# COMPONENTES DE UN JWT

## TEXT CLASS REVIEW

## TEMAS A TRATAR EN EL CUE

- Securización mediante JWT.
- Componentes de un JWT.
- Header.
- Algoritmos de encriptación.
- Payload.
- El elemento iat.
- Signature.
- El código Base64.
- Almacenamiento en LocalStorage/SessionStorage.
- Revisión de algunos de los campos estándares de la mensajería.

### COMPONENTES DE UN JWT

Un JSON Web Token está principalmente compuesto por tres partes:

- Encabezado o `header`.
- Contenido o `payload`.
- Firma o `signature`.

Estos tres componentes de un JWT están separados por puntos (`.`), así:

```
xxxxxx.yyyyyy.zzzzzz
```

JWT define la estructura de la información que enviamos de una parte a otra, y se presenta en dos formas: serializado o deserializado. El primer enfoque se utiliza principalmente para transferir los datos a través de la red con cada solicitud y respuesta; mientras que el segundo enfoque se usa para leer y escribir datos

en el token, y en éste, el JWT contiene solo el encabezado y el contenido. Ambos son objetos JSON simples.

**EL ENCABEZADO O HEADER:**

Se usa principalmente para describir el tipo de token, que en este caso es JWT, y el algoritmo usado para verificarlo. También puede contener datos sobre el tipo de medio o contenido de la información que enviamos. Esta información está presente como un objeto JSON, y luego, éste se codifica en BASE64URL. Las operaciones criptográficas en el encabezado definen si el JWT está firmado, sin firmar, o encriptado, y por lo tanto, qué técnicas de algoritmo usar. Un encabezado simple de un JWT se parece al siguiente código:

```
{
    "typ": "JWT",
    "alg": "HS256"
}
```

El `'typ'` y el `'alg'` son claves de objeto, que tienen diferentes valores y funciones diferentes. El `'typ'` nos da el tipo de encabezado de este paquete de información, mientras que el `'alg'` nos informa sobre el algoritmo de cifrado utilizado.

Algunos JWT también se pueden crear sin firma ni cifrado. Dicho token se denomina no seguro, y su encabezado debe tener el valor de la clave de objeto de `'alg'`, asignado como "ninguno". Así:

```
{
    "alg": "none"
}
```

**EL CONTENIDO O PLAYLOAD:**

La segunda parte del token es el contenido, que está compuesta por fragmentos de información (*claims*). Estos fragmentos son declaraciones sobre una entidad (generalmente el usuario), y datos adicionales. Existen tres tipos de fragmentos: registrados, públicos y privados.

Fragmentos de información registrados: se trata de un conjunto de información predefinida, que no son obligatorios pero se recomiendan para proporcionar un conjunto de fragmentos útiles e interoperables. Algunos de ellos son: `iss` (emisor), `exp` (tiempo de caducidad), `sub` (sujeto), `aud` (audiencia), entre otros.

Fragmentos de información públicos: éstos pueden ser definidos a voluntad por aquellos que usan JWT. Pero para evitar colisiones, deben definirse según el **IANA JSON Web Token Registry,** o como un URI que contenga un espacio de nombres resistente a colisiones.

Fragmentos de información privados: éstos son personalizados, creados para compartir información entre partes que acuerdan usarlos, y no son fragmentos de información ni registrados ni públicos.

El contenido de un JWT puede representarse de la siguiente manera:

```
{
    "userId": "b07f85be-45da",
    "iss": "https://provider.domain.com/",
    "sub": "auth/some-hash-here",
    "exp": 153452683
}
```

Este JWT contiene el `userId`, `iss`, `sub` y `exp`. Cada uno tiene un rol diferente. El `userId` almacena el identificador del usuario que se está almacenando; el `iss` muestra información acerca del emisor; `sub` significa asunto; y `exp` fecha de expiración o vencimiento.

El contenido o `playload`, generalmente contiene información del usuario, la cual está presente como un objeto JSON, y luego éste se codifica en BASE64URL. Se pueden utilizar tantos fragmentos de información como sea necesario dentro de esta parte del JWT, aunque a diferencia del encabezado, ningún fragmento es obligatorio.

JWT en forma serializada, representa una cadena con el siguiente formato:

```
[header].[payload].[signature]
```

Estos tres componentes conforman el JWT serializado. Ya se han descrito el encabezado y el contenido, y para qué son utilizados.

**LA FIRMA O SIGNATURE:**

Esta es la tercera parte de un JWT, y se utiliza para verificar la autenticidad del token. El encabezado y el contenido, los dos primeros componentes de un JWT, son codificados en BASE64URL, y se unen o concatenan con un punto (.); luego, se codifica mediante el algoritmo de cifrado definido en el encabezado

con una clave secreta. Posteriormente, esta firma se agrega al encabezado y al contenido usando un punto (.), y finalmente se completa nuestro token real con la estructura siguiente:

```
header.payload.signature
```

La sintaxis de la codificación se puede representar de la siguiente manera:

```
HASHINGALGORITM(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    secret)
```

Entonces, todos estos componentes anteriores juntos, son lo que constituye un JWT. Ahora, veamos cómo se mostrará nuestro token real:

**HEADER:**

```
{
    "alg": "HS256",
    "typ": "JWT"
}
```

**PAYLOAD:**

```
{
    "id": 123456789,
    "name": "Joseph"
}
```

**SECRET:** `secretword`

JSON WEB Token encriptado:

```
eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpZCI6MTIzNDU2Nzg5LCJuYW1lIjoiSm9zZ
XBoIn0.OpOSSw7e485LOP5PrzScxHb7SR6sAOMRckfFwi4rp7o
```

La firma se usa para verificar que el mensaje no se modificó en el camino que recorre, y en el caso de tokens firmados con una clave privada, también puede verificar que el remitente del JWT es quien dice ser.

**ALGORITMOS DE ENCRIPTACIÓN**

Un algoritmo de cifrado es un componente para la seguridad del transporte electrónico de datos. Ayudan a prevenir el fraude de datos, como el perpetrado por piratas informáticos que obtienen ilegalmente información electrónica. Éstos son parte de los protocolos de gestión de riesgos de cualquier empresa, y a menudo se encuentran en aplicaciones de software.

También ayudan en el proceso de transformación de texto sin formato en texto cifrado, y luego de vuelta a texto sin formato, con el fin de proteger los datos electrónicos cuando se transportan a través de redes.

La encriptación de datos permite:

- Confidencialidad: codifica el contenido del mensaje.
- Autenticación: verifica el origen de un mensaje.
- Integridad: demuestra que el contenido de un mensaje no ha cambiado desde que se envió.
- No rechazo: evita que los remitentes nieguen haber enviado el mensaje cifrado.

Hay tres componentes principales para cualquier sistema de cifrado: los datos, el motor de cifrado, y la gestión de claves.

Estos son los principales algoritmos de cifrado existentes:

**CLAVES SIMÉTRICAS:**

En la criptografía de cifrado simétrico, se utiliza la misma clave de cifrado para cifrar y descifrar los datos. Este medio de cifrado se utiliza principalmente para proteger los datos en reposo. Un ejemplo sería cifrar datos confidenciales en texto cifrado, mientras están almacenados en una base de datos, y descifrarlos en texto sin formato cuando un usuario autorizado accede a ellos, y viceversa.

**CLAVES ASIMÉTRICAS:**

Por otro lado, son un par de claves para el cifrado y descifrado de los datos. Ambas están relacionadas entre sí, y se crean al mismo tiempo. Se les conoce como clave pública y privada. La clave pública se usa principalmente para cifrar los datos, y se puede dar libremente, ya que se usará para cifrar datos, no para descifrarlos. Mientras que la clave privada se utiliza para descifrar los datos que su contraparte, la clave pública, ha cifrado. Esta clave debe protegerse, ya que es la única que puede descifrar los datos cifrados.

Generalmente, los algoritmos de cifrado utilizados en un entorno en el que se hace uso de JWT son HMAC con SHA-256(HS256), y firma digital con SHA-256(RS256).

A la hora de elegir el algoritmo de firma de token, hay que tener en cuenta que los algoritmos de clave simétrica son vulnerables a ataques de fuerza bruta si la clave usada no es lo suficientemente fuerte, con lo que hay que proporcionar suficiente complejidad si se eligen algoritmos de clave simétrica. Por otro lado, los algoritmos de clave asimétrica simplifican la custodia de la clave, ya que ésta sólo es necesaria en la parte servidor que genera el token.

**EL ELEMENTO IAT**

Es un fragmento de información que no es obligatorio incluir en un JWT, pero que su nombre es reservado. El estándar sugiere incluir una marca temporal o timestamp en inglés, llamado `iat` (issued at), para indicar el momento en el que el token fue creado.

**EL CÓDIGO BASE64**

Es un grupo de esquemas de codificación de binario a texto, que representa los datos binarios mediante una cadena ASCII. El término Base64 se origina de un sistema de codificación de transmisión de contenido MIME específico.

Los esquemas de codificación Base64 son comúnmente usados cuando se necesita codificar datos binarios, para que éstos sean almacenados y transferidos sobre un medio diseñado para tratar con datos textuales. Esto para asegurar que los datos se mantienen intactos y sin modificaciones durante la transmisión.

La codificación Base64 puede ser útil cuando se usa información de identificación bastante extensa en un entorno HTTP. Además, muchas aplicaciones necesitan codificar datos binarios de una manera conveniente para incluirlos en URL, incluso en campos de formularios web ocultos, y Base64 es una codificación conveniente para representarlos de manera compacta.

El uso de Base64 estándar en la URL requiere la codificación de los caracteres `'+'`, `'/'` y `'='`, en secuencias hexadecimales codificadas en porcentajes especiales, lo que hace que la cadena sea innecesariamente más larga. Por esta razón, existen Base64 modificado para variantes de URL, donde los caracteres `'+'` y `'/'` del estándar se reemplazan respectivamente por `'-'` y `'_'`, y así el uso de codificadores/descodificadores de URL ya no es necesario, y no tiene impacto en la longitud del valor codificado, dejando intacta la misma forma codificada para su uso en bases de datos relacionales, formularios web, e identificadores de objetos en general.

Algunas variantes permiten o requieren la omisión de los signos de relleno `'='`, para evitar que se confundan con los separadores de campo, o necesitan que dicho relleno esté codificado en porcentaje. Algunas bibliotecas codificarán `'='` a `'.'`.

**ALMACENAMIENTO EN LOCALSTORAGE/SESSIONSTORAGE**

Los objetos de almacenamiento web `localStorage` y `sessionStorage` permiten guardar pares de clave/valor en el navegador. Lo que es interesante sobre ellos es que los datos sobreviven a una recarga de página (en el caso de `sessionStorage`), y hasta un reinicio completo de navegador (en el caso de `localStorage`).

Las principales funcionalidades de `localStorage` son:

- Es compartido entre todas las pestañas y ventanas del mismo origen. De modo que si guardamos datos en una ventana, el cambio es visible en la otra.
- Los datos no expiran. Persisten reinicios de navegador, y hasta del sistema operativo.

El objeto `sessionStorage` se utiliza mucho menos que `localStorage`. Las propiedades y métodos son los mismos, pero es mucho más limitado:

- `sessionStorage` solo existe dentro de la pestaña actual del navegador.
- Otra pestaña con la misma página tendrá un almacenaje distinto.
- Los datos sobreviven un refresco de página, pero no a cerrar/abrir la pestaña.

En resumen, los objetos de almacenamiento web `localStorage` y `sessionStorage` permiten guardar pares de `clave/valor` en el navegador.

- Tanto la clave como el valor deben ser strings, cadenas de texto.
- El límite es de más de 5mb+; depende del navegador.
- No expiran.
- Los datos están vinculados al origen (dominio/puerto/protocolo).

**REVISIÓN DE ALGUNOS DE LOS CAMPOS ESTÁNDARES DE LA MENSAJERÍA**

El estándar define los siguientes campos que pueden incluirse en los tokens JWT:

| Código | Nombre | Descripción |
|---|---|---|
| iss | Issuer | Identifica el proveedor de identidad que emitió el JWT. |
| sub | Subject | Identifica el objeto o usuario en nombre del cual fue emitido el JWT. |
| aud | Audience | Identifica la audiencia o receptores para lo que el JWT fue emitido, generalmente el/los servidor/es de recursos (e.g. la API protegida). Cada servicio que recibe un JWT para su validación tiene que controlar la audiencia a la que el JWT está destinado. Si el proveedor del servicio no se encuentra presente en el campo aud, entonces el JWT tiene que ser rechazado. |
| exp | Expiration time | Identifica la marca temporal luego de la cual el JWT no debe ser aceptado. |
| nbf | Not before | Identifica la marca temporal en que el JWT comienza a ser válido. Éste no tiene que ser aceptado si el token es utilizado antes de este tiempo. |
| iat | Issued at | Identifica la marca temporal en que el JWT fue emitido. |
| jti | JWT ID | Identificador único del token, incluso entre diferentes proveedores de servicio. |

Los siguientes campos pueden ser utilizados en el encabezado:

| Código | Nombre | Descripción |
|---|---|---|
| typ | Token type | Si está presente, se recomienda utilizar el valor JWT. |
| cty | Content type | En casos normales, no es recomendado. En casos de firma o cifrado anidado, debe estar presente y el valor debe ser JWT. |
| alg | Message authentication code algorithm | El proveedor de identidad puede elegir libremente el algoritmo para verificar la firma del token, aunque algunos de los algoritmos soportados son inseguros. |