Tokens JWT

Transcribed on July 10, 2025 at 10:29 AM by Minutes AI

Speaker 1 (00:04)

Bienvenidos a esta nueva sesión.

En esta sesión vamos a conocer un uso de la criptografía, en este caso a través de los conocidos JSON Web Tokens.

Vamos a comprender qué son, por qué son importantes y qué componentes lo componen.

Los JSON Web Tokens o JWT son un estándar abierto definido en el RFC 7519.

En términos simples, son como pequeños paquetes de información que están encapsulados en un formato JSON, pero su verdadera magia reside en la capacidad para transferir esta información de manera segura entre diferentes partes.

¿Entonces, por qué son importantes estos tokens?

Los JSON Web Token son una herramienta fundamental en el ámbito de la autenticación y la autorización en el mundo de la seguridad informática.

Nos permiten verificar la identidad de un usuario y otorgarle acceso a recursos de manera segura, sin necesidad de almacenar una sesión en el servidor.

Están compuestos por tres partes diferenciadas que son la cabecera o el header, la carga útil o el payload y la firma o signature, que esta es la que asegura la integridad del token.

Entender estos componentes es crucial para comprender cómo funcionan y Logge Web Token y cómo podemos utilizarlos de manera efectiva en nuestros sistemas.

Vamos a continuación a desglosar la estructura interna de un yeso web token y entender cómo se compone cada una de estas partes.

Y comenzamos con la cabecera.

Esta sección contiene metadatos importantes sobre el token, como es el tipo de token que tiene y el algoritmo de cifrado utilizado.

Normalmente el tipo de token se JWt y el algoritmo de cifrado va a depender si se está utilizando clave simétrica o clave asimétrica.

Esencialmente, esta sección nos dice cómo debemos interpretar y verificar el token.

Pasamos al payload, que aquí es donde reside la información útil o la carga útil.

En este componente encontramos los llamados claims, que básicamente son declaraciones sobre la identidad del usuario y cualquier otro dato relevante para nuestra aplicación.

Esto podría incluir, por ejemplo, el nombre de usuario, los distintos roles que tiene, si es un administrador, si tiene permisos de lectura, de escritura e incluso información adicional personalizada.

Y finalmente llegamos a la tercera parte que es la firma.

Se trata de una firma digital que está generada mediante la combinación del header y del payload, junto con una clave secreta.

La firma se utiliza para verificar la integridad del token y garantizar que no ha sido alterado durante su trabaj transmisión.

También tenemos que conocer que estos componentes se representan como cadenas codificadas en formato base 64, pero que en realidad son únicamente objetos JSON.

Esto significa que podríamos pensar que se trata de una cadena de caracteres aleatoria a simple vista, pero realmente se puede decodificar fácilmente para inspeccionar su contenido en un formato legible.

Es importante entender la estructura interna de un JSON Web Token, ya que éste nos proporciona una visión clara de cómo se transmiten y cómo se verifican los datos de manera segura en las aplicaciones y en los servicios.

Ahora pasamos a hablar sobre el proceso completo de creación y verificación de un JSON Web token, desde su generación en el servidor hasta su uso por parte del cliente y su validación de nuevo en el servicio.

Comenzamos en primer lugar con la generación del JSON web token en el servidor.

Aquí el servidor va a combinar el header y el payload para formar el token.

Posteriormente, se va a firmar el token utilizando una clave secreta que únicamente conoce este servidor, creando así una firma única que se utilizará para verificar la autenticidad del token más adelante.

Una vez generado el token, el servidor lo va a mandar al cliente, generalmente como parte de una respuesta HTTP.

El cliente recibirá el token y lo almacena, listo para ser incluido en futuras solicitudes.

Cuando el cliente realiza una solicitud posterior, éste va a incluir el JSON web token en el encabezado de autorización, o incluso también podría ser incluido como una cookie HTTP only.

El servidor recibe el token y comienza el proceso de verificación.

En primer lugar, verifica la integridad del token, verificando la firma utilizando la misma clave secreta que se había utilizado anteriormente para generar el token.

Si la firma es válida, el servidor procede a decodificar el payload del JSON web token.

Aquí es donde se encuentran las declaraciones sobre la identidad del usuario y cualquier otro dato importante.

Con esta información decodificada, el servidor puede realizar las acciones por el notiente, como puede ser audiencia al usuario o autorizar el acceso a distintos recursos.

Como se puede ver, este proceso es simple pero fundamental para establecer una comunicación segura entre el servidor y el cliente.

Ahora vamos a profundizar en algunos casos de uso comunes de los JSON Web Tokens y cómo pueden aplicarse en diferentes escenarios.

Comenzamos con uno de los usos más fundamentales de los Hot Wt, qué es la autenticación de usuarios en aplicaciones web y en APIs.

Los JSON Web Token nos permiten autenticar a los usuarios de manera segura, eliminando la necesidad de almacenar una sesión en el servidor.

Esto significa que podemos verificar la identidad de un usuario sin tener que mantener un estado de sesión activo en el server, lo que simplifica considerablemente nuestra arquitectura y mejora la escalabilidad.

Otro caso de uso importante es la autorización y el control de acceso a recursos.

Los JWT contienen información de autorización que puede utilizarse para controlar quien tiene acceso a nuestros recursos protegidos dentro de una aplicación o un recurso API.

Esto nos permite implementar políticas de acceso granulares y garantizar que únicamente aquellos usuarios autorizados puedan acceder a la información o funcionalidades que necesita.

Los JSON Web Token también son útiles en el contexto del Single sign on y la Federación de identidad.

Con JWT podemos implementar un proceso de inicio de sesión único, lo que significa que un usuario puede iniciar sesión una única vez y acceder a múltiples servicios sin tener que volver a autenticarse en cada uno de ellos.

Esto simplifica la experiencia del usuario y reduce la carga de gestión en el para los usuarios y los administradores del sistema.

Por último, los JSON Web Token son excelentes para transferir información entre diferentes servicios de manera segura.

Dado que la información dentro del token está firmada y cifrada, podemos estar seguros de que no ha sido alterada durante la transferencia.

Esto los hace ideales para integraciones entre servicios donde necesitaríamos compartir información de manera segura y confiable.

Otra de las cosas que tenemos que tener en cuenta son algunas consideraciones de seguridad y buenas prácticas a la hora de utilizar jwt en las aplicaciones y en los servicios.

Comenzamos hablando sobre los algoritmos de cifrado recomendados para la generación y verificación de firmas en JSON Web Tokens.

Para firmas simétricas como Hmac, se recomienda utilizar el algoritmo HMA 256, mientras que para firmas asimétricas, como sería el caso de RSA, se aconseja el uso de RS.

Estos algoritmos son robustos y ampliamente aceptados en la industria, proporcionando un nivel adecuado de seguridad para nuestros tokens.

Como en otros casos, cuando hablamos de claves criptográficas, es crucial que manejemos nuestras claves secretas con sumo cuidado.

Esto incluye almacenarlas de formas seguras, protegiéndolas contra accesos no autorizados.

También debemos considerar la rotación periódica de estas claves para reducir el riesgo de compromiso de la seguridad en el largo plazo.

La seguridad de nuestras claves es fundamental para garantizar la integridad y la confidencialidad de nuestros JSON web tokens.

Otro aspecto importante es establecer fechas de expiración en estos tokens.

Limitar el tiempo de validez de un token reduce la ventana de oportunidad para posibles ataques.

Además, es recomendable implementar la renovación automática de los tokens antes de su expiración.

De esta manera nos garantizamos una experiencia de usuario sin interrupciones al tiempo que se mantiene muy alto el nivel de seguridad en nuestras aplicaciones.

Finalmente, es crucial estar al tanto de los posibles ataques contra los JSON Web Token y tomar medidas para mitigar estos riesgos.

Por ejemplo, podemos incluir información adicional en el Payload del JSON Web Token para validar la solicitud o verificar la audiencia para garantizar que el token sea utilizado por el destinatario previsto.

Estas medidas adicionales ayudan a fortalecer la seguridad de nuestros sistemas y protegerlos contra posibles vulnerabilidades.

A continuación vamos a mostrar de manera gráfica un depósito google de jwt para ver cómo podemos generar nuestros propios tokens y validarlos en un entorno de pruebas.

Me encuentro en el sitio web jwt IO donde podemos encontrar información detallada sobre los JSON Web Token.

Ya nada más entrar se nos está haciendo hincapié, se nos está indicando aquí el RFC 7519, que se recuerda que este es el RFC destinado a los JSON Web Topics.

Por otro lado, aquí podemos encontrar una sección de librerías en la cual podemos encontrar librerías para distintos lenguajes de programación, por ejemplo aquí tenemos.

Net, pero tenemos muchos más como podrían ser C, java, python, Kotlin, etc.

Nos vamos a ir por ejemplo al lenguaje de programación de Python y vemos como se nos están indicando cuatro librerías distintas y por supuesto de cada una de ellas se nos indica con qué cosas van a trabajar, tanto a nivel de claims, que sería esta parte de la izquierda, como a nivel de los algoritmos de firma que tendríamos aquí.

Si nos vamos al debugger, aquí podemos ver las distintas partes.

Por un lado vemos la cabecera señalado en rojo, aquí en lila tendríamos el payload o la carga útil y aquí en esta última parte tendríamos la firma y como también vamos a verificar la firma un poco más abajo.

Fijaos que también podríamos seleccionar el tipo de algoritmo si quisiéramos que sea simétrico o asimétrico dependiendo del que estemos seleccionando, pues vamos a ver cómo nos cambia.

Aquí tendríamos que indicar la clave pública y privada para poder generarlo, pero sin embargo si elegimos otro como el que teníamos, el HS, pues vemos que éste únicamente tenemos que indicar el secreto.

¿Qué vamos a encontrar aquí?

Pues en primer lugar encontramos el header, que en este header se nos está indicando el algoritmo, en este caso HS, y el tipo, que siempre va a ser Jsonway Token.

De hecho si vamos cambiando por aquí, si volvemos a cambiar aquí, vemos que lo único que cambia es esta información.

¿Qué pasa?

Recordad que esto puede ser recuperado ya que aquí lo tenemos representado en Base 64.

Si nos vamos por ejemplo a un decodificador de Base 64 y lo pegamos aquí, podemos ver que este resultado lo podemos leer de la misma manera.

Si aquí hiciéramos cualquier tipo de cambio, este valor va a ser el Base 64, mejor dicho este valor de la izquierda representa el Base 64 de esto que tenemos aquí a la derecha.

Vamos a volver a ponerlo bien, aquí lo tenemos y bueno esto sería con respecto a la carpeta, de esta manera cuando se vaya a validar el token ya se sabe con qué tipo de algoritmo va trabajar y va a validar.

Por supuesto que se trata de JSON Web Token.

Para un caso en el que siempre se trabaje con JSON Web Token con un algoritmo HS, la primera cabecera, perdón, la primera parte de base 64 siempre va a ser este string en base 64.

En segundo lugar tendríamos aquí la carga útil que para esta ocasión pues bueno, esta información de JSON Web Token, a ver si me deja subir, aquí tenemos el subject, que esto es información, cada uno podría poner la que quisiera, pero si revisamos el RFC es verdad que hay una serie de claims clave para poder trabajar de manera más ágil.

Pero bueno, vamos a entender primero que esto será un archivo JSON y que podremos crear lo que nosotros queramos.

Por ejemplo aquí tenemos este valor name, pues simplemente para ser saber este nombre de quién está llegando, pero podría ser cualquier otro tipo de identificador o por ejemplo aquí que se nos esté indicando cuándo ha sido lanzado o generado este token.

Pues esto es información que de nuevo destaco, se podría ver porque como veis se trata de base 64 y si lo ponemos aquí vamos a poder ver o leer esto.

Por tanto JSON Web Token no se recomienda que aquí se esté mandando ningún tipo de credencial ya que esta información puede ser vista.

Lo que sí que es interesante es que esta información va a poder ser validada y que si nos fijamos, si alguien cambiase algo aquí, vamos a suponer que nos venimos aquí y fijaos que yo quisiera cambiar, vamos a hacer esto mismo y lo que vamos a hacer va a ser un base 64 encode, vamos a ponerla aquí, imaginaos que yo digo oye es que en vez de ser este mi identificador pues lo voy a poner al revés, imaginaos que digo oye quiero hacerme pasar por esto, vamos a ver si esto, bueno esto como veis me estaría dando el balón en base 64, yo lo generaría y diría oye pues vamos a copiarlo y vamos a pegarlo aquí para sustituir a esta información.

¿Qué pasa?

Que la firma nos está dando inválida, es decir, no cuadran los campos con lo que está aquí escrito.

Esto que es lo que yo quiero validar no se corresponde con este token, este token aquí me lo está pintando bien, esta parte como veis sí que tiene que ver con lo que yo he escrito, pero la validación es incorrecta porque la parte importante que es la firma no cuadra.

¿Por qué?

Porque la gente que haya firmado esto no va a conocer la clave con la que hemos cifrado.

Imaginaos que yo ahora cojo y lo cambiase, lo voy a cambiar aquí para arreglar la firma, esta sería la firma correcta y fijaos que aquí yo voy a poder definir el secreto que yo quiera, incluso decir que quiero poner este secreto en base 64.

Y para que veáis que esto funciona, yo ahora mismo lo que tendría que hacer sería, para poder representar por ejemplo este valor, sería coger, en este caso esta es mi contraseña, el youbit secret, pero aquí pone, podríamos poner lo que sea, prueba JWT.

Fijaos que en este caso siempre que yo escribo aquí la firma me está cambiando.

¿Por qué?

Porque en el algoritmo HMA vamos a coger esta primera información, que si os fijáis es el header, un punto y el payload, es decir, literalmente vamos a coger este string y a este string le voy a pasar esta contraseña, por así decirlo.

Entonces si nos vamos por ejemplo a un validador de este algoritmo, fijaos que yo podría coger este de aquí, vamos a copiarlo y nos vamos a ir a un sitio web para poder calcular esto de manera online.

¿El hmax sea 256, no?

Pero perdonad porque, o sea, este es el resultado que me tiene que salir, pero yo lo que tengo que copiar es este apartado de aquí.

Vamos a ponerlo aquí, vamos a pegarlo, vamos a poner la contraseña prueba jwt y decimos que el resultado lo queremos en base 64 y aquí lo que encontramos es z vu y que termina en dmk y si volvemos a nuestra página web vemos que empezamos por zvuc y que terminamos en dmk, es decir, que vemos que es exactamente lo mismo.

Como podéis ver, pues si hay algún tipo de manipulación en el medio no van a ser capaces de volver a obtener la misma firma ya que necesita, necesitarían nuestra contraseña, que en este caso es esta de aquí.

Fijaos que aquí también nos dice, oye, pues la contraseña se puede poner en base 64 y vemos como en este caso también nos cambia, pero como digo, sería el mismo resultado de pasar esto base 64 y hacer el hmax sha de esto mismo.

Como veis en este sitio web jw tenemos el de google que nos va a ayudar bastante a poder comprender y a jugar con los JSON web token y ver cómo se utilizan, cómo se hace el encode y cómo se hace el decode y cómo se verifica la firma.

Para finalizar, vamos a ver las principales conclusiones de esta sesión.

En primer lugar, se ha conocido el fascinante mundo de los JSON Web Token y se ha explorado cómo se utilizan en la autenticación y autorización de aplicaciones web y servicios app.

Ya sabemos que los JSON Web Token son objetos JSON que contienen información y son firmados digitalmente para garantizar su integridad.

Hemos desglosado la estructura de un token, comprendiendo sus tres el header, el payload y la firma.

Se ha visto como cada una de estas las partes contribuye a la seguridad y la portabilidad del token, permitiendo la transferencia segura de información entre diferentes partes de un sistema distribuido.

Además, se ha examinado en detalle el proceso de creación y verificación de los tokens, desde la generación del token en el servidor hasta su validación en el cliente.

Se ha comprendido la importancia de utilizar algoritmos de piltrado robustos y de manejar adecuadamente las claves secretas para garantizar la seguridad del sistema.

También se han comentado diversos casos de uso, desde la autenticación de usuarios en aplicaciones web hasta la autorización de accesos a recursos protegidos.

Hemos visto como los JSON Web Token pueden facilitar la implementación de silla, permitiendo que los usuarios inicien sesión una vez y accedan a múltiples servicios sin necesidad de volver a autenticarse.

Por supuesto, no hay que descuidar la parte de seguridad y por ello se han comentado algunas buenas prácticas y consideraciones al trabajar con los tokens, destacando la importancia de establecer fechas de expiración en los tokens, así como de mitigar posibles ataques mediante la inclusión de información adicional en el payload y la verificación de la audiencia del token.

Todo esto se ha podido ver en funcionamiento gracias al debugger que está disponible online en el sitio web.

Información sobre los JSON Web Token en definitiva, y como se ha podido comprobar, los JWT ofrecen una forma segura y eficiente de transferir información entre diferentes partes de un sistema distribuido.

Entender cómo funcionan y cómo utilizarlos correctamente es fundamental para garantizar la seguridad y la integridad de nuestras aplicaciones y servicios.

En esto llegamos al final de la sesión.

Os esperamos en el siguiente vídeo.