1. Introduction

Dans les systèmes d'information modernes, la gestion de l'authentification et de l'autorisation est un enjeu majeur pour garantir la sécurité des utilisateurs et des données. Les jetons (ou tokens) sont un mécanisme largement utilisé pour gérer cette authentification et cette autorisation

Il existe trois types de jetons couramment utilisés dans les applications web modernes :

- ✓ Les Jetons de Session
- ✓ Les Jetons JWT (JSON Web Token)
- ✓ Les Jetons Uniques

2. Les Jetons de Session

Un jeton de session est un mécanisme simple permettant de maintenir une session active entre un client (l'utilisateur) et un serveur. Ce jeton est souvent stocké côté serveur et associé à un identifiant unique, qui permet au serveur de reconnaître l'utilisateur lors de ses prochaines requêtes.

2.1. Mode de fonctionnement

Lorsqu'un utilisateur se connecte, le serveur génère un identifiant de session unique et le stocke côté serveur, généralement dans une base de données ou dans une structure en mémoire (comme Redis).
Le serveur envoie cet identifiant de session au client sous forme de cookie.
À chaque requête suivante, le client renvoie ce cookie au serveur.
Le serveur utilise l'identifiant de session dans le cookie pour retrouver les informations associées à cette session, telles que les informations utilisateur (par exemple, l'ID de l'utilisateur, les droits d'accès, etc.).

2.2. Avantages

Les sessions sont **faciles à gérer**, surtout dans une architecture monolithique.
Les données sensibles ne sont pas envoyées directement au client ; elles restent stockées sur le serveur.
Le serveur peut facilement invalider ou modifier la session à tout moment (par exemple, lors de la déconnexion).

2.3. Inconvénients

Si l'application devient distribuée (multi-serveur), la gestion des sessions devient complexe. Il faut s'assurer que toutes les instances du serveur peuvent accéder à la même session, ou utiliser un mécanisme de synchronisation (comme Redis).
Les informations de session sont stockées côté serveur, ce qui peut entraîner un coût en termes de mémoire si le nombre d'utilisateurs est important.

2.4. Exemple d'instruction

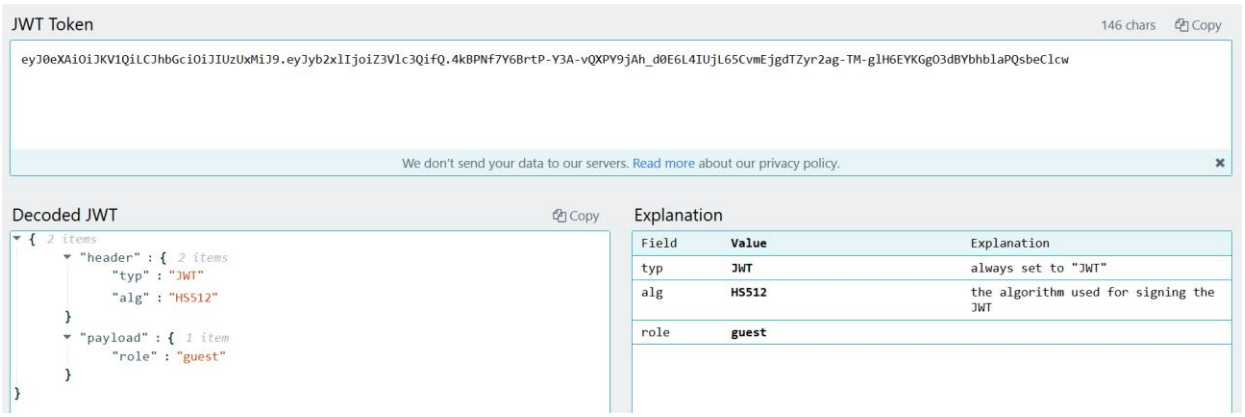
```
session_start();
if (isset($_SESSION['user_id'])) {
    $user_id = $_SESSION['user_id'];
    …
```

3. Les Jetons JWT (JSON Web Token)

Le JWT est un jeton d'authentification stateless (sans état). Contrairement aux jetons de session, il contient toutes les informations nécessaires à l'authentification dans le jeton lui-même, et il ne nécessite pas de stockage côté serveur. Cela permet une architecture plus scalable.

Un JWT est composé de trois parties :

Header (En-tête) : Contient des informations sur le type de jeton (JWT) et l'algorithme de signature utilisé (par exemple, HS256).
Payload (Données) : Contient les informations utilisateur, telles que l'ID de l'utilisateur, les rôles et les permissions. Ces informations peuvent être des claims (revendications) publiques ou privées.
Signature : Permet de vérifier que le jeton n'a pas été modifié. Elle est créée en signant le header et le payload avec une clé secrète.



4. Mode de fonctionnement

Authentification : Le client envoie ses informations d'identification (par exemple, nom d'utilisateur et mot de passe) au serveur.
Génération du JWT : Le serveur vérifie les informations d'identification et génère un JWT.
Transmission au Client : Le serveur envoie le JWT au client dans la réponse HTTP.
Stockage du JWT : Le client stocke le JWT (par exemple, dans le stockage local ou les cookies).
Requêtes Authentifiées : Pour chaque requête ultérieure nécessitant une authentification, le client inclut le JWT dans l'en-tête Authorization..

5. Avantages

Pas besoin de stocker des informations côté serveur. Chaque requête est indépendante et contient tout ce qui est nécessaire à la validation.
Le serveur n'a pas besoin de maintenir l'état de la session, ce qui facilite la scalabilité horizontale (multi-serveur).
Le JWT peut être utilisé pour l'authentification entre plusieurs services dans une architecture distribuée.

6. Inconvénients

Si un JWT est volé ou mal stocké, l'attaquant peut l'utiliser pour accéder aux ressources tant qu'il est valide (pas d'invalidation immédiate).

7. Les Jetons Uniques

Un jeton unique est un identifiant temporaire généré côté serveur pour chaque action ou requête. Contrairement aux JWT, un jeton unique est généralement utilisé pour vérifier l'authenticité d'une requête spécifique, comme une demande de modification ou une action sensible, et il n'est pas conçu pour gérer l'authentification sur plusieurs requêtes.

7.1. Mode de fonctionnement

Lors de la connexion ou d'une action, le serveur génère un jeton unique et l'associe à la session de l'utilisateur.
Ce jeton est ensuite envoyé au client, souvent sous forme de cookie ou de champ caché dans un formulaire.
Lorsque l'utilisateur effectue une action (par exemple, soumettre un formulaire), le jeton est renvoyé avec la requête.
Le serveur vérifie si le jeton renvoyé correspond à celui stocké dans la session. Si c'est le cas, l'action est validée. Sinon, la requête est rejetée.

7.2. Avantages

Le jeton unique permet de s'assurer que la requête provient bien de l'utilisateur légitime (contre les attaques de type CSRF par exemple).
Il est facile à mettre en œuvre, surtout dans des applications monolithiques.

7.3. Inconvénients

Contrairement au JWT, un jeton unique nécessite un stockage côté serveur, ce qui peut poser des problèmes de scalabilité.
Le jeton unique est généralement valide pour une seule action, ce qui implique qu'il doit être renouvelé pour chaque nouvelle requête nécessitant une vérification.

7.4. Exemple

La création

```
$_SESSION['token'] = bin2hex(random_bytes(32));
```

La vérification

```
if (!isset($_SESSION['token'])) {
    Erreur::envoyerReponse("Le jeton de vérification n'est pas présent sur le serveur.", 'global');
}
if (!isset($_REQUEST['token'])) {
    Erreur::envoyerReponse("Le jeton de vérification n'a pas été transmis.", 'global');
}
if ($_SESSION['token'] !== $_REQUEST['token']) {
    Erreur::envoyerReponse("Absence d'un jeton valide ….", 'global');
}
```