HINTS

- `HS256` y `RS256` son los dos algoritmos principales que utilizamos en la sección de encabezado de un JWT.
- Tenga en cuenta que los nombres de los fragmentos de información en el contenido tienen solo tres caracteres, ya que JWT debe ser compacto.
- IANA (*Internet Assigned Numbers Authority*): esta entidad desempeña un papel esencial en la gestión de Internet, ya que es responsable de asignar nombres y sistemas de números únicos que se usan de acuerdo con los estándares técnicos –protocolo de red– de Internet, y constituyen la base del direccionamiento de páginas web. En el caso de JWT, la IANA establece los nombres de los fragmentos públicos para así evitar colisiones.
- Tenga en cuenta que, para los tokens firmados, la información del contenido o playload, aunque está protegida contra la manipulación, puede ser leída por cualquiera. No se debe incluir información confidencial en el contenido, o en los elementos del encabezado de un JWT, a menos que esté encriptada.
- Si el algoritmo utilizado para codificar un token es `HMAC SHA256`, la firma se creará de la siguiente manera:

```
HMACSHA256(
    base64UrlEncode(header) + "." +
    base64UrlEncode(payload),
    secret)
```

- Un JWT se conforma de tres cadenas de URL Base64, separadas por puntos que se pueden pasar fácilmente en entornos HTML y HTTP, mientras que son más compactas en comparación con los estándares basados en XML, como SAML.
- Un JWT puede ser firmado tanto con un algoritmo simétrico, como uno asimétrico.
- Buenas prácticas de seguridad al utilizar JWT:
    - Siempre se debe verificar la firma de los tokens.
    - Evite las funciones de librerías que no verifican las firmas.
    - Comprobar que no se comparte la palabra secreta de las firmas simétricas.
    - Una configuración distribuida solo debe usar firmas asimétricas.

- Solo los servidores que tienen el `ACCESS_TOKEN_SECRET` compartido pueden descifrar el token JWT. Esto permite escalar múltiples servidores por separado, y sin la necesidad de tener una copia actual de la base de datos de ID de sesión en todos los servidores, o hacer que todos los servidores accedan a una base de datos de sesión común. Siempre que un servidor tenga el mismo `ACCESS_TOKEN_SECRET` que se usó para generar el token JWT, puede descifrar el token JWT, y obtener el ID de usuario.

LA DIFERENCIA ENTRE JWT Y COOKIES DE SESIÓN

Las cookies de sesión y JWT proporcionan una autenticación segura del usuario, pero son diferentes en los siguientes puntos:

- Firma criptográfica.
- JWT tiene una firma criptográfica, pero las cookies de sesión no.
- JSON no tiene estado.
- JWT no tiene estado, pues la declaración se almacena en el cliente, no en la memoria del servidor.
- La autenticación se puede realizar localmente, no en una base de datos del servidor, o en una ubicación similar donde debe pasar la solicitud. Esto significa que el usuario puede autenticarse varias veces sin tener que comunicarse con la base de datos del sitio o la aplicación, y sin consumir muchos recursos en el proceso.

ESCALABILIDAD

- Las cookies de sesión se almacenan en la memoria del servidor, lo que significa que si el sitio web o la aplicación es grande, consumirá muchos recursos. Dado que los JWT no tienen estado, en muchos casos pueden ahorrar recursos del servidor. Por lo tanto, JWT es más escalable que las cookies de sesión.
- JWT admite la autenticación entre dominios.
- Las cookies de sesión solo se pueden utilizar en Dominio de nodo único, o en subdominio. Si intentan acceder a través del tercer nodo, serán baneados. Este es un problema si desea que su sitio web establezca una conexión segura con otros sitios.
- Use JWT para resolver este problema, pues permite pasar múltiples nodos, y realice la autenticación de usuario, que es lo que se suele llamar “autenticación entre dominios”.

SELECCIÓN DE JWT Y COOKIES DE SESIÓN

Para sitios web pequeños y medianos, que solo necesitan iniciar sesión como usuarios y acceder a cierta información almacenada en la base de datos del sitio, las cookies de sesión suelen ser suficientes.

Si tiene sitios de nivel empresarial, aplicaciones o sitios cercanos, y necesita manejar una gran cantidad de solicitudes, especialmente de terceros o muchos terceros (incluidas las API en diferentes dominios), entonces JWT es más adecuado.

COOKIES DE SESIÓN FRENTE A TOKENS WEB JSON: VENTAJAS Y DESVENTAJAS

ENFOQUE DE COOKIES DE SESIÓN:

VENTAJAS

- Al cerrar la sesión, dado que el ID de sesión se elimina de la base de datos, tenemos un cierre de sesión de precisión. Es decir, el cierre de sesión se produce en el momento en que se elimina el ID de sesión de la base de datos de sesión.

DESVENTAJAS

- Necesita una búsqueda de BD adicional dentro de la tabla sessionId para obtener el ID de usuario, las búsquedas de BD son relativamente más lentas que la acción de descifrado de JWT.
- Los servidores individuales no se pueden escalar por separado, ya que tendrían que compartir la base de datos de sesión.

ENFOQUE JWT (TOKEN WEB JSON):

VENTAJAS

- Dado que el ID de usuario se obtiene descifrando el token JWT, no se requiere una llamada a la base de datos para obtener el ID de usuario, por lo que el enfoque de la sesión es un poco más rápido.
- Los servidores se pueden escalar por separado, sin necesidad de compartir sessionBD. Esto hace que el enfoque JWT sea una excelente opción para la arquitectura de microservicios.
- El mismo token se puede usar para autenticarse en diferentes servidores (siempre que el servidor tenga `access_token_secret`), sin necesidad de compartir sessionBD.
- Esto también permite un servidor de autenticación completamente separado, que puede ser el único responsable de emitir "tokens de acceso" y "tokens de actualización".

DESVENTAJAS

- Dado que los tokens JWT no se pueden "invalidar" (sin mantenerlos en una base de datos compartida) en el enfoque JWT, la precisión de la duración del cierre de sesión se establece por la duración del vencimiento del `access_token`. Sin embargo, la vida útil de `"access_token"` se puede mantener corta (normalmente de 10 a 15 minutos), de modo que los tokens se "invaliden" automáticamente después de la duración.
- Anti-patrón: algunas veces se almacena información adicional e innecesaria en el JWT. El token JWT debe contener principalmente información del usuario, y los datos autorizados a acceder por ese usuario deben aprovisionarse, y administrarse como un servicio separado en ese servidor respectivo.