Sécurisation des communications

I. Algorithmes de chiffrement

1. Principes du chiffrement

Alice veut transmettre un message secret à Bob via un réseau non sécurisé, comme Internet. C'est-à-dire que le message peut être intercepté par une autre personne. Un réseau sécurisé serait par exemple un câble unique allant directement de l'ordinateur d'Alice à celui de Bob sans intermédiaire et sans autre connexion.

Le message doit être chiffré à l'aide d'un algorithme de chiffrement et d'une clé.

Le chiffrement peut être :

- Symétrique : dans ce cas, Alice et Bob partagent la même clé secrète qui leur permet de chiffrer et de déchiffrer les messages.
- Asymétrique : Bob crée deux clés, une clé de chiffrement qu'il rend publique et une clé de déchiffrement qui reste privée (uniquement en possession de Bob). Alice récupère la clé publique et peut chiffrer les messages. Seul Bob, qui possède la clé privée, peut les déchiffrer.
  Si les échanges se font dans les deux sens par chiffrement asymétrique, Alice crée elle aussi deux clés, une clé de chiffrement publique, que Bob utilise pour chiffrer les messages et une clé de déchiffrement privée qui reste en sa possession.

Le chiffrement symétrique est généralement plus léger et plus rapide, mais il faut avoir partagé la clé avant l'échange. Pour cela, Bob crée deux clés de chiffrement asymétrique, transmet la clé publique à Alice. Alice crée une clé de chiffrement symétrique, la chiffre avec la clé publique et la transmet à Bob, qui la déchiffre avec sa clé privée. Alice et Bob échangent ensuite par chiffrement symétrique avec la nouvelle clé. Cette procédure est répétée régulièrement pour renouveler la clé.

2. Algorithmes de chiffrement symétrique (clé partagée)

DES

Le DES, Data Encryption Standard, fut adopté comme standard en novembre 1976 et recommandé par le gouvernement des États-Unis pour toutes les données non classifiées. Il s'agit d'une modification par la NSA d'un algorithme de chiffrement utilisé par IBM depuis 1971.

Il utilise des clés de 56 bits, ce qui donne 2^{56} clés possibles. C'était suffisant dans les années 70, mais en 1998, Deep Crack, une machine à 250 000 dollars construite spécialement à cette fin, déchiffre un message en 56 heures par force brute. En 1999, Deep Crack collabore avec Distributed.net, un projet mondial de calcul distribué, pour déchiffrer un message en 22 heures.
En 1999, DES est remplacé par Triple DES, qui consiste à appliquer trois fois DES avec deux ou trois clés différentes, puis en 2001 par AES.

Un algorithme de chiffrement est considéré comme efficace si on ne connait pas de meilleure technique que la force brute pour déchiffrer le message, c'est-à-dire tester toutes les clés jusqu'à trouver la bonne, ce qui correspond à 2^{64} tests avec une clé de 64 bits, et prendrait 584 ans en testant un milliard de clés par seconde… mais devient envisageable en distribuant le calcul entre de nombreux ordinateurs.

Remarque : pourquoi une clé de 56 bits ? La clé tient sur 64 bits mais seuls 7 bits par octet constituent la clé, le 8^{e} est utilisé pour le contrôle de parité : on le met à 0 ou 1 pour que chaque octet ait un nombre pair de bits à 1. Ainsi, en cas d'erreur de transmission sur la valeur d'un octet, la parité ne sera plus bonne et on pourra détecter l'erreur.

AES

L'algorithme de chiffrement AES, Advanced Encryption Standard, a été adopté par le gouvernement des États-Unis en mai 2002. Il a été développé par deux cryptographes belges. C'est le seul algorithme de chiffrement public approuvé par la NSA pour les données « Top Secret » et toujours le standard actuel pour les chiffrements symétriques. Il utilise des clés de 128, 192 ou 256 bits.

3. Algorithmes de chiffrement asymétrique (clé privée/clé publique)

RSA

Publié en 1978 par Rivest, Shamir et Adleman, c'est toujours l'algorithme de chiffrement asymétrique le plus utilisé, notamment dans le commerce électronique.

Génération des clés

On choisit deux nombres premiers distincts p et q.
On pose $n = pq$ et $\varphi(n) = (p-1)(q-1)$
On choisit un entier e strictement inférieur à $\varphi(n)$ et premier avec $\varphi(n)$.
Le couple (e, n) constitue la clé publique.

D'après le théorème de Bezout, il existe deux entiers d et k tels que $ed = 1 + k\varphi(n)$.
On détermine ce nombre d avec l'algorithme d'Euclide étendu.
Le couple (d, n) ou tout simplement d, puisque n est public, constitue la clé privée.

Chiffrement

Un message est converti en un nombre entier M strictement inférieur à n.
Ensuite, le message chiffré C s'obtient avec $C = M^e \% n$.

Déchiffrement

À partir du message chiffré C on retrouve le message clair M avec $M = C^d \% n$.

Casser RSA

Il faut connaître la clé privée d pour déchiffrer un message.
Or d se calcule sans problème à partir de e que l'on connaît et de $\varphi(n) = (p-1)(q-1)$.
Il suffit donc de déterminer p et q (de factoriser n en produit de facteurs premiers) pour déchiffrer le code.

Si $n = 33$ et $e = 7$, c'est très simple : $p = 3$ et $q = 11$ donc $\varphi(n) = 20$.
Alors $d = 3$ puisque $3 \times 7 = 1 + 1 \times 20$.
On a retrouvé la clé privée à partir de la clé publique !

Mais si n = 189831529945950222945862255570149911207968504515065194812841332487258915303580210693253366427739361931659023238752706132929091594805461098414087548655286637943647335183625968067409060147363574760665948490981220951865977436738924418376921819637107902805969670237487118826680795921260510901121091343625074018804473220239130809076561478191149659885680692527492767852158518862403410573156363617874333653029434948945665729104906621431376277843947101408261107161641173696200868415718206709956955841136973975509449074183505514858349590446747323115239125962035046622855739618936662867222112426747634811975145944656516053347 9 ?

Tous les algorithmes connus pour déterminer p et q à partir de n sont en temps exponentiel : leur durée d'exécution augmente exponentiellement avec la longueur de la clé. Le 2 décembre 2019, le plus grand nombre factorisé par force brute, en utilisant le calcul distribué était long de 795 bits. Les clés RSA font généralement 1024 ou 2048 bits.

En pratique, un algorithme génère les clés, et les stratégies permettant de casser le RSA s'appuient la connaissance des ces algorithmes de génération… Les algorithmes les plus efficaces sont bien sûr top secret.

Le code RSA ne résisterait pas à un ordinateur quantique avec l'algorithme de Shor.
Ces machines ne sont pas encore au point, mais risquent de remettre en question une grande partie des systèmes de sécurité mondiaux.

ECC
Elliptic Curve Cryptography est la cryptographie sur les courbes elliptiques. Une courbe elliptique est une courbe mathématique avec une équation du type $y^2 = x^3 + ax + b$.
Ces courbes ont plusieurs propriétés utilisées en cryptographie.
Un algorithme de chiffrement basé sur les courbes elliptiques avec une clé de 256 bits est comparable à un algorithme RSA avec une clé de 3072 bits.
Les signatures Bitcoin utilisent des courbes elliptiques. La NSA a recommandé l'utilisation d'algorithmes ECC avec des clés de 384 bits pour les informations top secret, mais craint depuis 2015 les attaques futures d'ordinateurs quantiques et travaille sur de nouveaux algorithmes résistants à ces machines.

II. Authentification et intégrité des données

1. Algorithmes de hachage

Un algorithme de hachage permet de vérifier l'intégrité du message.
Il calcule une empreinte correspondant à un message. Il est impossible de retrouver le message à partir de l'empreinte, et toute modification du message change complètement l'empreinte.

Exemple
```
$ echo "Débarquement le 6 juin" | sha1sum
b5639120642c38ffbf05da1cd06d489e7bd95265
$ echo "Débarquement le 5 juin" | sha1sum
a217d4afe3fbd150166a924ffbdd95879f308432
```

La valeur de hachage associée à un fichier ou un message peut être indiquée publiquement. Suite à une transmission, on peut s'assurer que celle-ci est inchangée, et donc que le fichier n'a pas été modifié.
Aussi, le stockage des mots de passe des utilisateurs d'un site peut créer des problèmes de sécurité. Il peut être intéressant de stocker plutôt leurs valeurs de hachage. Les connexions sont toujours sécurisées. Il suffit de comparer les valeurs de hachage lorsque l'utilisateur saisit son mot de passe, et en cas de piratage du site, les pirates ne pourront pas récupérer les mots de passe.

2. Signature électronique

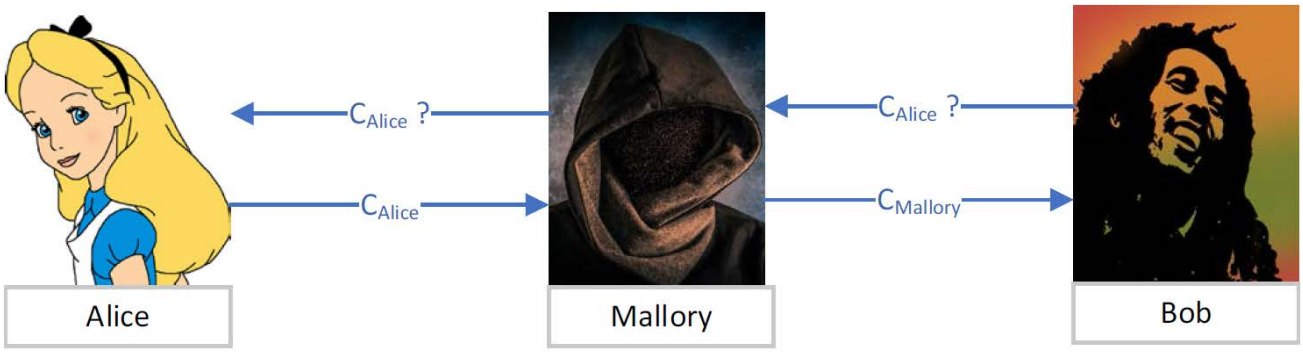
Alice va envoyer un message à Bob. Bob veut être certain que le message vient d'Alice et qu'il n'a pas été modifié.
Alice utilise un chiffrement asymétrique avec une clé de chiffrement privée et une clé de déchiffrement publique (c'est l'inverse pour transmettre des messages chiffrés). Alice transmet à Bob la clé publique et une fonction de hachage par un canal non sécurisé.

Alice crée le hachage du message et le chiffre avec sa clé privée. Elle transmet alors à Bob le message avec le hachage chiffré.
Bob déchiffre le hachage avec la clé publique. Si c'est bien le hachage attendu, il est certain que le message a bien été envoyé par Alice et qu'il n'a pas été modifié.
La signature électronique d'un document en France a la même valeur qu'une signature manuscrite, selon l'article 1367 du code civil.

3. Certificat électronique

Pour vérifier la signature d'Alice, Bob a besoin de sa clé publique C_{Alice}.
Comment Bob peut-il savoir qu'il a la bonne clé publique ?

Problème du Man-In-The-Middle
Mallory, qui veut intercepter les messages, peut s'insérer entre Alice et Bob.



Alice doit prouver son identité. Pour cela, elle passe par un tiers de confiance, une autorité de certification, qui va vérifier physiquement son identité et émettre un certificat électronique. Ce certificat contient :

- la clé publique d'Alice
- des informations d'identification (nom, adresse…)
- une signature de l'autorité de certification dont la clé publique est connue

Alice envoie ce certificat à Bob qui vérifie l'authenticité de la signature et obtient ainsi la clé publique d'Alice.

Les autorités de certification sont des organismes enregistrés et certifiés par des états ou des autorités de contrôle de l'Internet : Amazon Trust Services, CertEurope, Cisco, DigiCert, Google Trust Services LLC, Gouvernement français (ANSSI, DCSSI), Microsoft, Symantec, Visa…
Des clés publiques des principaux organismes sont installées avec les systèmes d'exploitation ou les navigateurs. On peut les consulter :

- sous Windows avec la commande certmgr.msc
- sous Linux avec ls /etc/ssl/certs
- avec Firefox : options → vie privée et sécurité → Afficher les certificats…
- avec Edge : paramètres → confidentialité, recherche et services → Gérer les certificats

III. Exemple du HTTPS

L'HTTPS suit le protocole http, mais rajoute une couche supplémentaire, le protocole TLS, qui a succédé à SSL, pour sécuriser les échanges. Plus généralement, les protocoles se terminant par s sont les protocoles sécurisés par une couche TLS : FTPS, SMTPS, IMAPS…

Initialement, le HTTPS était utilisé pour les transactions financières et la consultation de données privées (emails et réseaux sociaux), mais depuis 2016, suite à une campagne de l'Electronic Frontier Foundation, il est utilisé par de plus en plus de sites pour protéger l'authenticité des pages et des données des utilisateurs.
Le protocole HTTPS se distingue du HTTP :

- par une adresse qui commence par https:// au lieu de http://
- par le port, 443 au lieu de 80
- par un petit cadenas à gauche de la barre d'adresse, que l'on peut cliquer pour consulter le certificat.

Les échanges se déroulent suivant la procédure classique :

- Le client envoie Hello au serveur.
- Le serveur renvoie Hello avec son certificat, qui contient sa clé publique et la signature d'une autorité de certification (CA).
- Le client déchiffre la signature avec la clé publique du CA, vérifiant ainsi la validité du certificat, et envoie alors une clé de chiffrement symétrique chiffrée avec la clé publique du serveur.
- Le serveur déchiffre la clé de chiffrement symétrique avec sa clé privée.
- Les échanges suivants sont chiffrés avec la clé de chiffrement symétrique.

IV. Deux liens pour conclure ce chapitre

Une explication en vidéo
https://www.youtube.com/watch?v=7W7WPMX7arI

Les exercices du concours Alkindi
https://concours-alkindi.fr/main.html#/pagePrevious