NSI · TERMINALE / COURS

Sécuriser les communications

Astrolab Cours · astrolabcours.fr

01 Le problème

Quand tu tapes ton mot de passe sur un site, il traverse des dizaines de routeurs que n'importe qui peut écouter. Comment l'envoyer pour que seul le site puisse le lire, sans vous être jamais rencontrés pour convenir d'un secret ?

01 L'idée : le cadenas ouvert par la poste

Le site t'envoie un cadenas **ouvert** dont il garde seul la clé. Tu mets ton message dans une boîte, tu fermes le cadenas, tu renvoies. N'importe qui peut voir passer le cadenas ouvert, ça ne sert à rien : une fois fermé, seul le site peut l'ouvrir. Le cadenas, c'est la clé publique ; la clé du cadenas, c'est la clé privée.

01 Le chiffrement symétrique

> La **même clé** sert à chiffrer et à déchiffrer.

Le **chiffre de César** décale chaque lettre (clé 3 : A devient D). Le **XOR** combine chaque bit du message avec un bit de la clé ; appliquer deux fois le même XOR redonne le message. **AES** est l'algorithme symétrique actuel, très rapide et sûr.

Le problème : il faut d'abord se transmettre la clé en secret. Si on l'envoie en clair, un espion la récupère.

01 Le chiffrement asymétrique

> Chacun a deux clés liées : une **clé publique**, donnée à tout le monde, et une **clé privée**, gardée secrète. Ce qui est chiffré avec la clé publique ne se déchiffre qu'avec la clé privée.

RSA repose sur le fait qu'il est facile de multiplier deux grands nombres premiers, mais extrêmement long de retrouver ces deux facteurs à partir du produit.

HTTPS : les deux à la fois

L'asymétrique est sûr mais lent ; le symétrique est rapide. HTTPS combine les deux :

1. le serveur envoie sa clé publique, dans un **certificat** signé par une autorité de confiance ;
2. le navigateur choisit une clé symétrique, la chiffre avec la clé publique du serveur et l'envoie ;
3. toute la suite de l'échange est chiffrée en AES avec cette clé.

L'homme du milieu

Un attaquant peut se placer entre toi et le site et t'envoyer **sa** clé publique en se faisant passer pour le site.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/securiser-les-communications/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits

Le certificat sert à l'empêcher : il prouve que la clé publique appartient bien au site.

Exemple guidé

Un exemple de chiffrement asymétrique avec de petits nombres (le principe de RSA). La clé publique est $(n, e) = (55, 3)$, la clé privée $d = 27$.

1. **Chiffrer** le message $m = 7$ avec la clé publique : $7^3 = 343$, et $343 \bmod 55 = 13$. On envoie 13.
2. **Déchiffrer** avec la clé privée : $13^{27} \bmod 55 = 7$. En Python, `pow(13, 27, 55)` donne 7.
3. Pour trouver d, il faut connaître les deux nombres premiers dont le produit fait n (ici 5×11). Avec 55, c'est immédiat. Avec un n de plus de 600 chiffres, aucun ordinateur actuel n'y arrive en un temps raisonnable.

En pratique, ce calcul lent ne sert qu'à échanger une **clé symétrique**, avec laquelle la suite de la conversation est chiffrée, beaucoup plus vite.

Retour au problème

Le site envoie sa clé publique, ton navigateur s'en sert pour lui transmettre une clé secrète que lui seul peut lire. Le mot de passe voyage ensuite chiffré avec cette clé : les routeurs ne voient passer que du bruit.

Les pièges

- Chiffrer avec sa propre clé publique : seul soi-même pourra déchiffrer.
- Le cadenas de HTTPS garantit que la connexion est chiffrée, pas que le site est honnête.
- César se casse en essayant les 25 décalages : ce n'est pas un chiffrement sûr.

© Astrolab Cours · astrolabcours.fr/nsi/terminale/securiser-les-communications/ · Tous droits réservés : usage personnel uniquement, republication et usage commercial interdits