

# Getting started with apiman: Deploy a sample echo service

---

apiman is an API management system that addresses the following needs for enterprises:

- Publicize APIs to make them discoverable by API consumers
- Track API use and consumption for rate limiting and monetization

This article provides a rapid walk through that shows how to deploy a sample application that is named echo-service and to configure apiman to apply a rate-limiting policy.

## Overview

apiman provides enterprise capabilities and those capabilities are available through several configuration choices. To make the process of installing apiman, deploying echo-service, and configuring apiman easier to understand, this article shows these high-level steps:

1. Download and build the echo-service
2. Download and start apiman
3. Configure an API provider in apiman
4. Configure an API consumer in apiman
5. Configure a defined usage policy to perform rate-limiting
6. Use the echo-service with `curl` to trigger the rate-limiting policy

## Download and build the echo-service

Echo-service is part of the apiman-quickstarts project. Run the following commands to download the echo-service.

```
mkdir ~/demo
cd ~/demo
git clone git@github.com:apiman/apiman-quickstarts
cd apiman-quickstarts
git checkout tags/apiman-plugins-1.3.1.Final -b demo
```

Build the project with maven.

```
sudo docker run -it --rm -v "$(pwd)":/mnt -w /mnt maven:3-jdk-8 mvn package
```

After Docker downloads the maven image, maven builds the project and creates the `./echo-service/target/apiman-quickstarts-echo-service-1.3.1.Final.war` file. The echo-service WAR file is used in the next step.

## Download and start apiman

Run the following command to start the apiman service.

```
sudo docker run -it --rm \
  -p 8080:8080 -p 8443:8443 \
  -v "$(pwd)":/mnt -w /mnt \
  --name apiman-demo \
  apiman/on-wildfly10:latest
```

Several lines of log messages are written to the console. After the log messages pause, start a second terminal and run the following command to copy the echo-service WAR file into the deployment directory in the apiman-demo container.

```
# Open a new terminal to run this command.
sudo docker cp \
  ./echo-service/target/apiman-quickstarts-echo-service-1.3.1.Final.war \
  apiman-demo:/opt/jboss/wildfly/standalone/deployments
```

After you run the `docker cp` command, you can close the second terminal.

At this point, the echo-service is deployed and the apiman service is running. You can access the API management interface with a browser at <http://localhost:8080/apimanui>.

## Configure an API provider

For this demonstration, Priya Pai, a developer with the Acme enterprise uses apiman to make the echo-service API available.

1. Start a browser and access the API management interface at <http://localhost:8080/apimanui>.
2. Click the **Register** link and then add a user account for the API provider by specifying the following values:

Field	Value
<b>Username</b>	ppai
<b>First name</b>	Priya
<b>Last name</b>	Pai
<b>Email</b>	<a href="mailto:ppai@acme.com">ppai@acme.com</a>
<b>Password</b>	ppai

Click the **Register** button.

- On the dashboard page, click the **Create a New Organization** link and then add an organization for Acme by specifying the following values:

Field	Value
Organization Name	Acme
Description	Provider of RESTful web services

Click the **Create Organization** button.

- On the **Plans** page for the Acme organization, click the **New Plan** button and then add a plan by specifying the following values:

Field	Value
Plan Name	gold
Description	The gold plan is subject to rate limiting

Click the **Create Plan** button.

- On the gold plan page, click the **Policies** tab on the left and then click **Add Policy**.
- On the **Add Policy** page, select Rate Limiting Policy from the **Policy Type** menu and then add a policy by specifying the following values:

Field	Value
# of requests	5
Granularity	Client App
Period	Day

The number of requests is set very low so that the limit can be triggered quickly.

Accept the default values for the other fields and click the **Add Policy** button.

- Back on the gold plan page, click the **Lock Plan** button. Locking a plan prevents future policy changes and makes the plan available for use by APIs.
- From the breadcrumb navigation at the top of the page for the gold plan, click the Acme link.
- On the Acme organization page, select the **APIs** tab and then click the **New API** button. Add an API by specifying the following values:

Field	Value
API Name	echo-service

Field	Value
<b>Description</b>	Sample API for demonstration

Click the **Create API** button.

10. Configure the API details on the `echo-service` page by performing the following steps:

1. Select the **Implementation** tab on the left and then add the implementation by specifying the following values:

Field	Value
<b>API Endpoint</b>	<a href="http://localhost:8080/apiman-echo">http://localhost:8080/apiman-echo</a>

Accept the default values for the other fields and click the **Save** button.

2. Select the **Plans** tab on the left, enable the checkbox for the `gold` plan, and click the **Save** button.
3. Select the **Policies** tab on the left, click the **Add Policy** button, and add the API policy by specifying the following values:

Field	Value
<b>Policy Type</b>	BASIC Authentication Policy
<b>Authentication Realm</b>	Echo
<b>Forward Authenticated Username as HTTP Header</b>	X-Identity
<b>Identity Source</b>	Static
<b>Username : Password</b>	user1:pass1

Click the **Add** button to add the sample user and then click the **Add Policy** button to add the API policy.

11. At the top of the `echo-service` page, click the **Publish** button.

At the top-right of the page, select **ppai > Logout**. The next steps require logging on as an API consumer.

## Configure an API consumer

Chris Choi, a developer with the Omega enterprise, uses apiman to enable a client app for the echo-service.

1. On the apiman realm page, <http://localhost:8080/apimanui>, click the **Register** link and then add a user account for the API consumer by specifying the following values:

Field	Value
<b>Username</b>	cchoi

Field	Value
First name	Chris
Last name	Choi
Email	<a href="mailto:cchoi@omega.com">cchoi@omega.com</a>
Password	cchoi

Click the **Register** button.

2. On the dashboard page, click the **Create a New Organization** link and then add an organization for Omega by specifying the following values:


Field	Value
Organization Name	Omega
Description	Consumer of RESTful web services

Click the **Create Organization** button.

3. On the Omega organization page, select the **Client Apps** tab and then click **New Client App**. Add a client app by specifying the following values:

Field	Value
Client App Name	echo
Description	Receive a response that is identical to the request

Click **Create Client App**.

4. On the echo client app page, click the **Search for APIs to consume** link.
5. On the **Find an API** page, enter `echo-service` in the search field and click **Search**. Click the **Acme / echo-service** link from the search results.
6. On the **API Details** page for `echo-service`, click the **Create Contract** button for the `gold` plan.
7. On the **New Contract** page, accept the default values and click the **Create Contract** button.
8. Back on the echo client app page, click the **Register** button.
9. While you are still on the echo client app page, select the **APIs** tab. The table has a row for the **Acme / echo-service** API. Click the  button to view the API endpoint. Copy the URL from the **As Query Parameter** field. The value is similar to the following example:

```
https://localhost:8443/apiman-gateway/Acme/echo-service/1.0?
apikey=23945...591e427
```

At the top-right of the page, select **cchoi > Logout**. The next step is to use `curl` with the URL that you copied to access the echo-service. You can experience the rate-limiting policy after making several HTTP requests.

## Trigger the rate-limiting policy

The last step is to use `curl` with the echo-service to trigger the rate-limiting policy.

Create a sample file for the HTTP request payload.

```
cat << EOF > /tmp/payload.json
{"message": "hello apiman"}
EOF
```

The echo-service accepts the body of the HTTP request, hashes it with the SHA1 hash function, and then returns the hash value. You can calculate the hash with the `sha1sum` command. The hash value is `b4ff397ff32fcef37e89cad6086422b88142423`.

```
echo -n '{"message": "hello apiman"}' | sha1sum
```

Run the following command to use the echo-service. Use the URL that you copied from the echo client app page.

```
curl -k -u user1:pass1 -H "Content-Type: application/json" \
  --data @/tmp/payload.json \
  https://localhost:8443/apiman-gateway/Acme/echo-service/1.0?apikey=2392e3a9-
6d06-48b0-a290-601fa591e427
```

The service returns a response that is similar to the following. The value for the `bodySha1` field matches the output from the `sha1sum` command.

```
{
  "method": "POST",
  "resource": "/apiman-echo",
  "uri": "/apiman-echo",
  "headers": {
    "Accept": "*/*",
    "Connection": "Keep-Alive",
    "User-Agent": "curl/7.64.0",
    "X-Identity": "user1",
    "Host": "localhost:8080",
```

```
"Accept-Encoding": "gzip",
"Content-Length": "27",
"Content-Type": "application/json"
},
"bodyLength": 27,
"bodySha1": "b4ff397ff32fced37e89cad6086422b88142423"
}
```

Run the `curl` command five more times to trigger the rate limit. When you trigger the limit, apiman no longer sends your request to the echo-service. Instead, apiman provides a response that is similar to the following.

```
{
  "type": "Other",
  "failureCode": 10005,
  "responseCode": 429,
  "message": "Rate limit exceeded.",
  "headers": {
    "X-RateLimit-Limit": "5",
    "X-RateLimit-Remaining": "-1",
    "X-RateLimit-Reset": "40544"
  }
}
```

## Key benefits and next steps

This article shows how apiman enables the Acme enterprise to provide and manage a sample API. The Omega enterprise is able to discover and consume the sample API.

This article showed a few shortcuts with apiman that are easily fixed:

- apiman is easily configured to use LDAP so that users do not need to register themselves
- apiman supports several popular databases for persistence
- apiman supports deploying several API gateways for high-volume APIs

Some next steps that enterprises will want to take with apiman:

- Integrate security with a popular protocol such as OAuth
- Collect metrics for API consumption to initiate billing

For information about important concepts like organizations, policies, client apps, and so on, refer to the apiman documentation:

- [Installation Guide](#)
- [User Guide](#)
- [Production Guide](#)