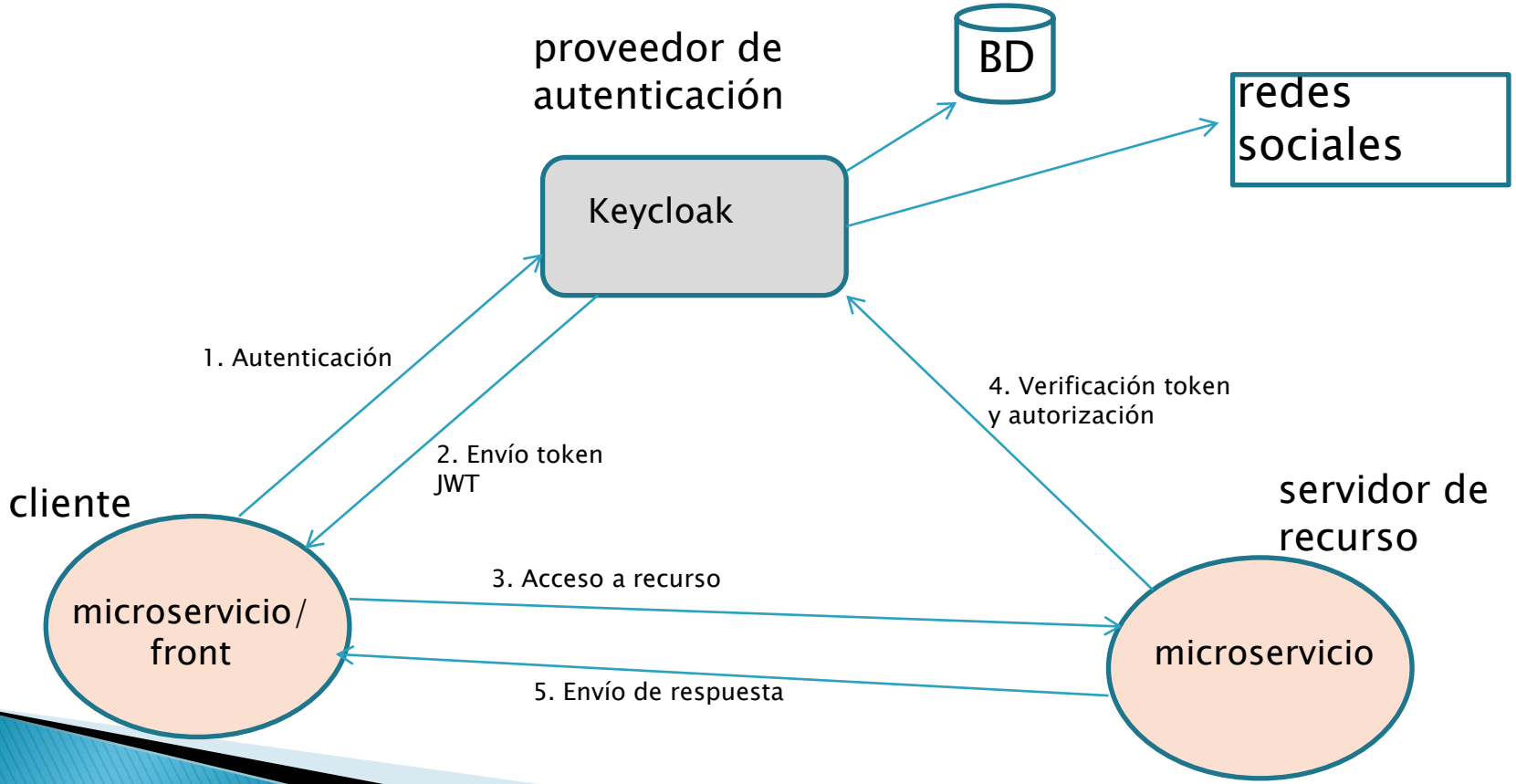
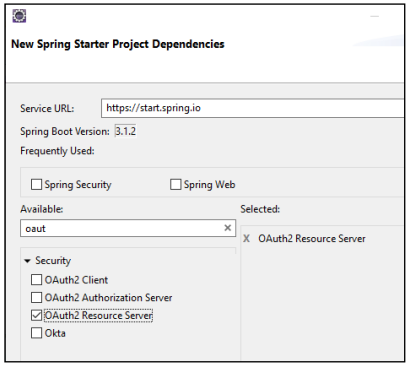
Integración Keycloak con microservicios

Esquema OAuth2



Keycloak en servidor de recurso

- El servidor de recurso necesita comunicar con Keycloak para la verificación del token JWT.
- Además de Spring Security requiere el starter oauth2 resource server:




```
<dependency>
   <groupId>org.springframework.boot</groupId>
   <artifactId>spring-boot-starter-oauth2-resource-server</artifactId>
</dependency>
```

Propiedades configuración

- En el archivo .properties se deben definir una serie de propiedades para que el servidor de recurso pueda validar el token, como los datos de conexión al reino de Keycloak y la clave de la firma:

```
spring.security.oauth2.resourceserver.jwt.issuer-uri=
 http://localhost:8080/realms/ContactosRealms
spring.security.oauth2.resourceserver.jwt.jwk-set-uri=
 ${spring.security.oauth2.resourceserver.jwt.issuer-uri}/protocol/openid-connect/certs
```

- Propiedades personalizadas adicionales:

```
keycloak.clientId=login
```

Conversor

- Componente utilizado para extraer la información del token:

Tipo entrada

Tipo salida

Inyección propiedades personalizadas

```
@Component
public class JwtAuthConverter implements Converter<Jwt, AbstractAuthenticationToken> {
    @Value("${keycloak.clientId}")
    private String clientId;

    @Override
    public AbstractAuthenticationToken convert(Jwt jwt) {
     …
    }
}
```

Genera un objeto con los datos del usuario extraídos del token

Configuración Spring security

- Mantiene la configuración de autorización.
- En la configuración para OAuth2, se indica el conversor utilizado:

Inyección componente conversor →

```
@EnableWebSecurity
@Configuration
public class SecurityConfig {
    @Autowired
    private  JwtAuthConverter jwtAuthConverter;
    @Bean
    public SecurityFilterChain filterChain(HttpSecurity http) throws Exception{
        http.csrf(cus->cus.disable())
        .authorizeHttpRequests(aut->
            aut.requestMatchers(HttpMethod.POST,"/contactos").hasRole("ADMIN")
            .requestMatchers(HttpMethod.DELETE,"/contactos/**").hasAnyRole("ADMIN","OPERATOR")
            .requestMatchers("/contactos").authenticated()
            .anyRequest().permitAll())
        .oauth2ResourceServer(oauth2ResourceServer->
            oauth2ResourceServer.jwt(jwt->jwt
                .jwtAuthenticationConverter(jwtAuthConverter)))
                .sessionManagement(sessionManagement->
                    sessionManagement.sessionCreationPolicy(SessionCreationPolicy.STATELESS));
        return http.build();
    }
}
```

Keycloak en el cliente

- El cliente no requiere ninguna dependencia ni configuración especial.
- Debe encargar de lanzar la petición post a Keycloak con los datos para realizar el proceso de autenticación y obtención del token
- Una vez recibido el token, lo incluirá en las cabeceras de las peticiones a microservicio