Session Connect Authenticator

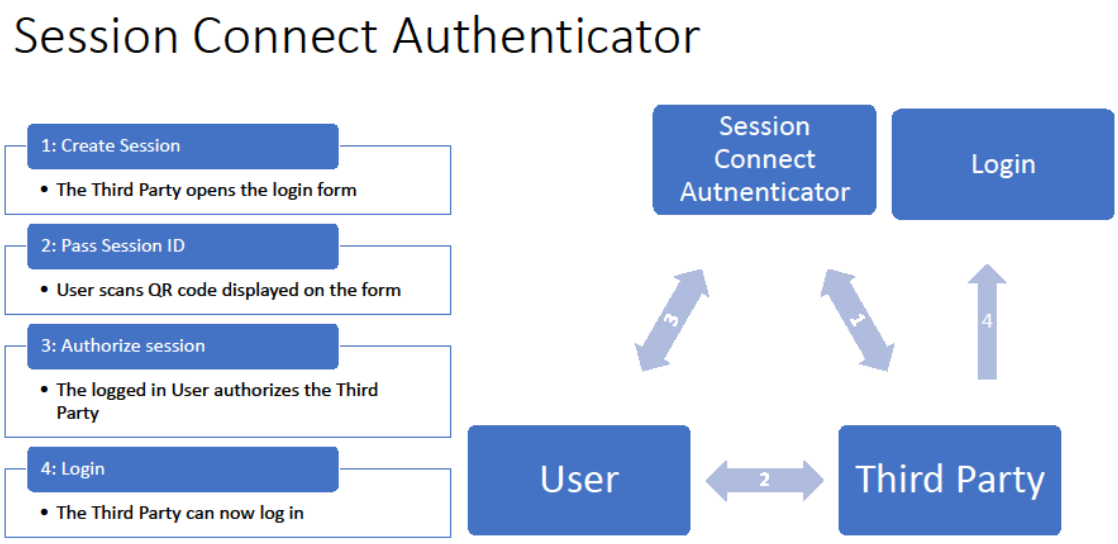
1 General

The Session Connect Authenticator consists oft wo JBoss modules, which can be added to a Keycloak server.

The authenticator enables a third party to log in with a user's account, without knowing his username and password. Therfore, the third party calls the authentication website with the query parameter „use_sessionconnect", e.g. http://mykeycloak.com/auth/realms/master/protocol/openid-connect/auth?client_id=...&redirect_uri=...&response_mode=fragment&response_type=code&scope=openid&use_sessionconnect.

The returned website contains a QR code. The user can scan this code using a special app, whereby he authorizes the third party to log in as him. Hereby, the user's app performs a PUT operation on the REST endpoint http://mykeycloak.com/auth/realms/master/sessionconnect/<QR-code-content>.When the third party then clicks the login button, he will be logged in as the user.

The following figure describes, how the Session Connect Authenticator works:



2 USING A CUSTOM FORM

The form, which will be displayed tot he Third Party can be configured. We recommend to use your own one, specifically tailored to your application's needs. Therfore, you can create your own Freemaker file named „sessionconnect-form.ftl". To use this form, you can replace the file „sessionconnect-form.ftl" in the „ftl" directory of the Authenticator project with your own one.

The Session Id will be passed to the form under the attribute name „session_id". Make sure the Session Id will be displayed in a format that can be processed by the user's application.

When submitting the form, the Keycloak server needs to know, which Session Id should be used to log in. Therfore, there must be an input element named „session_id" containing the Session Id.

The Third Party has to submit this form, after the user connected the displayed Session Id with his account. If you are displaying the form in a web browser, your form should therfore have a submit element. Alternatively, you can embed the form into your own user interface. In this case, you should submit the form programatically. If you are doing so, do not forget to add the Session Id as an input parameter to the submission.

3 Deploying The Keycloak Server

There are several possibilities to make the Session Connect Authenticator usable on a Keycloak server.

3.1 Deployment on IBM Cloud

Execute deploy.cmd in the Keycloak project.

Make sure, that the version numbers of the authenticator and the webservice in the beginning of the script are equal tot he version numbers defined in the build.gradle files in the corresponding projects. Furthermore, make sure that the version number in deployment.yml is equal tot he one defined in deploy.cmd.

3.2 Docker Image

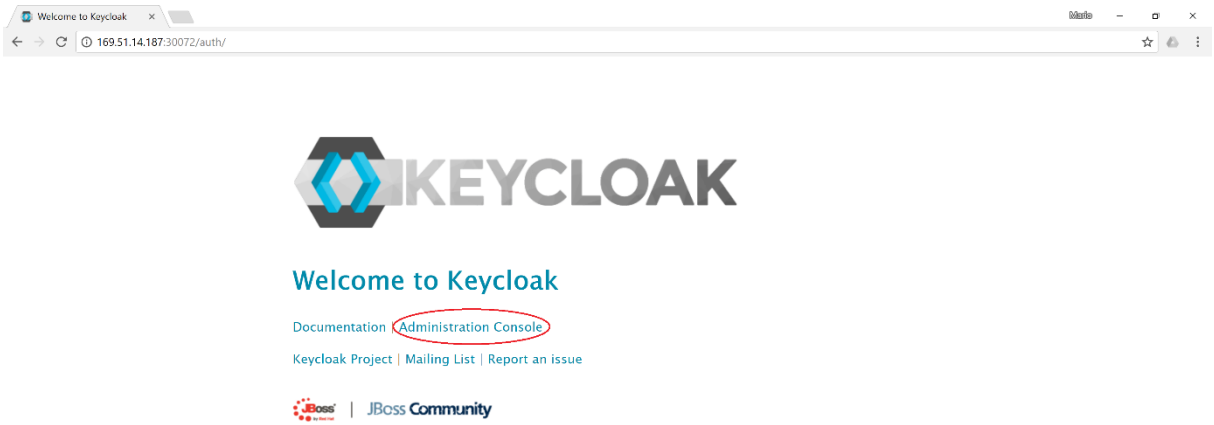
If you want to use a Keycloak server using the Session Connect Authenticator without a Kubernetes cluster on IBM Cloud, you can create and run a Docker image. Make sure, that the version numbers of the authenticator and the webservice in the beginning of the script are equal tot he version numbers defined in the build.gradle files in the corresponding projects. Then you can build the Authenticator and the Webservice project separately by calling „gradle build“ in the corresponding project directories. Now, you have to create a new directory „module“ in the Keycloak project and copy the .jar files of the builds into this directory, using the names „Authenticator.jar“ and „Webservice.jar“ respectively. After that, copy the ftl directory of the Authenticator into the Keycloak project. If you are using a custom form, replace the .ftl files in this directory with your own ones. Now you can create a Docker image using the Dockerfile in the Keycloak project.

3.3 Existing Keycloak Server

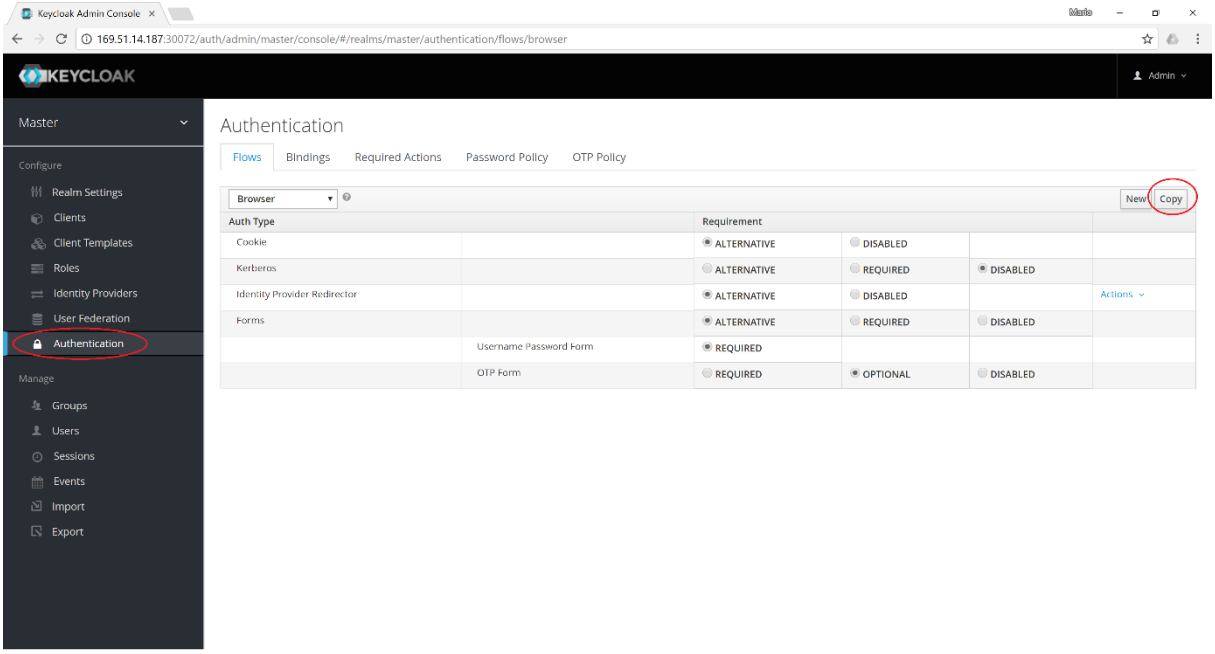
If you already have an existing Keycloak server, or you do not want to use Docker, you can add the Session Connect Authenticator as two JBoss modules to your Keycloak server. Therfore, build the Authenticator and the Webservice separately by calling „gradle build“ in their project directories. For each oft he two, create a new JBoss module for each oft he resulting .jar files by using the JBoss CLI. Add the modules tot he standalone.xml of your JBoss Server. You can find the exact commands in the Dockerfile in the Keycloak project.

4 CONFIGURING THE KEYCLOAK SERVER

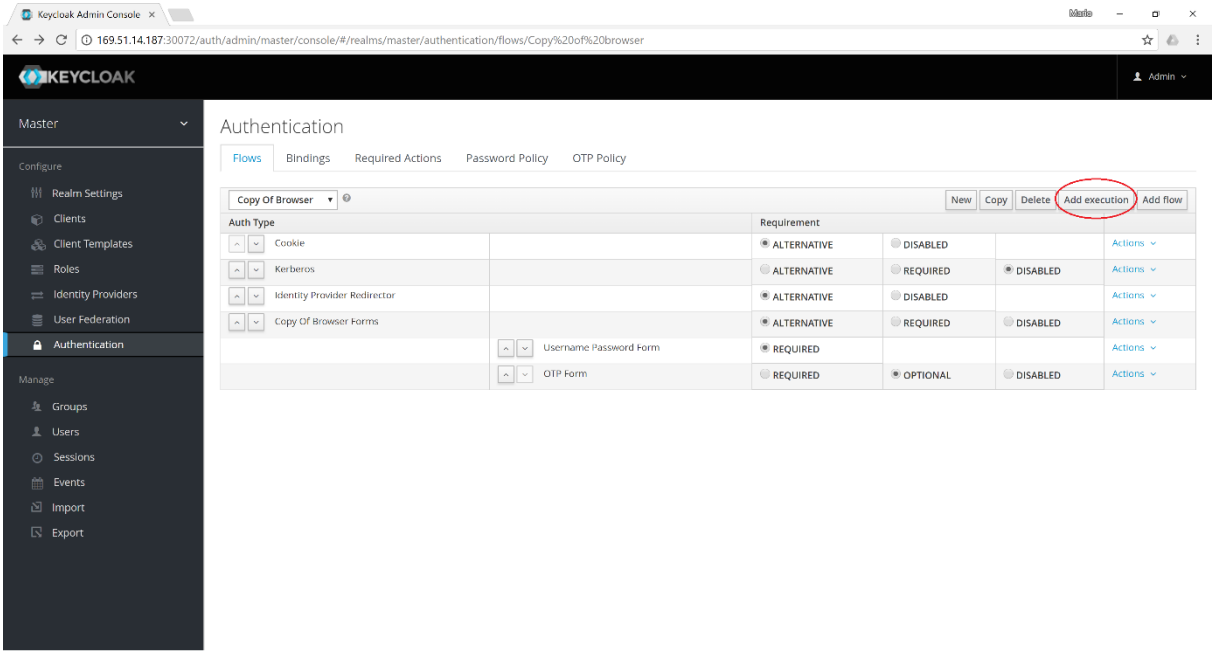
Open the administration console of your Keycloak server. If you created the server using the deploy script or the Dockerfile, you can log in with „admin“ as username and password:

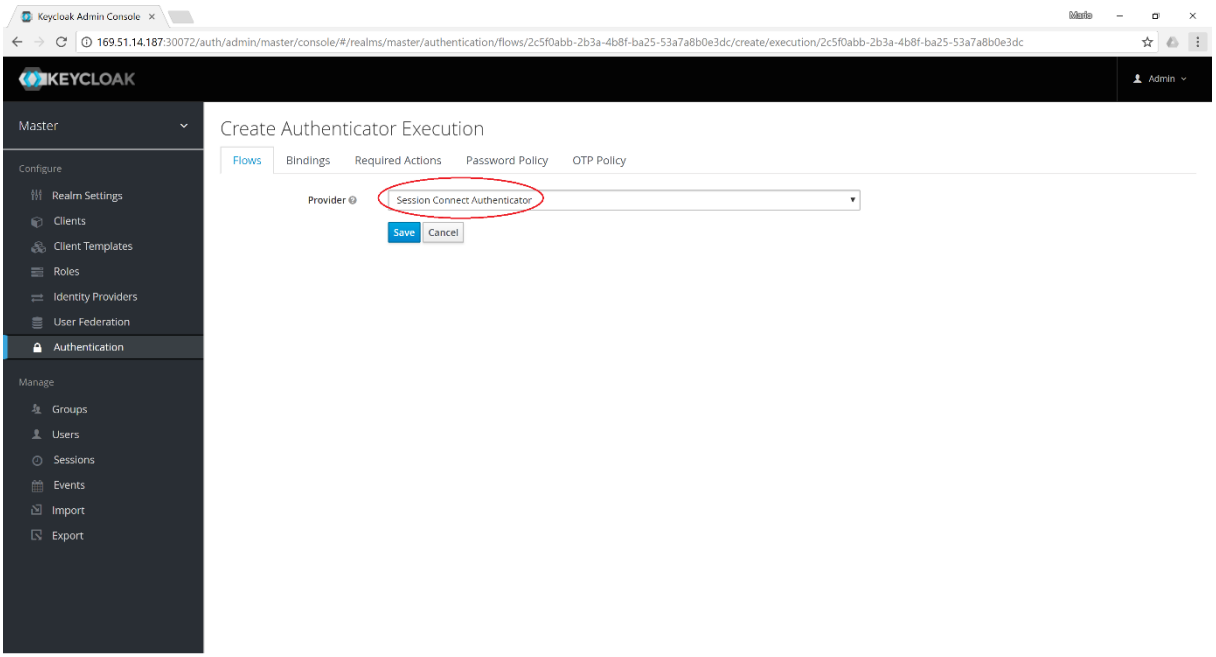


Now choose the entry „Authentication“ in the left context menu and copy the existing „Browser“ flow. Alternatively, you can define your own flow:

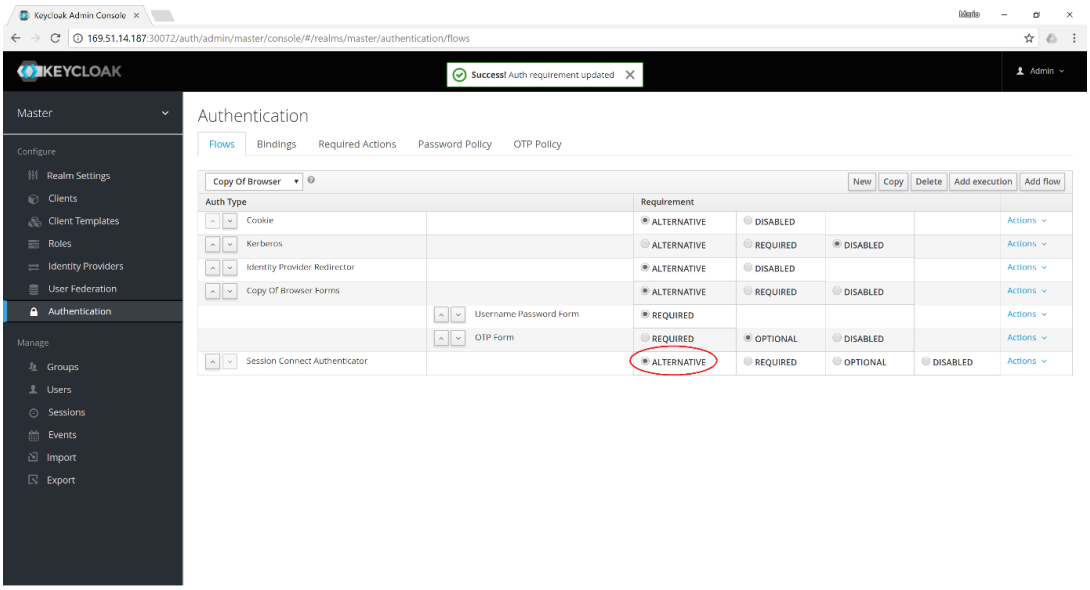


Now add the Session Connect Authenticator as a new execution.



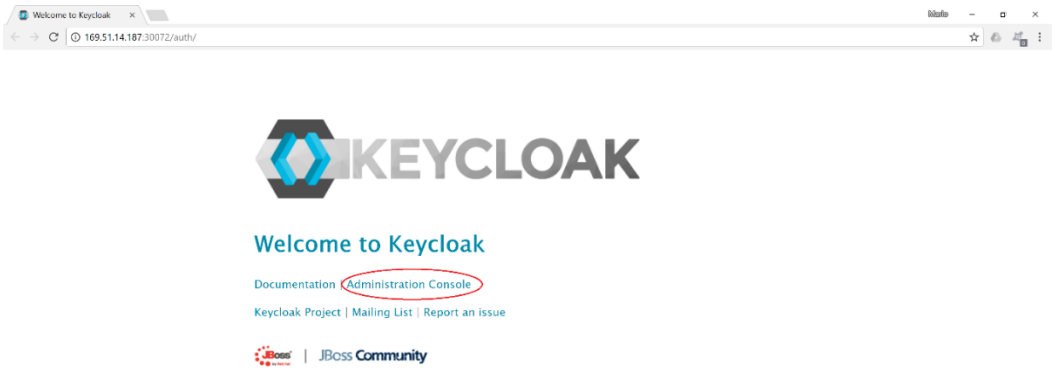


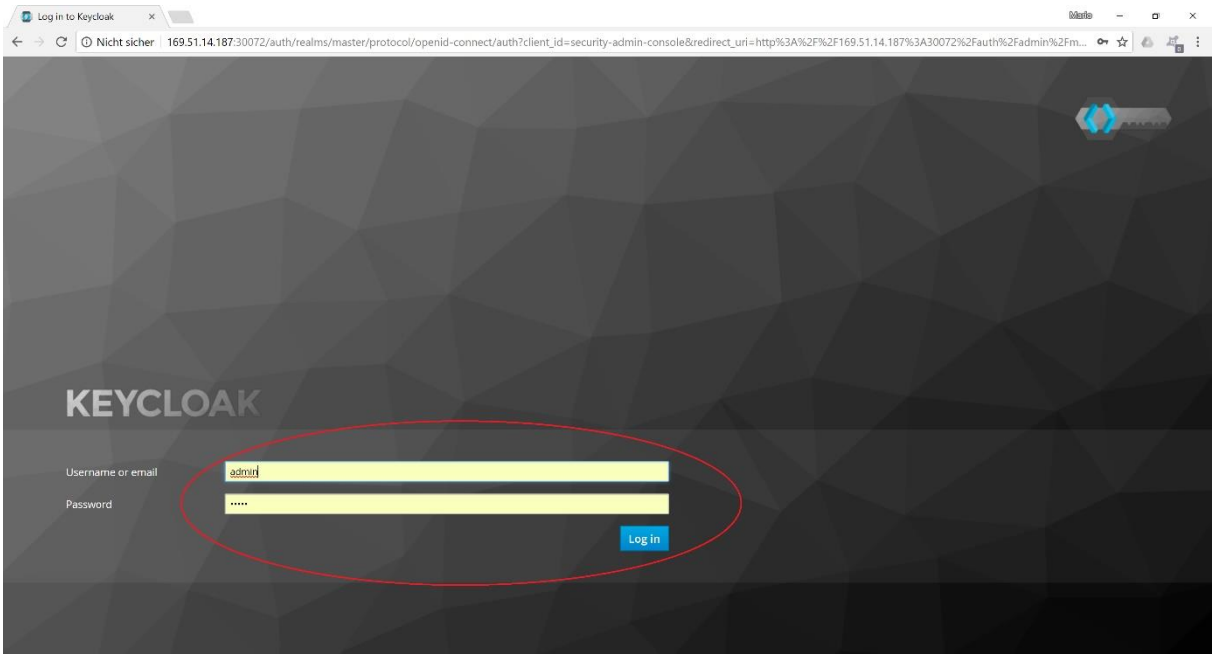
Now mark the Session Connect Authenticator as an alternative. This is neccessary, because there has to be another authenticator besides this one, which will be used by the user:



Now move the Session Connect Authenticator up in the priority list, so that it is above all alternative executions using a form. If you copied the „Browser“ flow, you have to move it above the execution „Copy of Browser Forms“. This is neccessary, because in contrast tot he Session Connect Authenticator, other form based authenticators cannot be skipped, so they would block the Session Connect Authenticator.

A user can now log into the Keycloak server using the following username and password: http://<keycloak-server>/auth/realms/master/protocol/openid-connect/auth?client_id=...&redirect_uri=...&response_mode=fragment&response_type=code&scope=openid:





If a third party wants to log in using the Session Connect Authenticator, he hast o add the query parameter „use_sessionconnect“ to the url: