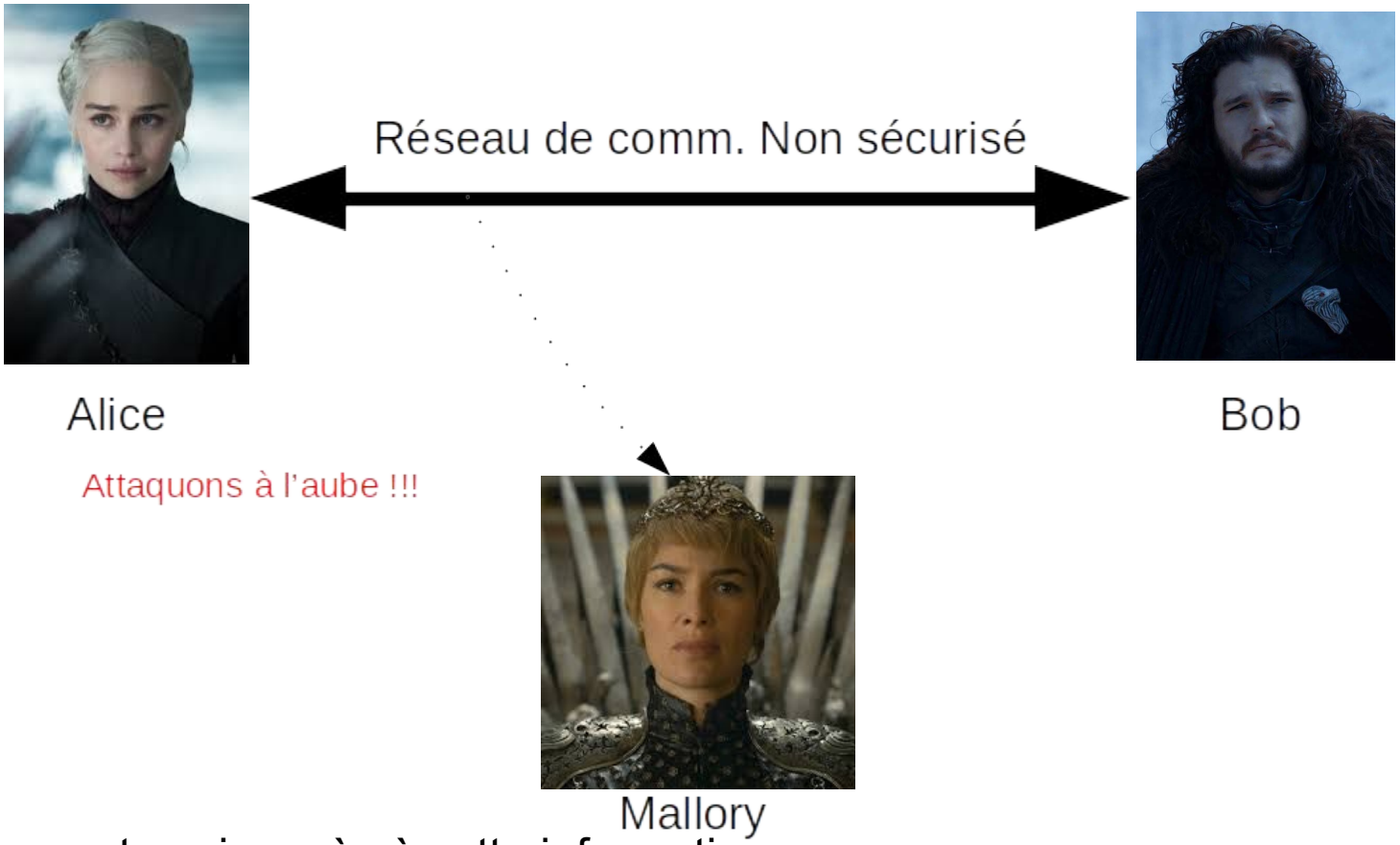
Cours 8

Sécurité des Communications

Contexte

Alice veut transmettre une information secrète à Bob (et seulement a Bob) en utilisant un réseau non sécurisé.



Mallory veut avoir accès à cette information.

Cryptographie

Un élément clé dans tous les systèmes de sécurité, essentiel pour assurer 4 objectifs :

- Confidentialité : seules les personnes autorisées ont accès aux données.
- Intégrité des données : seules les personnes autorisées peuvent modifier les données.
- Authentification : prouver l'identité.
- Non répudiation : l'émetteur d'un message ne peut pas dire qu'il ne l'a pas fait.

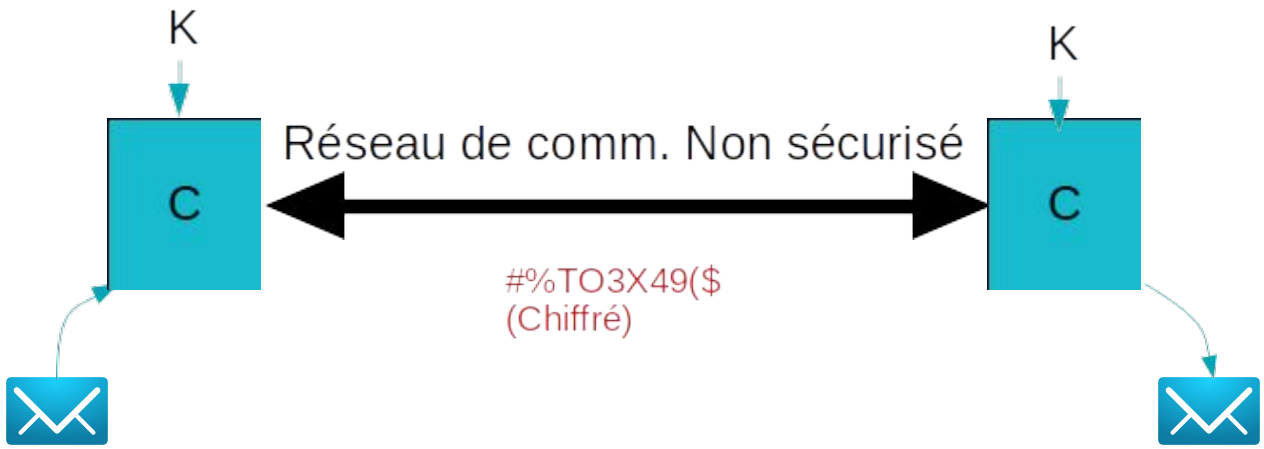
Utilisation du Chiffrement

Alice veut transmettre une information secrète à Bob (et seulement a Bob) en utilisant un réseau non sécurisé.

- Comment gérer les clés ?
- Quel algorithme utiliser ?

Chiffrement Symétrique

- Chiffrement et déchiffrement avec la même clé : $K_E=K_D$
- La clé doit être connue d'Alice et de Bob.
- Algorithmes : AES, DES, …



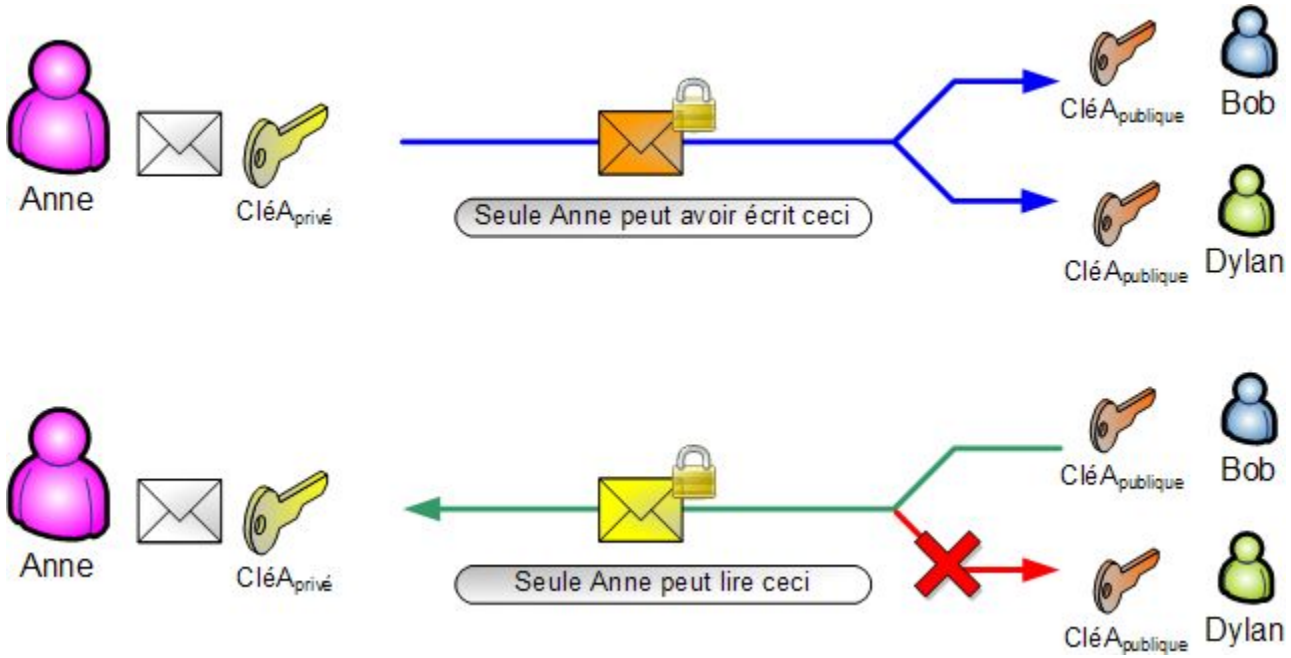
Alice

Attaquons à l'aube !!!

Attaquons à l'aube !!!

Chiffrement Asymétrique

- Clé de chiffrement et de déchiffrement différente : $K_E \neq K_D$
- Alice et Bob possèdent chacun une paire de clé (C,K) telles que :
  - K_{Alice} est privée à Alice et C_{Alice} est publique ;
  - Tout ce qui est chiffré avec C_{Alice} peut être déchiffré avec K_{Alice} ;
  - Tout ce qui est chiffré avec K_{Alice} peut être déchiffré avec C_{Alice} ;
  - De même pour Bob.
- Algorithmes : RSA, ECC, …



Chiffrement Asymétrique

Scénario simple

- Chiffrer avec la clé publique C
- Déchiffrer avec la clé privée K

Chiffrement Asymétrique

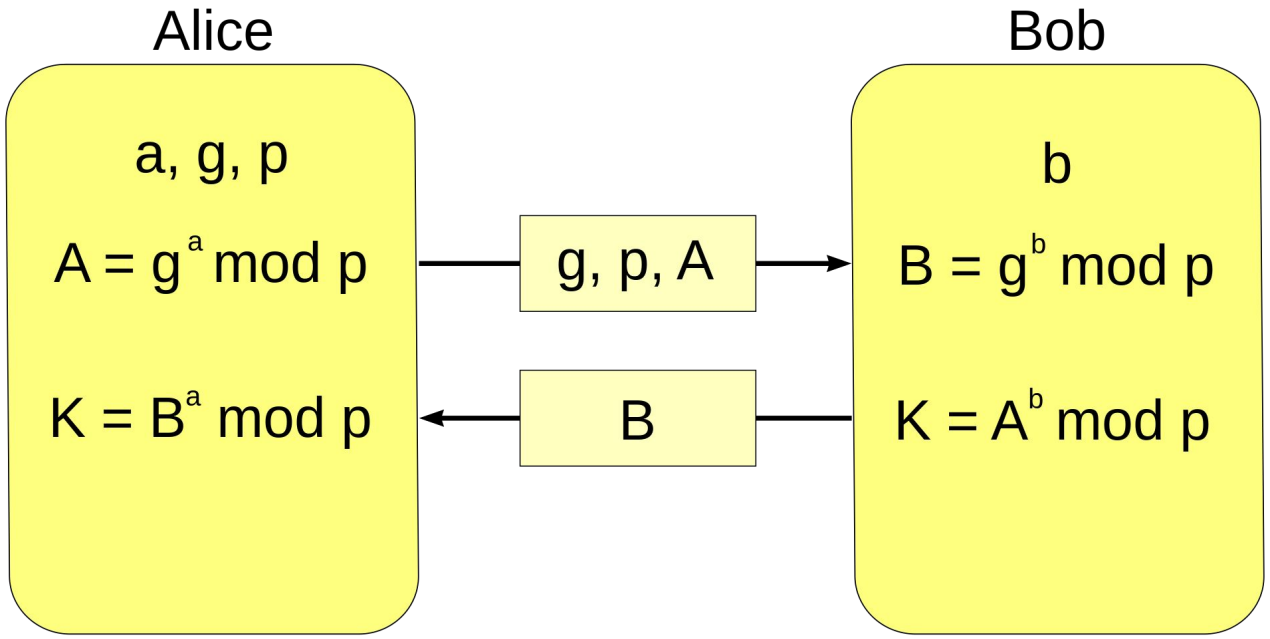
Scénario réaliste

- Générer une clé aléatoire secréte S (symétrique) → clé de session
- Chiffrer S avec C et l'envoyer ; Déchiffrer S avec K
- Utiliser S pour chiffrer le trafic
- Changer S régulièrement au cours de la session…

Confidentialité Persistante

- Que se passe-t-il si un adversaire découvre la clé privée de Bob ou Alice ?
- Comment ne pas compromettre la confidentialité des communications passées ?

Confidentialité Persistante (ou *Perfect Forward Secrecy*) : algorithme de Diffie-Hellman pour le calcul d'une clé de session inviolable…



$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$

Algorithmes de Hachage

Permettent la vérification de l'intégrité du message...

- Fonctions à sens unique calculant une empreinte / condensat du message
  - Facilité de calcul du hachage d'un message
  - Impossibilité de retrouver le message à partir du hachage
  - Impossibilité de construire deux messages ayant le même hachage
  - Impossibilité de modifier un message sans mise à jour du hachage
- Algorithmes : SHA256, SHA1, MD5, ...

Exemples :

```
$ echo "bonjour" | sha1sum
1F71E0F4AC9B47CD93BF269E4017ABAAB9D3BD63

$ echo "Attaquons à l'aube!!!" | sha1sum
8073B9D9B2EB74F31F9AE87359AF440883380D7E
```

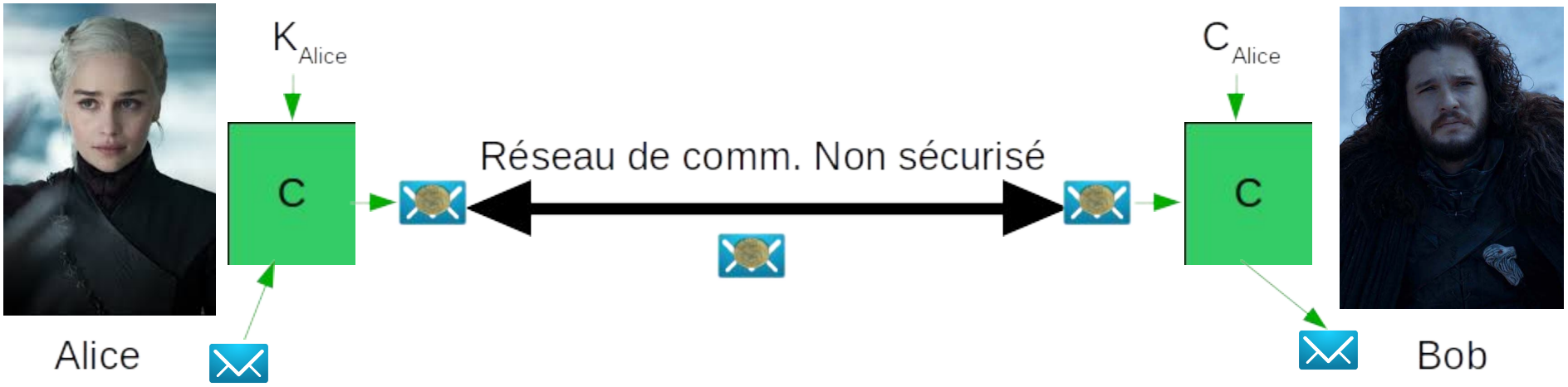
Signature Électronique

Permet de vérifier l'authenticité du message

- Générer le hachage H du message
- Chiffrer H avec K_{Alice} et envoyer le résultat avec le message

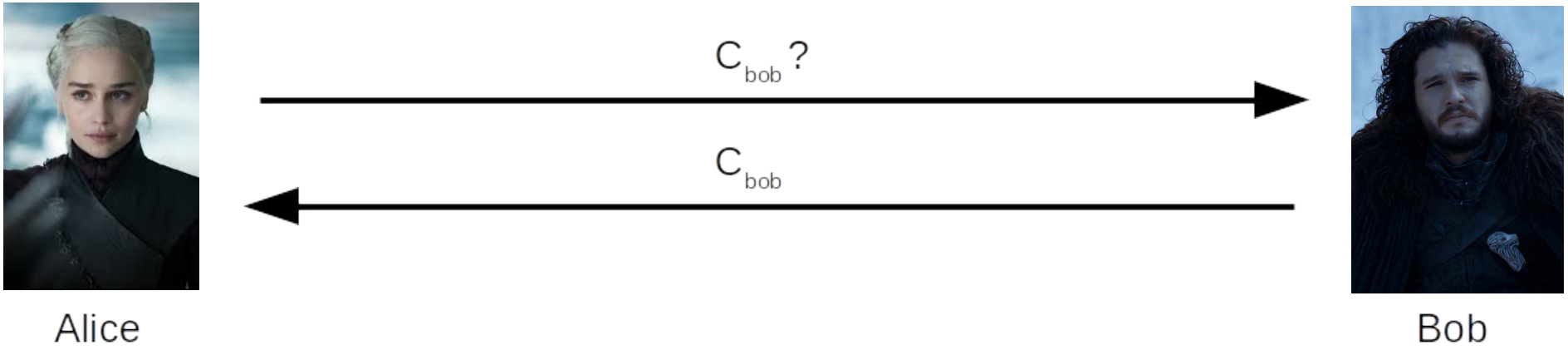
Bob peut vérifier la signature en utilisant C_{Alice}

- Bob est sûr que le message n'est pas corrompu si le résultat du déchiffrement est identique au hachage qu'il calcule
- Bob est sûr qu'Alice est l'émetteur du message



Certificats Électroniques

Que se passe-t-il si Alice n'a pas C_{bob} initialement ?



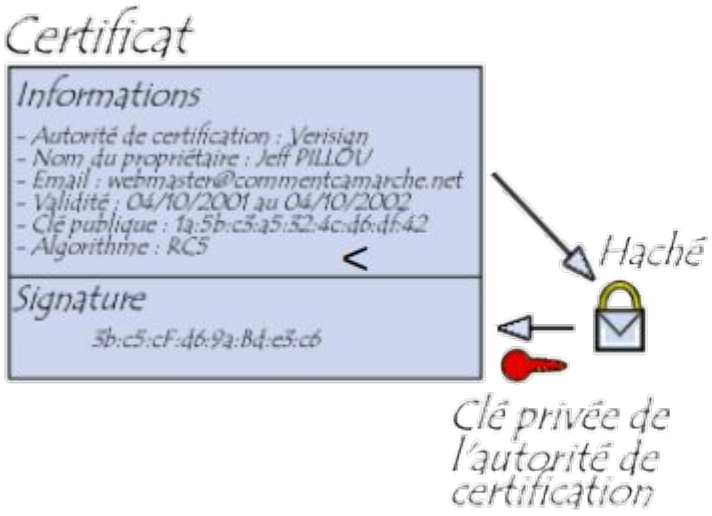
Problème du Man-In-The-Middle !

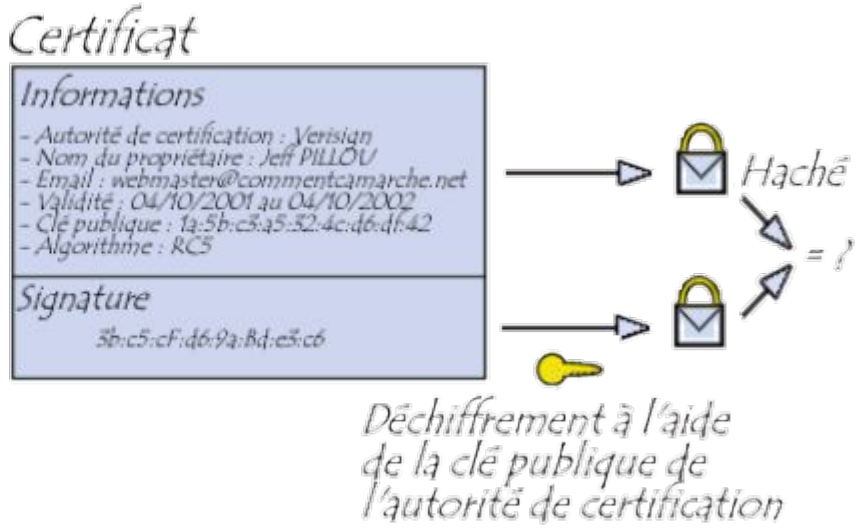
Certificats Électroniques

Un certificat contient :

- Une clé publique + une identité (dans un format clé/valeur)
- Une signature par une autorité de confiance (ou CA) dont la clé publique est connue
- Les clés publiques des CA sont pré-chargées dans votre système d'exploitation…

Vérification d'un certificat

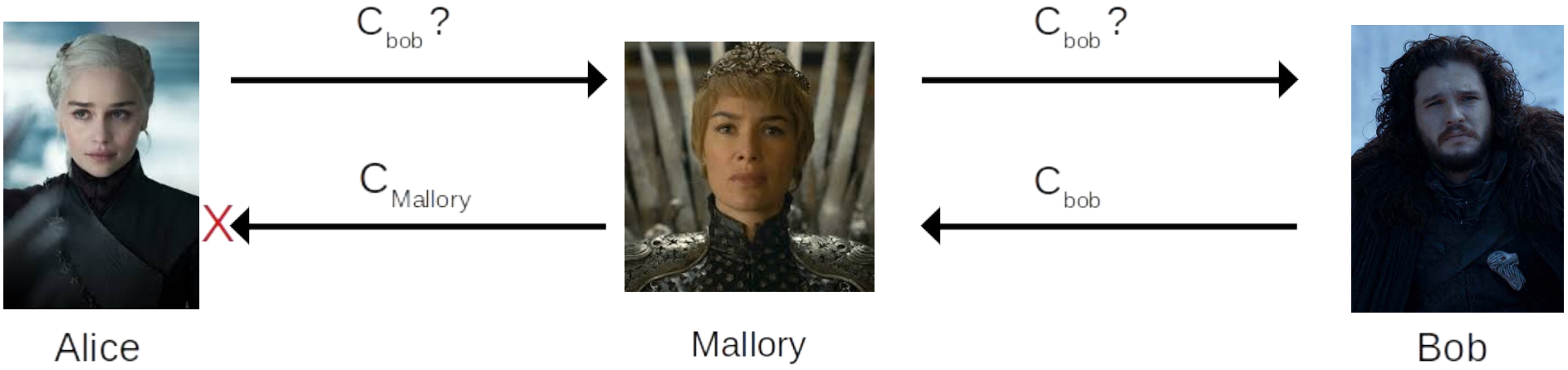




Certificats Électroniques

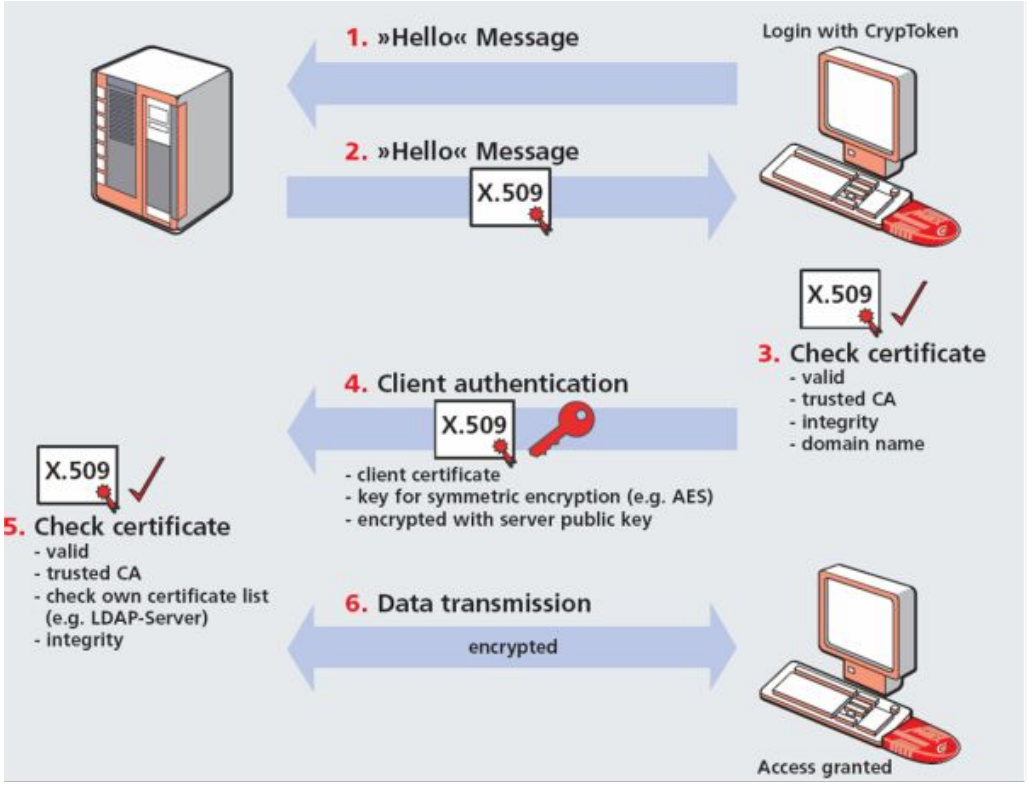
À la réception du certificat de Bob, Alice peut vérifier que le certificat appartient bien à Bob

- Mallory ne peut plus usurper l'identité de Bob...



SSL/TLS

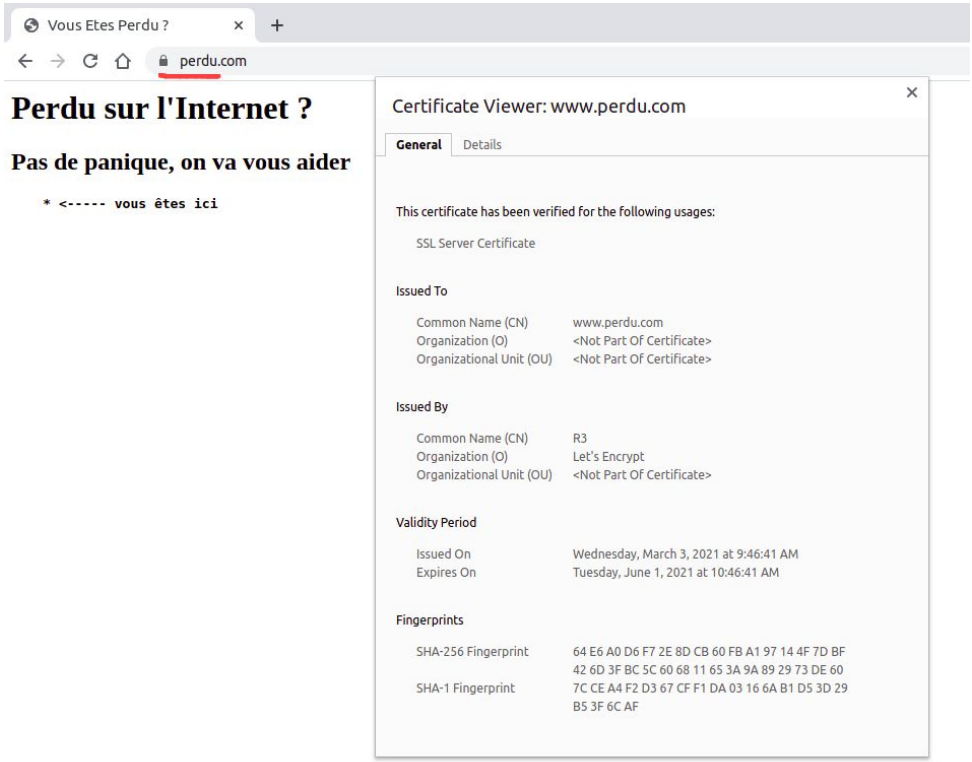
- Protocole de sécurisation des échanges sur Internet
- Basé sur l'utilisation de certificats
- Utilisé pour l'implémentation de versions sécurisées des protocoles standards (HTTPS, SMTPS, IMAPS, ...)



Source : wikipedia

HTTPS

- Utilisation transparente du protocole HTTP au-dessus de TLS/SSL (port 443 au lieu de 80)
- Authentification du serveur web via son certificat (signé du CA)
- Confidentialité et intégrité des données envoyées au serveur
- En général, pas d'authentification du client



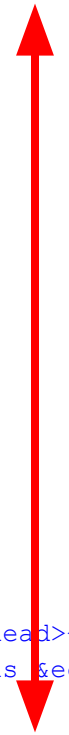
Démo HTTPS

https://rx2.gitlabpages.inria.fr/support/data/https.pcap

```
$ guntls-cli --crlf www.perdu.com
Resolving 'www.perdu.com:443'...
Connecting to '208.97.177.124:443'...
- Certificate type: X.509
- Got a certificate list of 2 certificates.
- Certificate[0] info:
 - subject `CN=www.perdu.com', issuer `CN=R3,O=Let's Encrypt,C=US', ...
- Certificate[1] info:
 - subject `CN=R3,O=Let's Encrypt,C=US', issuer `CN=DST Root CA X3,O=Digital Signature Trust Co.', ...
- Status: The certificate is trusted.
- Handshake was completed
- Simple Client Mode:
GET / HTTP/1.1
Host: www.perdu.com

HTTP/1.1 200 OK
Date: Sun, 28 Mar 2021 21:34:49 GMT
Server: Apache
Upgrade: h2
Connection: Upgrade
Last-Modified: Thu, 02 Jun 2016 06:01:08 GMT
ETag: "cc-5344555136fe9"
Accept-Ranges: bytes
Content-Length: 204
Cache-Control: max-age=600
Expires: Sun, 28 Mar 2021 21:44:49 GMT
Vary: Accept-Encoding,User-Agent
Content-Type: text/html

<html><head><title>Vous Etes Perdu ?</title></head><body><h1>Perdu sur l'Internet ?</h1><h2>Pas de panique, on
va vous aider</h2><strong><pre>    * <----- vous &ecirc;tes ici</pre></strong></body></html>
```



échanges sécurisés entre le client et le serveur web