Create secure API’s with Quarkus and JWT

This tutorial is related to building secure API’s with JWT and a maven project

Create your quarkus project

\* you can use their template : git clone <https://github.com/quarkusio/quarkus-quickstarts.git>

Firstly we need to add the dependencies relate to JWT:

<dependency>

<groupId>io.quarkus</groupId>

<artifactId>quarkus-smallrye-jwt-build</artifactId>

</dependency>

<dependency>

<groupId>io.quarkus</groupId>

<artifactId>quarkus-smallrye-jwt </artifactId>

</dependency>

Secondly we need to create our public and private keys:

1. Create a folder on your computer
2. Open cmd if you are using windows / terminal for linux and mac
3. Create the rsa with the command : openssl genrsa -out rsa.private 1024
4. Create the public key : openssl rsa -in rsa.private -out rsa.public -pubout -outform PEM
5. Create the private key: openssl genrsa -out rsa.private 1024

After this create your resource API

@GET  
@Path("/test")  
public Response test(){  
 return Response.*ok*().build();  
}

Add your roles to the endpoint with the following annotation: @RolesAllowed() and specify the roles that you want to use, in our case we will use @RolesAllowed(“test”)

Create the service that you want to create your key and add the following dependency and move the privateKey.pem that was created to its resources:

# Security  
smallrye.jwt.sign.key.location=privateKey.pem

Create the service responsible for the JWT token, here is an example:

@ApplicationScoped  
public class JwtAuthService {  
 public String generateJwt(String role) {  
 long duration = System.*currentTimeMillis*() + 3600;  
 return Jwt.*issuer*("https://test.com")  
 .subject("test")  
 .groups(role)  
 .expiresAt(duration)  
 .sign();  
 }  
}

Create also an API that returns the token that was created:

@Inject  
JwtAuthService jwtAuthService;  
  
@GET  
@Path("/jwt/{role})  
public Response createJwt(@PathParam("role") String role){  
 return Response.*ok*(jwtAuthService.generateJwt(role)).build();  
}

Go back to the project that has the api that you want to be secured and move the publicKey.pem to its resource folder and also add the dependency to application.properties:

mp.jwt.verify.publickey.location=publicKey.pem  
mp.jwt.verify.issuer=https://test.com

Now start both applications, firstly call the createJwt endpoint and get the key, secondly add the key to the rest call for the secured GET endpoint by attaching an Authorization policy with the required key.

If everything was followed properly the endpoint should work and return 200.

Aditionally Quarkus lets you specify also token algorithms as follows:

We can specify the algorithm for the signature in the src>java>main>resource>application.properties:

smallrye.jwt.new-token.signature-algorithm=PS256

Here we can also specify the encryption algorithm:

smallrye.jwt.new-token.key-encryption-algorithm=RSA-OAEP-256

smallrye.jwt.new-token.content-encryption-algorithm=A256CBC-HS512

Here is a list to the encryption algorithms:

|  |  |  |
| --- | --- | --- |
| smallrye.jwt.new-token.signature-algorithm | RS256 | Signature algorithm. This property  will be checked if the JWT signature  builder has not already set the  signature algorithm. |
| smallrye.jwt.new-token.key-encryption-algorithm | RSA-OAEP | Key encryption algorithm.  This property will be checked  if the JWT encryption builder  has not already set the key  encryption algorithm. |
| smallrye.jwt.new-token.content-encryption-algorithm | A256GCM | Content encryption algorithm.  This property will be checked  if the JWT encryption builder  has not already set the content  encryption algorithm. |