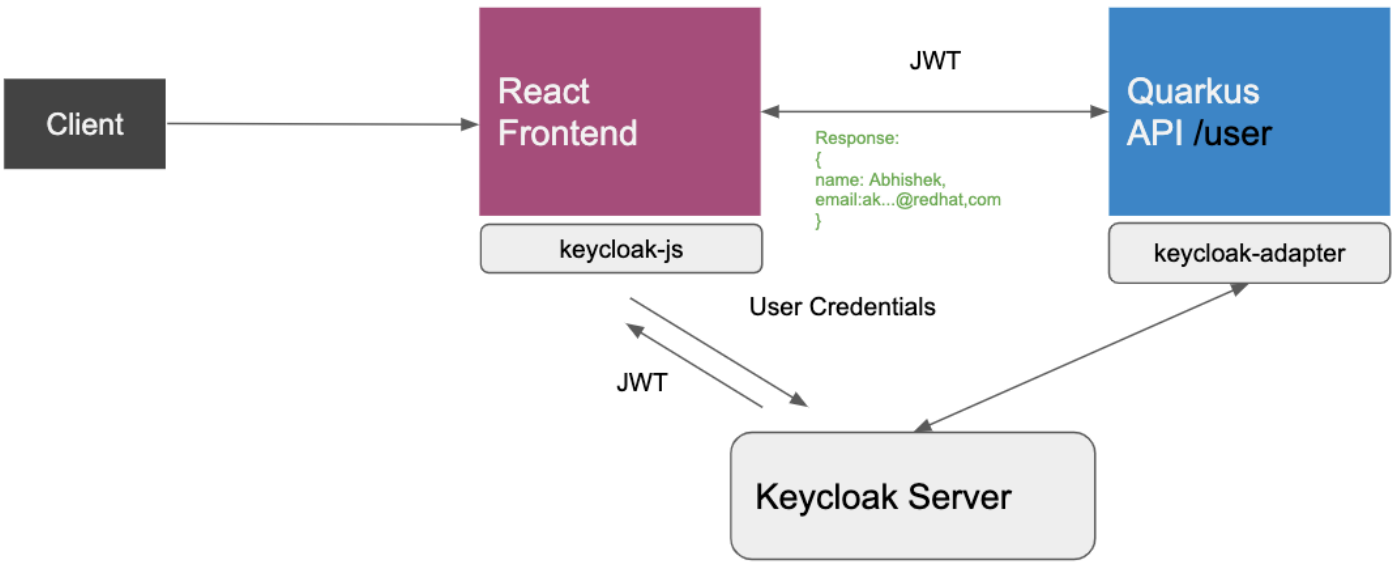
Quarkus and React integration with keycloak

Abhishek koserwal Follow
Nov 20, 2019 · 4 min read

In this post, we will learn about how to design a stateless secure enterprise application using Quarkus as backend and React as a frontend app; using keycloak as an IAM solution. If you are not familiar with keycloak basics, you can go through this post Essentials. This type of architecture is suitable for building cloud-native applications deployed as independent containers.

In this type of pattern, your backend only cares about a valid token (JWT) with each request from the frontend or any other service. We are not managing any kind of user session at the backend i.e we called it as a stateless architecture pattern.

If you are deploying this kind of pattern in Kubernetes/Openshift; you are free to scale individually containers without worrying about losing the user session or managing sticky sessions at the load balancer level.



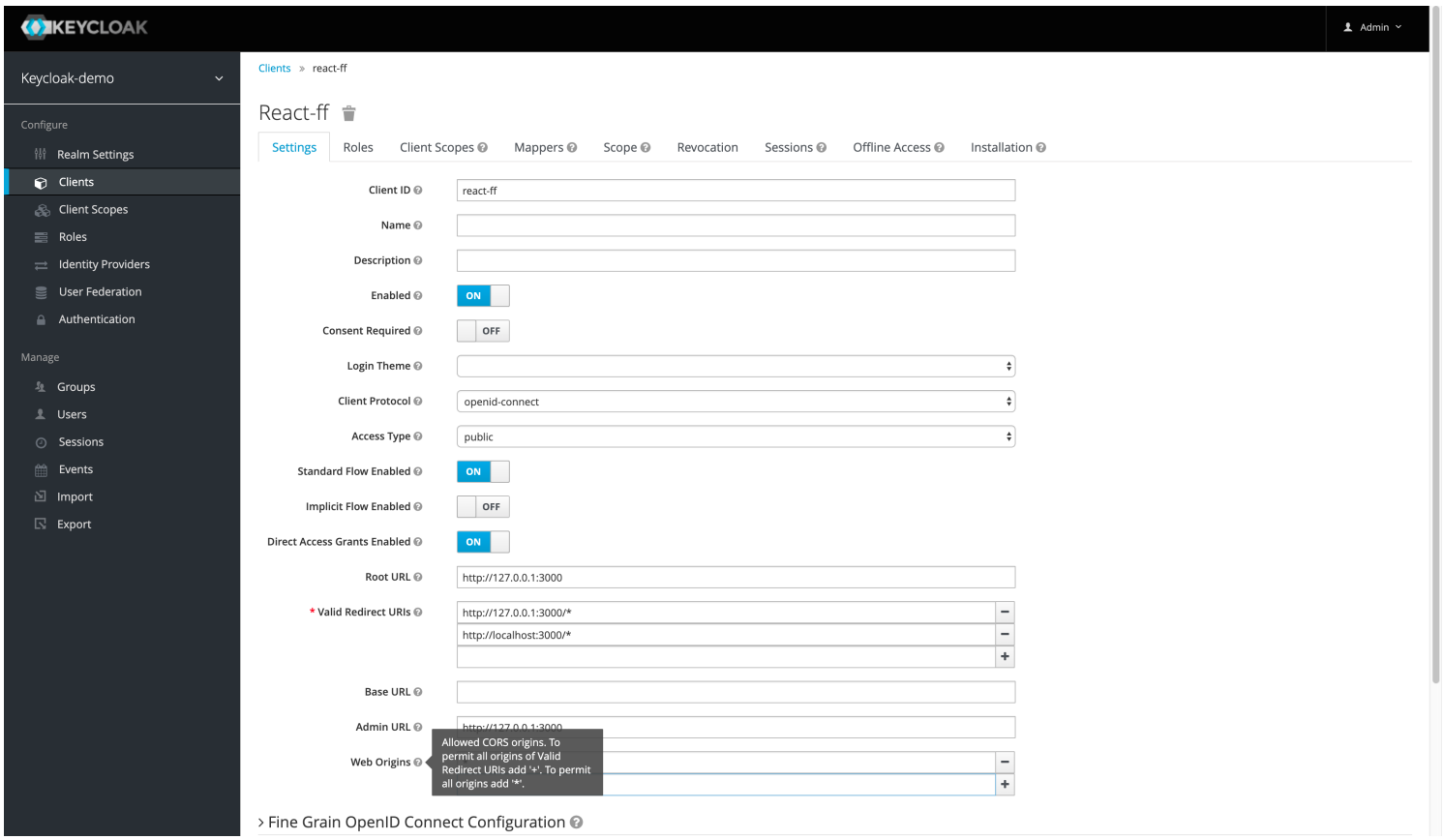
You can access the code from this repo: link. First, we will see the React app integration and then Quarkus backend.

REACT (16.12.0)

For understanding the react integration with keycloak in dept, you can go through this post: Secure React App with Keycloak. I will skip the basic part and directly jump to the client configuration for this scenario.

Client Configuration: using “keycloak-demo” as the realm

- **Client ID: react-ff**
- **Web Origin**: “*” (Only for the development & use “+” or domain name for production)
- **Root Url**: http://localhost:3000



App.js

I am using `React Hooks` to implement a simple GET request to our backend API with JWT token as a header option. For each request, you need to pass this header option otherwise you will get status: 401 (unauthorized).

```
function App() {
  const [data, setData] = useState({ users: [] });
  useEffect(() => {
    const fetchData = async () => {
      const result = await axios(
        'http://127.0.0.1:8080/user', { headers: { "Authorization": "Bearer " + localStc
      );
      setData(result.data);
    };
    fetchData();
  }, []);

  return (
    <div className="App">
      <header className="App-header">
        <h1>Secure React App</h1>
        <div>
          <img src={logo} className="App-logo" alt="logo" />
        </div>
        <div>

        <h2>Response from Quarkus API: /user </h2>

          <p>Name: {data.name}</p>
          <p>Email:{data.email}</p>

        </div>
      </header>
    </div>
  );
}

export default App;
```

App.js hosted with ❤️ by GitHub view raw

You can try running this application using

```
yarn start
```

Open the browser and hit: http://localhost:3000, it will redirect to keycloak server default login page for the realm: “keycloak-demo”.

After authentication, as you can see our application is working fine. But, we haven’t got any response from the API as we need to build it.

react-app

Let's now build a backend API using the Quarkus

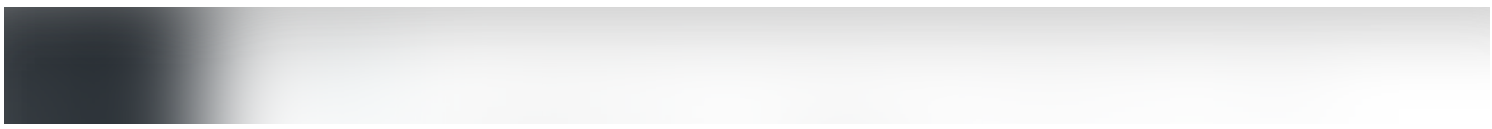
Quarkus (1.0.0.CR1)

We can start with the keycloak configuration for our API.

Client Configuration: using "keycloak-demo" as the realm

- **Client ID:** quarkus-bff
- **Access Type**: confidential (in case of our frontend react app we have kept this as public)

Once you save, you can see the **Credentials** tab on the client dashboard. Copy the client secret and keep it confidential.



You can quickly generate a code-base for you to work from https://code.quarkus.io/

Or refer the following code:

akoserwal/keycloak-integrations

You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or...

github.com

Adding the Dependencies:

```
<dependency>
   <groupId>io.quarkus</groupId>
   <artifactId>quarkus-oidc</artifactId>
 </dependency>
 <dependency>
   <groupId>io.quarkus.keycloak</groupId>
   <artifactId>quarkus-keycloak-adapter</artifactId>
   <version>1.0.0.Alpha1</version>
 </dependency>
```

API

It's a simple API, exposing as "<host>/user" endpoint. I am just returning a user object with a name and email as a response.

You need to use "@Authenticated" annotation to secure your end-point.

```
@Path("/user")
@Authenticated
@Produces(MediaType.APPLICATION_JSON)
@Consumes(MediaType.APPLICATION_JSON)
public class UserResource {

    private final User user;

    public UserResource() {
        user = new User();
        user.setEmail("abc..@redhat.com");
        user.setName("abhishek");
    }

    @GET
    public Response user(){
        return Response.ok(user).build();
    }
}
```

UserResource.java hosted with ❤️ by GitHub view raw

Adding the client configuration: application.properties

```
# security configurations
quarkus.oidc.client-id=quarkus-bff
quarkus.oidc.credentials.secret=<secret>
quarkus.oidc.auth-server-
url=http://localhost:8081/auth/realms/keycloak-demo
```

keycloak server URL will be: http://localhost:8081/auth/realms/<Your realm> (realm:keycloak-demo in this case)

This is all you need, now you can run the application

```
./mvnw compile quarkus:dev
```

Now if you try to access the react application, you will see the response from the API server.

Our backend is validating the JWT token using the key published by the keycloak server: (https://<keycloak-server>:<port>/auth/realms/${REALM}/protocol/openid-connect/certs) with every request from the frontend.

In conclusion, we have seen how to build a secure stateless application using react and quarkus. You can use follow a similar pattern with any other frontend frame/library and Quarkus as backend.

GitHub project link

Thank you for reading this post. If you like this post, give a Cheer!!!

Follow the Collection: Keycloak for learning more…

Happy Secure Coding ❤

Quarkus Keycloak React Jwt Stateless