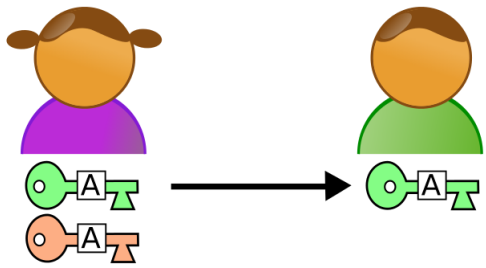
1. Chiffrement asymétrique

Lors du chiffrement asymétrique, deux clés sont utilisées, l'une privée (secrète) et l'autre publique. La clé publique permet de chiffrer le message, et la clé privée permet de le déchiffrer. Comme leur nom l'indique, la clé privée n'est pas transmise, contrairement à la clé publique.
Le principe a été imaginé en 1976 par Diffie et Hellman, mais sans algorithme correspondant. En 1978 Rivest, Shamir et Adleman inventent l'algorithme RSA. Les services secrets anglais avaient déjà imaginé des concepts semblables quelques années plus tôt, mais ont gardé ces recherches secrètes jusqu'en 1997.
Ces protocoles reposent sur l'asymétrie des fonctions utilisées (fonctions dites à sens unique). Par exemple, dans RSA on utilise le produit de deux nombres premiers. Autant il est facile d'effectuer une multiplication de deux nombres premiers, autant il est difficile d'obtenir la décomposition de ce produit en les deux nombres d'origine.

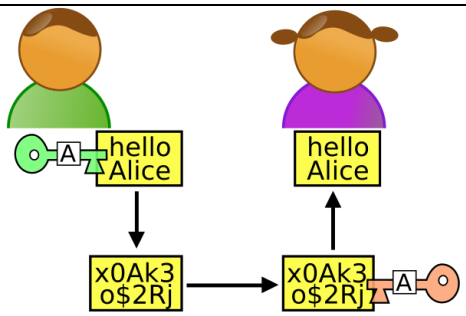
2. Principe

Alice souhaite recevoir un message secret de Bob, sur un canal susceptible d'être écouté par Ève.

- Alice génère deux clés (sa clé publique et sa clé privée).
- Elle transmet à Bob la clé publique et conserve précieusement la clé privée.
- Bob chiffre son message avec la clé publique et l'envoie à Alice.
- Alice déchiffre le message avec sa clé privée.
- Si Ève intercepte le message, elle ne peut pas le déchiffrer, ne disposant pas de la clé privée.



Génération des clés par Alice



Chiffrement par Bob et déchiffrement par Alice

On peut compléter ce protocole avec un mécanisme d'authentification (signature des messages) ou des certificats de sécurité. En effet, Alice ne peut pas savoir si un message reçu provient de Bob ou d'Ève, vu qu'Ève dispose également de la clé publique (cf. ci-après)

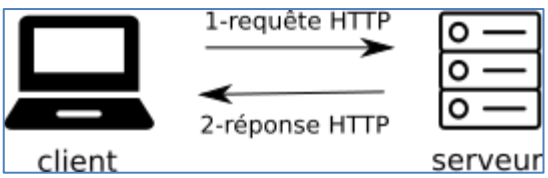
3. Protocoles hybrides

Les protocoles asymétriques sont difficiles à casser mais lents. Les protocoles symétriques sont rapides, mais nécessitent l'échange préalable des clés. Les protocoles hybrides utilisent les deux méthodes. RSA peut par exemple être utilisé pour échanger une clé de protocole symétrique.
Par exemple, TLS/SSL (le "https") utilise en partie un chiffrement asymétrique, pour initialiser la connexion client/serveur, et créer une clé symétrique utilisée pour la suite de la session. Ce protocole mélange donc les deux types de chiffrement. Cette méthode est couplée avec un mécanisme d'authentification : l'utilisation des certificats de sécurité, confiés à un tiers de confiance.

4. Le protocole HTTPS

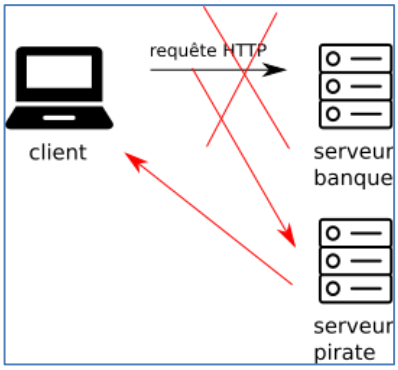
Nous allons maintenant voir une utilisation concrète de ces chiffrements symétriques et asymétriques : le protocole HTTPS.

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). Si nécessaire n'hésitez pas à consulter ce qui a été fait en première pour plus de détails.



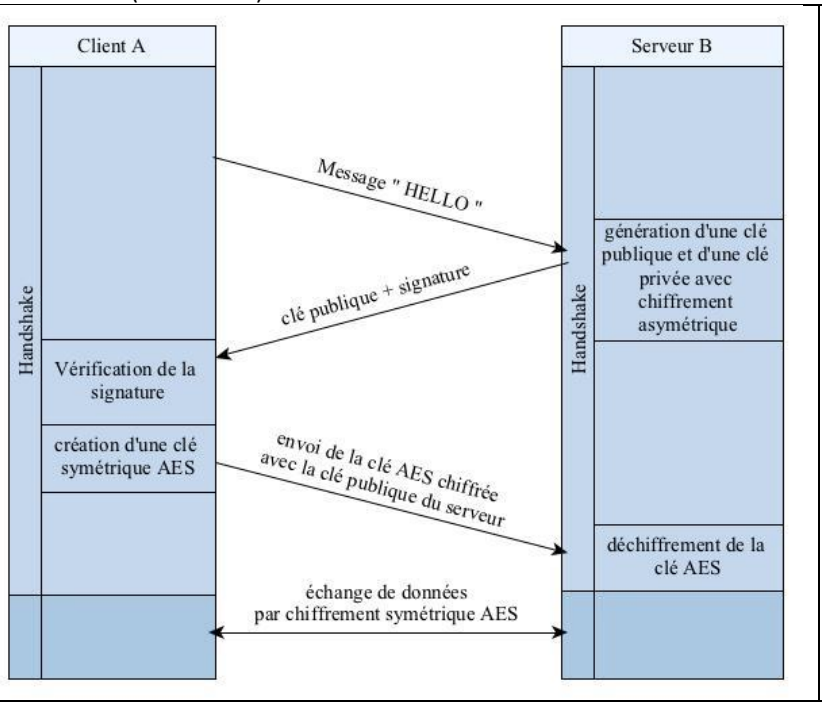
Le protocole HTTP pose 2 problèmes en termes de sécurité informatique :

- 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.



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 TSL (Transport Layer Security) anciennement connu sous le nom de SSL (Secure Sockets Layer)

Assurer la sécurité des machines reliées à Internet c'est être sûr de l'identité de la machine distante, sécuriser les échanges, limiter l'accès à certaines données. L'une des solutions possibles est la mise en œuvre du protocole HTTPS
Le protocole http permet d'échanger des données entre client et serveur. En première, on voit deux possibilités de http avec les méthodes GET et POST. Ce protocole n'est pas sécurisé, il n'est donc pas adapté à tous les échanges de données sur le web. https rajoute à http une couche de sécurité, la couche TLS (plus sûre que SSL).
Les protocoles de communication commencent par une mise en relation des entités. On appelle souvent cette mise en relation la « poignée de mains » (handshake).



- La vérification de la signature se fait auprès d'autorités extérieures (les tiers de confiance)
- La clé symétrique est une clé AES, qui est chiffrée avec la clé publique du serveur. Elle ne sert que pour une seule session : une autre clé est générée s'il y a déconnexion et reconnexion.
- La clé symétrique est déchiffrée par le serveur avec sa clé privée : on utilise deux protocoles successivement. Protocole asymétrique pour la poignée de mains, puis protocole symétrique pour l'échange des données.

Plus de détails, tout en restant dans la vulgarisation, sur cette bande dessinée (en anglais) : https://howhttps.works/
Test des vulnérabilités de votre navigateur : https://clienttest.ssllabs.com:8443/ssltest/viewMyClient.html