Getting started with Apiman

Getting started with Apiman

This article will help you to get up and running with **apiman**, an open-source API management and gateway offering.

Apiman overview

APIs are becoming pervasive in the modern enterprise. External facing APIs allow 3rd party access to your company's content and services, while the trend towards microservices results in loosely coupled services, connected by multiple internal APIs. Using an API gateway helps you to centralize the management of your APIs, providing support for common features such as security, quotas, logging and caching.

Apiman is an open-source, API management software offering that simplifies the control, distribution and monitoring of your APIs. It consists of two main components:

- An easy-to-use management interface that helps you to configure common policies for your APIs
- A runtime gateway component that executes the rules defined in the configured policies

Apiman features

The apiman software provides support for multiple policy rules that can be combined to achieve your desired outcome. It includes rules for:

- Securing access based on user identity (authentication), and based on user role (authorization)
- Limiting access using rate-limiting for fine-grained control, quotas for longer periods, and transfer qoutas for data-intensive operations
- Enabling/disabling access to subsets of an API
- Logging and caching
- Enabling/disabling access from specific IP addresses

Advantages of apiman

In addition to the normal advantages of using open source software, apiman provides benefits in other ways as well. It offers enterprise-grade security functionality, with a wide range of encryption, authentication, and authorization protocols supported. While the software provides many policies out-of-the-box, it is also extensible through the provision of a plug-in framework, allowing you to easily add support for the specific rules you require. Finally, all of the functionality provided by apiman in the manager UI is also accessible through a REST API, meaning you can easily automate common tasks.

Installing the software

This article describes getting started with apiman version 1.5.7, using WildFly version 10.1 as the host application server. The software runs on Linux, Windows and MacOS.

Prerequisites

- **Java JDK 1.8 or newer:** You can use OpenJDK or Oracle's JDK.
- **Git:** A git client is needed to download the quickstart sample used in this article.
- **Maven:** Apache Maven is used to build the quickstart sample. Version 3.6.3 is the latest release and recommended version, but any version from 3.3 up will work.

Installing WildFly

WildFly is an application development platform, based on Jakarta EE. It is highly configurable, and facilitates faster development and testing. It is an open source community project, sponsored by Red Hat™, and is available for use and distribution under the LGPL v2.1 license.

Use the following steps to download and configure the WildFly software:

1. Download the WildFly software from http://download.jboss.org/wildfly/10.1.0.Final/wildfly-10.1.0.Final.zip.
2. Extract the contents of the zip file. In this remainder of this article, it is assumed that the WildFly content has been extracted to `~/wildfly-10.1.0.Final`.
3. Use the `add-user.sh` script in the WildFy `bin` directory to configure a user, for example,

   ```
   $ cd ~/wildfly-10.1.0.Final
   $ ./bin/add-user.sh -u user1 -p password1
   ```

   If you are running Windows, you should use the `add-user.bat` or the `add-user.ps1` script, as appropriate.

Installing apiman

Use the following steps to download and deploy apiman:

1. Download the apiman software from http://downloads.jboss.org/apiman/1.5.7.Final/apiman-distro-wildfly10-1.5.7.Final-overlay.zip.
2. Extract the contents of the zip file. In the subsequent steps, it is assumed that the apiman content has been extracted to `~/apiman-distro-wildfly10-1.5.7.Final-overlay`.
3. Copy the contents of the apiman directory into the WildFly directory. This will result in an **overlay** of some apiman content on top of the existing WildFly content, in particular, the `modules`, `standalone` and `themes` folders.

   ```
   cp -r ~/apiman-distro-wildfly10-1.5.7.Final-overlay/.  ~/wildfly-10.1.0.Final/
   ```

4.

Check that the overlay has worked correctly by locating the apiman configuration file in the correct the WildFly folder:

```
$ ls ~/wildfly-10.1.0.Final/standalone/configuration/standalone-apiman.xml
```

Running the software

Start the WildFly server, using the apiman configuration file:

```
$ cd ~/wildfly-10.1.0.Final
$ ./bin/standalone.sh -c standalone-apiman.xml
```

If the server has started up correctly, you should see the address of the WildFly administration console in the log information that is displayed on the screen:

```
13:26:45,008 INFO  [org.jboss.as] (Controller Boot Thread) WFLYSRV0051: Admin console
listening on http://127.0.0.1:9990
```

NOTE Server logs are available at `~/wildfly-10.1.0.Final/standalone/log/server.log`.

To confirm that WildFly has started up correctly, navigate to http://127.0.0.1:9990 in your browser and log in with the credentials you specified when running the `add-user` script earlier. Check that the apiman software has been deployed correctly by navigating to the page http://127.0.0.1:9990/console/App.html#standalone-deployments.

Access the apiman user interface by navigating to http://127.0.0.1:8080/apimanui and logging in with the default credentials:

- **Username or email:** admin
- **Password:** admin123!

WARNING This article shows you how to get up and running quickly with apiman and it is not intended to be a guide for good practices. For more information on deploying and hardening apiman in a production environment, see the Apiman Production Guide.

Deploy the quickstart sample API

The apiman project provides some examples to help you get started. The echo-service quickstart is a simple service that responds to a HTTP request with a copy (echo) of the request configuration.

Download the quickstart project

Use your git client to download the `apiman-quickstarts` project:

```
$ cd ~
$ git clone https://github.com/apiman/apiman-quickstarts.git
```

Build the sample API

Use Maven to build the sample API:

```
$ cd ~/apiman-quickstarts/echo-service
$ mvn package
```

Check that the build worked by locating the generated `war` file:

```
$ ls ./target/apiman-quickstarts-echo-service-1.3.1.Final.war
```

Deploy the sample API

Copy the war file to the application server:

```
$ cd ~/apiman-quickstarts/echo-service/target/
$ cp ./apiman-quickstarts-echo-service-1.3.1.Final.war \
  ~/wildfly-10.1.0.Final/standalone/deployments/
```

Test the sample API

Use your browser to access the API at http://localhost:8080/apiman-echo. The service should return a response that is a copy (echo) of the request:


```
{
  "method" : "GET",
  "resource" : "/apiman-echo",
  "uri" : "/apiman-echo",
  "headers" : {
    "Accept" :
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/
*;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "Upgrade-Insecure-Requests" : "1",
    "User-Agent" : "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML,
like Gecko) Chrome/85.0.4183.121 Safari/537.36",
    "Connection" : "keep-alive",
    "Sec-Fetch-Dest" : "document",
    "Sec-Fetch-Site" : "none",
    "Host" : "localhost:8080",
    "Accept-Language" : "en-GB,en-US;q=0.9,en;q=0.8,de;q=0.7",
    "Accept-Encoding" : "gzip, deflate, br",
    "dnt" : "1",
    "Sec-Fetch-Mode" : "navigate"
  },
  "bodyLength" : null,
  "bodySha1" : null
}
```

Configure an API provider

Now that you have the sample API working, you can use apiman to configure access to the API using the following steps:

1. Create an organization and add a plan containing a policy
2. Configure an API, specifying the target API implementation and the plan

Create a provider organization

Create a new organization to manage your APIs with the following configuration:

- **Organization Name:** ProviderOrg
- **Description:** A sample provider organization

Create a new plan

Create a new plan within the specified organization with the following configuration:

- **Organization:** ProviderOrg
- **Plan Name:** Gold
- **Initial Version:** 1.0 (default)
- **Description:** Most expensive plan

Add a policy to the plan

Adding a policy to a plan allows the policy's functionality to be applied to the API invocation as part of the overall policy chain. In this example, a rate-limiting policy is created, to allow a maximum of 10 requests from a client application in one hour:

- **Policy Type:** Rate-limiting policy

  Rate Limiting Policy Configuration
  - **# of requests:** 10
  - **Granularity:** Client App
  - **Duration:** Hour

Lock the plan

You must lock a plan to make it available to be included in APIs. Locking a plan renders it immutable, requiring a new version to be created in order to make changes to the plan.

Click the "Lock" button and the plan status will change to `Locked`.

Create an API

From the ProviderOrg page, click on the "APIs" tab and create a new API with the following configuration:

- **API Name:** echo

Initial Version: 1.0 (default)

- **Description:** The echo service

Configure the API implementation

On the "Implementation" tab for the API, configure the details of the real API being managed, as shown below, and then save the implementation:

- **API Endpoint:** http://localhost:8080/apiman-echo
- **API Type:** REST (default)
- **API Content Type:** JSON (default)
- **API Security:** None (default)

Configure the API plan

On the "Plans" tab, you can configure which plans are available to a client application. If the API is marked as "Public", it can be invoked without sending an API Key. This example shows you how to use API keys, so do not mark the API as public in this case.

Choose the Gold plan that you created earlier and press "Save".

Publish the API

To make the API available to consumers, you need to click the "Publish" button.

Configure an API consumer

Now that you have finished publishing the API through the provider organization, you can configure a consumer organization and a client application to access the configured API.

Create the consumer organization

Create a new organization to consume the published API, with the following configuration:

- **Organization Name:** ConsumerOrg
- **Description:** A sample consumer organization

Create a client application

Access the "Client Apps" tab for the consumer organization, and create a new client app with the following configuration:

- **Organization:** ConsumerOrg
- **Client App Name:** EchoApp
- **Initial Version:** 1.0 (default)
- **Description:** A client application for consuming the echo API

Create a contract

A contract links a consumer organization's client app to a specific plan offered by the provider organization's API.

1. From the EchoApp page, choose "Search for APIs to consume". You can enter the term "echo" in the search box to find the echo API.
2. Click on the link to the echo service (or navigate directly to http://127.0.0.1:8080/apimanui/api-manager/browse/orgs/ProviderOrg/echo/1.0).
3. Create a new contract for the Gold plan that you created earlier. After clicking on the "Create Contract" button on the Gold plan, you will be presented with a summary of the contract details:
   - **Client App:** ConsumerOrg/EchoApp 1.0
   - **Plan:** Gold
   - **API:** ProviderOrg/echo 1.0
4. Click "Create Contract" if you are happy with the details in the summary page.
5. Click the "Register" button to register the application with the API Gateway, so that the gateway can act as a proxy for the API.

Consume the API

Once you have registered the client application with the gateway, you are ready to test that the client can access the API through the gateway, and that the Gold policy is being enforced correctly.

1. Navigate from the ConsumerApp page to the EchoApp page, and then access the "APIs" tab to list details of the APIs that the client can consume.

   Alternatively, navigate directly to the APIs page for the client app using the URL http://127.0.0.1:8080/apimanui/api-manager/orgs/ConsumerOrg/clients/EchoApp/1.0/apis
2. Click the information symbol at the right hand side of the API details, to determine how to invoke the API. This will display a URL for the `apiman-gateway` including the API key required, of the form:

   ```
   https://localhost:8443/apiman-gateway/ProviderOrg/echo/1.0?apikey=591c4999-c9d7-4513-a395-79cd903309fc
   ```
3. Use the URL to invoke the API. The response should be a JSON structure echoing the request configuration, similar to the following:

```
{
  "method" : "GET",
  "resource" : "/apiman-echo",
  "uri" : "/apiman-echo",
  "headers" : {
    "Accept" :
"text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apn
g,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9",
    "User-Agent" : "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36
(KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36",
    "Connection" : "keep-alive",
    "Sec-Fetch-Dest" : "document",
    "Sec-Fetch-Site" : "none",
    "Host" : "localhost:8080",
    "Accept-Encoding" : "gzip, deflate, br",
    "dnt" : "1",
    "Pragma" : "no-cache",
    "Sec-Fetch-Mode" : "navigate",
    "Cache-Control" : "no-cache",
    "Upgrade-Insecure-Requests" : "1",
    "Sec-Fetch-User" : "?1",
    "Accept-Language" : "en-GB,en-US;q=0.9,en;q=0.8,de;q=0.7"
  },
  "bodyLength" : null,
  "bodySha1" : null
}
```


4. Continue to invoke the API until you hit the rate limit, as shown in the response below:


```
{"type":"Other","failureCode":10005,"responseCode":429,"message":"Rate limit
exceeded.",
 "headers":{"X-RateLimit-Limit":"10","X-RateLimit-Remaining":"-1","X-RateLimit-
Reset":"3088"}}
```


Summary

This article has shown you how to get up and running with apiman, an open-source API management and gateway offering. You should now be able to publish an API for a provider organization with a rate-limiting policy, and test that policy using a client application associated with a consumer organization. If you would like to know more about apiman functionality in general and how to progress to using the software in a production environment, check the Resources section below.

This article is based on the "Crash Course in Apiman" document which provides a more comprehensive introduction to apiman. It is available on the apiman website at http://www.apiman.io/latest/crash-course.html.

Resources

Apiman project page: http://apiman.io/

Apiman GitHub repository: https://github.com/apiman/apiman

Apiman user guide: https://apiman.gitbooks.io/apiman-user-guide/content/

Apiman production guide: https://apiman.gitbooks.io/apiman-production-guide/content/