Introduction.

A. Dragut Univ. Aix-Marseille
Cours de cryptographie Chapitre I

Jusqu'au début de la seconde guerre mondiale, la cryptographie a utilisé des méthodes rudimentaires. Avec l'apparition de la communication radio et plus tard du reseau Internet, il est devenu indispensable de pouvoir chiffrer des messages et de les rendre inintelligibles. La "cryptographie" désigne l'ensemble des techniques permettant de préserver la sécurité et la confidentialité des données, aussi que leur intégrité et leur authenticité. On distingue entre la "stéganographie" qui dissimule l'existence même d'information secrète et la "cryptographie" proprement dite, qui transforme un message clair en un cryptogramme, tandis que la "cryptanalyse" analyse des textes chiffres pour retrouver l'information dissimulée.

1.1 Principaux mots-clé

Texte en clair : l'information qu'Alice souhaite transmettre à Bob. Le texte en clair est formé par les données lisibles et compréhensibles sans intervention spécifique.

Le *cryptage/ou chiffrement* est le processus de transformation d'un message M de manière à le rendre incompréhensible à tout le monde sauf aux personnes auxquelles les informations sont destinées.

Processus de chiffrement :
- un message M
- fonction de chiffrement E
- Génération d'un chiffre (message chiffré) $C = E(M)$

Le processus inverse de reconstruction du texte en clair à partir du message chiffré est appelé le *décryptage/déchiffrement.*

Processus de déchiffrement :
- un message chiffré $E(M)$ (E est injective)
- fonction de déchiffrement D (D est surjective)
- $D(C) = D(E(M)) = M$

Definition. *Une application $E : X \mapsto Y$ est dite injective si pour tous $x_1, x_2 \in X$ avec $x_1 \neq x_2$ nous avons $E(x_1) \neq E(x_2)$.*

Definition. *Une application $D : C \mapsto M$ est dite surjective si tout élément de l'ensemble d'arrivée M a au moins un antécédent par D, c'est-à-dire si pour tout $m \in M$, il existe au moins un $c \in C$, $D(c) = m$.*

En pratique E et D sont paramétrées par des clefs K_d et K_e. Dans l'informatique contemporaine il y a essentiellement trois types de cryptographies :
- système à clé secrète (symétrique) : $K_e = K_d = K$ (le modèle le plus simple est celui de Jules César) – voir figure 1.1

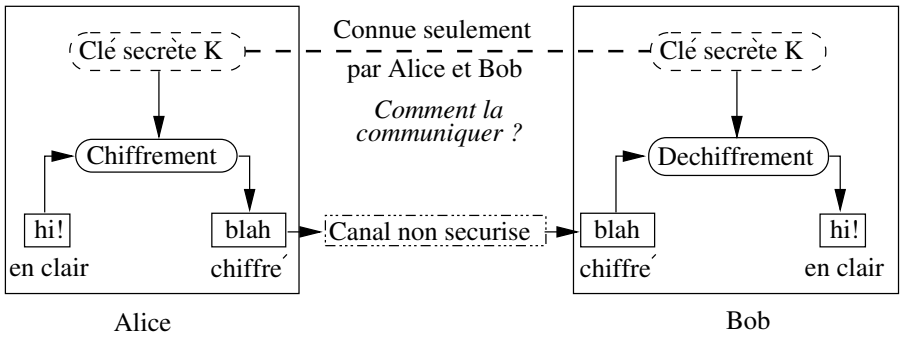

FIGURE 1.1 – Principe du chiffrement à clé privée

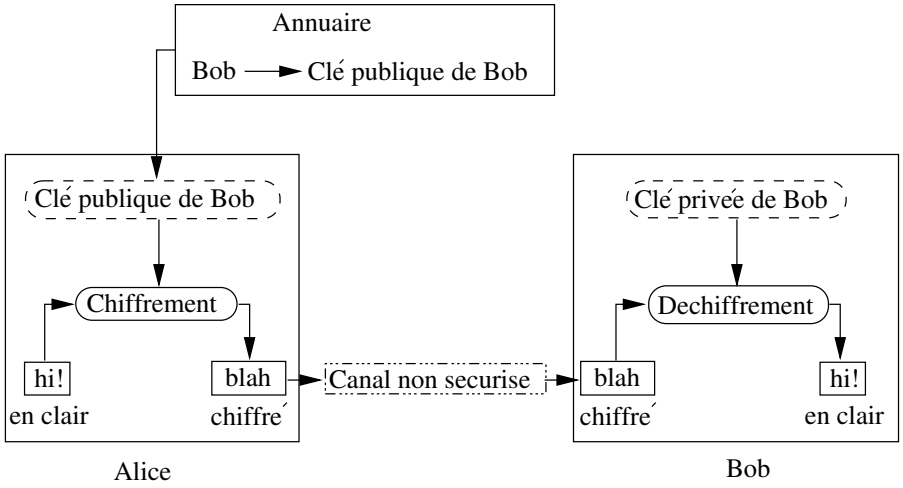

FIGURE 1.2 – Principe du chiffrement à clé publique

– système à clé publique (asymétrique) : $K_e \neq K_d$ (le modèle le plus simple est appelé RSA)
– système hybride (ex : PGP).

et deux types de fonctionnement :

– par flot : chaque nouveau bit est manipulé directement (RC4, Bluetooth, GSM)
– par bloc : chaque message est découpé en blocs (DES,AES, IDEA, RC6,BLOWFISH)

La **cryptographie à clé secrète** (symétrique) nécessite la connaissance de la clé par l'émetteur et par le destinataire. La nécessité de distribution de cette clé sans qu'aucune personne ne l'intercepte entre les intervenants constitue un inconvénient inhérent au système. La fonction de chiffrement doit être inversible et le protocole de cryptage doit prévoir une tierce personne ou un moyen de communication sécurisé. Les principaux algorithmes à clé privée utilisés actuellement sont : Blowfish, DES, 3DES, IDEA, RC2, RC5, RC6, AES(Rijndael).

La **cryptographie asymétrique, à clé publique**, élimine les problèmes liés à la distribution des clés dans le cryptage à clé secrète. Elle utilise une paire de clés pour le cryptage,

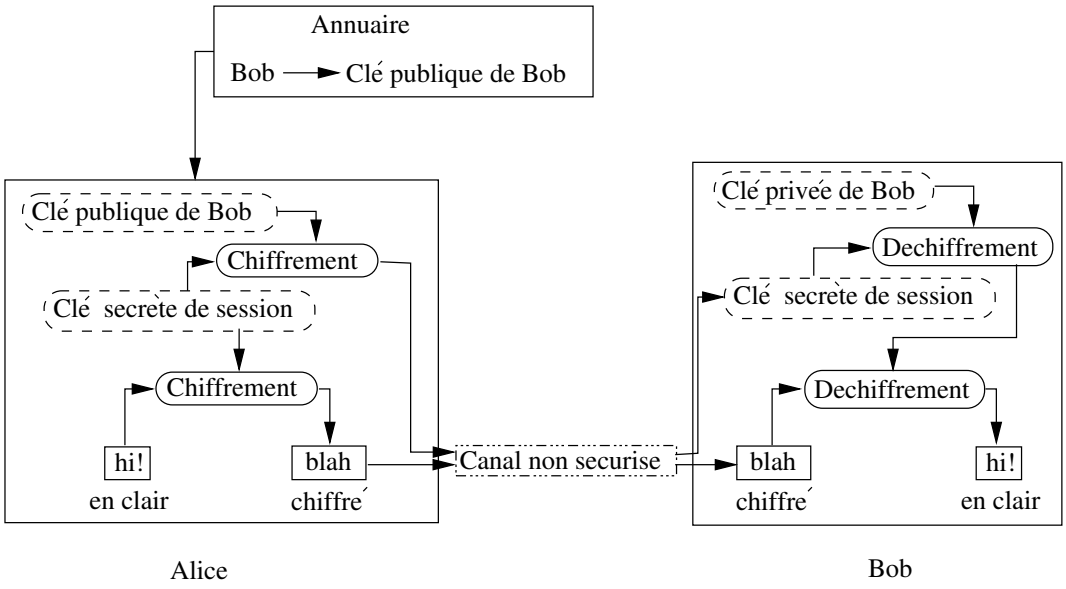

FIGURE 1.3 – Principe du chiffrement PGP

une clé publique qui crypte des données, une clé privée ou secrète correspondant pour le décryptage. Il est impossible de deviner la clé privée à partir de la clé publique. Tout utilisateur disposant d'une copie de la clé publique pourra ensuite crypter des informations que seul le propriétaire de la clé privée pourra déchiffrer. Seulement le cryptage est possible à l'aide de la clé publique. Le décryptage ne sera pas possible au moyen de la clé publique. En fait, apr'es avoir crypté un message avec la clé publique, celui qui a crypté n'est pas capable de décrypter son propre message. Seulement avec la clé privée correspondante le décryptage sera possible. Les principaux algorithmes à clé publique sont RSA et El Gamal.

Le **PGP (Pretty Good Privacy) est un système de cryptographie hybride** Les données sont d'abord compressées pour réduire le temps de transmission et pour surtout pour réduire des modèles existantes dans le texte en clair qui peuvent être exploités en cryptanalyse. Après le PGP crée une clé de session qui est secrète à usage unique. Un algorithme de cryptage symétrique crypte le texte en clair en utilisant la clé de session. La clé de session est aussi cryptée par un algorithme asymétrique a clé publique. Le texte chiffré et la clé de session chiffré sont transmises au destinataire.

1.2 Principaux personnages impliquées dans la cryptographie (selon Ron Rivest et Bruce Schneier)

Utilisateurs légitimes :

– *Alice et Bob* (A et B) : Alice veut envoyer un message à Bob.

– *Trent* (trusted arbitrator) un tiers parti neutre qui joue un rôle exact imposé par le protocole choisi

Adversaires :

– *Eve*(eavesdropper), un attaquant passif qui peut écouter les messages d'Alice et de Bob, sans pouvoir les modifier.
– *Mallory*(malicious attacker ou man in the middle) ou *Oscar* (opponent) ou *Trudy*(intruder), un attaquant actif, qui peut non seulement écouter et enregistrer les messages d'Alice et de Bob, mais aussi les modifier et substituer les siens. Il menace l'intégrité de l'information par altération, destruction, retardement, répétition (engorgement), répudiation des données. De plus il a lieu une usurpation d'identité : l'information reçue est interprétée comme provenant d'une personne autre que son véritable auteur.

1.3 Cryptanalyse

1.3.1 Quelques attaques élémentaires basés sur le résultat du chiffrement

Une tentative de déchiffrement s'appelle attaque. Le principe de Kerkhoff (1883) part de l'hypothèse que l'adversaire connaît tous les détails du système de cryptage utilisé sauf la clé utilisée.

1. Texte chiffré connu : seul le chiffré $C = y$ est connu d'Oscar
2. Texte clair connu : Oscar a obtenu le chiffré $C = y$ et le message $M = x$ correspondant
3. Texte clair choisi : pour tout message $M = x$, Oscar peut obtenir le chiffré $C = y$
4. Texte chiffré choisi : pour tout chiffré $C = y$, Oscar peut obtenir le message $M = x$

Garantir la confidentialité demande qu'il soit impossible de trouver le message $M = x$ à partir de $y = E(x)$. La force d'un système de chiffrement repose sur : la force de l'algorithme, la longueur de la clé utilisée (e.g. $K = 128$ bits) et dans le cas du chiffrement par bloc la taille du bloc (e.g. 64 ou 128 bits). Si l'algorithme est parfait, la seule attaque possible est l'attaque à force brute (ou attaque exhaustive) consistant tout simplement à essayer toutes les clés possibles. Mais aucun moyen de chiffrement n'a été démontré sûr, à l'exception du masque jetable dont le défaut est d'avoir une clé de chiffrement aussi longue que le message à chiffrer, ce qui en limite grandement l'emploi. Shannon a démontré que lorsque la clé de chiffrement est plus courte que le message à chiffrer, le message chiffré révèle forcément des détails sur la structure du message en clair.

L'**attaque par analyse fréquentielle** examine les répétitions des lettres du message crypté afin de trouver la clé. Elle est principalement utilisée contre les chiffrements monoalphabétiques qui substituent chaque lettre par une autre.

L'**attaque par indice de coïncidence** permet de calculer la probabilité de répétitions des lettres du message chiffré. Il est souvent couplé avec l'analyse fréquentielle. Cela permet

de savoir si le chiffrement d'un message est mono-alphabétique ou poly-alphabétique ainsi que la longueur probable de la clé.

L'**attaque par mot probable** consiste à supposer l'existence d'un mot probable dans le message crypté. Il est donc possible d'en déduire la clé du message si le mot choisi est correct. Ce type d'attaque a été menée contre la machine Enigma durant la Seconde Guerre mondiale.

L'**attaque par dictionnaire** consiste à tester tous les mots d'une liste comme mot clé. Elle est souvent couplée à l'attaque par force brute.

L'**attaque par force brute** consiste à tester toutes les solutions possibles de mots de passe ou de clés. C'est le seul moyen de récupérer la clé dans les algorithmes les plus modernes et encore inviolés comme AES.

Definition. [Simple substitution] *Soit* Π *une permutation de l'alphabet* $\{a, b, \ldots, z\}$*. Pour un message* x *le cryptage est* $e_\pi(x) = \pi(x)$*.*

L'espace des clés étant grand : il y a $26! \simeq 4.03 \cdot 10^{26}$ permutations. On emploie une attaque statistique, i.e. on détermine les fréquences des lettres ou on identifie le mot le plus probable.

Definition (***[Cas particuliers de chiffrements avec substitution]***)**.**

[Chiffrement avec déplacement] *Soient* $x, y, k \in \mathbb{Z}_{26}$*.*

Cryptage : $e_k(x) \equiv x + k \bmod 26$

Décryptage : $d_k(y) \equiv y - k \bmod 26$

[Chiffrement affine] *Soient* $x, y, a, b \in \mathbb{Z}_{26}$ *tels que la clé* $k = (a, b)$ *satisfait* $\text{pgcd}(a, 26) = 1$*.*

Cryptage : $e_k(x) = y \equiv a \cdot x + b \bmod 26$

Décryptage : $d_k(y) = x \equiv a^{-1} \cdot (y - b) \bmod 26$

Exemple.

[Chiffrement avec déplacement] *Considérons la clé* $k = 17$*, et le text en clair*

$$attack = x_1, x_2, \ldots, x_6 = 0, 19, 19, 0, 2, 10$$

Le texte chiffré est alors calculé comme étant

$$y_1, y_2, \ldots, y_6 = 17, 10, 10, 17, 19, 1 = rkkrtb$$

[Chiffrement affine] *Considérons la clé* $k = (a, b) = (9, 13)$*, et le texte en clair*

$$attack = x_1, x_2, \ldots, x_6 = 0, 19, 19, 0, 2, 10$$

L'inverse a^{-1} *de* a *existe et est donnée par* $a^{-1} = 3$*. Le texte chiffré est alors calculé comme étant*

$$y_1, y_2, \ldots, y_6 = 13, 2, 2, 13, 5, 25 = nccnfz$$

Dans les deux cas pour un attaque on peut utiliser : ***l'analyse fréquentielle*** *ou la* ***force brute*** *– l'espace des clés étant très petit on peut essayer toutes les clés possibles.*

1.3.2 Attaques basés sur la structure interne de la méthode de chiffrement

La **cryptanalyse linéaire** consiste à faire une approximation linéaire de la structure interne de la méthode de chiffrement. Elle remonte à 1993 et s'avère être l'attaque la plus efficace sur DES.

La **cryptanalyse différentielle** est une analyse statistique des changements dans la structure de la méthode de chiffrement après avoir légèrement modifié les entrées. Avec un très grand nombre de perturbations, il est possible d'extraire la clé. Cette méthode date du début des années 90 mais on sait maintenant que les concepteurs de DES connaissaient une variante de cette attaque nommée attaque-T. Les algorithmes récents (AES, IDEA, etc.) sont conçus pour résister à ce type d'attaques.

Attaquer un chiffrement assuré par la cryptographie asymétrique est plus complexe et lie à l'existence de fonctions à sens unique (one way functions). La conjecture $P \neq NP$ permet de supposer qu'il existe des problèmes calculatoires d'un grand intérêt qui sont "difficiles". Un problème est dans la classe de complexité P s'il a un algorithme lui trouvant la solution en un temps polynomial (en la taille de l'entrée). Par exemple, trier n entiers prend $n \log n$ étapes. Un problème est dans la classe NP s'il existe un algorithme qui peut vérifier en un temps polynomial si une solution candidate l'est vraiment ; si la solution candidate ne l'est en fait pas, le temps ne doit pas nécessairement être polynomial. Pour beaucoup de problèmes "intéressants" qui sont dans NP, on ne connaît par contre pas d'algorithme polynomial pour trouver des solutions : on ne sait donc pas si elles sont dans P. De plus, parmi ces problèmes il y en a qu'on a montré "équivalentes" à toute autre problème de NP – on les appelle NP-complète. Ceci veut dire qu'il suffit de trouver un algorithme polynomial pour un problème NP-complet pour que cet algorithme serve à résoudre tous les problèmes de NP (avec des transformations polynomiales pour passer d'un problème à l'autre).

Definition. *On appelle des* fonctions à sens unique*, les fonctions « faciles » à calculer et « difficile » à inverser.*

L'existence des fonctions à sens unique est une condition nécessaire à l'existence de la plupart des primitives cryptographiques connues (chiffrement et signature numérique). Dans les systèmes de chiffrement asymétrique, l'utilisateur légitime est capable de déchiffrer les messages (en utilisant sa clé privée), alors que pour un adversaire, qui n'a pas cette information privée, la tâche de décryptage ne doit pas être exécutée en temps polynomial par des machines déterministes (ou même probabiliste). La condition nécessaire $P \neq NP$ n'est pas suffisante. Elle implique seulement que le système de chiffrement est difficile à casser dans le pire des cas. Il ne dit rien sur le fait que le système soit facile à casser dans presque tous les cas. Par conséquent, la difficulté dans le pire cas n'est pas une bonne mesure pour la sécurité. À part les fonctions à sens unique, il y encore deux notions de sécurité pour un système de chiffrement asymétrique : la « sécurité sémantique », et la « non-malléabilité ».

Definition. [**Sécurité sémantique. Indistinguabilité des chiffrés**] *Un système de chiffrement à clé publique est dit* sémantiquement sûr *si aucun attaquant probabiliste en temps*

polynomial ne peut apprendre un seul bit d'information sur le clair à partir du chiffré, excepté sa longueur. Cette notion n'est valable que pour des schémas de chiffrement probabilistes.

Definition. [La non-malléabilité] *La* non-malléabilité *formalise l'incapacité d'un adversaire à obtenir un nouveau chiffré y_0 à partir d'un chiffré y de telle sorte que les clairs x et x_0 des chiffrés y et y_0 soient significativement reliés.*

Une relation entre les clairs peut être $x_0 = x + 1$, par exemple. Cette notion capture la notion de résistance du chiffré à une tentative de modification.

1.3.3 Fonctions conjecturées à sens unique : Le problème de la factorisation entière

Soit p et q deux nombres premiers de longueur de bits K. On dit que K représente le paramètre de sécurité et on prend en général $K = 280$ opérations. Pour fixer un ordre de grandeur sur K, on a estimé l'âge de l'univers à 4.10^{11} années, ce qui représente environ 287 instructions sur un ordinateur cadencé à 1 Ghz.

Definition. [Le problème de la factorisation entière] *Étant donné un entier N, calculer des facteurs premiers non triviaux de N (différent de 1 et N).*

Ce problème est facile si N est pair. Les instances difficiles de ce problème semblent être les nombres de la forme $N = pq$ où p et q sont de grande taille équivalente. On peut donc conjecturer que pour p et q deux nombres premiers de longueur K la fonction $f(p, q) = pq$ est à sens unique : c.à-d., il est facile de calculer $N = pq$, étant donnés p et q, en temps $O(K^2)$, mais il n'existe pas d'algorithme probabiliste en temps polynomial qui, sur une entrée aléatoire N de la forme pq, permet de calculer p et q en temps raisonnable, pour des paires (p, q) choisies au hasard parmi les grands nombres premiers. De même, il est facile de générer p et q et de tester leur primalité en temps $O(K^3 \log(K))$.

Remarque. *Pour ces calculs avec des nombres très grands (beaucoup plus grands que ceux de la représentation machine câblée usuelle, c.à-d. 32bit ou 64bit), on peut utiliser des bibliothèques dédiées à de tels calculs, comme la `GMP` (GNU Multiple Precision Arithmetic Library) ainsi que les routines pour des nombres très grands de la bibliothèque crypto `OpenSSL`.*

`GMP` offre un grand nombre de fonctions hautement optimisées pour l'utilisation dans des programmes `C` ou `C++`. En lançant la commande shell `info gmp` on peut en lire la documentation.

`OpenSSL` est un toolkit crypto qui implémente la couche Secure Sockets Layer (SSL v2/v3) ainsi que la couche Transport Layer Security (TLS v1) des protocoles réseau, ainsi que les standards crypto afférents requis. `OpenSSL` est très largement utilisé et donc très bien debuggé. Le protocole a bien entendu besoin de cryptographie, et `OpenSSL` implémente ses propres procédures pour larges nombres, dans sa bibliothèque crypto. En lançant la commande shell `man 3 crypto` on obtient des renseignements généraux sur la bibliothèque, tandis qu'avec la commande shell `man 3 bn` n obtient des renseignements sur les procédures pour larges nombres.

Definition. **[Le problème RSA]** *Étant donné* (N, e, y) *avec* $y \in \mathbb{Z}_N^*$ *et* $N = pq$*, trouver* x *tel que* $y \equiv x^e \pmod N$.

Un nombre N *de la forme* pq *où* p *et* q *sont deux grands nombres premiers est appelé un* module RSA.

L'algorithme de cryptage (RSA) de Rivest, Shamir et Adleman est basé sur le fait que sans connaître la factorisation de N pour tout $y \in \mathbb{Z}_N^*$ arbitraire le problème RSA était « difficile ».En revanche, si on sait résoudre RSA, on ne sait pas si l'on peut factoriser N. La clé publique RSA est la paire d'entiers (N, e). Le createur de la clé privée est d ets le seul qui connaît la factorisation de $N = pq$. Il est le seul à pouvoir calculer $\phi(N) = (p-1)(q-1)$ et par conséquent à pouvoir inverser e pour obtenir sa d, l'inverse $\pmod{\phi(N)}$ de e. Désormais, toute personne ayant accès à la clé publique de Bob peut lui envoyer des messages sécurisés. A aucun moment, Alice n'est intervenue dans la création de la clé publique de Bob.

Trouver l'inverse modulo se fait en utilisant le Théorème Bachet-Bézout et l'algorithme d'Euclide étendu ou bien en utilisant Le petit Théorème de Fermat et son corollaire d'Euler (voir chapitre : Anneaux modulo N).

1.3.4 Fonctions conjecturées à sens unique : Le problème du logarithme discret

Definition. *Un groupe cyclique* G*, est un groupe dans lequel il existe un élément* g *tel que tout élément du groupe puisse s'exprimer sous forme d'un multiple/puissance de* g*, cet élément* g *est appelé générateur du groupe et on note* $G =< g >$.

En notation additive : $(G, +)$ $[n]g$
En notation multiplicative : $(G, \cdot)$ g^n

Definition. *Soit* G *un groupe et* $g \in G$*, alors le groupe sous-groupe* H *généré par* g*, noté* $H =< g >$*, est le plus petit sous-groupe de* G *contenant* g.

Definition. *L'ordre d'un élément* g *d'un groupe* G *est l'ordre du sous-groupe* $H =< g >$ *généré par cet élément. L'ordre de* g *est noté* $ordre(g)$ *ou* $o(g)$*. Si l'ordre est fini, il est le plus petit entier* $m > 0$

- *En notation additive :* $mg = 0$
- *En notation multiplicative :* $g^m = 1$

On peut dire que si l'ordre de g ***est fini*** $ordre(g) = |H| = | < g > |$ ***la cardinalité sous-groupe*** $H =< g >$ ***généré par*** g**.**

Exemple. *Trouver l'ordre de l'élément* 2 *dans*

- $F_{11}^* = (\mathbb{Z}/11\mathbb{Z})^* = \{1, 2, ..., 10\}$ *le groupe multiplicatif de restes modulo* 11
- $F_{17}^* = (\mathbb{Z}/17\mathbb{Z})^* = \{1, 2, ..., 16\}$ *le groupe multiplicatif de restes modulo* 17

Dans $\mathbb{F}_{11}^*$ nous avons $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$, $2^4 \equiv 5$, $2^5 \equiv 10$, $2^6 \equiv 9$, $2^7 \equiv 7$, $2^8 \equiv 3$, $2^9 \equiv 6$. Mais $2^{10} \equiv 1$ et on commence à répéter les éléments.

Donc $ordre(2) = 10$ et il génère tout le groupe G.

Dans $\mathbb{F}_{17}^*$ nous avons $2^0 = 1$, $2^1 = 2$, $2^2 = 4$, $2^3 = 8$,$2^4 = 16$, $2^5 \equiv 15$, $2^6 \equiv 13$, $2^7 \equiv 9$. Mais $2^8 \equiv 1$ et on commence à répéter les éléments. Donc 2 n'est pas un générateur pour $\mathbb{F}_{17}^*$. Il génére que le sous-groupe $H =< 2 >= \{1, 2, 4, 8, 9, 13, 15, 16\}$ et son ordre est la cardinalité de ce sous-groupe, donc 8.

Le problème DLP peut être formulé pour un groupe générique H et $< g >= G \subset H$. On va étudier après le DLP sur des groupes particuliers : le groupe multiplicatif $(\mathbb{Z}/p\mathbb{Z})^*$ et le groupe additif de courbes elliptiques. Pour certains groupes le problème du logarithme discret est "difficile" : soit on ne connaît aucun algorithme sous-exponentiel qui résout le problème DLP, soit on ne connaît aucun algorithme qui résout le problème DLP pour des tailles grandes (disons de 1024 bits et plus).

Definition. [Le problème du logarithme discret DLP] *Soit H un groupe et g un générateur d'ordre q d'un sous-groupe $G \subseteq H$. Étant donnés $y \in G$, trouver x tel que* $y = g^x \Leftrightarrow \log_g(y) = x$.

Le système de chiffrement El Gamal est basé sur le problème du logarithme discret, c'est à dire d'inverser l'exponentielle modulaire. La clé privée de de ElGamal est a. La clé publique de ElGamal est $A = g^a \pmod p$, où $g \in H$ avec un ordre p grand. Décrypter clé publique de ElGamal en partant de sa clé publique revient à résoudre un problème du logarithme discret pour $y = A$.