Tema 4: Mecanismos de autenticación, autorización y control de acceso

Los servicios REST/JSON gestionan eventos corporativos. Los conceptos que se estuduan son los aplicables a otro tipo de servicios como SOAP, Thrift, gRPC, etc… Un servicio REST es un servicio que implementa la lógica de negocio, pero que no tiene interfaz gráfica y es utilizado por un servicio o aplicación que si tiene interfaz gráfica. Un ejemplo de esto sería una aplicación para ver el tiempo en el móvil, la aplicación contactaría con un servicio REST por HTTP para solicitar la información climatológica y el REST devolvería la información en un formato estructurado como por ejemplo XML o JSON. Este mismo servicio rest podría ser utilizado por otras aplicaciones con una interfaz gráfica diferente.

Vamos a ver un ejemplo base. Normalmente dentro de estos servicios suelen haber roles como:

- User
- Manager

El servicio atiende a peticiones HTTP para:

- [Manager] Crear un evento
- [Manager] Manager
- [No autentciado] Buscar eventos
- [No autenticado]Obtener información de un evento
- [User]responder a un evento
- [User y Manager]Recuperar las respuestas de un usuario

Dependiendo de lo que se vaya viendo en cada apartado se analizará este ejemplo base o versiones extendidas de este.

Autenticación y Autorización

Muchas funcionalidades requieren que el usuario de la aplicación cliente se autentique. Se reciben el nombre de usuario y la contraseña. Normalmente, el punto de acceso de autenticación podría devolver un toquen que autoriza a esa aplicación cliente a hacer peticiones al servicio REST.El token debe ser seguro. Un atacante no debería tener forma de generar un token arbitrario válido.

Cuando el servicio recibe una petición debe comprobar que el token es válido y si lo es, comprobar que no esté caducado. Tras eso se revisa si el usuario que ha realizado la petición está autorizado.

Autenticación en HTTP

La cabecera debe ser ASCII estádar (7 bits, del 0 al 127) si hay simbolos extraños en la contraseña puede haber problemas, por ello normalmente se codifican las contraseñas en base 64.

JSON Web Token

define un formato compacto para transmitir información en JSON de forma segura entre 2 partes. Está dividido en 3 partes

- Cabecera: Es un JSON en formato base64URL que indica el tipo de token (formada por 2 elementos: TYP(tipo) y ALG)
- Cuerpo: Es un JSON en formato base64URL que contiene el cuerpo del mensaje (Formado por sub(subject), roles(rol que tiene) y exp(contenido del mensaje))
- Firma: En formato base64URL

```
{
  "typ": "JWT"
  "alg": "HS256"
}   |
    |     {
    |      "sub":"bob"
    |      "roles":["MANAGER"]
    |      "exp": 020399
    |     }  |
    |        |
    ˇ        ˇ
Cabecera.cuerpo.firma
    |   |   |      ^
    |   |   |      |
    |   |   |   Base64URL
    |   |   |      ^
    |   |   |      |
    |   |   |      |
    ---------->Algoritmo de firma
```

Se usa el formato base64URL para evitar el uso de caracteres extraños que puedan dar problemas en la comunicación.

Tipos de campos de cabecera

- Registrados: estandarizados, aunque no son obligatorios (sub para identificar username, id.... exp para indicar fecha)
- Públicos: Los puede definir cualquiera. Se suele recomendar usar el formato definido por la IANA.
- Privados: Los puede definir cualquiera, son campos que se utilizan en un contexto local

Firma

Se puede firmar con criptografía simétrica y asimétrica. Si se usa simétrica, la firma es un MAC: Message Authentication Code, mientras que si es asimétrica se la llama firma digital, donde hay una tercera firma como conocida de no repudio, que solo conoce la parte privada, de forma que el

firmante no puede negar el haber firmado algo. La firma se suele poder validar con la clave pública.

Ejemplo

En la autenticación se comprueba si usuario y contraseña son correcto, si lo son, se toman el rol, usuario y demás y se generan la cabecera y el mensaje. En caso de no ser correcto se devuelve error 401. Si todo es correcto se nos devuelve una respuesta.

OAuth

OpenID Connect

SAML

Kerberso y NPNEGO

Control de acceso

Impedir a otros usuarios que puedan acceder a funcionalidades como las de administrador.

Control de acceso basado en roles

Control de acceso basado en atributos

From:
https://knoppia.net/ - **Knoppia**

Permanent link:
https://knoppia.net/doku.php?id=app:tema4&rev=1731603648

Last update: **2024/11/14 17:00**