Les concepts de la cryptographie

Arthur Garnier

$\boxed{\text{Message en clair}} \Rightarrow \boxed{\text{Algo de chiffrement}} \Rightarrow \boxed{\text{Algo de déchiffrement}} \Rightarrow \boxed{\text{Message authentifié}} \Rightarrow$ Message reçu $\Rightarrow$ Code correcteur d'erreur $\Rightarrow$ Message chiffré + authentifié

Si la clé de chiffrement est identique à la clé de déchiffrement, c'est une clé symétrique. Sinon elle est asymétrique.

1 Chiffrement

1.1 Le principe de Kerckhoffs ($\approx$ 1880)

L'algorithme de chiffrement est connu de l'attaquant. Ce principe implique que la sécurité du message ne dépend que du fait que l'attaquant ne connait pas la clé.

- Cacher l'algorithme utilisé d'apporte pas beaucoup de sécurité, car le retrouver n'est pas très difficile en pratique. On peut reconnaitre la forme du chiffré, utiliser le programme qui chiffre pour déduire l'algorithme utilisé (rétro-ingénierie), pot de vin, ...
- Standardisation : Tout le monde utilise les mêmes, qui sont sûres
- Possibilité open source

1.2 Attaques

Un attaquant C (chiffré), il veut le message M (clair), l'objectif pour l'attaquant est de trouver la clé de déchiffrement.

La force brute : J'essaie toutes lec clés unes par unes. Cette méthode fonctionne sur n'importe quel système de chiffrement.

Attaque par canaux auxiliaires : Mesure de la consommation d'énergie, de la puissance, du temps pris par le chiffrement, rayonnement électomagnétique, ...

Les parades au brute force :

- Ralentir l'algo de chiffrement : Pb les utilisateurs sont pénalisés
- Limité le nombre d'essai : Pb pas souvent applicable

Estimer la sécurité : Nombre d'opération ou proba

- Nombre d'opération :
- Processeur 2GHz $\approx$ 2 milliards d'opérations par seconde $= 2 * (2^{10})^3 = 2^{31}$
- Supercalculateur $= 2^5 5$ ops/seconde
- Toutes les machines pour miner Bitcoin $= 2^{57}$ ops/seconde
- Tous les appareils du monde : 2^{63} ops/seconde

- Secondes dans une année = $365 * 24 * 3600 \approx 2^{25}$
- Age de l'univers : 14 milliards d'annnées $\approx 2^{34}$
- Tous les appareils sur terre pendant l'age de l'univers : $2^{63}2^{25}2\{34\} = 2^\{122\}$ ops Essayer une clé : 128 ops = 2^7

Avec toute la puissance pendant toute l'existance de l'univers : $2^{122}/2^7 = 2^{115}$ clés essayés

En pratique, sécurité $=2^{128}$ clés (clé de 128 bits). Une autre mesure souvent utilisée = 2^{128} ops pour casser le système

Le niveau top secret de la NSA = 256 bits

2 Analyse fréquentielle

Si une lettre est toujours remplacée par la même lettre, on analyse les fréquences et on décode.

Le problème est que ce sont des valeurs moyennes. Si le texte est trop petit, l'ordre n'est pas forcément respecté.

3 Meet in the middle (/!\ ≠ Man in the middle)

≈ On coupe le calcul en deux.

Plusieurs méthodes :

- Si un bloc est chiffré de la même façon
- Réponse non chiffrée à un mail chiffré
- Messages stéréotypés
- Forcer quelqu'un à envoyer un message connu

Si on a $C = chif_{k2}(chif_{k1}(m))$ alors $Dec_{k2}(C) = chif_{k1}(m)$

Cout =

- Bruteforce = Nombre possibilité k2 * nombre possibilité k1 = $2^{56} * 2^{56} = 2^{112}$
- Meet in the middle = $2^{56} + 2^{56} = 2^{57}$

Pour cette même raison :

$C = DES_{k3}(DES_{k2}(DES_{k1}(m)))$ Donc $Dec(C) = Chif(Chif(m))$

On a donc 2^{56} chiffrements et $2^{112} \rightarrow 2^{112} + 2^{56}$

3DES a une sécurité de 112 bits.

4 Chiffrement asymétrique

La clé de chiffrement est différente de la clé de déchiffrement (elles sont même totalement décorrellées)

Un autre nom est un chiffrement par clé publique (la clé de chiffrement) et clé privée (la clé de déchiffrement).

N'importe qui peut chiffrer un message qui vous est destiné (car la clé de chiffrement est publique). Vous êtes le **seul** à pouvoir le déchiffrer grâce à la clé privée.

4.1 Intéressant (mais vulnérable parfois) : Diffie-Hellman

Problème mathématique utilisé : Logarithme discret

Un problème difficile à résoudre.

Arithmétique module m :

Les calculs modulo m ramènent toujours tout entre 0 et m. Le problème de calculer x dans : $10^x = 3 modulo 17$

$10^2 = 100 - 5 * 17 = 15$

Ce problème est très compliqué.

Dans le cas du logarithme discret :

Alice et Bob choisissent un nombre g et un nombre n standard.

- Alice : a=random() et $g^a mod n$
- Bob : b=random() et $g^b mod n$

- Alice : $(g^b)^a mod n$
- Bob : $(g^a)^b mod n$

Si la NSA intercepte les communications, elle a g^a et g^b mais elle ne saura pas calculer g^{ab}

Le seul moyen serait de trouver a à partir de $g^a \rightarrow$ logarithme discret

Si la NSA se place entre Alice et Bob, elle peut modifier a en c et b en d par exemple et donc lire et re-chiffrer les messages à la volée. Alice et Bob penseront que les chiffres venant de l'autre sera c et d, alors qu'ils sont en réalité envoyé par la NSA.

En pratique, Diffie-Hellman est utilisé sur un canal sécurité avec des clés éphémères (24h) : Perfect Forward Secrecy

4.2 RSA

Algorithme :

- Choisir 2 nombres premiers : ex $p = 7$ et $q = 11; N = 7 * 11 = 77$
- Calculer $(p-1) * (q-1) = 6 * 10 = 60$
- Choisir e et d tels que $e * d = 1 modulo 60$
- Clé publique = $N = 77$ et $e = 17$
- Clé privée = $d = 53$

Calculer d à partir de e et N $\rightarrow$ très difficile

Message en clair = 5

$5^e mod N$?

$5^{17} mod 77 = 5 * 5^{16} mod 77 = 5 * (5^2)^8 mod 77 = 5 * 25^8 mod 77 = 5 * (25^2)^4 mod 77 = 5 * 9^4 mod 77 = 5 * 4^2 mod 77 = 3$

5 chiffré donne 3

Pour déchiffré : $3^5 3 mod 77 = 5$

Donc :

- Chiffré : $m^e mod N$

- Déchiffré = $m^d mod N$

L'idée : Calculer d à partir de N et e, c'est aussi dur que factoriser N.

Factoriser un nombre de 128bits = 2^{33} opérations.

Pour factoriser $N = p \times q$

- Essayer tous les p : RSA-1024 = 2^{512} essais
- Essayer tous les nombre premiers = $\frac{2^{512}}{512}$ essais (Faire une list de nombres premiers dans un certain intervalle : Crible d'Erathostène = Beaucoup de mémoire et d'opérations)
- NFS : Algorithme pour aller plus vite.

4.3 Courbes elliptiques

Définir des opérations (compliquées) + logarithme discret.

$9 \times 9 \times 9 \times 9x9...x9$ = truc

En faisant l'opération x fois, avec des opérations compliquées (pas un simple multiplication)

Moins de structure et donc moins d'attaque, c'est donc plus sûr.

	RSA	Courbes elliptiques
2^{76}	1024	173
\$2^{106}	2048	230
2^{146}	4096	313

Utilisées dans :

- Passports allemands
- Cartes d'identité en Autriche
- Gmail
- TOR
- What's app
- SSH
- iCloud

Problème : brevets sur courbes elliptiques (Certicom racheté par Blackberry). Ca a donc freiné l'adoption car le brevet RSA expirait à ce moment là.