps fini $\mathbb{F}_{2^{128}} = \mathbb{F}_2[X]/(X^{128} + X^7 + X^2 + X + 1)$. Il est recommandé par le NIST et la NSA, c'est en fait un standard. Il intègre l'authentification des données, mais il est impératif d'utiliser des blocs de 128 bits. Il possède les mêmes propriétés que le mode CTR, et un bon rapport vitesse/ressources nécessaires. Enfin, la valeur initiale du compteur doit changer à chaque chiffrement et est libre de droit.

4 Chiffrement par flot

Rappel. Le principe du chiffrement par masque jetable consiste à XORé le message et une clé de même longueur, qu'on jette ensuite. C'est le seul système de chiffrement que l'on a montré être sûr. Il est cependant inutilisable puisque s'échanger la clé entre Alice et Bob pose le même problème que s'échanger le message en premier lieu. Il est donc en pratique inutilisable.

Le chiffrement par flot est une méthode cherchant à se rapprocher du chiffrement par masque jetable et sa sûreté. On génère un flux de bits aléatoires qui constituent la clé, que l'on XORe avec le message (ou autre). Le chiffrement par flot est essentiellement utilisé pour les communications sans fils (GSM, WIFI, Bluetooth, ...) car le chiffrement par flot a l'avantage (contrairement au chiffrement par bloc en mode CFB ou OFB) de se dérouler à un débit *élevé*, et s'implémente en hardware plutôt qu'en software mais demande une implémentation spécifique (*i.e* qu'on ne peut pas réutiliser les implémentations existantes).

Pour générer des flux de bits aléatoires, on utilise un générateur aléatoire. Un *vrai* générateur aléatoire doit être non-déterministe (*i.e* qu'il ne doit pas pouvoir être reproduit de manière fiable). Cependant, on possède principalement des générateurs *pseudo-aléatoires* qui permettent de produire une suite qui *a priori* semble aléatoire et satisfait des tests probabilistes, doit être imprévisible (*i.e* étant donné l'algorithme et les premiers bits de la clé, on ne doit pas être capable de déterminer la suite). Cependant, ces générateurs sont déterministes.

En pratique, les générateurs aléatoires sont difficiles à réaliser puisque les ordinateurs sont déterministes [6]. Cependant, le monde réel est aléatoire (bruit dans un ordinateur, temps de lecture d'un fichier, vitesse de frappe au clavier) mais ces flux aléatoires ont des débits très lents. Autre problème : un ordinateur n'a qu'un nombre fini d'états possibles. Ce sont donc des générateurs pseudo-aléatoires *périodiques* (et on va donc potentiellement utiliser deux fois la même clé). De grandes périodes sont donc nécessaires.

Le principe d'un générateur pseudo-aléatoire dans un chiffrement par flot est le suivant : on se munit d'une clé ou d'une *seed*, qui permet à l'algorithme de généré une suite chiffrante, qu'on l'on XORe avec le message chiffré.

Type de chiffrement par flot. On présente plusieurs algorithmes utilisés en chiffrement par flot.

— *Chiffrement synchrone.* La séquence de clé est indépendante du message. On a $k_i = f(k_{i-1})$ et $c_i = k_i \oplus m_i$. Ce chiffrement a l'avantage de ne pas être susceptible à la propagation d'erreur (mais alors une modification est facile et indétectable). Cependant, le chiffré devient totalement faux si il y a une désynchronisation (une clé ne chiffre plus le bon message avec

6. et on ne fait pas vraiment de cryptographie sans ordinateur.

décalage). Cet inconvénient peut permettre une attaque par omission ou insertion puisque la désynchronisation peut être visible lorsque l'on perturbe le système de messages. Enfin, réutiliser le même flux de clé casse le système. Une solution est de changer la *seed*, et d'utiliser un générateur avec un grande période. Remarquons que ce mode est équivalent à l'OFB.

— *Chiffrement asynchrone.* Les bits de clé dépendent des bits de clés précédents et des bits du message (un chiffrement *auto-synchrone* ne dépend que de cela). On a

$$k_i = f(k_{i-1}, c_{i-t}, \dots, c_{i-1}) \quad \text{et} \quad c_i = k_i \oplus m_i.$$

Il a pour avantage de se synchroniser automatiquement, mais peut être sujet à des propagations d'erreurs, et il est possible de rejouer les données dans une attaque par rejeu. Remarquons que ce mode est équivalent au CFB.

Exemples de générateurs.

— *Générateurs linéaires congruentiels.* On définit une suite $(z_n)_n$ avec $z_n \cong (az_{n-1} + b) \mod m$, où m est public et a, b, et z_0 forment la clé. Si a, b, et m sont chois correctement, le générateur sera de période maximale m. Ce générateur a l'avantage d'être rapide puisqu'il demande peu d'opérations, et a une bonne répartitions. Cependant, il est cryptographiquement mauvais (puisqu'il est linéaire, il peut être sujet à des attaques à clairs connus). Une combinaison de générateurs linéaires (ou polynomiaux) congruentiels ont un meilleur comportement statistique, de plus grandes périodes, mais ne sont pas pour autant plus sûrs.

— *Registres à décalage à rétroaction.*

— *Registres à décalage à rétroaction linéaire* (LFSR).

On appelle polynôme de rétroaction le polynôme de $\mathbb{F}_2[X]$

$$F(X) = 1 + c_1 X + c_2 X^2 + \cdots + c_n X^n.$$

Théorème 4.1. *La suite est une m-suite si son polynôme de rétroaction est le polynôme minimal d'un générateur d'un générateur du groupe cyclique $\mathbb{F}_{2^n}^*$ (i.e qu'il est primitif). Autrement dit, F est irréductible, divise $X^{2^n-1}+1$ et F ne divise pas X^d+1 si d divise 2^n-1.*

Remarque. Générer un polynôme primitif n'est pas chose aisée. Le mieux est encore de choisir un polynôme au hasard et de vérifier qu'il est primitif.

Exemple. Le polynôme $X^{16} + X^{14} + X^{13} + X^{11} + 1 \in \mathbb{F}_2[X]$ est primitif [7].

On utilise souvent des polynômes clairsemés qui sont plus rapides mais moins sûrs .

Définition 4.1. *La complexité linéaire d'une suite périodique s est la longueur du plus petit $LFSR$ qui l'engendre. On la note $\mathcal{L}(s)$.*

Remarque. Si s est une m-suite de période $2^L - 1$, sa complexité linéaire est L.

7. Merci Alexis.

Algorithme de Berkelamp-Massey. Cet algorithme permet de calculer en un temps quadratique la complexité linéaire et d'un LFSR engendrant un flot donné. Il suffit de connaître $2\mathcal{L}(s)$ bits consécutifs de la suite pour la retrouver entièrement.

5 RSA

5.1 Principe

Le système a pour clé privée deux nombres premiers p et q *très grand*, ainsi qu'un entier d premiers avec $(p-1)(q-1)$. Ces trois entiers sont strictement confidentiels. On dispose aussi d'une clé publique : $n := pq$, et e l'inverse de d modulo $(p-1)(q-1)$. La clé publique peut et doit être divulguée.

On peut facilement déduire la clé publique de la clé privée (algorithme d'Euclide étendu). Il est cependant très difficile étant donné n et e de retrouver d, p, ou q. En effet, les algorithmes de factorisations actuels sont totalement incapables de gérer des nombres de l'ordre de n lorsque p et q sont choisis très grands.

Ce système a été inventé en 1978 par Rivest, Shamir, et Adleman.

5.2 Cryptage et décryptage

On suppose que Bob veut envoyer un message m (un entier dans $[\![1, n]\!]$, où $n = pq$) à Alice.

— Bob récupère (sur la page web d'Alice, ou sur le serveur de certificat d'authentification) les entiers n et e qui font parties de la clé publique d'Alice ;

— Bob crypte le message en calculant le chiffré $c = m^e \mod n$;

— Bob envoie c à Alice.

De son côté, Alice déchiffre le message en sachant que $m = c^d \mod n$. Alice est en fait la seule personne capable de déchiffrer c puisque personne d'autre ne sait [8] ou ne peut calculer d. En fait, $x \longmapsto x^e \mod n$ est une fonction à sens unique.

Proposition 5.1. *Soit $n \in \mathbb{N}^*$. On note $\varphi(n)$ le nombre d'entiers dans $[\![1, n]\!]$ premiers avec n (ou l'ordre de $(\mathbb{Z}/n\mathbb{Z})^\times$). Pour tout $a \in \mathbb{N}$ premiers avec n, on a*

$$a^{\varphi(n)} \equiv 1 \mod n.$$

En particulier, si $n = p$ et est premier, on a

$$a^{p-1} \equiv 1 \mod p$$

(le petit théorème de Fermat). Si $n = pq$, où p et q sont premiers, on a

$$a^{(p-1)(q-1)} \equiv 1 \mod n.$$

Théorème 5.1 (des restes chinois)**.** *Soient m_1 et m_2 deux entiers premiers entre eux. Le système*

$$\begin{cases} x \equiv a_1 \mod m_1 \\ x \equiv a_2 \mod m_2 \end{cases}$$

possède une unique solution, qui est $a_1 n_1 m_2 + a_2 n_2 m_1$ où n_2 est l'inverse de m_1 modulo m_2 et n_1 est l'inverse de m_2 modulo m_1.

8. Techniquement, il y a toujours quelqu'un d'autre.

Statistiquement, on peut supposer que m et q sont premiers entre eux. En effet, en pratique, $n > 2^{1024}$ et $p, q > 2^{512}$. Il y a q multiples de $q < n$, donc p possibilités pour que $m \wedge q \neq 1$. Il y a donc une chance de $p/n = 1/q < 1/2^{512} \approx 1/10^{150}$. Et tant mieux car si ce n'est pas le cas, on a $m \wedge n = q$, on pourrait donc trouver q (et donc p) lors d'une attaque à clair connu.

5.3 Décryptage efficace

Un décryptage classique consiste à calculer $c^d \mod n$. Cependant, il existe d'autres moyens plus efficaces.

— Calculer $c^d = c^{d_p} \mod p$ où $d_p \equiv \mod p - 1$;

— calculer $c^d = c^{d_q} \mod q$ où $d_q \equiv \mod p - 1$;

— retrouver $c^d \mod n$ en utilisant le théorème des restes chinois.

5.4 Génération de clé

Pour le chiffrement RSA, nous avons besoin d'être capable de

— trouver deux nombres premiers p et q très grands ;

— trouver d premier avec $(p-1)(q-1)$ (il suffit de calculer $d \wedge ((p-1)(q-1))$) ;

— trouver l'inverse de d modulo $(p-1)(q-1)$ (*via* l'algorithme d'Euclide étendu).

On doit donc être capable en premier de générer de très grands nombres premiers. Pour cela, on utilise des tests de primalités. Le problème est que l'on sait selon le théorème des nombres premiers que le nombre d'entiers premiers inférieurs ou égal à un entier N est de l'ordre de $N/\ln(N)$ (ils se raréfient donc). De plus, les tests de primalités sont lents. Les tests probabilistes (Fermat, Miller-Rabin) sont plus pratiques dans leur efficacités, mais ne sont pas déterministes et ne montrent pas exactement qu'un nombre donné est premier.

5.5 Attaque par factorisation

L'attaque évidente sur le système RSA est de factoriser n. On a plusieurs possibilités.

— *Force brute.* Diviser n par tous les entiers plus petit que $\sqrt{n}$.

— *Crible d'*Ératosthène. Si 2 ne divise pas n, il n'est pas nécessaire de tester la division par les nombres pairs. La remarque suit pour tous les nombres premiers qui suivent.

— *Crible quadratique (QS).* On utilise les corps quadratiques (extensions quadratiques de $\mathbb{Q}$, tous de la forme $\mathbb{Q}(\sqrt{d})$ où d est un entier sans facteur carré) pour trouver x et y tels que $x^2 = y^2 \mod n$. Alors, $(x-y) \wedge n$ divise n. Cet algorithme a une complexité sous-exponentielle : $e^{\log(n)^{1/2}\log(\log(n))^{1/2}}$. En effet, si $x^2 - y^2 = kn$ où $k \in \mathbb{Z}$ et $x \neq \pm y \mod n$ (on les prend plus petit que n strictement), alors n divise $(x-y)(x+y)$ mais ne divise ni $x-y$ ni $x+y$. Ainsi, $(x-y) \wedge n \in \{p, q\}$. On dispose de quelques astuces pour trouver efficacement y^2. On peut par exemple le chercher petit (puisque y^2 ne dépasse pas n et il y a plus de petits carrés que de grands carrés). On se place généralement dans un corps quadratique pour obtenir plus de carrés (par exemple, 2 est un carré dans $\mathbb{Q}(\sqrt{2})$). Pour trouver x, on le choisit proche de $\sqrt{n}$ de sorte que x^2 dépasse n de peu. Par exemple si $n = 9127$, on prend x proche de $\sqrt{9167} \approx 95,7$. On choisit $x = 96$ et alors $x^2 = 9167 + 49$, donc on peut prendre $y = 49$. On a donc $x^2 = 96^2 = y^2 = 7^2 \mod 9167$, et $(96-7)(96+7) = 9167k$ pour un certain $k \in \mathbb{Z}$. On a donc montrer que $9167 = 89 \times 103$.

5.5.1 Méthode $p-1$

Définition 5.1. *Un entier s est dit être B-lisse si pour tout entier premier q et α entier tel que q^α divise s, alors $q^\alpha \leq B$.*

Supposons que p divise n et que $p-1$ est B-lisse. Si a est un entier premier avec p, et si

$$k = \prod_{q^\alpha \leq B} q^\alpha,$$

alors $a^k - 1 \equiv 0 \mod p$. Ainsi, $n \wedge (a^k - 1)$ est un facteur de n.

Pseudo-algorithme. On a la méthode suivante.

— Choisir une borne $B \in \mathbb{N}^*$ raisonnable ;
— choisir au hasard un entier a et calculer $b = a^k - 1 \mod n$;
— calculer $g = b \wedge n$;
— si $g \neq 1$, on renvoie g. Sinon, on recommence avec un B plus grand.

5.5.2 Méthode ρ-Pollard

Soit $(u_k)_k$ une marche aléatoire définie récursivement sous la forme $u_{k+1} = f(u_k) \mod n$.

5.5.3 Équivalences et réciproques

Nous savons que savoir factoriser permet de casser le code RSA. Cependant, la réciproque est-elle vraie ? En fait, trouver p et q est équivalent à trouver d. Le sens direct est trivial, mais pas la réciproque. L'entier $k = ed-1$ est multiple de $(p-1)(q-1)$, donc pour tout $g \in (\mathbb{Z}/n\mathbb{Z})^\times$, on a $g^k = 1$, donc $g^{k/2}$ est une racine carrée de 1. Il y a au plus quatre racines de 1 dans $(\mathbb{Z}/n\mathbb{Z})^\times$: 1, -1, x, et $-x$ avec x tel que $x \equiv 1 \mod p$ et $x \equiv -1 \mod q$. Si $g^{k/2} = \pm 1$, on choisit un autre g. Autrement, on sait que $n \wedge (x-1)$ est un facteur de n. Pour un g choisi au hasard, il y a une chance sur deux que $g^{k/2} = \pm 1$.

5.5.4 Pertinence du choix de p et q

Si $p-1$ est B-lisse, alors la factorisation est simple. Si p et q sont trop proches, on remarque les faits suivants. D'abord, on a toujours

$$\frac{(p+q)^2}{4} - n = \frac{(p-q)^2}{4}.$$

Soit x le plus petit entier plus grand que $\sqrt{n}$. Si $x^2 - n$ est un carré parfait noté y^2, alors $p = x+y$ et $q = x-y$. Si ce n'est pas le cas, on réitère en remplaçant x par $x+1$. Ceci est l'idée centrale des méthodes de cribles pour factoriser (qui consiste à trouver des égalités non pas dans $\mathbb{Z}$ mais dans des anneaux plus complexes (voir des corps)). En pratique, les nombres premiers choisis au hasard ne sont jamais susceptibles d'être faible face à ce type d'attaque.

5.6 Attaque sur la forme des messages

De petits messages sont prévisibles (par exemple "oui" ou "non"). Si le message crypté est connu et est petit, on peut facilement crypter tous les clairs possibles et le comparer au chiffré intercepté. Autre problème avec les petits messages : si $m < n^{1/e}$, on peut retrouver m en prenant la e-ième racine (réelle) du chiffré. Dans les applications pratiques du RSA, on a $n > 2^{1024}$ et e (premier avec $(p-1)(q-1)$) est choisi le plus petit possible [9].

9. Dans la pratique, il y a une asymétrie entre les moyens de calculs d'Alice et de Bob.

5.7 Attaque sur l'ordre de grandeur de e

Pour avoir des opérations efficaces sur les clés publiques (chiffrement, vérification de signature), il est pratique de choisir e le plus petit possible. Cependant, ce choix pratique peut faire l'objet d'une attaque. Supposons que l'on souhaite envoyer un message m à trois personnes (avec trois modules publics différents n_1, n_2, et n_3) avec $e = 3$. Évidemment, on a $m < n_i$ pour $i \in \{1, 2, 3\}$. Un attaquant peut alors remarquer que $c_i = m^3 \mod n_i$ et peut facilement résoudre le système

$$\begin{cases} x = c_1 \mod n_1, \\ x = c_2 \mod n_2, \\ x = c_3 \mod n_3. \end{cases}$$

Ainsi, l'attaquant peut déterminer m^3 modulo $n_1 n_2 n_3$ (par unicité de la solution). Or, $m^3 < n_1 n_2 n_3$ en tant qu'entier. L'attaquant connaît donc m en prenant la racine cubique réelle de m^3.

Comment se parer face à ce type d'attaque ? On peut par exemple oublier l'idée de prendre e petit et plutôt le prendre aléatoire, ou bien ne jamais envoyer plusieurs fois le même message. En réalité, aucune de ces deux solutions n'est satisfaisante. En général, on choisit

$$e = 65537 = 2^{16} + 1,$$

qui constituent un bon compromis. Il permet d'obtenir de bonne performance (en général cinquante fois plus rapide qu'un exposant aléatoire). De plus, il est rare d'envoyer le même message à plusieurs personnes différentes (ou du moins un grand nombre suffisant pour revenir à la méthode décrite plus haut).