Octopus Framework

Rudy De Busscher

Version 0.2, ??/??/2017

Table of Contents

Release notes

0.2

1. Split into different modules (Core, JSON, Non-Web [Java SE, …], Web [JSF, JAX-RS])
2. Octopus-jwt-support for handling JSON supporting plain, JWS and JWE.
3. Octopus-json is optimized smart-json code
4. MicroProfile JWT Auth for Rest (POC)
5. OfflineToken for standalone Java SE (POC)

0.1

1. POC integration Apache Shiro into Octopus

Modules

List of Maven modules

Artefact	SE, CDI, EE	info
be.atbash:octopus-utils	SE	Needs to be moved externally (Use by Config, Jerry, …)
be.atbash.json:octopus-json-smart	SE	Very light-weight JSON reader/writer (based on json-smart). Can be externalized if needed.
be.atbash.ee.security:octopus-jwt-support	SE, CDI	Java Beans to/from JSON/JWS/JWE. Can be externalized if needed. octopus-keys can be extracted from it.
be.atbash.ee.security:octopus-core	SE, CDI	All Octopus classes usable in Java SE and Java EE environment.
be.atbash.ee.security:octopus-common-se	SE, CDI	All Octopus classes Specific for Java SE
be.atbash.ee.security:octopus-se-standalone	SE, CDI	Specific for Java SE CLI programs
be.atbash.ee.security:octopus-token-generator	SE, CDI	Contains class to generate the Offline Token (for SE usage).
be.atbash.ee.security:octopus-common-web	EE (Web)	All Octopus classes Specific for Java EE (Web - Servlets)
be.atbash.ee.security:octopus-rest	EE (JAX-RS)	Specific for JAX-RS
be.atbash.ee.security:octopus-jsf7	EE (JSF)	Specific for JSF
be.atbash.ee.security:octopus-mp	EE (JAX-RS)	Support for MP JWT Auth tokens

octopus-utilities contains for the moment the JavaFX app to maintain JWK files.

Java EE

JAX-RS

Core

FIXME

MP Auth token

Creation of the token can be done be using the **be.atbash.ee.security.octopus.token.MPJWTTokenBuilder** class.

Maven artefact be.atbash.ee.security:octopus-mp contains the *mpUser* filter.

Java SE

OfflineToken

Offline token can be used for standalone Java SE programs.

A token can be generated which will be only valid for a certain computer.

Besides the Processor Id and the first disk UUID, also a pass phrase is required (when multiple users are using the program on the same laptop/desktop.)

Steps (example flow, final programs not created yet)

1. Program **LocalSecret** (*examples/local-secret*) generates the token which is user dependent for a certain machine(Standalone program run by the end-user)
2. Program **CreateOfflineTokenFile** (*examples/se-cli*) generates the offline token (here stored within the *<user_home>/octopus.offline.token* file)
3. Program **SecuredCLI** uses the offline token to authenticate/authorize using Octopus.