10

Implementing security at backend service

Java Backend Developer I

Objetivos

Comprender los conceptos:

- Architecture Components
  - AuthenticationProvider
- Authentication Mechanisms
  - OAuth 2.0 Login
  - OAuth 2.0 Client
  - OAuth 2.0 Resource Server

Agenda

Revisión de los siguientes conceptos:

- Architecture Components
  - AuthenticationProvider
- Authentication Mechanisms
  - OAuth 2.0 Login
  - OAuth 2.0 Client
  - OAuth 2.0 Resource Server

Architecture Components

AuthenticationProvider

Si hablamos de seguridad dentro de la capa de negocio, es muy habitual, vincular el concepto con **Spring Security**.

Este framework facilita las tareas a adoptar como medidas de seguridad en aplicaciones Java, ya sean aplicaciones standalone o aplicaciones web. La arquitectura de **Spring Security** está fuertemente basada en interfaces y en patrones de diseño, proporcionando las implementaciones más comúnmente utilizadas y numerosos puntos de extensión donde se pueden añadir nuevas funcionalidades.

Architecture Components

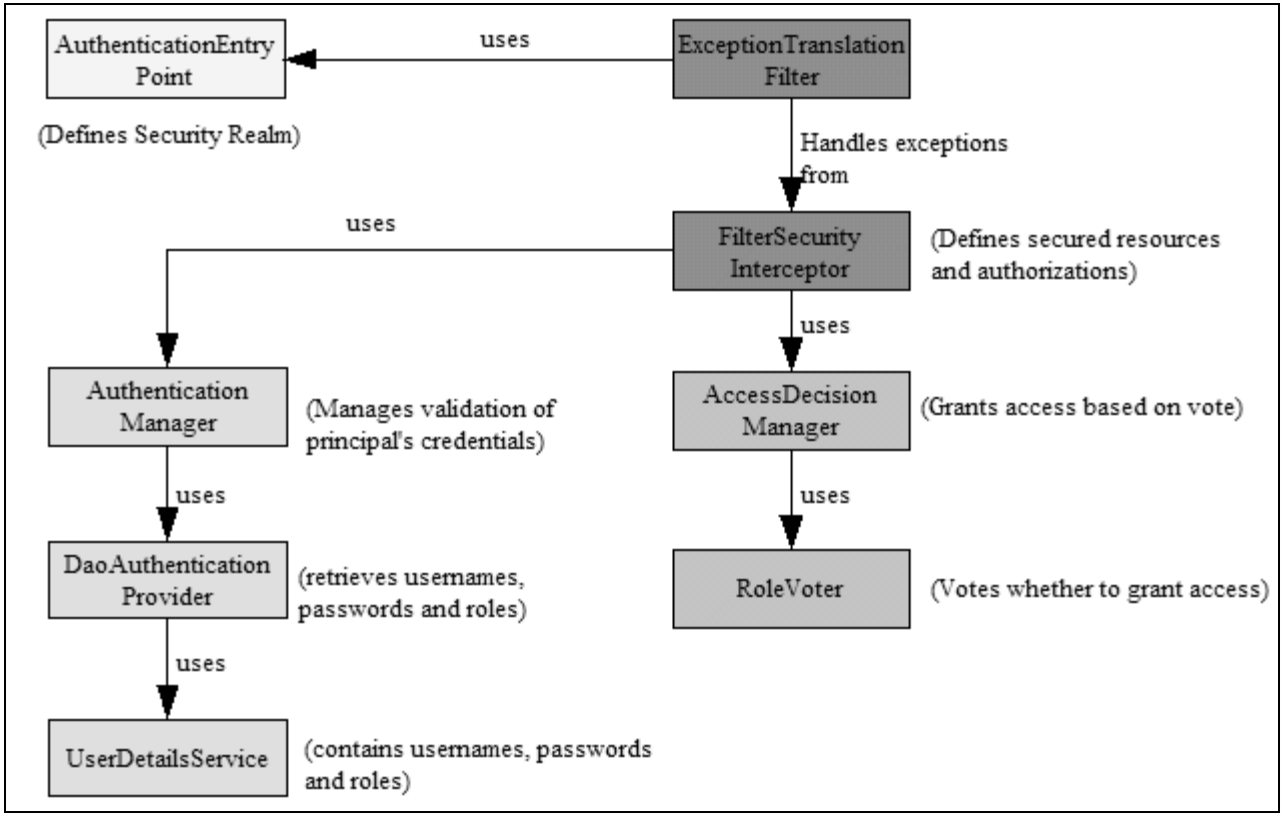
AuthenticationProvider

Para utilizar los servicios de autenticación, por lo general será necesario que configurar un filtro web, junto con un **AuthenticationProvider** y **AuthenticationEntryPoint**. En el **web.xml** de la aplicación se necesita un sólo filtro a fin de utilizar el **FilterChainProxy.** Casi todas las aplicaciones tendrán una entrada.

Architecture Components

AuthenticationProvider

Esquema:



Authentication Mechanisms

OAuth 2.0

Concepto

OAuth 2 es una estructura (framework) de autorización que le permite a las aplicaciones obtener acceso limitado a cuentas de usuario en un servicio HTTP, como Facebook, GitHub..

Delega la autenticación del usuario al servicio que aloja la cuenta del mismo y autoriza a las aplicaciones de terceros el acceso a dicha cuenta de usuario.

Authentication Mechanisms

OAuth 2.0

OAuth 2 proporciona flujos de autorización para aplicaciones web y de escritorio; y dispositivos móviles.

A continuación se describe los roles del **OAuth** los cuales son cuatro:

- Propietario del recurso
- Cliente (Client)
- Servidor de recursos (Resources Server)
- Servidor de autorización

Authentication Mechanisms

OAuth 2.0

Propietario del recurso: *Usuario*

El propietario del recurso es el “usuario” que da la autorización a una aplicación, para acceder a su cuenta. El acceso de la aplicación a la cuenta del usuario se limita al “alcance” de la autorización otorgada (e.g. acceso de lectura o escritura).

Authentication Mechanisms

OAuth 2.0

Servidor de Recursos / Autorización: *API*

El servidor de recursos aloja las cuentas de usuario protegidas, y el servidor de autorizaciones verifica la identidad del usuario y luego genera **tokens** de acceso a la aplicación.

Desde el punto de vista del desarrollador de una aplicación, la API del servicio atiende tanto a los roles de recursos como a los de autorización. Nos referiremos a ambos roles combinados, como al rol de servicio o de API.

Authentication Mechanisms

OAuth 2.0

Cliente: *Aplicación*

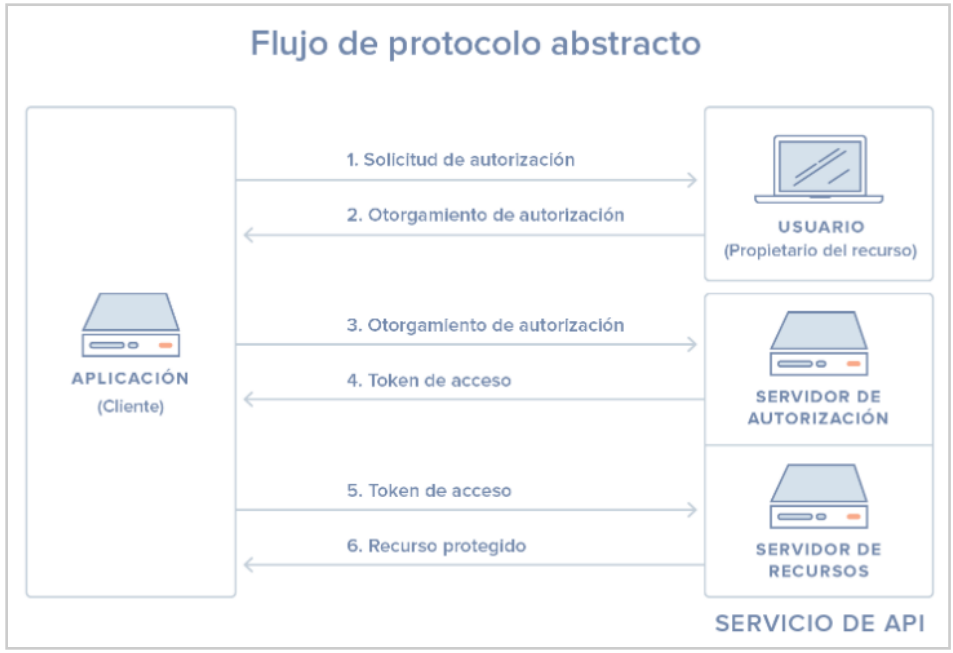
El cliente es la *aplicación* que desea acceder a la cuenta del *usuario*. Antes de que pueda hacerlo, debe ser autorizado por el usuario, y dicha autorización debe ser validada por la API.

Authentication Mechanisms

OAuth 2.0

Flujo de protocolo abstracto

Diagrama de interacción entre los roles:



Lecturas adicionales

Para obtener información adicional, puede consultar los siguientes enlaces:

- https://oauth.net/2/
- https://aaronparecki.com/oauth-2-simplified/

Resumen

En este capítulo, usted aprendió:

- Architecture Components
  - AuthenticationProvider
- Authentication Mechanisms
  - OAuth 2.0 Login
  - OAuth 2.0 Client
  - OAuth 2.0 Resource Server