

Prévention

Avoir les réflexes préventifs



Compétence demandée :
**Comprendre les mécanismes
préventifs**

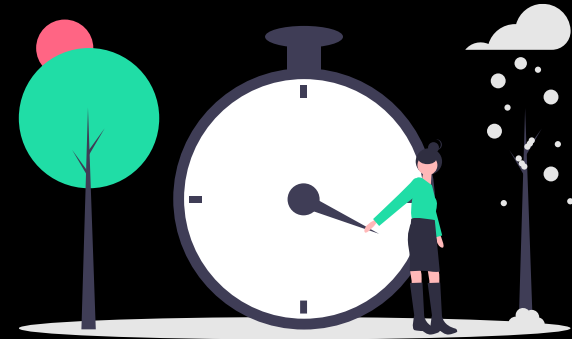
RAPPEL SUR LES VULNERABILITES

Qui peut expliquer les
vulnérabilités suivantes ?



1. Les attaques MITM
2. Le phishing
3. Les attaques CSRF
4. Les modifications du DOM
5. Les injections
6. Le brute force

A votre avis, quelles sont les
mécanismes préventifs ?



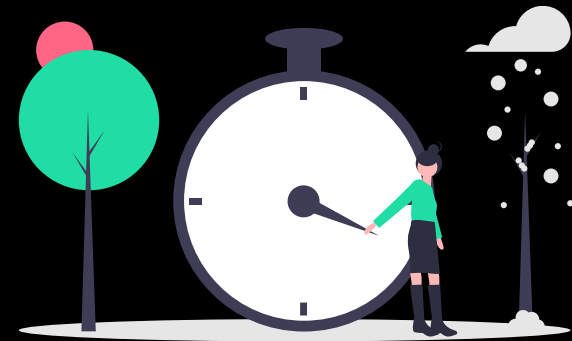
1. Les attaques MITM
2. Le phishing
3. Les attaques CSRF
4. Les modifications du DOM
5. Les injections
6. Le brute force

1. Les attaques MITM : HTTPS
2. Le phishing : Eduquer les visiteurs
3. Les attaques CSRF : Token CSRF
4. Les modifications du DOM : Faire les vérifications au backend
5. Les injections : Figer la forme de la requete SQL et désactiver les balises HTML
6. Le brute force : Mettre une limite sur le nombre d'essais

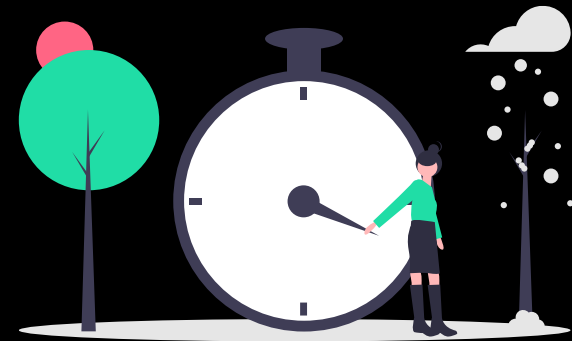


1. Les attaques MITM : HTTPS
2. Le phishing : Eduquer les visiteurs
3. Les attaques CSRF : Token CSRF
4. Les modifications du DOM : Faire les vérifications au backend
5. Les injections : Figer la forme de la requete SQL et désactiver les balises HTML
6. Le brute force : Mettre une limite sur le nombre d'essais

A votre avis, qu'est-ce que
HTTPS ?



Avant d'y répondre, à votre avis,
qu'est-ce qu'un protocole ?



Un **protocol** est une **convention de communication entre logiciels**. Il existe plusieurs protocoles (donc plusieurs conventions de communication) dont les noms sont **HTTP**, **HTTPS**, FTP, SFTP, SSH etc.

Le protocol HTTP

Le **protocol HTTP** spécifie les conventions suivantes dans une communication :

- Le **destinataire** est spécifié par une URL (un lien)
- Une **méthode** spécifie le moyen de transport des données (GET ou POST)

Le protocol HTTPS

A votre avis, qu'est-ce que
HTTPS a en plus que HTTP ?



Le **protocol HTTPS** spécifie les conventions suivantes dans une communication :

- Le **destinataire** est spécifié par une URL (un lien)
- Une **méthode** spécifie le moyen de transport des données (GET ou POST)
- Les données seront **cryptées par une clé symétrique**

A votre avis qu'est-ce qu'une clé ?



La cryptographie symétrique

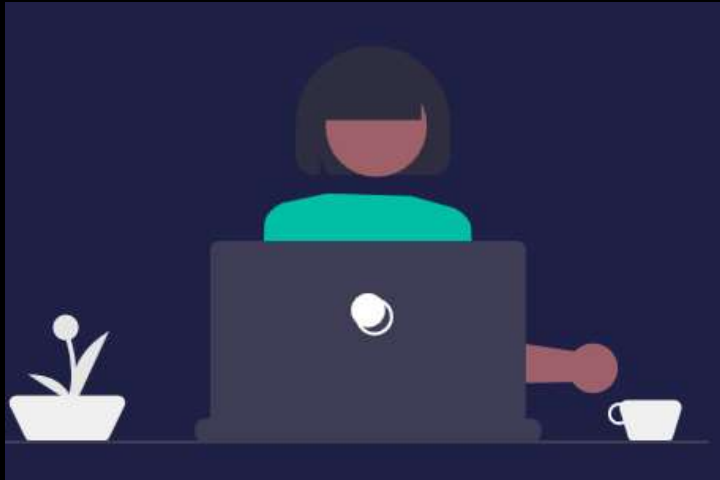
Un petit exemple ...

Pour simplifier le problème, nous allons prendre un exemple basé sur le **décalage des lettres** dans un message.

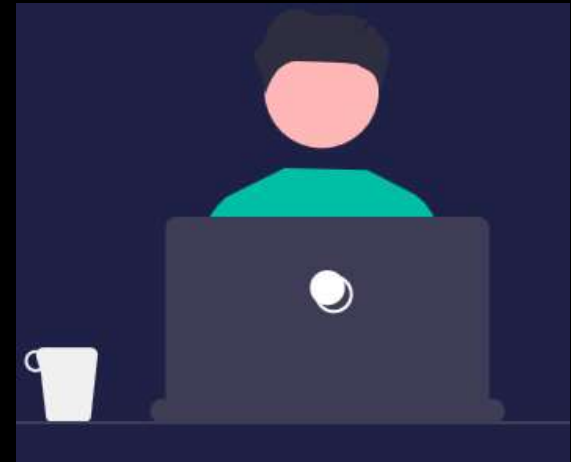
Dans la réalité, c'est beaucoup plus compliqué et implique beaucoup de mathématiques !

Je vous présente ...

Alice



Bob



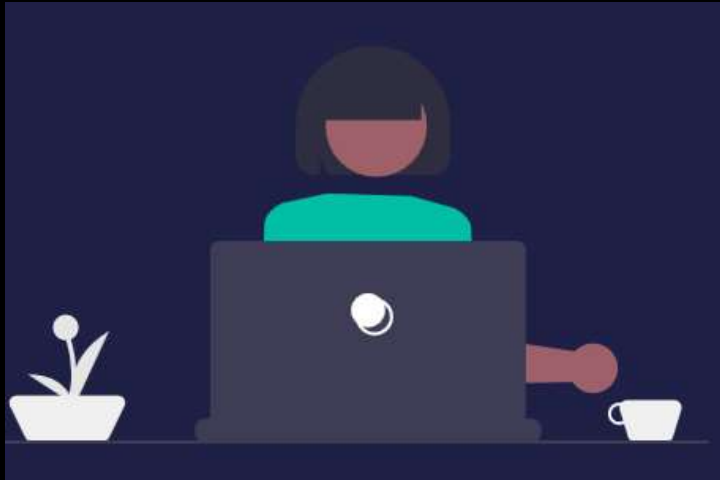
Besoin : **Alice souhaite communiquer avec Bob** sur la
sale coupe de cheveux de son responsable Colin

Problème : **Alice a peur** que son responsable Colin fasse
une attaque MITM et lise son message pour Bob

Solution : **Alice communique une clé et un cipher** à Bob
pour cacher leur message (crypter le message)

Exemple cryptographie symétrique

Alice



Alice choisit « 5 » comme **clé**

Alice choisit « le decalage de lettres » comme **cipher**

Alice transmet ces 2 choses à Bob à l'avance en cachette

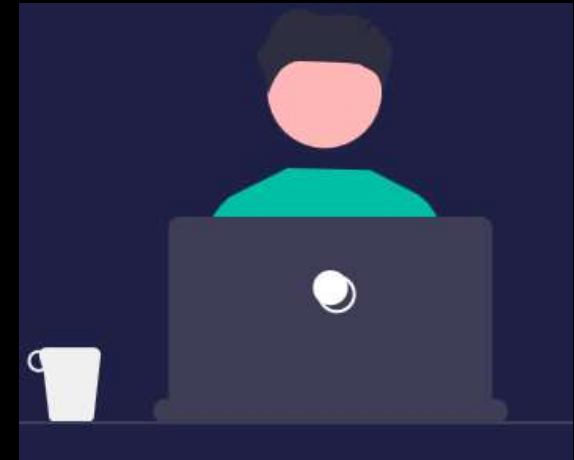
Exemple cryptographie symétrique

Bob dit :

Ok Alice, la clé est « 5 »

Ok Alice, le cipher est le
« décalage de lettres »

Bob



1. Processus de chiffrement symétrique

Alice



Alice veut dire « Colin a une sale coupe » à Bob.

Elle va exécuter le cipher avec la clé, donc elle va décaler les lettres de 5 et obtenir :
« Htqns%f%zsj%xfqj%htzuj »

Pourquoi « Colin a une sale
coupe » donne
« Htqns%f%zsj%xfqj%htzuj »
une fois chiffré ?



On utilise le **cipher** (c'est la méthode de cryptage) :
« décalage de lettres »

On utilise la **clé** (c'est un nombre) pour savoir de
combien de lettres on doit décaler : c'est 5

Donc « Colin », donne « Htqns » car :
« C » 5 lettres après, donne « H »
« o » 5 lettres après donne « t »,
« l » 5 lettres après donne « q »,
« i » 5 lettres après donne « n »,
« n » 5 lettres après donne « s »

Et ainsi de suite ...

Donc le message :

« Colin a une sale coupe »

Est crypté en :

« Htqns%f%zsj%xfqj%htzuj »

Alice a chiffré son message !

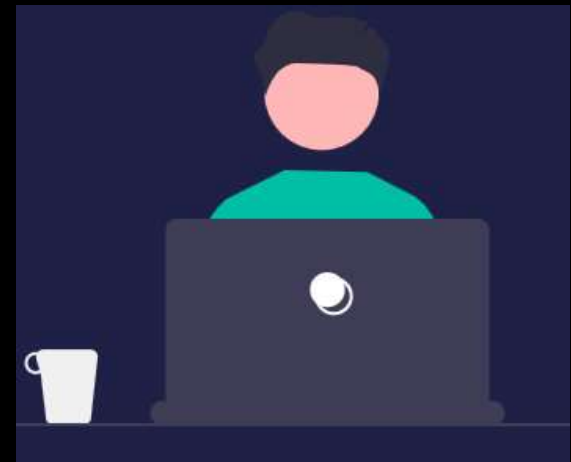
Alice



« Colin a une salle
coupe »

« Htqns%f%zsj%xfqj%
htzuj »

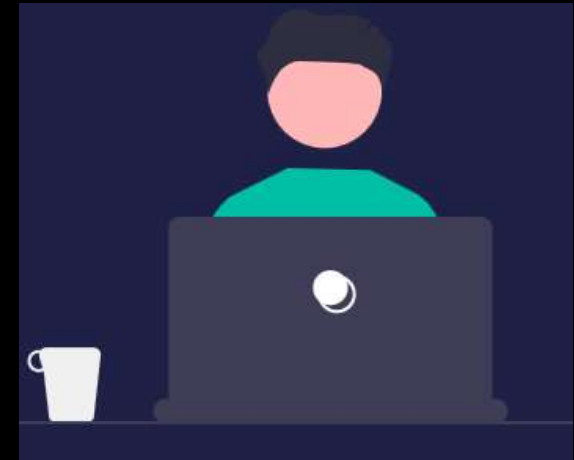
Bob



2. Processus de déchiffrement symétrique

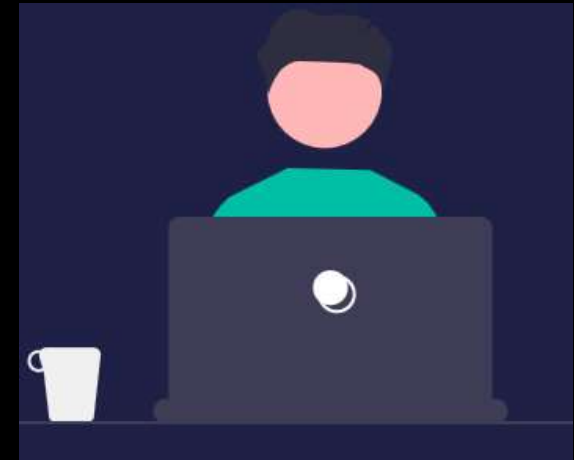
Bob a reçu
«Htqns%f%zsj%xfqj%htzuj » de
la part d'Alice, il va donc utiliser
la clé et le cipher pour déchiffrer
le message.

Bob



Bob fait donc l'opération inverse :
Donc « Htqns », donne « Colin » car :
« H » 5 lettres avant, donne « C »
« t » 5 lettres avant donne « o »,
« q » 5 lettres avant donne « l »,
« n » 5 lettres avant donne « i »,
« s » 5 lettres avant donne « n »

Bob



Bob a déchiffré le message !

Alice



« Colin a une salle
coupe »

« Htqns%f%zsj%xfqj%
htzuj »

Bob



« Colin a une salle
coupe »

Que se passe-t-il si Colin fait
une attaque MITM ?



Colin ne va rien comprendre !!

Lorsque les messages sont cryptés, une attaque MITM est-elle possible ?



Il n'est pas possible (ou très difficilement)
de faire une attaque MITM lorsque la
communication est chiffrée
symétriquement

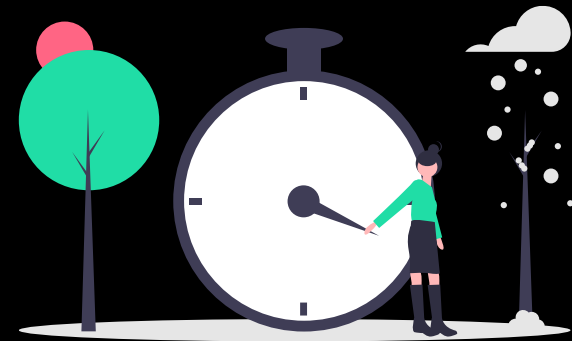
Fin de l'exemple sur la
cryptographie symétrique

On appelle **cryptographie symétrique** toute cryptographie entre 2 parties (comme Alice et Bob) impliquant **1 même clé des 2 côtés** (Alice utilise la même clé que Bob)

Quelqu'un voit-il un problème ?
Est-ce que la cryptographie
symétrique suffit ?

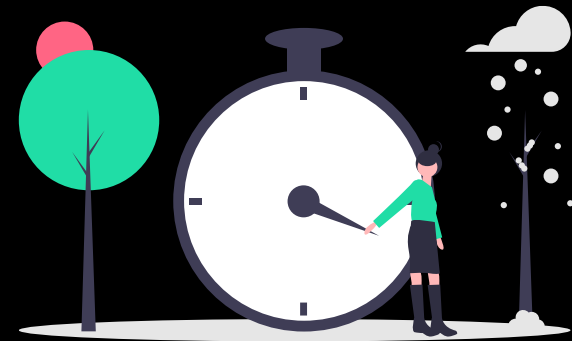


Sauf ... que se passe-t-il si Colin
fait une attaque MITM au
moment où Alice donne la clé
et la cipher à Bob ?



Colin sera aussi dans la capacité de
déchiffrer les messages !! Zut !! =(

A votre avis, comment peut-on empêcher Colin de voir l'échange de la clé symétrique de Alice vers Bob ? (attention c'est plus dur)



Pour s'assurer que Alice donne sa clé et son cipher en cachette à Bob, nous allons utiliser la **cryptographie asymétrique** !

La cryptographie **asymétrique est lente** (alors que la cryptographie **symétrique est rapide**), mais assure à 100% (en théorie) que les informations sont données en cachette

L'idée globale devient ainsi que :

1. Alice choisisse la clé et le cipher
2. Alice envoie tout ça à Bob en **cryptographie asymétrique** (même si c'est lent) 100% en cachette
3. Bob recoit la clé et le cipher
4. Alice et Bob communique en **cryptographie symétrique** avec cette clé et le cipher (c'est rapide)

Pre-master key

L'idée globale devient ainsi que :

1. Alice choisisse la clé et le cipher
2. Alice envoie tout ça à Bob en **cryptographie asymétrique** (même si c'est lent)
3. Bob reçoit la clé et le cipher
4. Alice et Bob communique en **cryptographie symétrique** avec cette clé et le cipher (c'est rapide)

Negotiation
process

Bulk encryption-
decryption process

Une cryptographie asymétrique se base sur
2 clés différentes :

- Une clé dite publique
- Une clé dite privée

Nous n'allons pas voir la cryptographie asymétrique qui est plus compliquée et nécessite davantage de mathématiques.







Ce processus global est le mécanisme du chiffrement des données utilisées par HTTPS :

1. Alice choisit la clé et le chiffre
2. Alice envoie tout ça à Bob en **cryptographie asymétrique** (même si c'est lent)
3. Bob reçoit la clé et le chiffre
4. Alice et Bob communiquent en **cryptographie symétrique** avec cette clé et le chiffre (c'est rapide)

On remplace Alice par le navigateur et Bob par le serveur et on obtient :

1. ~~Alice~~ Le navigateur choisisse la clé et le cipher
2. ~~Alice~~ Le navigateur envoie tout ça ~~à Bob~~ au serveur en **cryptographie asymétrique** (même si c'est lent)
3. ~~Bob~~ Le serveur recoit la clé et le cipher
4. ~~Alice et Bob~~ Le navigateur et le serveur communique en **cryptographie symétrique** avec cette clé et le cipher (c'est rapide)

La seule clé nécessaire pour faire démarrer tout le processus de cryptographie des données est la clé publique de la cryptographie asymétrique.

Tout le reste en découle automatiquement entre le navigateur et le serveur.

La clé publique est détenue par le serveur et est consignée dans un fichier appelé : **certificat SSL**

Le certificat SSL contient donc la clé publique qui permet tout le processus de cryptographie des données entre le navigateur et le serveur

Le **SSL** (Secure Sockets Layer) et **son**
successeur TLS (Transport Layer Security)
sont des protocoles cryptographiques utilisés
pour **crypter les communications** entre
logiciels

HTTPS = HTTP + TLS !

Conclusion : comment faire
pour parer les attaques MITM ?



Utiliser le protocole **HTTPS** !

