How can I read all users using keycloak and spring?

Asked 6 years, 5 months ago Modified 1 year, 7 months ago Viewed 44k times

15

I'm using `keycloak 3.4` and `spring boot` to develop a web app. I'm using the Active Directory as User Federation to retrieve all users information.

But to use those information inside my web app I think I have to save them inside the "local-webapp" database.

So after the users are logged, how can I save them inside my database?

I'm thinking about a **scenario** like: "I have an object A which it refers to the user B, so I have to put a relation between them. So I add a foreign key."

In that case I need to have the user on my DB. no?

EDIT

To avoid to get save all users on my DB I'm trying to use the Administrator API, so I added the following code inside a controller.

I also created another client called `Test` to get all users, in this way I can use `client-id` and `client-secret.` Or is there a way to use the `JWT` to use the admin API?

The client:

```
     Keycloak keycloak2 = KeycloakBuilder.builder()

.serverUrl("http://localhost:8080/auth/admin/realms/MYREALM/users")
                          .realm("MYREALMM")
                          .username("u.user")
                          .password("password")
                          .clientId("Test")
                          .clientSecret("cade3034-6ee1-4b18-8627-2df9a315cf3d")
                          .resteasyClient(new
ResteasyClientBuilder().connectionPoolSize(20).build())
                          .build();

 RealmRepresentation realm2 = keycloak2.realm("MYREALMM").toRepresentation();
```

the error is:

```
2018-02-05 12:33:06.638 ERROR 16975 --- [nio-8080-exec-7] o.a.c.c.C.[.[.[/].
[dispatcherServlet]    : Servlet.service() for servlet [dispatcherServlet] in
context with path [] threw exception [Handler dispatch failed; nested exception
is java.lang.Error: Unresolved compilation problem:
    The method realm(String) is undefined for the type AccessTokenResponse
] with root cause

java.lang.Error: Unresolved compilation problem:
    The method realm(String) is undefined for the type AccessTokenResponse
```

Where am I doing wrong?

EDIT 2

I also tried this:

```
@Autowired
private HttpServletRequest request;

public ResponseEntity listUsers() {
    KeycloakAuthenticationToken token = (KeycloakAuthenticationToken)
request.getUserPrincipal();
    KeycloakPrincipal principal=(KeycloakPrincipal)token.getPrincipal();
    KeycloakSecurityContext session = principal.getKeycloakSecurityContext();

    Keycloak keycloak = KeycloakBuilder.builder()
                                       .serverUrl("http://localhost:8080/auth")
                                       .realm("MYREALMM")

.authorization(session.getToken().getAuthorization().toString())
                                       .resteasyClient(new
ResteasyClientBuilder().connectionPoolSize(20).build())
                                       .build();

    RealmResource r = keycloak.realm("MYREALMM");
    List<org.keycloak.representations.idm.UserRepresentation> list =
keycloak.realm("MYREALMM").users().list();
    return ResponseEntity.ok(list);
```

but the authorization is always `null`. Why?

EDIT 3 Following you can find my spring security config:

```
    @Configuration
@EnableWebSecurity
@EnableGlobalMethodSecurity(prePostEnabled=true)
@ComponentScan(basePackageClasses = KeycloakSecurityComponents.class)
@KeycloakConfiguration
public class SecurityConfig extends KeycloakWebSecurityConfigurerAdapter {

    @Override
    protected void configure(HttpSecurity http) throws Exception {
         super.configure(http);

        http.httpBasic().disable();
        http
        .csrf().disable()
        .authorizeRequests()
            .antMatchers("/webjars/**").permitAll()
            .antMatchers("/resources/**").permitAll()
            .anyRequest().authenticated()
        .and()
        .logout()
            .logoutUrl("/logout")
            .logoutRequestMatcher(new AntPathRequestMatcher("/logout", "GET"))
            .permitAll()
            .logoutSuccessUrl("/")
            .invalidateHttpSession(true);
    }

      @Autowired
```

```
        public KeycloakClientRequestFactory keycloakClientRequestFactory;

        @Bean
        public KeycloakRestTemplate keycloakRestTemplate() {
            return new KeycloakRestTemplate(keycloakClientRequestFactory);
        }

    @Autowired
    public void configureGlobal(AuthenticationManagerBuilder auth) {

        KeycloakAuthenticationProvider keycloakAuthenticationProvider =
keycloakAuthenticationProvider();
        SimpleAuthorityMapper simpleAuthorityMapper = new
SimpleAuthorityMapper();
        simpleAuthorityMapper.setPrefix("ROLE_");
        simpleAuthorityMapper.setConvertToUpperCase(true);

keycloakAuthenticationProvider.setGrantedAuthoritiesMapper(simpleAuthorityMapper);
        auth.authenticationProvider(keycloakAuthenticationProvider);
    }

    @Bean
    public KeycloakSpringBootConfigResolver keycloakConfigResolver() {
        return new KeycloakSpringBootConfigResolver();
    }

    @Bean
    @Override
    protected SessionAuthenticationStrategy sessionAuthenticationStrategy() {
        return new RegisterSessionAuthenticationStrategy(new
SessionRegistryImpl());
    }

    @Override
    public void configure(WebSecurity web) throws Exception {
        web
           .ignoring()
           .antMatchers("/resources/**", "/static/**", "/css/**", "/js/**",
"/images/**", "/webjars/**");
    }

     @Bean
     @Scope(scopeName = WebApplicationContext.SCOPE_REQUEST, proxyMode =
ScopedProxyMode.TARGET_CLASS)
     public AccessToken accessToken() {
         HttpServletRequest request = ((ServletRequestAttributes)
RequestContextHolder.currentRequestAttributes()).getRequest();
         return ((KeycloakSecurityContext) ((KeycloakAuthenticationToken)
request.getUserPrincipal()).getCredentials()).getToken();
     }

}
```

EDIT 4

These are the properties inside the `applicatoin.properties`

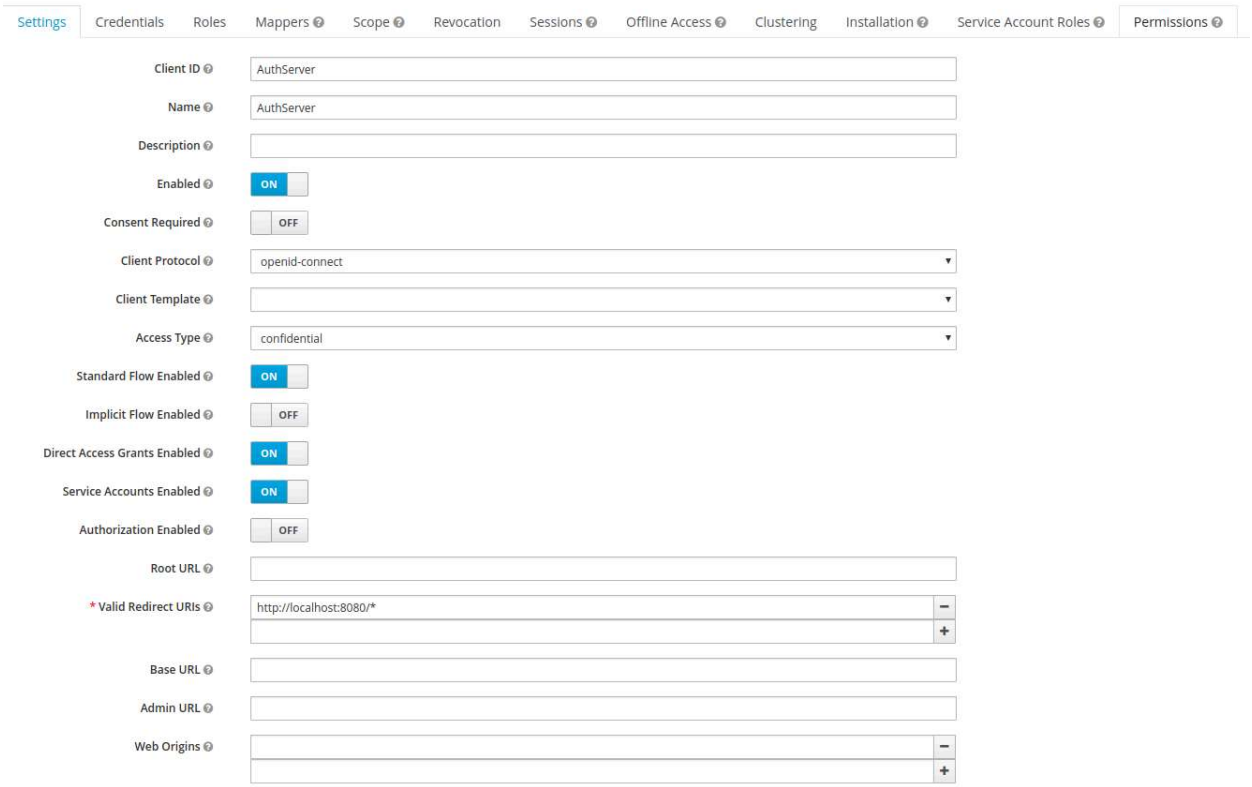
```
#########################################
#             KEYCLOAK                  #
#########################################

keycloak.auth-server-url=http://localhost:8181/auth
keycloak.realm=My Realm
```

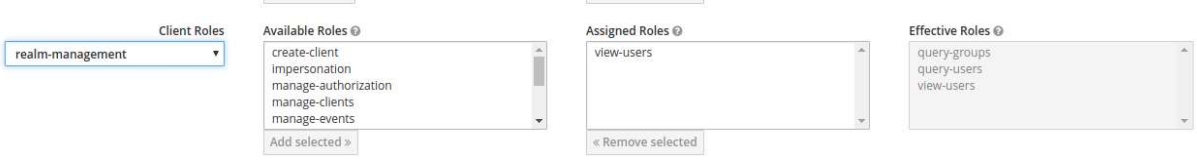
```
keycloak.ssl-required=external
keycloak.resource=AuthServer
keycloak.credentials.jwt.client-key-password=keystorePwd
keycloak.credentials.jwt.client-keystore-file=keystore.jks
keycloak.credentials.jwt.client-keystore-password=keystorePwd
keycloak.credentials.jwt.alias=AuthServer
keycloak.credentials.jwt.token-expiration=10
keycloak.credentials.jwt.client-keystore-type=JKS
keycloak.use-resource-role-mappings=true
keycloak.confidential-port=0
keycloak.principal-attribute=preferred_username
```

EDIT 5.

This is my keycloak config:



the user that I'm using to login with view user permission:



EDIT 6

This the log form keycloak after enabling logging:

```
2018-02-12 08:31:00.274 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.adapters.PreAuthActionsHandler       : adminRequest
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.274 3DEBUG 5802 --- [nio-8080-exec-1]
.k.a.t.AbstractAuthenticatedActionsValve : AuthenticatedActionsValve.invoke
/utente/prova4
2018-02-12 08:31:00.274 3DEBUG 5802 --- [nio-8080-exec-1]
```

```
o.k.a.AuthenticatedActionsHandler        : AuthenticatedActionsValve.invoke
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.274 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.AuthenticatedActionsHandler        : Policy enforcement is disabled.
2018-02-12 08:31:00.275 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.adapters.PreAuthActionsHandler       : adminRequest
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.275 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.AuthenticatedActionsHandler        : AuthenticatedActionsValve.invoke
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.275 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.AuthenticatedActionsHandler        : Policy enforcement is disabled.
2018-02-12 08:31:00.276 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.adapters.PreAuthActionsHandler       : adminRequest
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.276 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.AuthenticatedActionsHandler        : AuthenticatedActionsValve.invoke
http://localhost:8080/utente/prova4
2018-02-12 08:31:00.276 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.AuthenticatedActionsHandler        : Policy enforcement is disabled.
2018-02-12 08:31:10.580 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.s.client.KeycloakRestTemplate      : Created GET request for
"http://localhost:8181/auth/admin/realms/My%20Realm%20name/users"
2018-02-12 08:31:10.580 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.s.client.KeycloakRestTemplate      : Setting request Accept header to
[application/json, application/*+json]
2018-02-12 08:31:10.592 3DEBUG 5802 --- [nio-8080-exec-1]
o.k.a.s.client.KeycloakRestTemplate      : GET request for
"http://localhost:8181/auth/admin/realms/My%20Realm%20name/users" resulted in 401
(Unauthorized); invoking error handler
2018-02-12 08:31:10.595 ERROR 5802 --- [nio-8080-exec-1] o.a.c.c.C.[.[.[/].
[dispatcherServlet]    : Servlet.service() for servlet [dispatcherServlet] in
context with path [] threw exception [Request processing failed; nested exception
is org.springframework.web.client.HttpClientErrorException: 401 Unauthorized]
with root cause

org.springframework.web.client.HttpClientErrorException: 401 Unauthorized
    at
org.springframework.web.client.DefaultResponseErrorHandler.handleError(DefaultRespor
~[spring-web-4.3.13.RELEASE.jar:4.3.13.RELEASE]
    at
org.springframework.web.client.RestTemplate.handleResponse(RestTemplate.java:707)
~[spring-web-4.3.13.RELEASE.jar:4.3.13.RELEASE]
```

spring spring-boot spring-security authorization keycloak

Share Edit Follow

edited Dec 8, 2021 at 8:32

Chris K

357 ● 1 ● 4 ● 19

asked Feb 2, 2018 at 13:04

Teo

3,193 ● 2 ● 30 ● 59

What kind of authorization / authententication protocols are you using for your application? Do you relay on Keycloak as a OpenID Connect provider? – dchrzascik Feb 2, 2018 at 20:00

What is the keycloak api do you use to fetch users that are synced with LDAP? – Sindhu Arju Mar 14, 2019 at 12:34

2 Answers

Sorted by: Highest score (default)

23

In order to access the whole list of users, you must verify that the logged user contains at least the `view-users` role from the `realm-management` client, see this answer I wrote some time ago. Once the user has this role, the JWT she retrieves will cointain it.

As I can infer from your comments, you seem to lack some bases about the `Authorization` header. Once the user gets logged in, she gets the signed JWT from keycloak, so as every client in the realm can trust it, without the need to ask Keycloak. This JWT contains the access token, which is later on required in the `Authorization` header for each of user's request, prefixed by the `Bearer` keyword (see **Token-Based Authentication** in https://auth0.com/blog/cookies-vs-tokens-definitive-guide/).

+50

So when user makes the request to your app in order to view the list of users, her access token containing the `view-users` role already goes into the request headers. Instead of having to parse it manually, create another request yourself to access the Keycloak user endpoint and attach it (as you seem to be doing with `KeycloakBuilder`), the Keycloak Spring Security adapter already provides a `KeycloakRestTemplate` class, which is able to perform a request to another service for the current user:

SecurityConfig.java

```
@Configuration
@EnableWebSecurity
@ComponentScan(basePackageClasses = KeycloakSecurityComponents.class)
public class SecurityConfig extends KeycloakWebSecurityConfigurerAdapter {

    ...

    @Autowired
    public KeycloakClientRequestFactory keycloakClientRequestFactory;

    @Bean
    @Scope(ConfigurableBeanFactory.SCOPE_PROTOTYPE)
    public KeycloakRestTemplate keycloakRestTemplate() {
        return new KeycloakRestTemplate(keycloakClientRequestFactory);
    }

    ...
}
```

Note the scope for the template is `PROTOTYPE`, so Spring will use a different instance for each of the requests being made.

Then, autowire this template and use it to make requests:

```
@Service
public class UserRetrievalService{

    @Autowired
    private KeycloakRestTemplate keycloakRestTemplate;

    public List<User> getUsers() {
        ResponseEntity<User[]> response =
```

```
keycloakRestTemplate.getForEntity(keycloakUserListEndpoint, User[].class);
        return Arrays.asList(response.getBody());
    }

}
```

You will need to implement your own `User` class which matches the JSON response returned by the keycloak server.

Note that, when user not allowed to access the list, a 403 response code is returned from the Keycloak server. You could even deny it before yourself, using some annotations like: `@PreAuthorize("hasRole('VIEW_USERS')")`.

Last but not least, I think @dchrzascik's answer is well pointed. To sum up, I would say there's actually another way to avoid either retrieving the whole user list from the keycloak server each time or having your users stored in your app database: you could actually cache them, so as you could update that cache if you do user management from your app.

EDIT

I've implemented a sample project to show how to obtain the whole list of users, uploaded to Github. It is configured for a confidential client (when using a public client, the secret should be deleted from the application.properties).

See also:

- https://github.com/keycloak/keycloak-documentation/blob/master/securing_apps/topics/oidc/java/spring-security-adapter.adoc

Share Edit Follow

edited Apr 22, 2019 at 21:14

answered Feb 7, 2018 at 8:29

Aritz
31.4k ● 16 ● 141 ● 222

Thank you very much! really! but it downs't work yet :(:(If I use your project or if I use my project I obtain `Unauthorized` :(I added 2 images (**EDIT 5**) about my kecycloak config.. maybe it is there the error.. I'm losing hope :(– Teo Feb 10, 2018 at 9:14

I checked but it doesn't log anything.. That's a problem I think – Teo Feb 10, 2018 at 9:58

The error trace doesn't provide further info about the issue... Just to give it another try, could you test it with a completely new realm whose name doesn't contain spaces? That's the only difference I can see with my current test setup (apart from the KC version, I have tested against 2.2.1) – Aritz Feb 12, 2018 at 8:02

Glad to know it ;-) It seems that the adapter is messing up in some way with the space-containing realm name. – Aritz Feb 12, 2018 at 8:24

1 Ah ok.. I didn't think about the space as the possible cause.. Well after a while we fixed it :) – Teo Feb 12, 2018 at 11:36

9

I suggest double checking if you really need to have your own user store. You should relay solely on Keycloak's users federation to avoid duplicating data and hence avoiding issues that comes with that. Among others, Keycloak is responsible for managing users and you should let it do its job.

Since you are using OIDC there are two things that you benefit from:

1. In the identity token that you get in form of JWT you have a "sub" field. This field uniquely identifies a user. From the OpenID Connect spec:

   > REQUIRED. Subject Identifier. A locally unique and never reassigned identifier within the Issuer for the End-User, which is intended to be consumed by the Client, e.g., 24400320 or AItOawmwtWwcT0k51BayewNvutrJUqsvl6qs7A4. It MUST NOT exceed 255 ASCII characters in length. The sub value is a case sensitive string.

   In keycloak, "sub" is just a UUID. You can use this field to correlate your "object A" to "user B". In your DB this would be just a regular column, not a foreign key.

   In Java, you can access this JWT data using security context. You can also take a look at keycloak's authz-springboot quickstart where it is shown how you can access KeycloakSecurityContext - from there you can get an IDToken which has a getSubject method.

2. Keycloak provides Admin REST API that has a users resource. This is OIDC supported API so you have to be properly authenticated. Using that API you can perform operations on users - including listing them. You can consume that API directly or through use of Java SDK: keycloak admin client.

   In this scenario, you should use the JWT that you get from user in request. Using JWT you are sure that someone who is making a request can list all users in that realm. For instance, please consider following code:

```
@GetMapping("/users")
public List<UserRepresentation> check(HttpServletRequest request){
    KeycloakSecurityContext context = (KeycloakSecurityContext)
request.getAttribute(KeycloakSecurityContext.class.getName());

    Keycloak keycloak = KeycloakBuilder.builder()
                                    .serverUrl("http://localhost:8080/auth")
                                    .realm("example")
                                    .authorization(context.getTokenString())
                                    .resteasyClient(new
ResteasyClientBuilder().connectionPoolSize(20).build())
                                    .build();

   List<UserRepresentation> list = keycloak.realm("example").users().list();

   return list;
}
```

   In that case we are using HttpServletRequest and token that it contains. We can get the same data through use of `org.springframework.security.core.Authentication` from

spring security or directly getting an Authorization header. The thing is that KeycloakBuilder expects a string as a 'authorization', not an AccessToken - this is the reason why you have that error.

Please keep in mind that in order for this to work, user that is creating a requests, has to have a 'view-users' role from 'realm-management' client. You can assign that role to him in 'Role Mapping' tab for that user or some group to which he belongs.

Besides, you have to be properly authenticated to benefit from security context, otherwise you will get a null. Exemplary spring security keycloak configuration class is:

```
@Configuration
@EnableWebSecurity
@ComponentScan(basePackageClasses = KeycloakSecurityComponents.class)
class SecurityConfig extends KeycloakWebSecurityConfigurerAdapter {

@Autowired
public void configureGlobal(AuthenticationManagerBuilder auth) throws Exception {
    KeycloakAuthenticationProvider keycloakAuthenticationProvider =
keycloakAuthenticationProvider();
    keycloakAuthenticationProvider.setGrantedAuthoritiesMapper(new
SimpleAuthorityMapper());
    auth.authenticationProvider(keycloakAuthenticationProvider);
}

@Bean
public KeycloakSpringBootConfigResolver KeycloakConfigResolver() {
    return new KeycloakSpringBootConfigResolver();
}

@Bean
@Override
protected SessionAuthenticationStrategy sessionAuthenticationStrategy() {
    return new RegisterSessionAuthenticationStrategy(new SessionRegistryImpl());
}

@Override
protected void configure(HttpSecurity http) throws Exception {
    super.configure(http);
    http.authorizeRequests()
        .antMatchers("/api/users/*")
        .hasRole("admin")
        .anyRequest()
        .permitAll();
}
}
```

Share Edit Follow

edited Feb 6, 2018 at 15:55

answered Feb 3, 2018 at 10:18

dchrzascik

341 ● 1 ● 5

Actually I have the same problem as Droide.. Should I configure keycloak in some way to pass that header? Sorry, but what is it the authorization string? – Teo Feb 6, 2018 at 8:28

I gave example of directly passing a header as I didn't have a full project setup. Nevertheless idea is the same, you should be able to grab access token from KeycloakSecurityContext. Later I will try to prepare some sample that shows that. For the sake of clarity, are you using spring security with Keycloak or directly some keycloak java adapter? – dchrzascik Feb 6, 2018 at 8:31

Please let me know if my example has helped, if not, then there is some misconfiguration in your project. If this is the case, I will break down the example into details. – dchrzascik Feb 6, 2018 at 19:10

@dchrzascik thank yo ufor the answer.. But nothing change! I mean I already have that configuration as you can see inside **EDIT 3**.. – Teo Feb 7, 2018 at 7:00