Sécurité

Pr. Laurent Réveillère

Université de Bordeaux

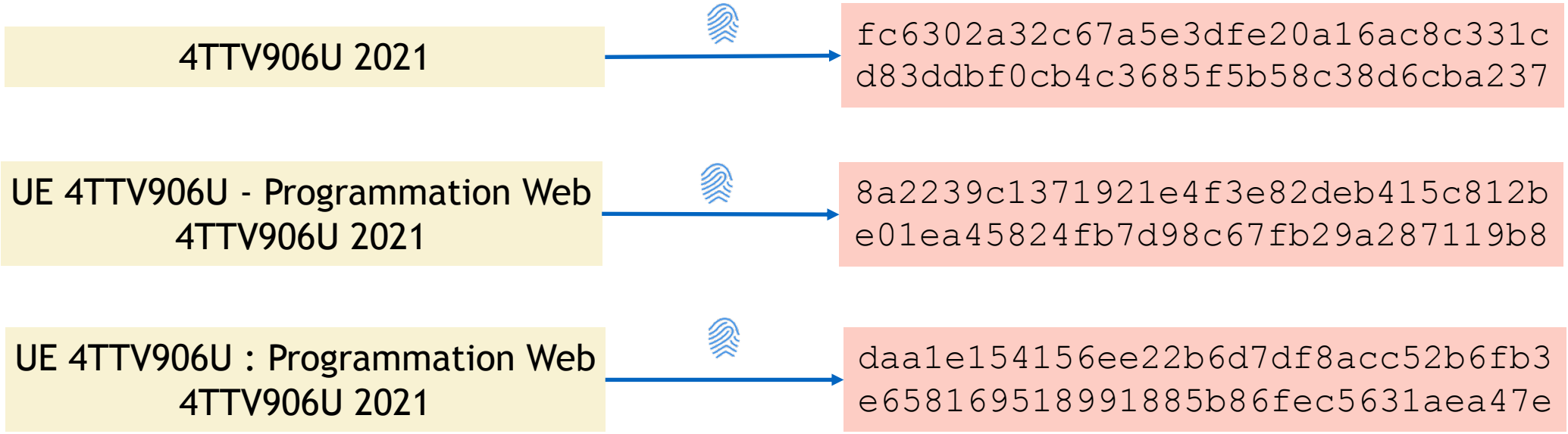
université de BORDEAUX

laurent.reveillere@u-bordeaux.fr

http://www.reveillere.fr/

Empreinte numérique

- Résumer en un petit nombre fixe d'octets le contenu d'un texte de taille arbitraire
- Garantir l'intégrité d'un message : le même message aura toujours la même empreinte (presque unique)



Fonctions de hachage

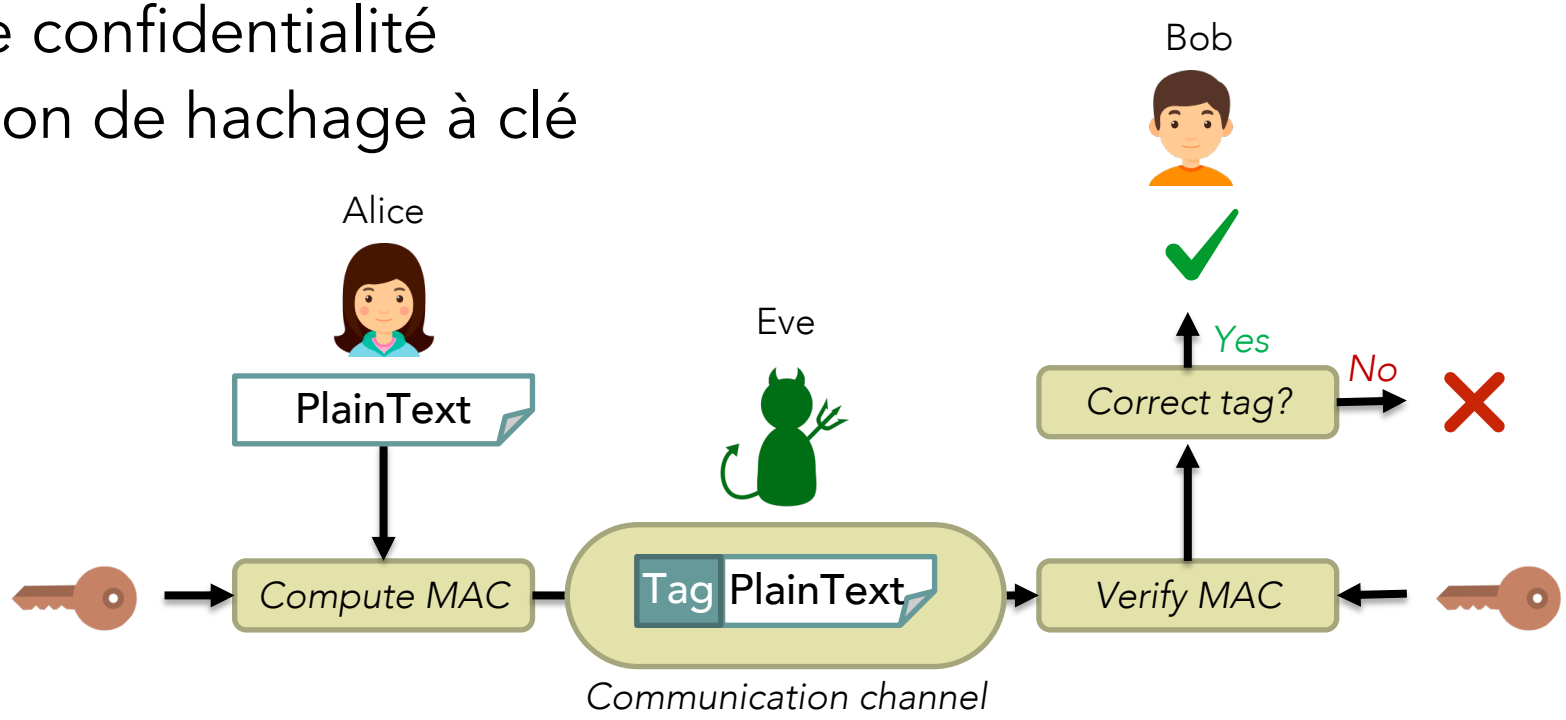
- Fonction cryptographique qui produit en sortie une donnée de taille fixe quelque soit les données en entrée
- Propriétés
  - Impossible à inverser (fonction à sens unique)
  - Déterministe
  - Résistant aux collisions
    - La probabilité que deux messages aient le même hachage est négligeable

Contrôlez l'intégrité des messages

- Message Authentication code (MAC)
  - Code d'authentification du message : garantie de non-modification des données
- CBC-MAC
  - Utilisation du dernier bloc du chiffrement du message en AES-CBC
  - Suppose un message de taille connu
- Hashed-MAC
  - Utilisation d'une fonction de hachage
  - Besoin de protéger le code MAC par un secret partagé

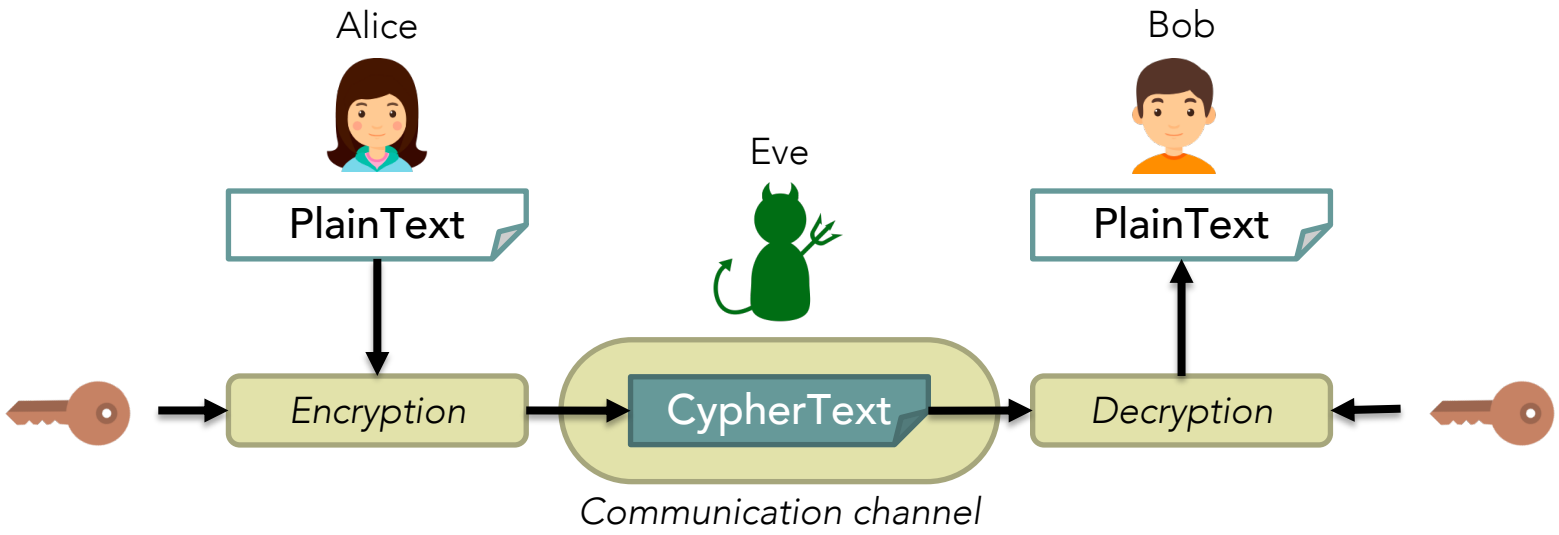
HMAC avec clé secrète

- Message authentication code (MAC)
  - Étiquette permettant de vérifier l'intégrité et l'authenticité du message
  - La clé est nécessaire pour produire un MAC correct
  - Pas de confidentialité
  - Fonction de hachage à clé



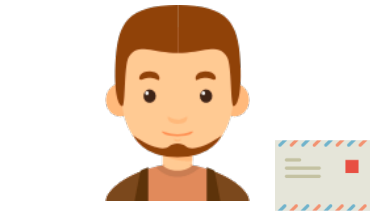
Cryptographie symétrique

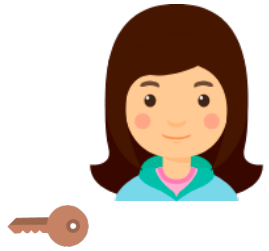
- Toutes les parties ont une clé secrète partagée
  - Même clé pour le cryptage et le décryptage
- Exemple: AES (advanced encryption standard)



Cryptographie asymétrique

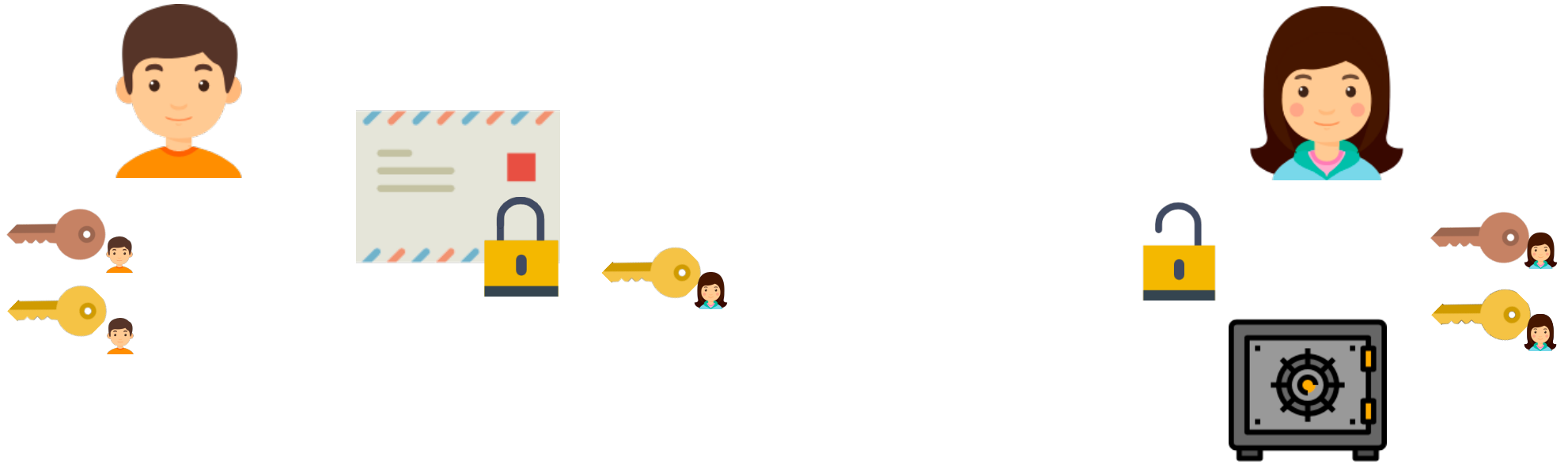
- Métaphore de la boîte aux lettres





Cryptographie asymétrique (RSA - Rivest-Shamir-Adleman)

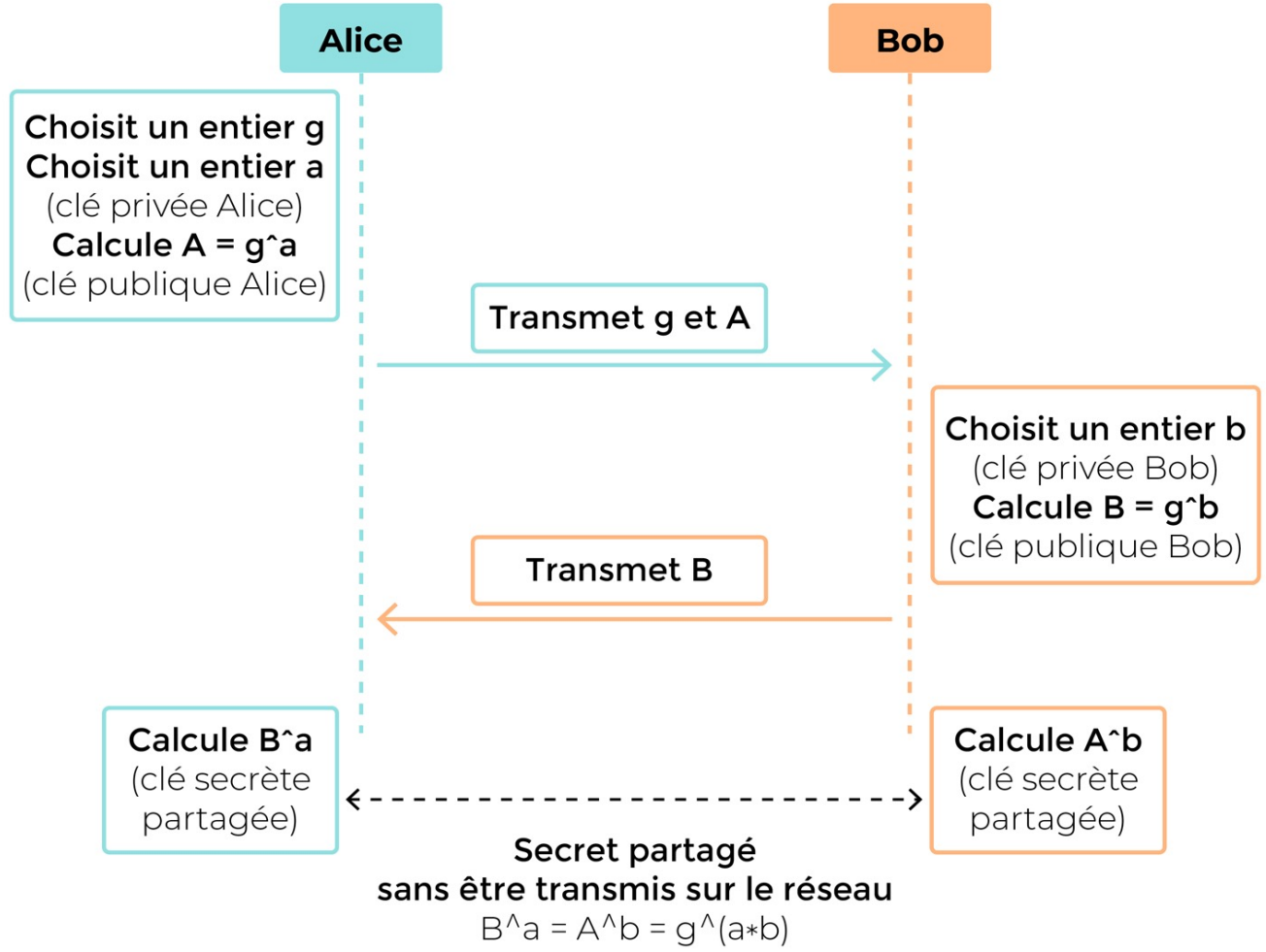
- Système basé sur la factorisation des grands nombres
- Chaque partie possède une paire de clés
  - Clé publique : non secrète, utilisée pour le cryptage ( )
  - Clé privée : secrète, utilisée pour le décryptage ( )



Échange de clés Diffie-Hellman

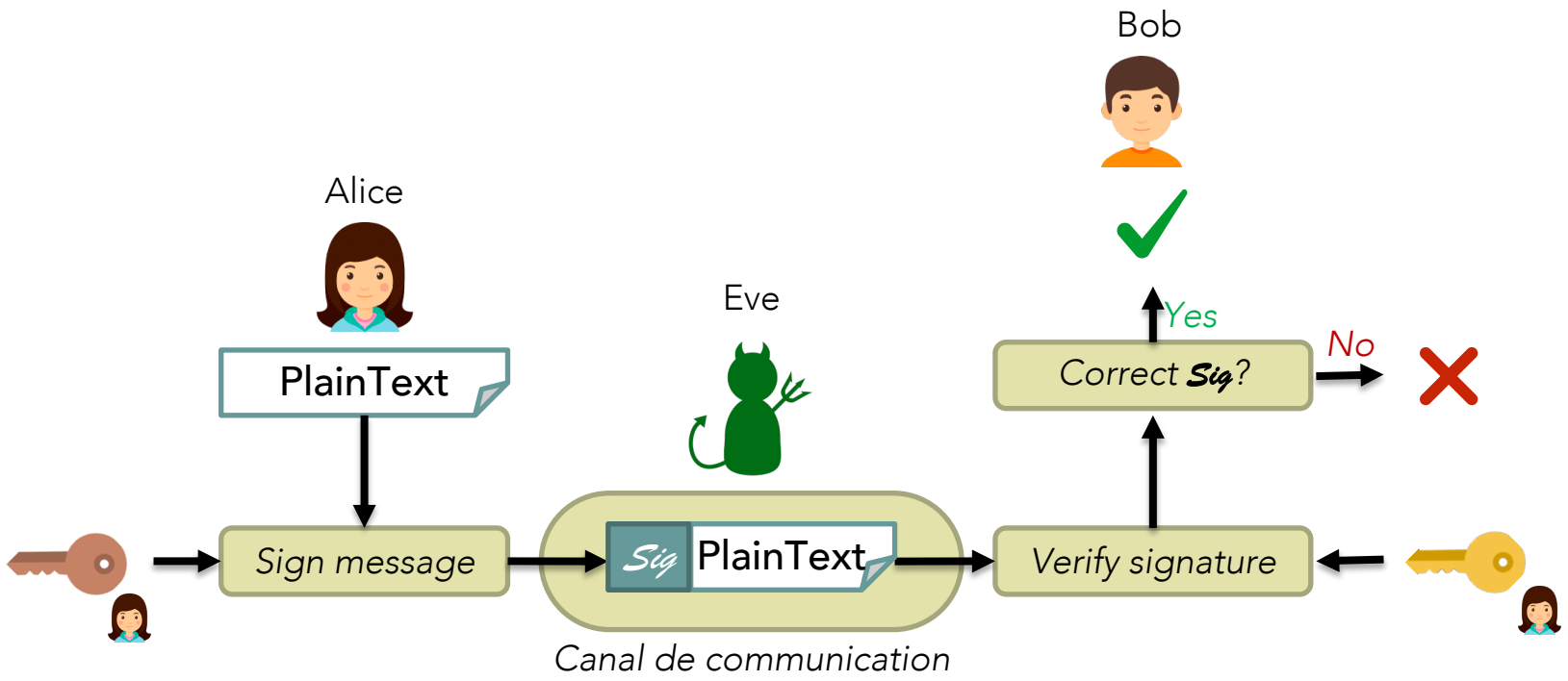
- Système basé sur les propriétés mathématiques de la fonction exponentielle et de sa réciproque, le logarithme discret

  C'est compliqué mais joli!



Signature numérique

- HMAC + chiffrement asymétrique
  - Signer avec la clé privée de l'expéditeur
  - Vérifier avec la clé publique de l'expéditeur



TLS

- Successeur de SSL
- Permet :
  - la **confidentialité** des données avec le chiffrement symétrique ;
  - l'**intégrité** des données avec un code MAC ;
  - l'**échange d'une clé de session** avec le chiffrement asymétrique ;
  - l'**authentification du serveur** avec les certificats signés par une autorité de certification.
- Utilisation possible
  - HTTP + TLS = HTTPS

Contrôle d'accès

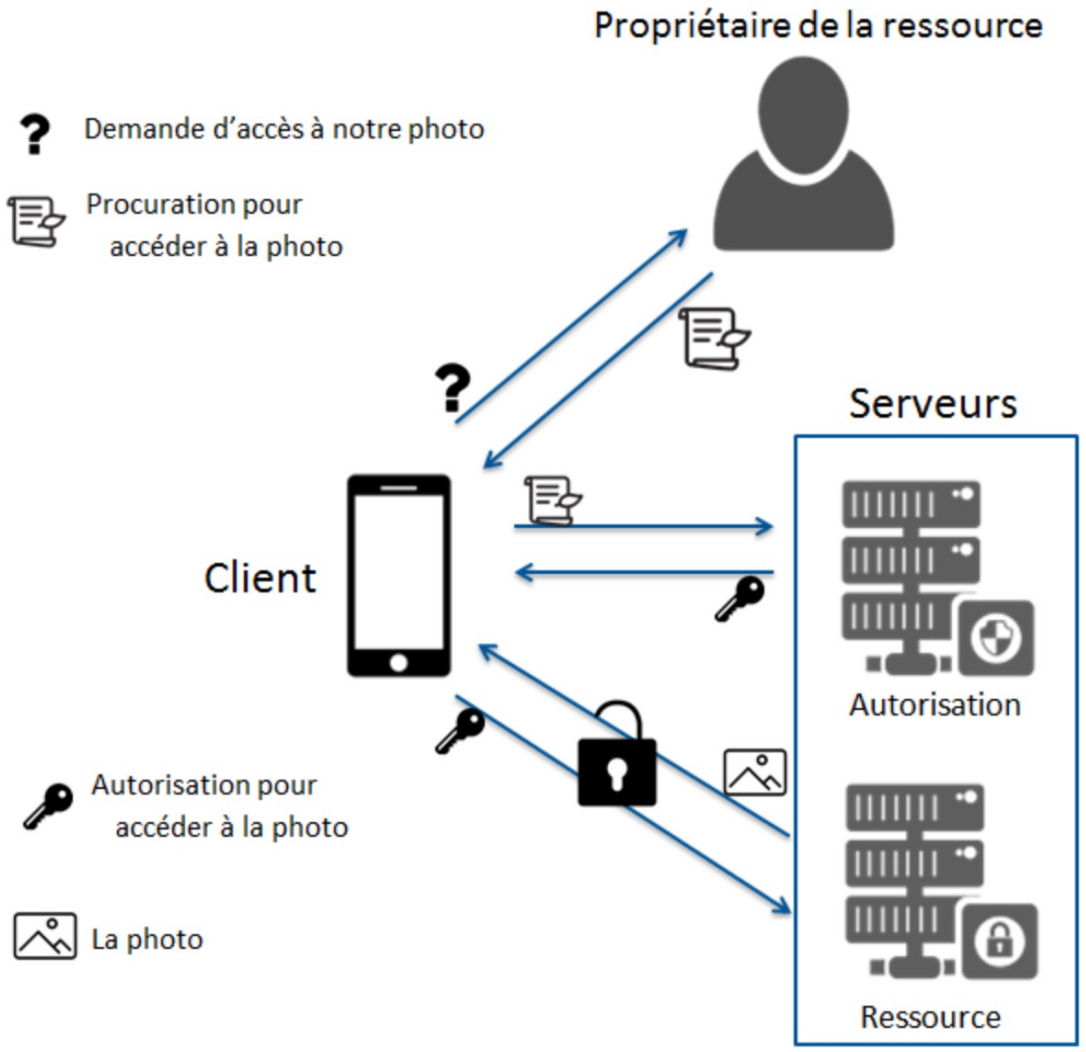
- Garantir qu'un utilisateur particulier a les permissions suffisantes pour
  - Créer, lire, mettre à jour ou supprimer une **ressource sécurisée** ;
  - Accéder à une **fonction sécurisée** ;
  - Réaliser un **processus métier** protégé.

Stocker des mots de passe

- En clair
  - À proscrire !!!!
  - Mais ça existe encore beaucoup …
- De manière chiffrée
  - Utilisation d'un chiffrement robuste comme AES256
  - Peut sembler une bonne pratique, mais sensible au vol de clé secrète
- De manière hachée
  - Meilleure méthode car non-réversible
  - Sel (chaîne aléatoire pour chaque mdp) et poivre (chaîne globale)
  - Rend les attaques par force brutes plus difficiles

Oauth 2.0

- Acteurs - rôles
  - **Propriétaire** des données (vous) ;
  - **Serveur de ressources** qui héberge les accès dont l'accès est protégé (p.ex. profil Facebook) ;
  - **Client** qui souhaite accéder au serveur de ressources ;
  - **Serveur d'autorisation** qui délivre des jetons au client.



Gestion de l'authentification

- JWT (JSON Web Tokens)
  - Gestion sans état de l'authentification
  - Pour chaque demande de connexion, le serveur génère un jeton
  - Le serveur envoie le jeton généré au client
  - Le client stocke le jeton en local
  - Il envoie le jeton dans les requêtes suivantes

- Bibliothèques NodeJS
  - https://github.com/kelektiv/node.bcrypt.js
  - https://github.com/auth0/node-jsonwebtoken
  - http://www.passportjs.org
  - ….