Comme des informations confidentielles circulent dans les réseaux, la sécurité des communications est devenue une préoccupation importante des utilisateurs et des entreprises. Tous cherchent à se protéger contre une utilisation frauduleuse de leurs données ou contre des intrusions malveillantes dans les systèmes informatiques.

Sources : Claude Chaudet, Benjamin Monmège, Didier Müller, David Roche, wikipedia

1. Introduction

Soit 2 personnes, Anne et Bob, qui cherchent à s'envoyer des messages par l'intermédiaire d'un réseau informatique. Anne et Bob ne désirent pas qu'une tierce personne soit capable de lire les messages si par hasard ces derniers devaient être interceptés.

Les caractéristiques que l'on cherche à protéger :

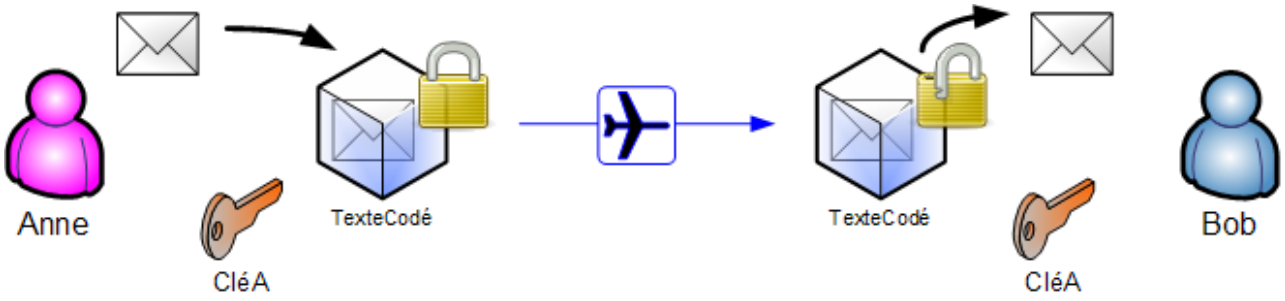
1. La confidentialité : les données échangées ne peuvent être connues que de l'expéditeur et du destinataire. Pour ce faire, Alice va chiffrer le message. Toute personne qui ne possédera pas le moyen de déchiffrer ce message chiffré se verra dans l'impossibilité de comprendre le contenu du message.
2. L'authentification : les interlocuteurs sont réellement qui ils prétendent être. Un certificat numérique est alors nécessaire, car il permet d'utiliser une signature numérique comme moyen d'authentifier des informations numériques.
3. L'intégrité : toute modification (involontaire ou malveillante) est détectée. Dans ce cas, il faut calculer la somme de contrôle (checksum, en anglais) ou « empreinte » du message et la comparer avec celle du message original.

2. La confidentialité

L'idée de chiffrer des messages (de les rendre illisibles pour des personnes non autorisées) ne date pas du début de l'ère de l'informatique. En effet, dès l'antiquité, on cherchait déjà à sécuriser les communications en chiffrant les messages sensibles. Nous nous intéresserons ici uniquement aux communications ayant lieu par l'intermédiaire d'un réseau informatique.

2.1. Le chiffrement symétrique

Pour chiffrer un message, Anne va utiliser une suite de caractère que l'on appelle "clé de chiffrement". Dans le cas du chiffrement symétrique, cette clé de chiffrement sera aussi utilisée par Bob pour déchiffrer le message envoyé par Anne. Dans ce cas, la clé de chiffrement est identique à la clé de déchiffrement.



Ainsi, Anne peut aussi déchiffrer un message en provenance de Bob avec la même clé. Mais il faut au préalable trouver un moyen sûr de transmettre la clé à l'abri des regards. La situation peut cependant devenir complexe, si Anne doit envoyer un message chiffré à Bob et à Charlie mais qu'elle ne souhaite pas donner la même clé à Charlie. Plus le nombre de personnes est grand, plus il est difficile de gérer les clés symétriques. D'autant qu'il faut au préalable trouver un moyen sûr de transmettre la clé.

La méthode la plus utilisée en matière de chiffrement symétrique se nomme AES[1]. C'est le même fonctionnement dans le cadre du protocole TLS/SSL.

TLS[2] est le successeur du protocole SSL[3]. Le protocole TLS/SSL est un système de règles communes suivies aussi bien par les clients que par les serveurs lorsqu'un client visite un site web sécurisé par HTTPS.

Avant que le client et le serveur ne puissent initier une communication sécurisée, ils doivent passer par une procédure nommée « négociation TLS » – le résultat étant une clé de session, permettant des transmissions de données chiffrées entre le serveur et le client. La clé de session est symétrique comme elle est utilisée par le client et par le serveur pour chiffrer et déchiffrer des transmissions, de sorte que seul ceux ayant la clé de session identique puisse déchiffrer et lire ces contenus.

2.2. Le chiffrement asymétrique

Le gros problème avec le chiffrement symétrique, c'est qu'il est nécessaire pour A et B de se mettre d'accord à l'avance sur la clé qui sera utilisée lors des échanges. Le chiffrement asymétrique permet d'éviter ce problème.

La propriété des algorithmes asymétriques est qu'un message chiffré par une clé privée sera lisible par tous ceux qui possèdent la clé publique correspondante. À l'inverse, un message chiffré par une clé publique n'est lisible que par le propriétaire de la clé privée correspondante.

Ainsi avec sa clé privée, Anne :

- signe ses messages ;
- lit (déchiffre) les messages qui lui sont adressés.

Le chiffrement asymétrique repose sur des problèmes très difficiles à résoudre dans un sens et faciles à résoudre dans l'autre sens.

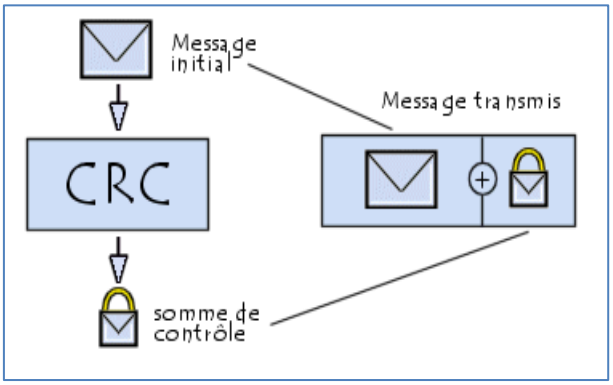
Prenons un exemple : l'algorithme de chiffrement asymétrique RSA[4], est très couramment utilisé, notamment dans tout ce qui touche au commerce électronique. RSA se base sur la factorisation des très grands nombres premiers. Si vous prenez un nombre premier p (par exemple p = 16813007) et un nombre premier q (par exemple q = 258027589), il facile de déterminer n le produit de p par q (ici on a p x q = n avec n = 4338219660050123). En revanche si on vous donne n (ici 4338219660050123) il est très difficile de retrouver A et B dans un temps "raisonnable".

[1] Advanced Encryption Standard
[2] Transport Layer Security (sécurité de couche de transmission)
[3] Secure Sockets Layer (couche de sockets de sécurité)
[4] du nom de ses 3 inventeurs : Rivest Shamir et Adleman

3. Intégrité

Une somme de contrôle est un moyen simple pour garantir l'intégrité de données en détectant une altération lors d'une transmission de données.

Le principe est d'ajouter aux données des éléments dépendant de ces dernières — on parle de redondance — et simples à calculer. Cette redondance accompagne les données lors d'une transmission ou bien lors du stockage sur un support quelconque. Plus tard, il est possible de réaliser la même opération sur les données et de comparer le résultat à la somme de contrôle originale, et ainsi conclure sur la corruption potentielle du message.



Le **contrôle de redondance cyclique** (noté **CRC**[5]) est un moyen de contrôle d'intégrité des données puissant et facile à mettre en œuvre. Il représente la principale méthode de détection d'erreurs utilisée dans les télécommunications.

Les **fonctions de hachage cryptographique** représentent un autre moyen qui, à une donnée de taille arbitraire, associe une image de taille fixe, et dont une propriété essentielle est qu'elle est pratiquement impossible à inverser. Si l'image d'une donnée par la fonction se calcule très efficacement, le calcul inverse d'une donnée d'entrée ayant pour image une certaine valeur se révèle impossible sur le plan pratique. Pour cette raison, on dit d'une telle fonction qu'elle est *à sens unique*.

La valeur de sortie est souvent appelée valeur de hachage, empreinte numérique, empreinte, ou encore haché[6].

Les algorithmes MD5[7], SHA-1[8], SHA-256, SHA-512… sont des fonctions de hachage cryptographique qui permettent d'obtenir une empreinte numérique d'un fichier ou d'un message avec une très forte probabilité (quasi unique) que deux fichiers différents donnent deux empreintes différentes.

L'algorithme MD5 calcule une empreinte de 128 bits (ou 32 caractères hexadécimaux), tandis que le SHA-1 calcule une empreinte de 160 bits (ou 40 caractères hexadécimaux).

Exemple : empreinte du texte « Hello world! »

- hachage MD5 : 86fb269d190d2c85f6e0468ceca42a20.
- hachage SHA-1 : d3486ae9136e7856bc42212385ea797094475802.

Attention : les fonctions de hachage MD5 et SHA1 sont **obsolètes**. Pour plus d'informations : Chiffrer, garantir l'intégrité ou signer | CNIL.

SHA1 va être progressivement remplacé par d'autres fonctions plus sécurisées, comme le SHA2 ou SHA3, ou encore bcrypt qui est considérée comme une des meilleures fonctions de hachage à ce jour. En outre, chrome et Firefox n'acceptent plus les certificats SHA1.

4. Authentification

En se référant au précédent paragraphe nous voyons rapidement que, lorsqu'une entité (entreprise, association, individu, service public, etc.) veut sécuriser ses communications (entrantes et sortantes) auprès d'un large public, le chiffrement le plus simple est l'asymétrique à clé publique : l'entité n'a qu'à diffuser sa clé publique à l'ensemble de son audience.

Le problème vient de la transmission de la clé publique. Si celle-ci n'est pas sécurisée, un attaquant peut se positionner entre l'entité et son public en diffusant de fausses clés publiques (par le biais d'un faux site web

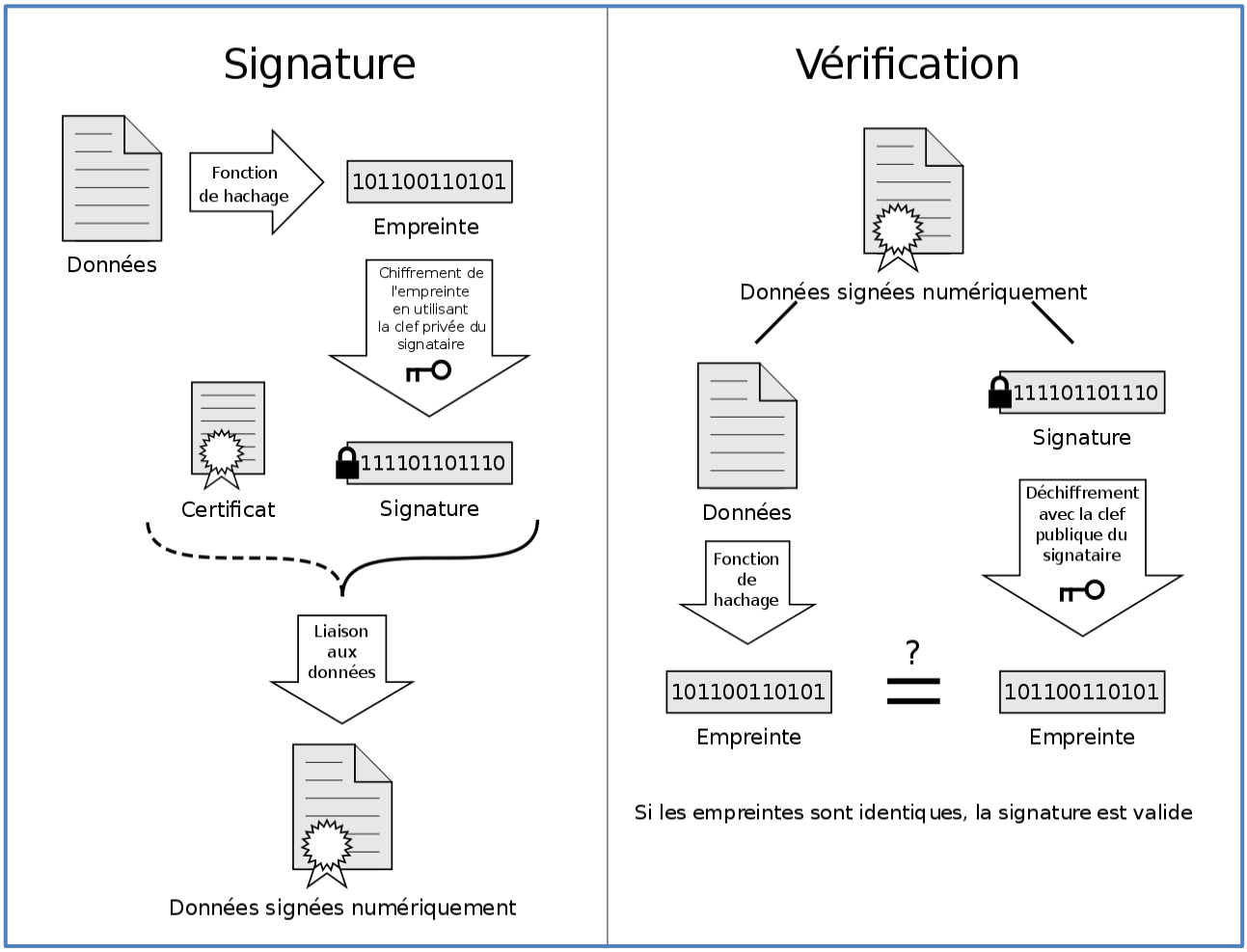
[5] Cyclic Redundancy Check
[6] message digest ou digest, hash
[7] Message Digest 5
[8] Secure Hash Algorithm

par exemple) puis intercepter toutes les communications, lui permettant d'usurper l'identité du diffuseur de clés publique et de créer une attaque de l'homme du milieu.

Les certificats résolvent le problème du canal sécurisé grâce à la signature de tiers de confiance.

Un certificat électronique est un ensemble de données contenant :

- au moins une clé publique ;
- des informations d'identification, par exemple : nom, localisation, adresse électronique ;
- au moins une signature (construite à partir de la clé privée) ; de fait quand il n'y en a qu'une, l'entité signataire est la seule autorité permettant de prêter confiance (ou non) à l'exactitude des informations du certificat.



Les certificats électroniques respectent des standards spécifiant leur contenu de façon rigoureuse. Les deux formats les plus utilisés aujourd'hui sont :

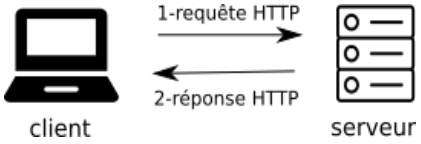
- X.509, défini dans la RFC 5280
- OpenPGP, défini dans la RFC 4880

5. Le protocole HTTPS

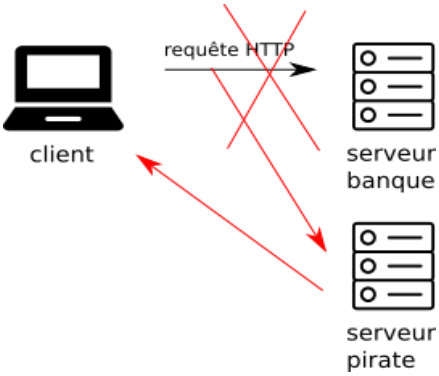
Nous allons maintenant voir une utilisation concrète de ces chiffrements symétriques et asymétriques : le protocole HTTPS.

Avant de parler du protocole HTTPS, petit retour sur le protocole HTTP : un client effectue une requête HTTP vers un serveur, le serveur va alors répondre à cette requête (par exemple en envoyant une page HTML au client).

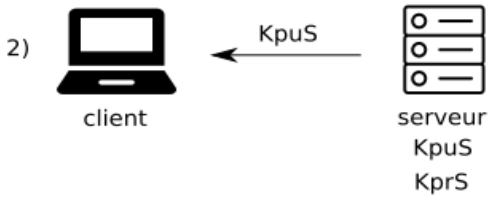
Le protocole HTTP pose 2 problèmes en termes de sécurité informatique :



➢ Un individu qui intercepterait les données transitant entre le client et le serveur pourrait les lire sans aucun problème (ce qui serait problématique notamment avec un site de e-commerce au moment où le client envoie des données bancaires)



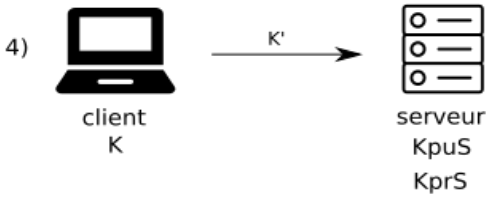
➢ grâce à une technique qui ne sera pas détaillée ici (le DNS spoofing), un serveur "pirate" peut se faire passer pour un site sur lequel vous avez l'habitude de vous rendre en toute confiance : imaginez vous, voulez consulter vos comptes bancaires en ligne, vous saisissez l'adresse web de votre banque dans la barre d'adresse de votre navigateur favori, vous arrivez sur la page d'accueil d'un site en tout point identique au site de votre banque, en toute confiance, vous saisissez votre identifiant et votre mot de passe. C'est terminé un "pirate" va pouvoir récupérer votre identifiant et votre mot de passe ! Pourquoi ? Vous avez saisi l'adresse web de votre banque comme d'habitude ! Oui, sauf que grâce à une attaque de type "DNS spoofing" vous avez été redirigé vers un site pirate, en tout point identique au site de votre banque. Dès vos identifiant et mot de passe saisis sur ce faux site, le pirate pourra les récupérer et se rendre avec sur le véritable site de votre banque. À noter qu'il existe d'autres techniques que le DNS spoofing qui permettent de substituer un serveur à un autre, mais elles ne seront pas évoquées ici.



Le protocole HTTPS est donc la version sécurisée de HTTP, le but de HTTPS est d'éviter les 2 problèmes évoqués ci-dessus. HTTPS s'appuie sur le protocole TLS anciennement connu sous le nom de SSL.

Comment chiffrer les données circulant entre le client et le serveur ?

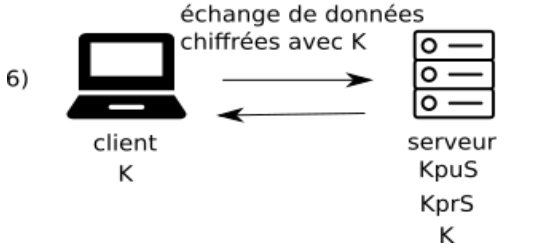
Les communications vont être chiffrées grâce à une clé symétrique. Problème : comment échanger cette clé entre le client et le serveur ? Simplement en utilisant une paire clé publique / clé privée !



Voici le déroulement des opérations :

1. le client effectue une requête HTTPS vers le serveur, en retour le serveur envoie sa clé publique (KpuS) au client
2. le client "fabrique" une clé K (qui sera utilisé pour chiffrer les futurs échanges), chiffre cette clé K avec KpuS et envoie la version chiffrée de la clé K au serveur
3. le serveur reçoit la version chiffrée de la clé K et la déchiffre en utilisant sa clé privée (KprS). À partir de ce moment-là, le client et le serveur sont en possession de la clé K
4. le client et le serveur commencent à échanger des données en les chiffrant et en les déchiffrant à l'aide de la clé K (chiffrement symétrique).

Ce processus se répète à chaque fois qu'un nouveau client effectue une requête HTTPS vers le serveur.



Comment éviter une attaque DNS Spoofing

Pour éviter tout problème, il faut que le serveur puisse justifier de son "identité" (voici la preuve que je suis bien le site de la banque B et pas un site "pirate"). Pour se faire, chaque site désirant proposer des transactions HTTPS doit, périodiquement, demander (acheter dans la plupart des cas) un certificat d'authentification (sorte de carte d'identité pour un site internet) auprès d'une autorité habilitée à fournir ce genre de certificats (chaque navigateur web possède une liste des autorités dont il accepte les certificats). Comme dit plus haut, ce certificat permet au site de prouver son "identité" auprès des clients.

Le serveur envoie ce certificat au client en même temps que sa clé publique (étape 2 du schéma ci-contre).

En cas d'absence de certificat (ou d'envoi de certificat non conforme), le client stoppe immédiatement les échanges avec le serveur. Il peut arriver de temps en temps que le responsable d'un site oublie de renouveler son certificat à temps (dépasse la date d'expiration), dans ce cas, le navigateur web côté client affichera une page de mise en garde avec un message du style "ATTENTION le certificat d'authentification du site XXX a expiré, il serait prudent de ne pas poursuivre vos échanges avec le site XXXX".