Artículo
Heloisa Paiva · 24 ene, 2023 Lectura de 7 min

Autenticación por token - Conceptos básicos que necesitas para empezar a desarrollar

Por qué he decidido escribir esto

Recientemente tuve el reto de crear un método de autenticación seguro para autorizar el acceso a algunos datos, pero desafortunadamente no tenía ninguna experiencia con esas configuraciones de seguridad y sentí que me faltaban algunos conceptos básicos para entender mejor la documentación oficial.

Después de estudiar y lograr entregar las clases que me pidieron hacer, me gustaría compartir un poco de mi nuevo conocimiento, que me ayudó a seguir los temas de la documentación.

Empezando por lo básico: la santísima trinidad de los servidores

Primero, es importante entender de qué estamos hablando exactamente. En general, tenemos datos que pueden ser sensibles, o por cualquier razón necesitan protección. Hay personas (usuarios) que podrán verlos, otras podrán cambiarlos y otras no tendrán ningún tipo de acceso. Para ocuparse de los usuarios, el acceso y los datos, tendremos tres servidores: el del cliente, el de recursos y el de autorización.

Así que básicamente este es el flujo:

El servidor del cliente envía una solicitud al servidor de autorización, que hará la validación – comprueba si puede autorizarla o no; hablaremos de esto más adelante. Una vez que el servidor decide que esta ok permitir el acceso, envía un token de vuelta al cliente. El cliente ahora puede enviar ese token al servidor de recursos, que lo reconocerá y devolverá al cliente la información que necesita.

Para estar seguros de que todo está claro antes de pasar al siguiente paso, vamos a ponernos en el lugar de los servidores:

- **Soy el servidor de autorización:** quiero recibir información de un cliente y si lo reconozco, le daré la llave (el token) para acceder a los datos. Soy el recepcionista.
- **Soy el servidor de recursos:** quiero recibir un token. Si es válido, quien lo envió podrá acceder a los datos que yo guardo. Soy el banco.
- **Soy el servidor del cliente:** quiero acceder a los datos desde el servidor de recursos, pero no sé la llave. Así que me voy a identificar al servidor de autorización, para que me de la llave. Entonces, la mostraré al servidor de recursos, que me llevará a los datos que necesito. Soy el cliente.

Un paso más: el token y la validación

Pero... ¿qué es esa validación y ese token? Bueno, es simple. El token es una cadena de caracteres, que el servidor de recursos interpretará como el tipo de acceso que el cliente puede tener. Así que, si tengo acceso de visualización, mi token va a decir al servidor de recursos " ¡Ella puede hacer un SELECT! " - y otras cosas como quién soy, el tiempo que puedo usar el token, etc. - o si tengo cualquier tipo de acceso, el token va a chivar todo sobre mí al servidor de recursos y le dirá " ¡Ella puede hacer lo que quiera en esta tabla! "

En la práctica, el servidor de autorización va a usar algún tipo de cifrado para cambiar " Hola, soy Heloisa, mi contraseña para este servidor es JuroQueSoyHonesta.123 " por " iuiDBIldb3287$@*++==" o algo similar, que significa " Servidor de recursos, dale acceso SELECT (o el que corresponda) a quien te envíe esto". El token podría ser una cadena de caracteres que diga literalmente " ACCESO SELECT " , pero eso no tendría garantía de seguridad.

Pero no siempre es todo tan fácil, así que vamos a complicarlo un poco. Está claro que el servidor de autorización no va a dar el token a cualquiera que pregunte, porque hay que ser muy muy especial para acceder a estos datos, ¿cierto? Para resolver eso, el servidor tiene una sencilla tabla con los nombres de usuarios, sus correspondientes contraseñas y los accesos permitidos. El cliente debe identificarse con un nombre de usuario y una contraseña, para que se produzca la validación. Esa validación es literalmente verificar la tabla de usuarios y contraseñas. La parte que vamos a complicar un poco es cómo el cliente envía esa información.

Hay algunas maneras:

- El cliente envía una solicitud " Hola Autorización, quiero identificarme " , y el servidor de autorización le pide el nombre de usuario y la contraseña. Después de la validación, devuelve un código de autorización, que el cliente envía al servidor de autorización para tener el token necesario para el servidor de recursos.
- El mismo proceso anterior, pero en vez de recibir primero un código de autorización, el cliente recibe directamente el token.
- El cliente envía el nombre de usuario y la contraseña en una solicitud al servidor de autorización, que hace la validación y envía el token.
- No hay usuario, el cliente envía una identificación de cliente (Client ID y Client Secret) para obtener el token.

Cada método tiene sus pros y sus contras, dependiendo del nivel de seguridad que los datos necesitan. Hablaremos sobre ello más tarde.

Una pequeña comparación para aclararlo todo

Eso me dio la imagen de Hagrid y Harry Potter entrando al banco Gringotts para conseguir la Piedra Filosofal. Excepto que Dumbledore es el servidor cliente, que envía a Harry, el usuario, y Hagrid, la solicitud del cliente, diciendo " Hola, necesitamos unos datos de la caja fuerte " . La caja fuerte es el servidor de recursos y los duendes son el servidor de autorización, validando si Hagrid puede obtener los datos para Dumbledore o no. Y claro, la llave para la caja fuerte es el token. Si no habéis visto Harry Potter, debéis de estar mucho más confundidos que antes, pero ahora podéis poner la peli pensando " Es para aprender, no estoy procrastinando... " . Esta es una buena comparación, porque Harry solo sabe lo que hay en la caja fuerte una vez que la abre, que es el propósito de la autenticación. Además, Harry no hace nada, excepto decir " Hola, soy Harry Potter y no tengo ni idea de lo que pasa, pero de alguna manera tuve acceso a una riquísima caja fuerte que da la casualidad de que es mía! " . Date cuenta de que para la caja fuerte de Dumbledore, solo tenía acceso SELECT, porque podía ver lo que había adentro, pero no podía hacer nada. Para su propia caja fuerte, tenía acceso ALL, porque podía sacar o depositar cosas, permitir acceso a personas de confianza, etc...

Para el tercero método, hubiera ido con Hagrid pero se habría ido de compras a Sortilegios Weasley, mientras Hagrid sacaba el paquete. Y para el último método, Dumbledore hubiera enviado a Hagrid solo al banco.

Pero basta de Harry Potter, vamos a volver al trabajo.

Manos a la obra: la práctica

Ahora que tenemos los conceptos claros, necesitamos hacerlos funcionar. Para empezar, pregúntate " ¿que servidor estoy desarrollando? "

Se estás desarrollando más de uno, puede ser útil hacer uno cada vez. Responder a esa pregunta te llevará a centrarte en lo que esta clase necesita hacer exactamente:

- **Servidor de recursos:** recibe una solicitud con un token y verifica si es válido. Si lo es, realiza la acción requerida por la solicitud (por ejemplo, SELECT * FROM bd.tabla) y devuelve la información obtenida.
- **Servidor de autorización:** depende del tipo de validación elegido. Para el primero, si recibe una solicitud vacía, pide al usuario un nombre de usuario y contraseña. Después, realiza la validación y si está ok, devuelve un código de autorización. Si recibe una solicitud con un código de autorización válido, devuelve el token. Para el segundo, recibe una solicitud vacía y pide al usuario un nombre de usuario y contraseña, realiza la validación y si está todo ok, devuelve el token. Para el tercero, recibe una solicitud con un nombre de usuario y contraseña, realiza la validación y si es ok, devuelve el token. Para el último, recibe una solicitud con un Client ID y un Client Secret, realiza la validación y si es ok, devuelve el token.
- **Servidor del cliente:** depende del tipo de validación que use el servidor de autorización. Para el primero, envía una solicitud al servidor de autorización y si el usuario tiene acceso, recibe un código en la respuesta, envía ese código en otra solicitud al servidor de autorización y recibe el token. Finalmente, envía el token al servidor de recursos y recibe datos. Para el segundo, envía una solicitud al servidor de autorización y si el usuario tiene acceso, recibe un token en la respuesta, envía el token al servidor de recursos y recibe lo los datos. Para el tercero, recibe un nombre de usuario y contraseña (puede ser desde otro servidor, pidiéndoselo al usuario, o puede ser un nombre de usuario y contraseña fijos, etc.), los envía en una solicitud al servidor de autorización y si el usuario tiene acceso, recibe el token y blablabla al servidor de recursos y vuelve con los datos otra vez. Para el último, tiene un Client ID y Client Secret que serán enviados al servidor de autorización, que devuelve un token y – blablabla una vez más.

Sobre los métodos de validación

La mayor parte de la seguridad se debe al método de validación. Si recibes órdenes específicas para desarrollar en un tipo de validación o en otro, no necesitas preocuparte de esto, pero es interesante saber por qué tu jefe hizo esa elección.

Si eres el que debe elegir, o solamente quieres aprender, aquí tienes más información sobre autenticación por token:

De StackOverflow: Proper way to send username and password from client to server

De StackExchange: Why shouldn't my client just send the user's username and password with every request?

No las voy a comentar porque está fuera del ámbito de este artículo y además creo que merece la pena leer los enlaces al completo.

Fin

Muchas gracias por leerme y espero que el artículo os resulte útil.

Podéis preguntarme si tenéis dudas o contactarme para que os ayude en algún caso especifico. ¡Estaré encantada de ayudar!

#Autenticación #Control de acceso #Innovatium #InterSystems IRIS

URL de fuente:https://es.community.intersystems.com/post/autenticaci%C3%B3n-por-token-conceptos-b%C3%A1sicos-que-necesitas-para-empezar-desarrollar