UNITECH

Autenticación con Tokens JWT

UNITECH

Generación de Token

El siguiente ejemplo de generación de un token JWT utiliza el algoritmo HMAC (Hash-based Message Authentication Code) para garantizar la integridad y autenticidad de los datos contenidos en el token. Sin embargo, es importante destacar que JWT es flexible en términos de algoritmos de firma, lo que significa que podrían utilizarse otros algoritmos criptográficos como RSA, según los requisitos de seguridad y las preferencias del desarrollador. La elección del algoritmo de firma adecuado dependerá de factores como el nivel de seguridad requerido y las implementaciones disponibles en la plataforma o lenguaje de programación que se esté utilizando.

C#

Dependencia

```
<PackageReference Include="JWT" Version="10.1.1" />
```

Código

```
using JWT.Algorithms;
using JWT.Builder;

// usuario
var subject = "test-usuario";

// secreto
var secret = "secret*123";

// crear token HMAC
var token = JwtBuilder.Create()
                      .WithAlgorithm(new HMACSHA256Algorithm())
                      .WithSecret(secret)
                      .AddClaim("exp",
DateTimeOffset.UtcNow.AddMinutes(5).ToUnixTimeSeconds())
                      .AddClaim("sub", subject)
                      .AddClaim("iat",
DateTimeOffset.UtcNow.ToUnixTimeSeconds())
                      .Encode();

Console.WriteLine(token);
```

Java

Dependencia

```
<dependency>
<groupId>com.auth0</groupId>
<artifactId>java-jwt</artifactId>
<version>4.4.0</version>
</dependency>
```

Código

```
import java.util.Date;
import java.util.concurrent.TimeUnit;

import com.auth0.jwt.JWT;
import com.auth0.jwt.algorithms.Algorithm;

// usuario
String subject = "test-usuario";

// secreto
String secret = "secret*123";

Algorithm algorithm = Algorithm.HMAC256(secret);
String token = JWT.create()
.withSubject(subject)
.withIssuedAt(new Date())
.withExpiresAt(new Date(System.currentTimeMillis() +
TimeUnit.MINUTES.toMillis(5)))
.sign(algorithm);
```

Invocación a Aplicación

Para autenticar en la aplicación, se debe realizar una invocación mediante una solicitud HTTPS a la URL proporcionada en el servidor de la aplicación, utilizando el siguiente formato:

https://hostname/app?jwt={{token}}

En esta solicitud, hostname representa el nombre de host del servidor donde reside la aplicación, y {{token}} es el JWT (JSON Web Token) generado previamente que contiene la información de autenticación. Al incluir este token en la URL como un parámetro, la aplicación puede verificar y autenticar al usuario basándose en la información contenida en el JWT, permitiendo así el acceso seguro a los recursos y funcionalidades correspondientes.

UNITECH

Validación de Token

Cuando la aplicación recibe el token JWT y procesa la solicitud, extrae el contenido del token, donde el "subject" (sujeto) que contiene el nombre de usuario o identificador único del usuario autenticado. Este nombre de usuario se convierte en un elemento esencial para la identificación del usuario dentro de la aplicación.

Además, la aplicación verifica la firma del token JWT. Esta verificación garantiza que el token no haya sido alterado y que realmente provenga de una fuente confiable. Si la firma es válida, se puede confiar en la autenticidad del token y, por lo tanto, en la identidad del usuario.

Finalmente, la aplicación verifica que el token no esté expirado. Cada JWT contiene una marca de tiempo que indica cuándo fue emitido y cuándo expirará. La aplicación compara esta información con la hora actual para asegurarse de que el token esté dentro de su período de validez. Si el token ha caducado, se rechaza y no se permite el acceso a la aplicación.

En conjunto, estas acciones permiten a la aplicación autenticar de manera segura al usuario, obtener su identidad a partir del "subject" y asegurarse de que el token no haya sido manipulado, lo que proporciona una base sólida para la autorización y el acceso a las funcionalidades y recursos de la aplicación.

C#

Dependencia

```
<PackageReference Include="JWT" Version="10.1.1" />
```

Código

```
using JWT.Algorithms;
using JWT.Builder;

// secreto
var secret = "secret*123";

// validar token HMAC
var json = JwtBuilder.Create()
                     .WithAlgorithm(new HMACSHA256Algorithm())
                     .WithSecret(secret)
                     .MustVerifySignature()
                     .Decode(token);
```

```
Console.WriteLine(json);
```

Java

Dependencia

```
<dependency>
<groupId>com.auth0</groupId>
<artifactId>java-jwt</artifactId>
<version>4.4.0</version>
</dependency>
```

Código

```
import java.util.Date;
import java.util.concurrent.TimeUnit;

import com.auth0.jwt.JWT;
import com.auth0.jwt.JWTVerifier;
import com.auth0.jwt.DecodedJWT;
import com.auth0.jwt.algorithms.Algorithm;

// secreto
String secret = "secret*123";

Algorithm algorithm = Algorithm.HMAC256(secret);
            JWTVerifier verifier =
JWT.require(algorithm).acceptLeeway(1).acceptExpiresAt(5).build();

DecodedJWT decodedJWT = verifier.verify(token);
```