Authentification

Séquence 2

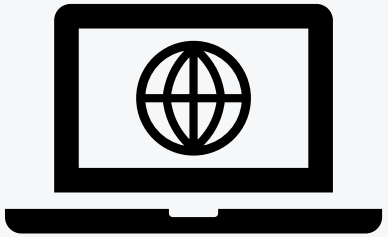
Les objectifs

- Objectifs

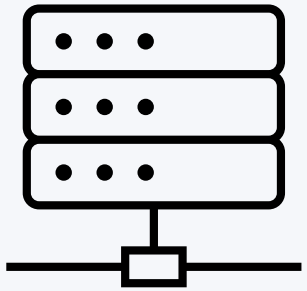
- Expliquer le principe d'une session
- Connaître les dangers liés à la gestion de mot de passe
- Décrire le contenu d'un token JWT
- Valider un token JWT à l'aide de jwt.io

LES SESSIONS

Le principe

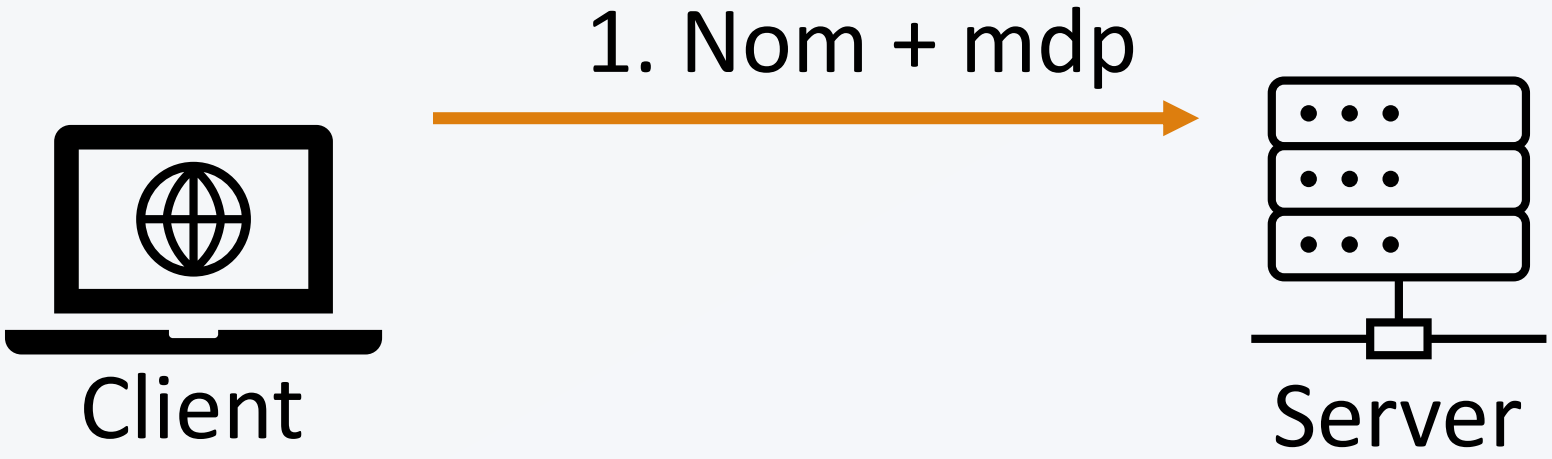


Client

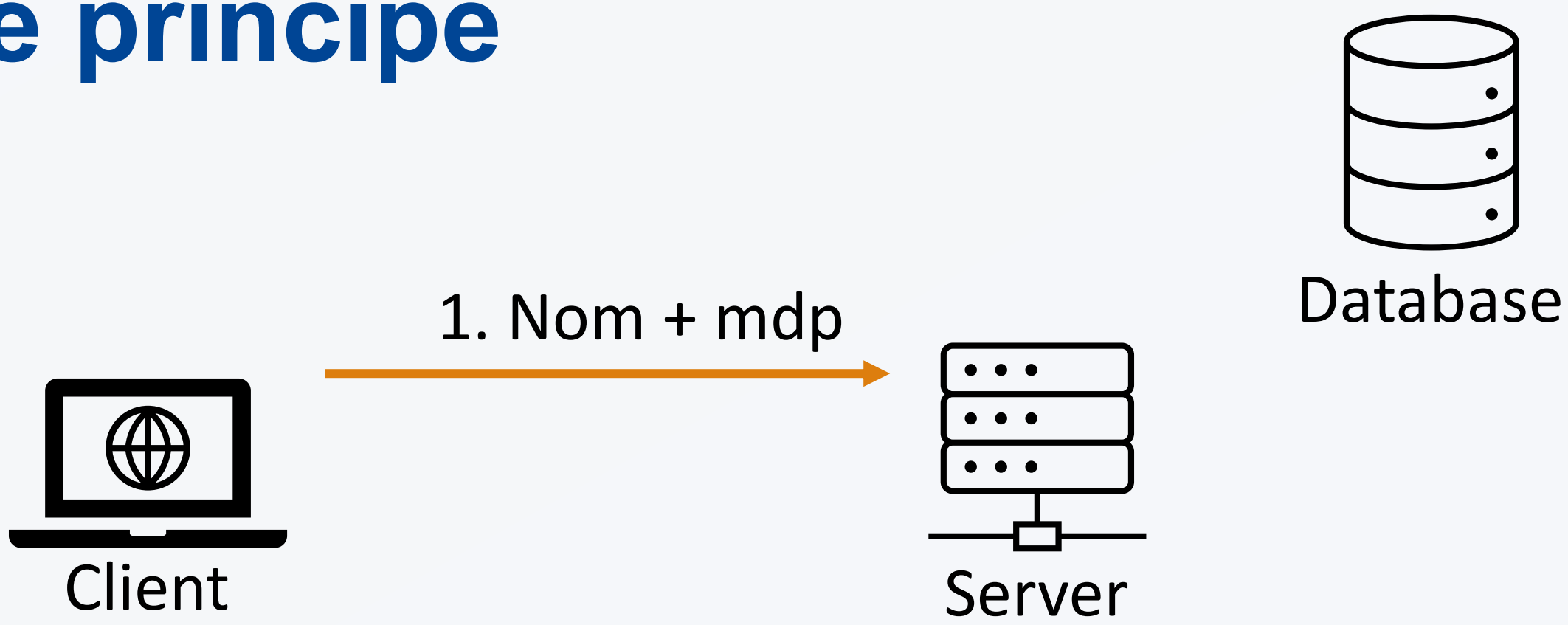


Server

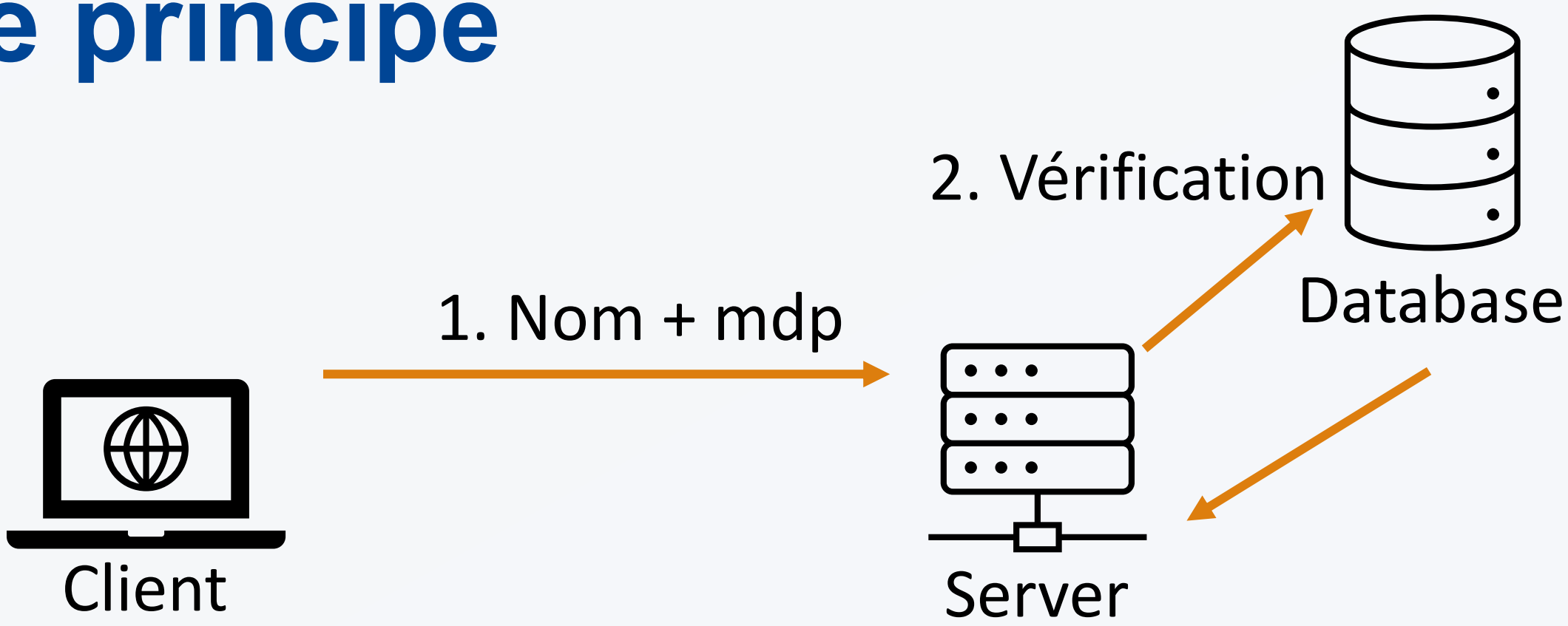
Le principe



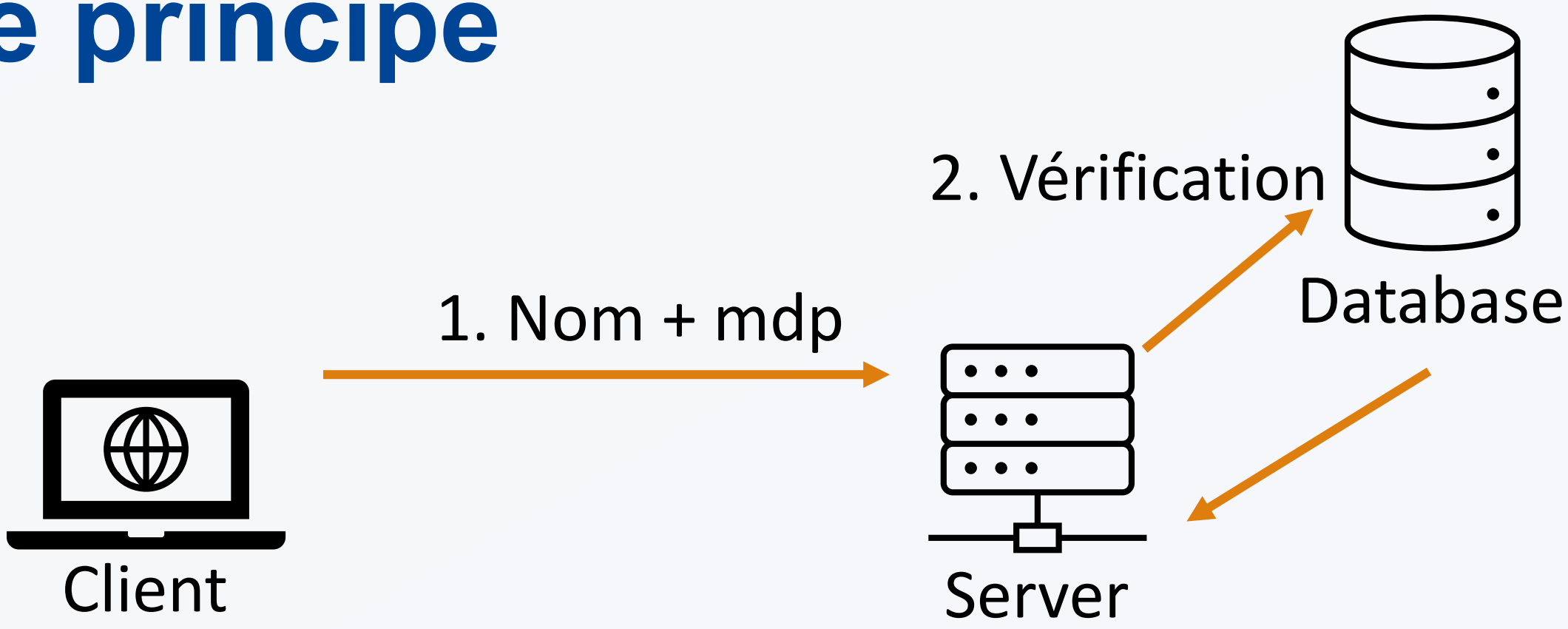
Le principe



Le principe



Le principe



Le mot de passe salé et haché correspond au contenu de la db ?

BRAINSTORM

- Comment est-ce que l’on maintient la connexion «ouverte» entre le client et le serveur ?

Protection du mot de passe

RISQUES

- Les personnes de la compagnie peuvent voir votre mot de passe.
- Un mot de passe stocké en **clair** peut être réutilisé tel quel (même site ou autre site)
- Un mot de passe haché peut être comparé à d'autres hash.

MOYEN DE PRÉVENTION POUR LES UTILISATEURS

- Activer la double authentification
- Choisir un mot de passe différent pour chaque site

MOYENS TECHNIQUES

- Sel
- Poivre

- Ces deux éléments sont ajoutés au mot de passe avant hachage afin de rendre le hash unique de tous les autres utilisateurs.

LE SEL

- Le sel est une chaîne de caractères aléatoires ajoutée au mot de passe avant le hash.
- Le sel est propre à chaque utilisateur (chacun son sel)
- Le sel est stocké en clair dans la base de donnée
- On considère qu'il y a peu de risque si le sel fuite.
- Le sel vise à protéger vos données contre une attaque par table précalculée (rainbow table)

LE POIVRE

- Le sel est l'option la plus connue, mais on peut l'améliorer avec un poivre.
- Le poivre est aussi une chaîne de caractères que l'on ajoute au mot de passe.
- … Mais le poivre est identique pour tous les utilisateurs
- … et le poivre n'est jamais stocké dans la base de données
- … (le mettre dans une variable d'environnement est la bonne pratique)

HASH TABLE

- Il s’agit d’une table avec des hash précalculés qui pourraient permettre de remonter vers le mot de passe original.
- En général, on y retrouve les hash de toutes les combinaisons jusqu’à une certaine longueur.
- On y retrouve aussi les hash des mots de passe les plus courants.

RAINBOW TABLE

- Il s’agit d’une table de hachage optimisée en taille
- Le but est toujours de retrouver le mot de passe original en clair.

PROTECTION PAR LE SEL

- Si vos mots de passe ne sont pas protégés par un sel, ils sont vulnérables aux rainbow tables.
- Les rainbow tables devraient être recalculées en y incluant le sel (et il faudrait le faire pour tous les sels)
- Le sel rend donc la génération de rainbow tables beaucoup plus complexe et lent.

Password Cheat Sheet

- https://cheatsheetseries.owasp.org/cheatsheets/Password_Storage_Cheat_Sheet.html

Activité 1 : Protéger un mot de passe

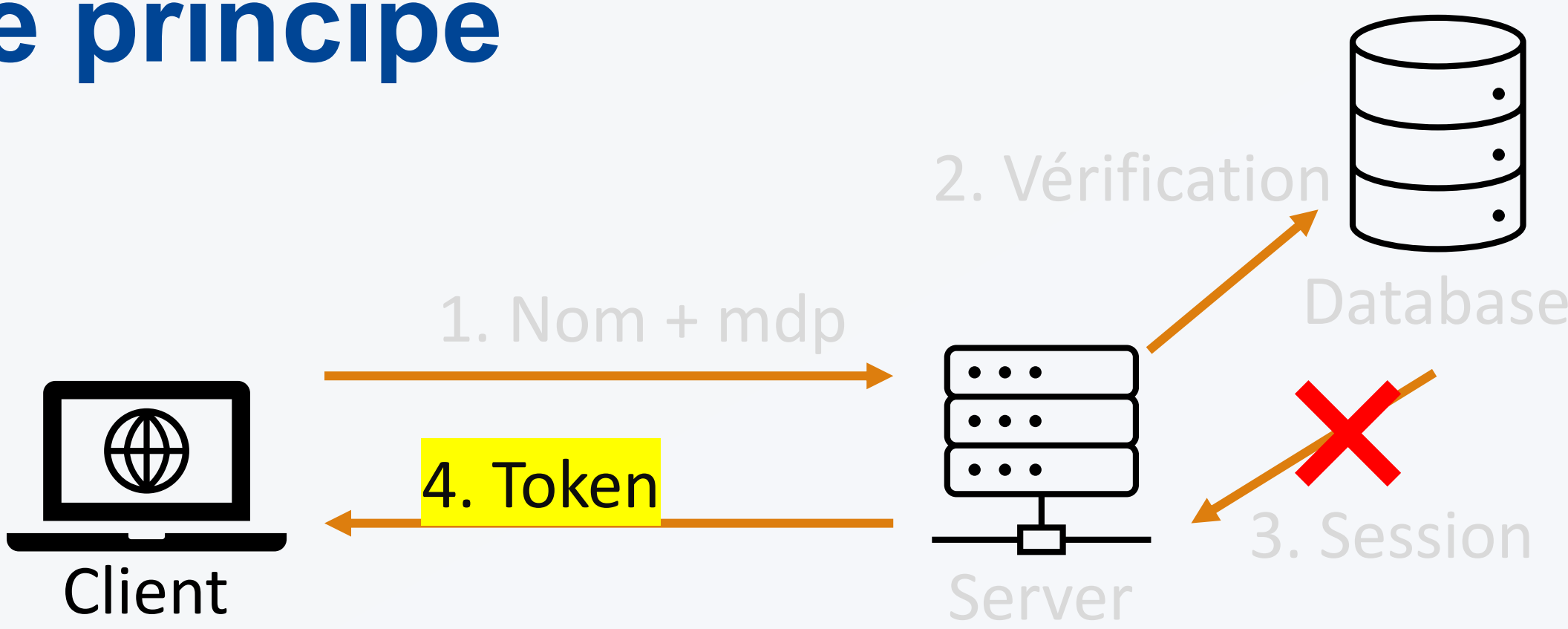
E-183-ALL-Salt.docx

LES TOKENS

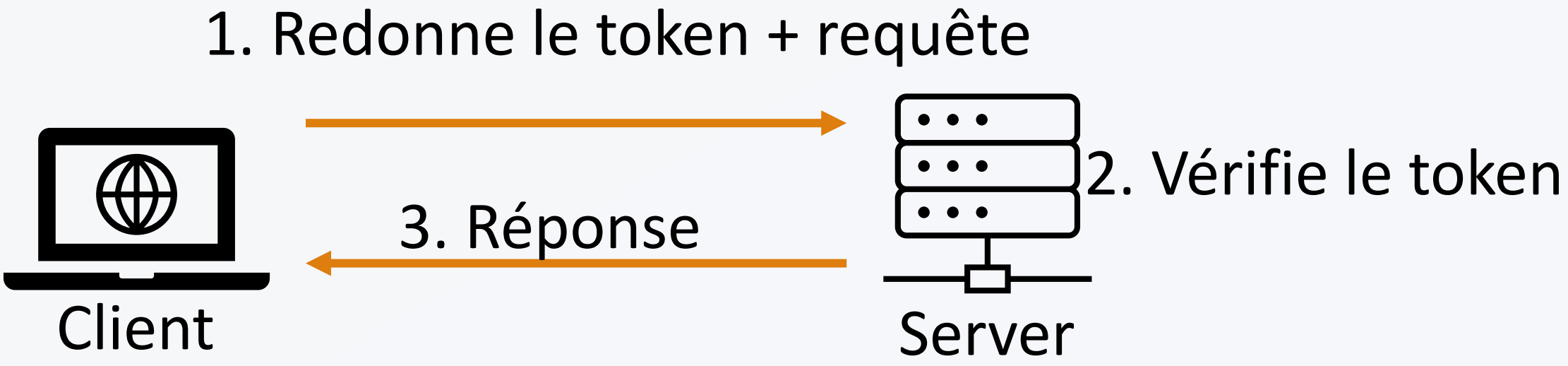
Ce que l’on cherche à résoudre

- Diminuer la complexité au niveau serveur
- Approcher la sécurité de manière globale
- Le protocole http est par nature **stateless**

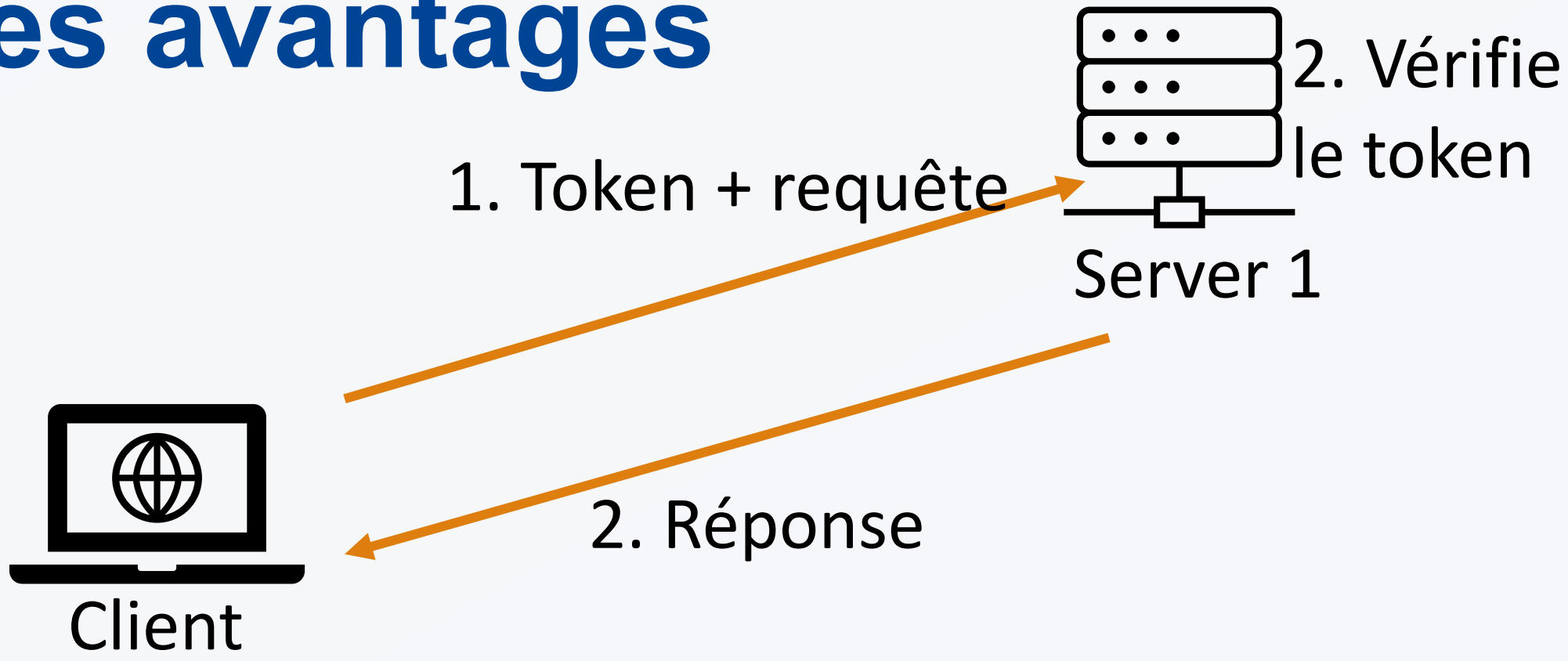
Le principe


2. Vérification
Database
1. Nom + mdp
4. Token
Client
Server
3. Session

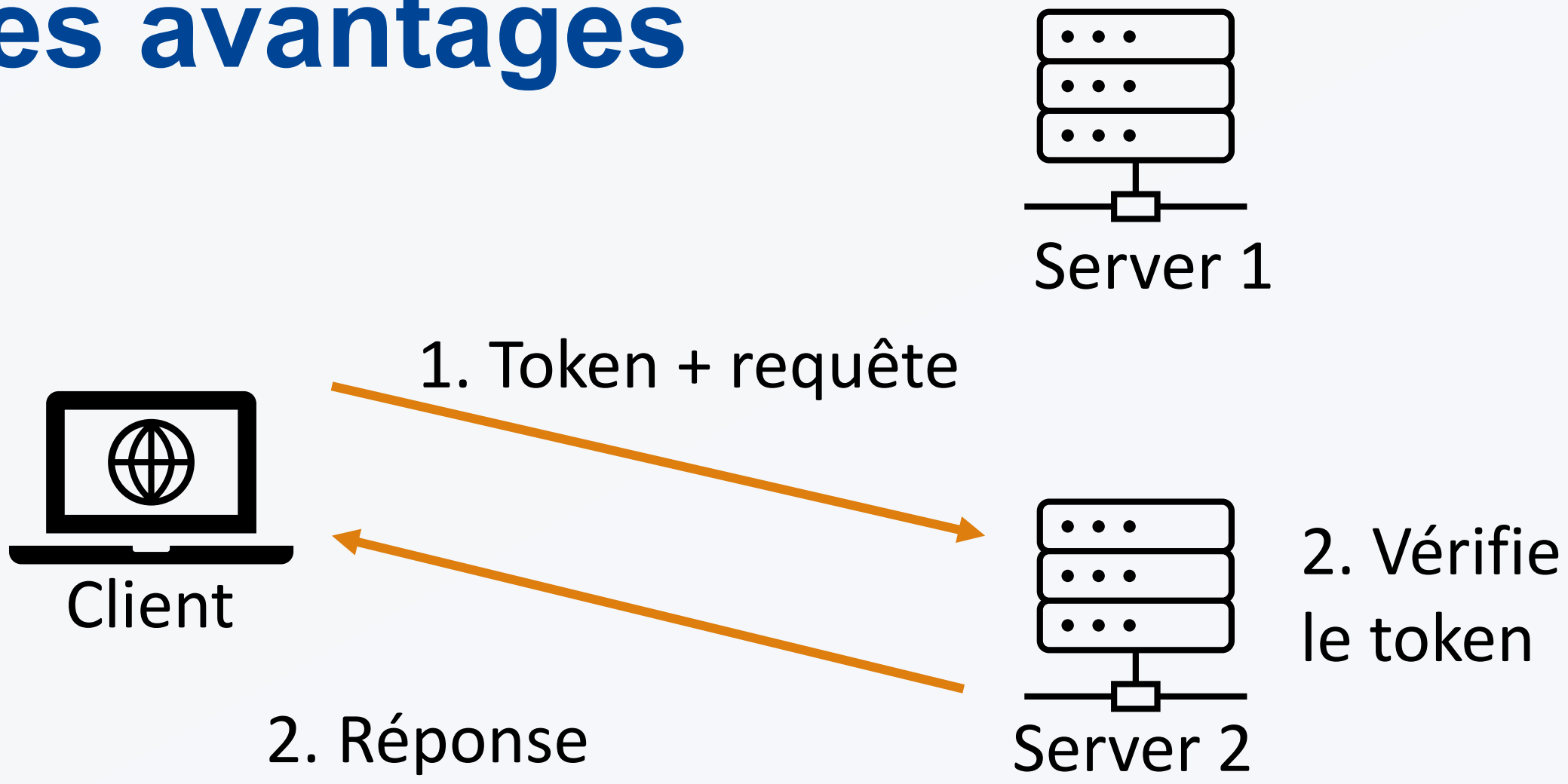
Le principe


1. Redonne le token + requête
2. Vérifie le token
3. Réponse
Client
Server

Les avantages



Les avantages



Pourquoi utiliser un jeton ?

- Réduire la charge sur le serveur
- Simplifier la gestion des sessions

Contenu du jeton

- **Header**
  - Type de token (JWT)
  - Algorithme de chiffrement
- **Payload**
  - Les données en claires
- **Signature**
  - Permet au serveur de vérifier que le token est valide

ETML

Activité 1 : Les jetons JWT

E-183-ALL-JWT-TOKEN.docx

RS256 vs HS256

HS256

- HMAC with SHA-256

- HMAC (**Hash-Based Message Authentication Code**)

- Algorithme de chiffrement symétrique
- Une clé unique génère et valide la signature
- Seul le serveur émetteur du jeton est capable de prouver que le jeton n’a pas été forgé.

RS256

- RSA Signature with SHA-256
- Algorithme de chiffrement asymétrique
- Une paire de clé privée / publique

ETML

Activité 2 : Les jetons JWT

E-183-ALL-JWT-TOKEN-Pratique.docx

Sites utiles

- https://jwt.io
- https://www.npmjs.com/package/jsonwebtoken
-

Par quoi je commence

- Démarrer le serveur
- Tester les URLs avec un client HTTP
- Dans le code
  - Passer en revue les fichiers et comprendre leur fonction
  - Identifier les fichiers que vous devez modifier

Conclusion