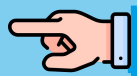
JWT

JWT y la seguridad en servicios WEB (Parte I)

{desafío} latam_

Implementar mecanismos de seguridad a un servicio REST de un servidor Express utilizando JWT de acuerdo al entorno Node.js.

- Unidad 1:
  API REST
- Unidad 2:
  Subida de archivos al servidor
- Unidad 3:
  JWT



Te encuentras aquí

{desafío}
latam_

¿Qué aprenderás en esta sesión?

- *Identifica las componentes principales del protocolo JWT y la estructura básica de la mensajería para la securización de servicios REST.*

- *Reconoce la necesidad del uso de seguridad en servicios Web y los mecanismos necesarios para su implementación.*

{desafío} latam_

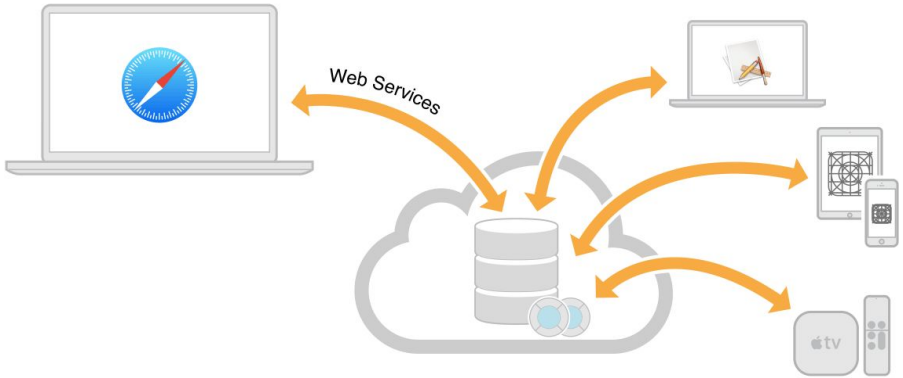
¿Que se les viene a la mente cuando hablamos de Token?

/* Seguridad en servicios web */

{desafío}
latam_

Seguridad en servicios web

Cada uno de nosotros estamos representados en la mayoría de los sistemas por una propiedad identificadora como bien puede ser la IP, rut, email o un ID/PIN generado por una aplicación. Los servicios web, así como lo muestra la siguiente imagen, se pueden interpretar como sistemas autónomos que centralizan las consultas realizadas desde diferentes dispositivos identificando sus necesidades y las credenciales de los usuarios.



/* JWT */

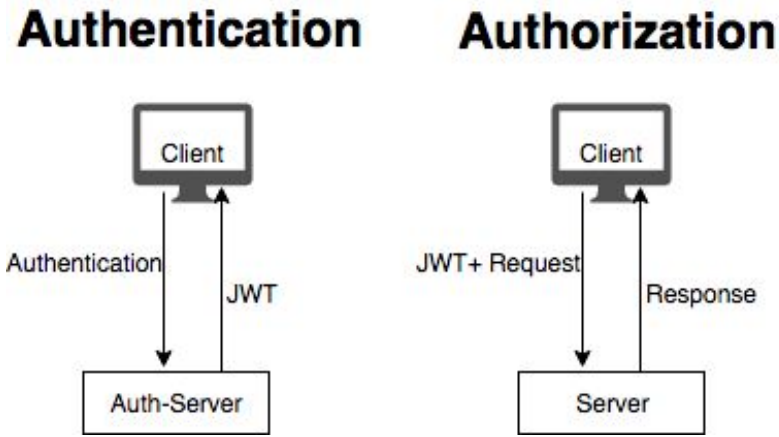
JWT

Por sus siglas JSON WEB TOKEN, es una tecnología que usa JSON y el algoritmo HS256 para la generación de tokens que contienen datos ocultos que son usados para la validación y autorización de usuarios durante la comunicación entre dos aplicaciones.

¿Qué es un token?

Los token en pocas palabras, son “firmas digitales” que representan un conjunto de datos codificados bajo un algoritmo determinado.

Lo ocupamos para autorizar a un usuario a consumir un recurso o contenido restringido. La siguiente imagen muestra la comunicación en una arquitectura cliente-servidor y cómo interviene el JWT.



¿Qué es un token?

Observa como la autenticación y la autorización son dos etapas diferentes que manejan procesos distintos.

En la autenticación, como primera acción el servidor confirma la existencia de un usuario en la base de datos validando por ejemplo, la coincidencia del correo con la contraseña, luego de esto, devuelve un token generado con la información detallada de dicho usuario (datos personales) y es entonces cuando el cliente con el token en su poder realiza consultas al servidor esperando ser verificado y autorizado para procesar su petición.

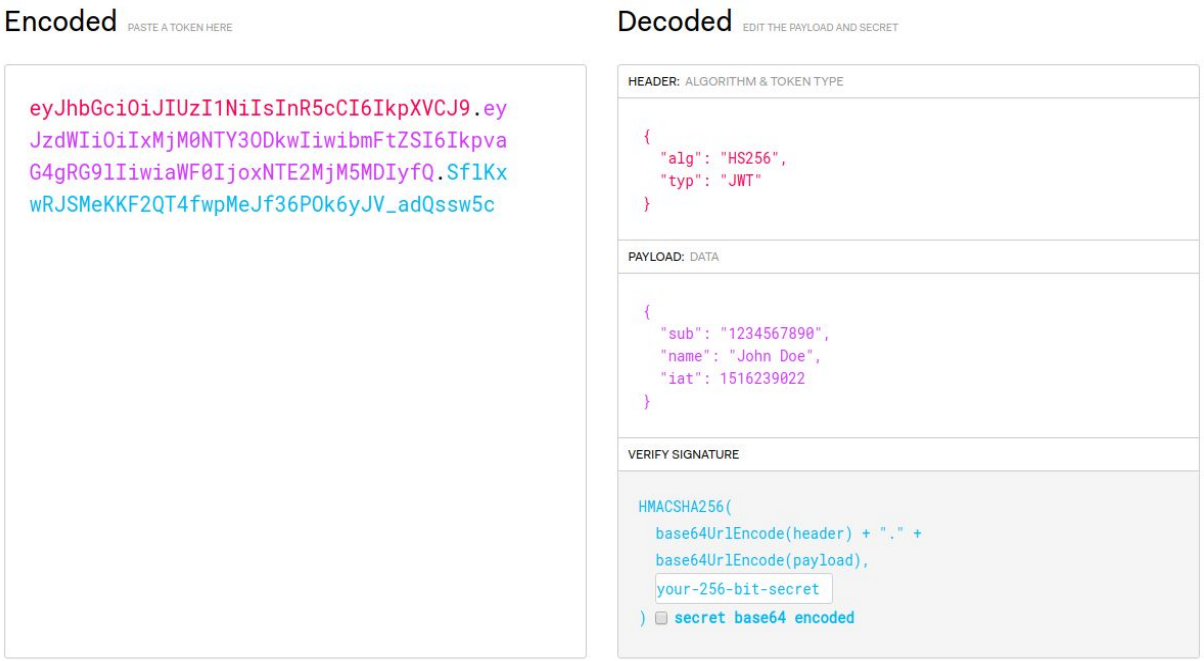
/* Estructura de un JWT */

Estructura de un JWT

Te has preguntado en estos minutos ¿Cómo se ve un JWT? ¿Qué forma tiene? En la siguiente imagen vemos su estructura y cómo está compuesta.

Estructura de un JWT

En la siguiente imagen te muestro la interfaz disponible en el sitio oficial de JWT, que puedes utilizar para entender mejor cómo funciona esta tecnología.



{desafío} latam_

Estructura de un JWT

Propiedades

Header

alg: Algoritmo usado para la firma o encriptación. Puede ser HS256 o RS256

typ: Tipo de token.

Payload

sub: Subject o ID de referencia del token.

name: Un nombre cualquiera. Es un ejemplo de envío de datos, en este caso simplemente un nombre.

iat: En inglés Issued at. Identifica el momento en el que se emitió el JWT.

Verify signature

Verify signature: Especificaciones de la codificación y decodificación del token en base64.

/* El modelo stateless */

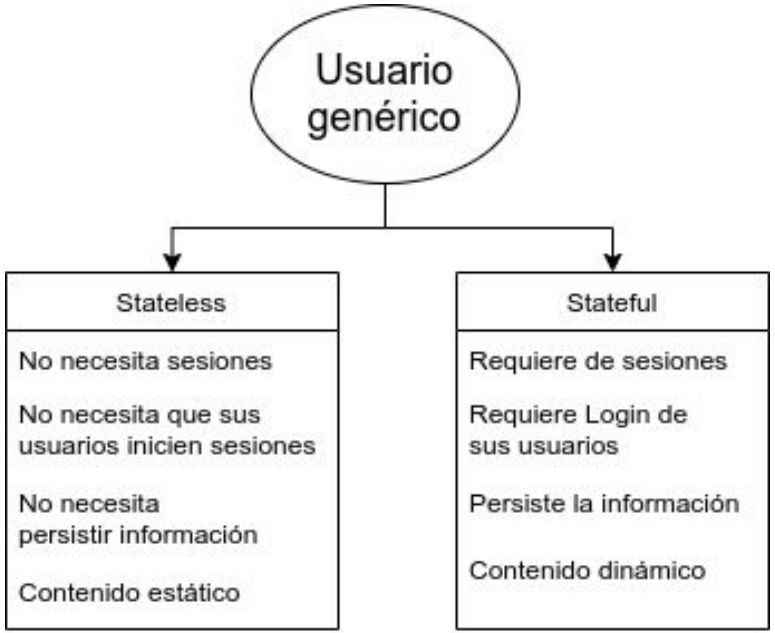
El modelo stateless

Cuando hablamos de la comunicación entre 2 aplicaciones que fueron diseñadas bajo la arquitectura cliente-servidor, denominamos con la palabra “estado” a los diferentes escenarios que puedan suceder durante esta “conversación”.

En un modelo basado en estados(stateful), para persistir las sesiones de los usuarios es necesario que éstos contengan la data que está siendo enviada desde el cliente, para ser procesados y alojados en el lado del servidor.

El modelo stateless

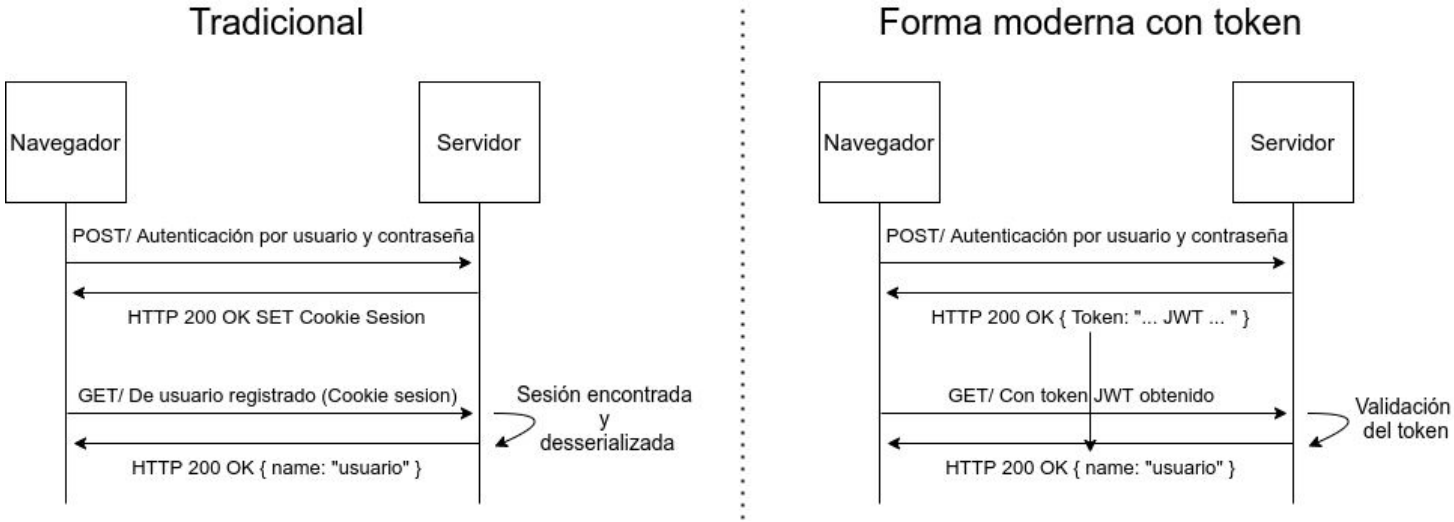
JWT se considera “stateless”, puesto que no necesita almacenar ni persistir su rastro en ningún lado, lo cual marca una fuerte diferencia con las estrategias de persistencia de usuarios como las cookies y las variables de sesiones. JSON Web Token contiene por sí mismo lo que necesita para ser verificado, ya que el algoritmo que lo firma incluye esta cualidad haciéndolo infalsificable



/* Ciclo de vida de un JWT */

Ciclo de vida de un JWT

En la siguiente imagen, veremos la diferencia entre la autenticación y autorización tradicional, además de la forma moderna al utilizar JWT en una arquitectura cliente-servidor.



{desafío} latam_

Ciclo de vida de un JWT

Ahora analicemos las siguientes diferencias:

- En el modelo tradicional el servidor recibe una consulta POST con los datos de un usuario y luego de almacenarlos devuelve una Cookie. En la versión moderna con token, lo que sucede es que el servidor devuelve un token generado con los datos firmados del usuario.

- Posteriormente en ambos modelos se realiza otra consulta, pero en esta ocasión de tipo GET. En el modelo tradicional el servidor debe buscar y entrar en la sesión previamente almacenada y devolver los datos al cliente, mientras que con JWT solo se necesita verificar la validez del token para devolver los datos decodificados.

Resumen

- Los **token** en pocas palabras, son “firmas digitales” que representan un conjunto de datos codificados bajo un algoritmo determinado, los utilizamos para autorizar a un usuario a consumir un recurso o contenido restringido.

- La estructura de un token se divide en:
  - Header
  - Payload
  - Verify signature

¿Existe algún concepto que no hayas comprendido?

Próxima sesión...

- *JWT y la seguridad en servicios WEB (Parte II)*

{desafío}
latam_

Academia de
talentos digitales