Texto Introductorio OAuth 2.0 scopes

OAuth 2.0 scopes

Keycloak es, fundamentalmente, un servidor de autorización basado en OAuth 2.0. A OAuth 2.0 ya lo conocemos y lo revisamos a fondo en clases pasadas, pero repasemos un aspecto importante. En OAuth hay dos tipos de aplicaciones: **clientes** y **servidores de recursos**. Vimos también que los **tokens** se generan para que los clientes puedan actuar en nombre de un usuario y están limitados al alcance (**scope**) al que el usuario otorgue su consentimiento. Por otro lado, los servidores de recursos son los que, basándose en ese token, deciden si un cliente puede acceder a un recurso protegido.

¿Pero qué es el scope? Es el alcance de la autorización y se usa para restringir el acceso a los recursos. Cuando se solicita un access token del servidor de autorizaciones, la aplicación cliente incluirá un scope como parámetro de la petición, especificando una lista de scopes (alcances) o determinados accesos a los recursos del usuario, que el token generado debe tener.

Por defecto, los clientes en Keycloak no están configurados para pedir consentimiento del usuario. Esto sucede porque usualmente se emplea dentro de un ámbito empresarial, donde los clientes se encuentran dentro de los límites de la organización y los recursos a los cuales se puede acceder no requieren del consentimiento del usuario. En su lugar, requieren de un permiso de acceso que se define de acuerdo a roles, atributos específicos de un usuario o incluso grupos de usuarios (como veremos más adelante).

La **autorización mediante los scopes de OAuth 2.0** se basa solamente en el consentimiento del usuario. Es una mejor estrategia a la hora de integrar servicios de terceros en nuestra aplicación y, en este caso, el usuario tendrá la decisión a la hora de autorizar a una aplicación de terceros a acceder a sus recursos. Para implementar esta estrategia se deberá poner énfasis en proteger la información de los usuarios. Debemos entender que esta estrategia de autorización se enfoca en proteger a los recursos de otros clientes (aplicaciones o servicios). Esto difiere de las estrategias que vimos anteriormente, como RBAC, en la cual se protege a los recursos de los usuarios.