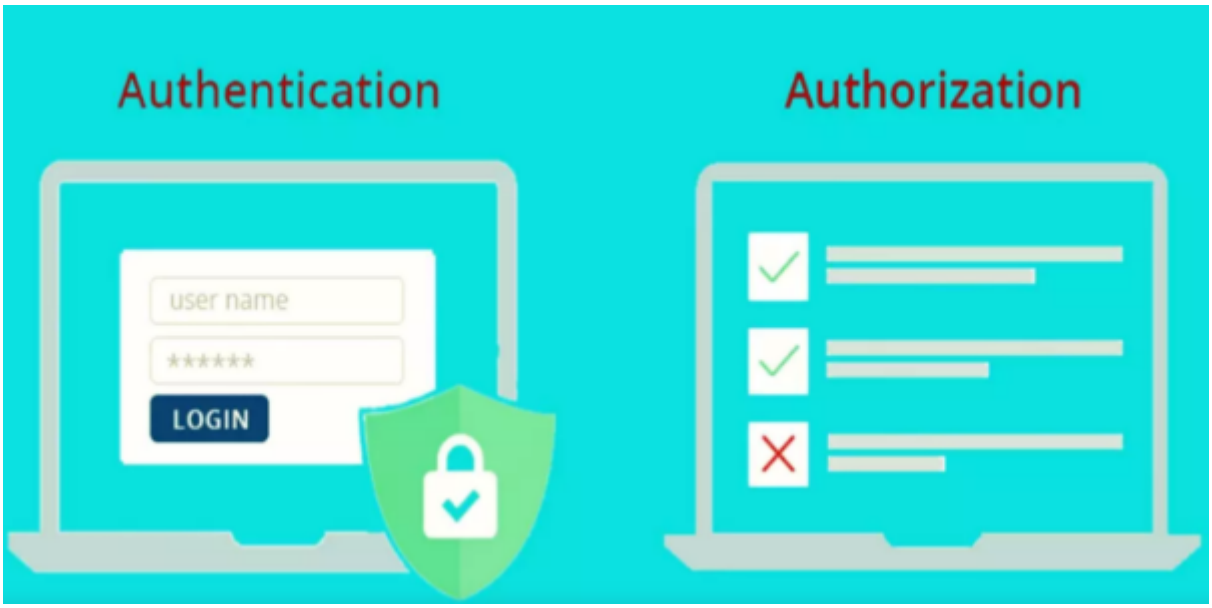
UTN FACULTAD REGIONAL CÓRDOBA
INGENIERÍA EN SISTEMAS DE INFORMACIÓN

Desarrollo de Software
Notas de Clase

Clase 14

Autenticación y. Autorización (*)



- ``La autenticación confirma que los usuarios son quienes dicen ser, validando la identidad del usuario, por diferentes métodos (algo que sabemos, algo que tenemos, algo que somos).
- La autorización otorga a los usuarios autenticados permiso para obtener acceso a determinados recursos, según su identidad.

En qué consiste la autenticación (*)

La autenticación es el proceso de identificar a los usuarios y garantizar que los mismos sean quienes dicen ser. Esto evita que cualquiera pueda entrar en un determinado sistema o iniciar sesión en alguna plataforma de forma indebida, sin que realmente sea el usuario legítimo que tiene el poder para hacerlo.

¿Cuál podría ser una prueba de autenticación? La más utilizada es usuario/contraseña. Podemos decir que es la principal barrera para evitar intrusos. Si el usuario conoce su par de credenciales (nombre de usuario y contraseña), el sistema entenderá que la identidad de dicho usuario es válida. En consecuencia, podrá acceder al recurso o conjunto de recursos deseado.

Las credenciales suelen consistir en un nombre de usuario (que puede ser una dirección de correo electrónico o un nombre único elegido por el usuario) y una contraseña. La mayoría de las aplicaciones web y sitios web requieren que los usuarios se registren y creen una cuenta antes de poder utilizarlos. En el proceso de registro, se solicita al usuario que proporcione sus credenciales, que luego se almacenan en una base de datos.

Es importante que los usuarios elijan contraseñas seguras y únicas para proteger sus cuentas de posibles ataques. Además, es responsabilidad de los propietarios de la aplicación web o sitio web garantizar que las

credenciales de los usuarios se almacenen de forma segura y se utilicen correctamente para proteger la privacidad y la seguridad de los usuarios.

Hay una tendencia visible a métodos adicionales de autenticación, para evitar únicamente el uso de las contraseñas. Uno de estos métodos de autenticación adicional son los OTP (OTP – One Time Passcodes) que son series alfanuméricas que llegan por mensaje de texto (SMS), correo electrónico o lo generamos con una aplicación autenticadora como Google Authenticator, Authy o Latch. El OTP se utiliza para la autenticación MFA, es decir, multifactor. Es un paso extra que nos da una mayor seguridad a la hora de autenticar a un usuario.

En qué consiste la autorización (*)

La autorización es lo que define a qué recursos de sistema el usuario autenticado podrá acceder. Que haya logrado pasar la instancia de la autenticación, no significa que podrá utilizar el sistema por completo como super administrador. De acuerdo a una serie de reglas, normas y regulaciones propias de cada red interna, se determina que el usuario A tendrá acceso a los recursos X e Y. Sin embargo, el usuario B sólo podrá acceder al recurso Z.

Si fuese un usuario administrador, tendría acceso tanto a los recursos X, Y y Z como a los recursos 1, 2, y 3, que son exclusivos de aquellos que tienen permisos y privilegios de administrador.

Métodos de autenticación más utilizados

Ya hemos comentado uno de los métodos más utilizados: el MFA o autenticación Multifactor. Pasemos a citar otros métodos populares:

Sin contraseña o Passwordless

Este es uno de los métodos modernos más prácticos. Un ejemplo de aplicación es el uso de un enlace mágico (magic link). Consiste en que, cada vez que quieras iniciar sesión a un recurso o servicio, se enviará a tu correo electrónico un enlace que te permitirá acceder sin necesidad de contraseña. Este es un método recomendable, ya que se necesita del acceso al correo electrónico y, por ende, hay más garantías de asegurar que es el propio usuario quien está accediendo.

Por redes sociales

Sin duda, ya habrás utilizado este método. Varias aplicaciones y servicios te dan como opción iniciar sesión directamente con alguna cuenta social. La ventaja principal es que no hace falta crear una cuenta aparte de forma manual, directamente los datos de esa cuenta social hacen ese paso al iniciar la sesión. Las plataformas sociales más utilizadas son Facebook, Twitter y la cuenta Google. De esta forma podremos iniciar sesión en programas o páginas de forma más rápida, sin tener que registrarnos.

Autenticación API

Este es el proceso de certificar la identidad de un usuario que quiera acceder a recursos y/o servicios en el servidor. Para tener en cuenta, alguna de las APIs de autenticación más populares son: autenticación básica por HTTP, de Core (núcleo) API y OAuth.

Autenticación Biométrica

Se vale de las huellas dactilares para validar la identidad del usuario. El caso de uso más popular es en los lugares de trabajo, en donde tanto para registrar la entrada como salida, se posa el dedo para validar la huella dactilar. Esa huella es validada mediante un previo registro de la misma que se almacena en la base de datos. Es cada vez más utilizado este método también en dispositivos móviles, para iniciar sesión, realizar un pago, etc.

Métodos de autorización más utilizados

Lo que comentaremos a continuación, es información especialmente útil para todo desarrollador aplicaciones y servicios web en general. Recordemos que parte fundamental de una gran experiencia de usuario es que los mismos puedan acceder con confianza y seguridad a todos los recursos necesarios en todo momento que precisen disponer de los mismos.

Autorización HTTP

Así como la autenticación, existe la autorización de tipo HTTP. ¿En qué consiste? La persona ingresa su nombre de usuario y contraseña para poder autenticarse. Es bueno tener presente que este método no implica a las cookies, IDs de sesiones o páginas de inicio de sesión. Este puede ser usado por servidores para revisar solicitudes, y por parte de un cliente para proveer información de autentificación. Para realizar esto, el servidor responde al usuario con un «Unauthorized», el cual también le proporciona al usuario toda la información para que conozca el método de autorización. Luego, el cliente que se quiera autorizar puede hacerlo con una solicitud «Authorization», y sus credenciales. Lo más normal es que el cliente haga una solicitud mediante un usuario y contraseña, en donde se incluye un encabezado «Authorization» directo al servidor.

Autorización API

Así como la autenticación, existe la autorización de tipo API. Cuando el usuario intenta ganar acceso a recursos de un sistema durante su registro, se genera una clave API. Esa misma clave se empareja con un token (una ficha identificadora) que se encuentra oculto. Entonces, esa combinación de clave API y token oculto es la que se utiliza constantemente cada vez que el usuario se autentica e ingresa a su entorno de recursos y servicios que puede utilizar. Estas se implementan para definir lo que el cliente debe hacer para enviar una solicitud al servidor, y para definir la respuesta que este recibe.

OAuth 2.0

Este método permite que la API se autentique y acceda a los recursos del sistema que necesita. La versión 2.0 de OAuth es uno de los métodos más seguros tanto de autenticación como autorización. Funciona de forma que el usuario delega la función de realizar algunas acciones, a las cuales da su consentimiento para realizarlas a su nombre. Un ejemplo, pueden ser las aplicaciones que genera un post en Twitter de forma automática.

Autorización JWT (*)

JWT es el acrónimo de "JSON Web Token", que es un estándar de token de seguridad abierto que se utiliza para la autenticación y autorización en aplicaciones web y móviles.

La autorización JWT se refiere al proceso de utilizar un JWT para verificar la identidad de un usuario y conceder acceso a recursos o funciones específicas dentro de una aplicación. Un JWT es un objeto JSON codificado que contiene información sobre el usuario y otros datos relevantes, y está firmado digitalmente para garantizar su autenticidad e integridad.

Cuando un usuario inicia sesión en una aplicación, el servidor genera un JWT y lo devuelve al cliente, quien lo incluye en todas las solicitudes posteriores. El servidor entonces verifica la firma del JWT y extrae la información relevante para determinar si el usuario está autorizado para realizar la acción solicitada.

La autorización JWT es una forma segura y eficiente de autenticar y autorizar a los usuarios en aplicaciones web y móviles. Al utilizar un JWT, los desarrolladores pueden evitar la necesidad de almacenar información de autenticación en el servidor, lo que reduce la carga en el servidor y mejora la escalabilidad de la aplicación.

JWT y una implementación sencilla en NodeJs - Express

En Node.js, se pueden utilizar varios paquetes para implementar la autorización JWT. Uno de los paquetes más populares es "jsonwebtoken".

Para utilizar "jsonwebtoken", primero debe instalarlo utilizando el siguiente comando en la terminal:

```
npm install jsonwebtoken
```

Luego, puede utilizar el paquete para generar tokens de autenticación JWT utilizando una clave secreta. Por ejemplo, para generar un JWT que expire en una hora, puede usar el siguiente código:

const jwt = require('jsonwebtoken');

```
const token = jwt.sign({
  username: 'admin',
  role: 'administrador'
}, 'claveSecreta', { expiresIn: '1h' });
```

En este ejemplo, el método sign() toma un objeto de datos que se utilizará para generar el token, la clave secreta que se utilizará para firmar el token y un objeto de opciones que incluye la duración de la validez del token. Esta funcionalidad en su forma mas simple la ejecutaremos dentro del proceso de Login, luego de validar las credenciales del usuario, que normalmente consisten en nombre de usuario y una contraseña.

Una vez que se ha generado el token, puede incluirlo en las solicitudes HTTP que realice su aplicación. El servidor de la aplicación puede verificar la autenticidad del token utilizando el siguiente código:

```
const token = req.headers.authorization.split(' ')[1];

jwt.verify(token, 'claveSecreta', function(err, decoded) {
  if (err) {
    // El token no es válido
  } else {
```

```
    // El token es válido, se pueden acceder los datos del usuario mediante
decoded
    console.log(decoded.username);
  }
});
```

En este ejemplo, el método verify() se utiliza para verificar la validez del token. Si el token es válido, los datos del usuario se pueden acceder a través del objeto "decoded". Si el token no es válido, se devuelve un error. Esta funcionalidad en su forma mas simple la utilizaremos simpre antes de conceder acceso a algun recurso de acceso restrigido.

Ejercicio Seguridad 1: Implementar en un aplicacion backend node-express, la funcionalidad de autenticacion:Login/Logout y de autorizacion mediante un middleware que permita proteger ciertos recursos segun el rol otorgado a cada Usuario. Este ejercicio esta planteado como un paso a paso y constituye la etapa 5 del "Paso a paso Backend"

Ejercicio Seguridad 2: Sobre la funcionalidad desarrollada en el ejercio anterior, compruebe mediante postman el correcto funcionamiento de la ruta protegida /api/jwt/articulos. Para lo cual debera seguir los siguientes pasos:

1. Loguearse mediante la ruta /api/login con el usuario: "admin", clave: 123, para obtener el token que usara en el paso siguiente.
2. Solicitar la ruta segunra /api/jwt/articulos, pasando en el header: **authorization: bearer** (donde es el token obtenido en el paso anterior)
3. Repetir el paso 1 y 2 para el usuario: "juan", clave: 123 **Nota:** puede ver una implementacion en html + js vanilla que invoca al login y a la ruta segura en: https://stackblitz.com/edit/dds-backend?file=public%2Flogin.html