



Apiman article

John-Paul Doran

Table of Contents

1. Abstract	1
2. OVERVIEW	2
3. ADMINISTERING WILDFLY 10 AND APIMAN	3
3.1. PREREQUISITES	3
3.2. INSTALL AND CONFIGURE APIMAN ON WILDFLY 10	3
3.3. CREATE A NEW USER IN WILDFLY 10	3
3.4. STARTING THE SERVER	4
3.5. INSTALL ECHO-SERVICE QUICKSTART	5
4. APIMAN USE CASE	6
4.1. CREATE PROVIDER AND CONSUMER USER PROFILES	7
4.2. CONFIGURATION FOR THE PROVIDER	8
4.2.1. Configure the Provider's Organization, Plan and Policy	8
4.2.2. Create a new API	10
4.3. CONFIGURATION FOR THE CONSUMER	12
4.3.1. Register the application with the API Gateway	15
4.3.2. Test the rate limiting policy	16
5. REFERENCE	16

1. Abstract

This article provides a basic example of Apiman, and WildFly application server. Based on echo-service quickstart, demonstrating how to configure an API Provider and Consumer, through the API Manager. And test a defined usage policy, through the API Gateway.

2. OVERVIEW

In an ever-growing world of modern technology, from Cloud-based services to IoT. There has never been a greater need to secure services. API Management is becoming a necessity to secure business assets. The vulnerabilities that can arise from malicious hackers. If they can gain access to APIs. Could have a serious impact on an organization. With a powerful Open Source API Management platform, such as Apiman. It makes it possible, to secure these assets. Through the use of an API Manager, with a rich user interface. The Governance of APIs is straightforward. Through a provider role. Organizations created, have Plans and Policies applied to them. Providing authentication, and IP filtering for security. These Policies are then Governed by the API Gateway at runtime. Apiman has a quota-based management system, for API requests. This provides metric based statistics. Which allows for the Provider, to adjust APIs, to support consumer trends such as that of application developers.

3. ADMINISTERING WILDFLY 10 AND APIMAN

To run this demonstration. You will need to install Java JDK, and Apache Maven. Download and install WildFly 10 application server, and the latest version of Apiman.

3.1. PREREQUISITES

To build and run the echo-service quickstart demonstration, install the following prerequisites:

- A supported version of the Java Developer Kit (JDK). See the Supported [Configurations](#) page for details.

```
$ sudo yum install -y java-1.8.0-openjdk-devel
```

- Apache Maven 3.3.x or later. See the Maven [Installation](#) page.

3.2. INSTALL AND CONFIGURE APIMAN ON WILDFLY 10

Apiman uses WildFly 10 as a runtime environment. This section outlines how to download and unarchive both Apiman and WildFly 10 on Fedora Linux.

1. Download the **.zip** archive files:

- [WildFly 10.1.0.Final](#)
- [Apiman 1.4.3.Final overlay for WildFly 10](#)
- Alternatively download the **.zip** files using **cURL**:

```
$ curl http://download.jboss.org/wildfly/10.1.0.Final/wildfly-10.1.0.Final.zip  
-o wildfly-10.1.0.Final.zip  
$ curl http://downloads.jboss.org/apiman/1.2.9.Final/apiman-distro-wildfly10-  
1.2.9.Final-overlay.zip -o apiman-distro-wildfly10-1.2.9.Final-overlay.zip
```

2. Unzip each **.zip** archive file into the **wildfly-10.1.0.Final** directory:

```
$ unzip wildfly-10.1.0.Final.zip  
$ unzip -o apiman-distro-wildfly10-1.2.9.Final-overlay.zip -d wildfly-10.1.0.Final
```

3.3. CREATE A NEW USER IN WILDFLY 10

- WildFly does not provide a default password. Create a new user to access the console:

```
$ cd path/to/work/dir/wildfly-<version>/bin  
$ ./add-user.sh
```

- Create your **username**, and **password**.
- Select the type of user **a) Management User** or **b) Application User**.
- Choose a **group**, or default to blank and click enter.

For more detailed instructions see WildFly Server [Administration Guide](#).

NOTE

For this demonstration. We use the username: **Admin** and password: **Admin123** with default values, or select **yes**.

3.4. STARTING THE SERVER

This section outlines how to start the Wildfly server and standalone version of Apiman. Access the Management UI, and shutdown the server when finished.

1. Start the Wildfly server and Apiman:

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

2. Access <http://localhost:8080/apimanui/> in your browser.
3. Stop the Wildfly server and Apiman:

```
$ ./bin/jboss-cli.sh --connect command=:shutdown
```

NOTE

Press **Ctrl+C** on the keyboard to stop the server.

3.5. INSTALL ECHO-SERVICE QUICKSTART

1. Download quickstarts into the **apiman** directory:

```
$ cd /wildfly-10.1.0.Final/apiman/  
$ git clone https://github.com/apiman/apiman-quickstarts.git
```

```
[john@fedoralinux apiman]$ git clone https://github.com/apiman/apiman-quickstarts.git  
Cloning into 'apiman-quickstarts'...  
remote: Counting objects: 556, done.  
remote: Total 556 (delta 0), reused 0 (delta 0), pack-reused 556  
Receiving objects: 100% (556/556), 75.60 KiB | 766.00 KiB/s, done.  
Resolving deltas: 100% (268/268), done.
```

Figure 1. Clone Quickstarts

2. Echo quickstarts directory structure

```
echo-service/  
├── pom.xml  
├── README.md  
├── src  
│   ├── main  
│   │   ├── java  
│   │   │   ├── io  
│   │   │   │   ├── apiman  
│   │   │   │   │   ├── quickstarts  
│   │   │   │   │   │   ├── echo  
│   │   │   │   │   │   │   ├── EchoResponse.java  
│   │   │   │   │   │   │   └── EchoServlet.java  
│   │   └── webapp  
│   │       ├── WEB-INF  
│   │       │   ├── jboss-web.xml  
│   │       │   └── web.xml  
└── target  
    └── apidocs
```

1. Change to the **echo-service** directory, and build the API, with Maven:

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

```
o-service/target/apiman-quickstarts-echo-service-1.3.1.Final-javadoc.jar  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 29.202 s  
[INFO] Finished at: 2018-07-25T20:21:37+01:00  
[INFO] -----  
[john@fedoralinux echo-service]$
```

Figure 2. Successful Maven build

4. APIMAN USE CASE

This section outlines how to create new users, for API service **Provider** and **Consumer** roles. It demonstrates how to configure basic authentication Policies, a username, and password, to get access to an API.

Some of the main concepts of Apiman, used in this demonstration, are Policies, Organization, Plans, and Applications:

- You can apply **Policies** to applications, plans, and services. This ensures the governance of the services. You can apply Policies to API requests at runtime, by the API Gateway.
- An **Organization** is a Role based container, for all entities, such as policies plans and APIs.
- A **Plan** allows for multiple levels of access to a service. By applying multiple policies to the plan.
- Client **Applications** consume APIs. Users can create Contracts to set up communication between the application and service.

Table 1. Overview of the API Managers and API Gateways function

API Manager	API Gateway
Configures Policies for Applications, Plans and Services	The API Gateway enforces configured Policies at runtime
Provider and Consumer roles can manage and configure APIs. through a UI	Security, Limiting, and Modification policies are applied

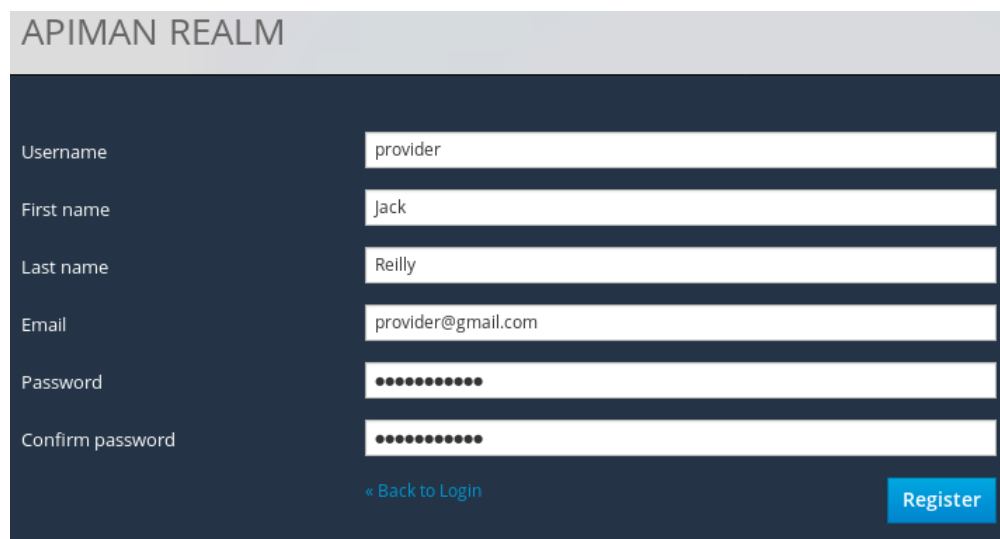
Table 2. Outline of the Provider and Consumer use case for this demonstration

Provider Role	Consumer Role
Creates the organization ABC APIs	Creates the organization AJAX API Consumer
Creates a Gold plan	Create a client application eat-echo
Applies a rate-limiting policy	Consumes the echo API
Creates the API called echo	Apply a contract
Applies the APIs endpoint	Register the application with the API Gateway
Adds an authentication policy	Selects which plan to apply
Publish the API	Create a client application eat-echo to consume API services, Governed by a Contract.

4.1. CREATE PROVIDER AND CONSUMER USER PROFILES

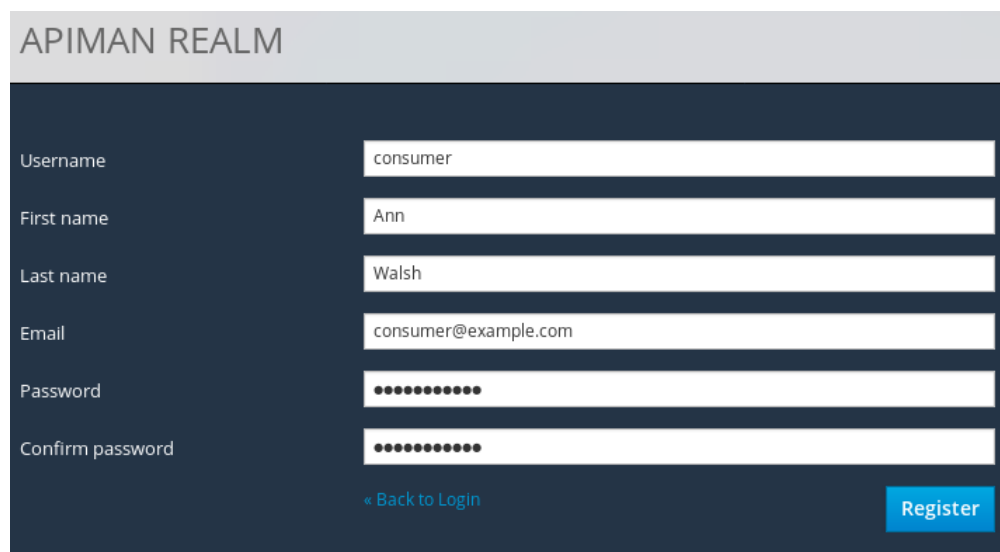
Create Provider and Consumer user accounts in the API Manager UI:
<http://localhost:8080/apimanui/>

- Select **New user/Register**, to register, the new users:



The screenshot shows the 'APIMAN REALM' registration page. It features a dark blue header with the text 'APIMAN REALM'. Below the header, there is a form with the following fields: Username (filled with 'provider'), First name (filled with 'Jack'), Last name (filled with 'Reilly'), Email (filled with 'provider@gmail.com'), Password (masked with dots), and Confirm password (masked with dots). At the bottom of the form, there is a link '« Back to Login' and a blue 'Register' button.

Figure 3. Provider account



The screenshot shows the 'APIMAN REALM' registration page. It features a dark blue header with the text 'APIMAN REALM'. Below the header, there is a form with the following fields: Username (filled with 'consumer'), First name (filled with 'Ann'), Last name (filled with 'Walsh'), Email (filled with 'consumer@example.com'), Password (masked with dots), and Confirm password (masked with dots). At the bottom of the form, there is a link '« Back to Login' and a blue 'Register' button.

Figure 4. Consumer account

4.2. CONFIGURATION FOR THE PROVIDER

4.2.1. Configure the Provider's Organization, Plan and Policy

- Login to the Provider account. Select **Create a New Organization**.

Organization Name

ABC APIs

Description

ABC provider of great APIs.

Create Organization

Cancel

Figure 5. Create the Organization, for the API Provider.

- Select **Plans** from the menu. Click **New Plan** and enter plan details.

Organization

ABC APIs ▾

/

Plan Name

Gold

Initial Version

1.0

Description

Gold standard plan.

Create Plan

Cancel

Figure 6. Create a gold plan

- Select **Policies** from the menu. Click **Add Policy** and add the policy configuration.

Policy Type

 Rate Limiting Policy ▼

Rate Limiting Policy Configuration

I want to limit request rates to requests per Client App ▼ per Day ▼

Configure the rate limiting related response headers below - these headers will convey useful information to clients such as imposed limits and when the rate period will be reset. You may override the default header names by supplying your own in the fields below (or leave them blank to accept the defaults).

Limit Response Header

Remaining Response Header



Reset Response Header

Add Policy

Cancel

Figure 7. Create a gold plan policy

- Click **Lock Plan** in the header


Version: 1.0 ▼
New Version


Description: Gold standard plan.

Created on 2018-07-24

Created by [provider](#)

Status: LOCKED

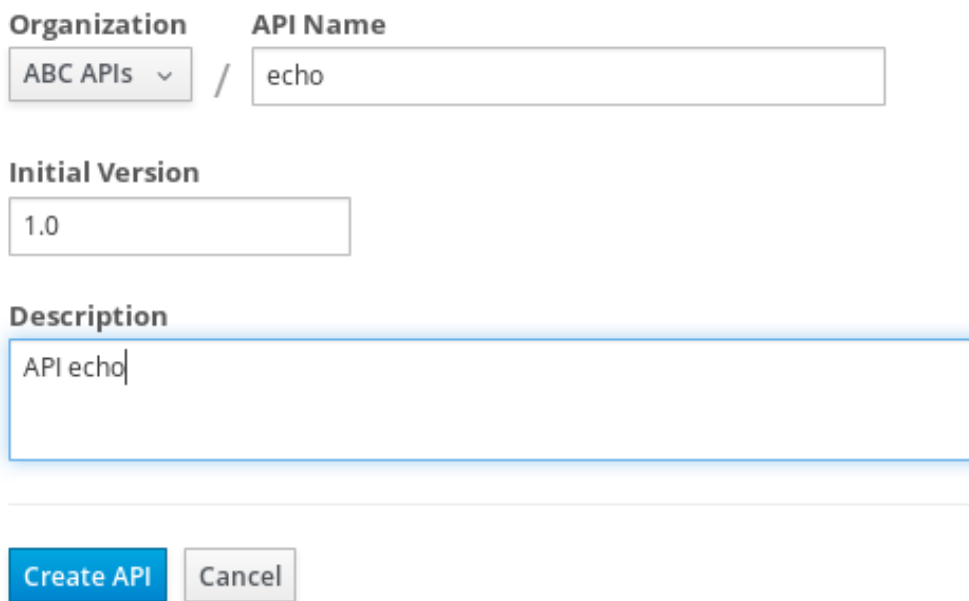
[Create a new version of this Plan \(New Version\)](#)

Figure 8. Lock the plan

NOTE Create multiple plans the same way as required.

4.2.2. Create a new API

- Select **APIs** from the menu. Click **New API** to add the API configuration.



Organization ABC APIs / **API Name** echo

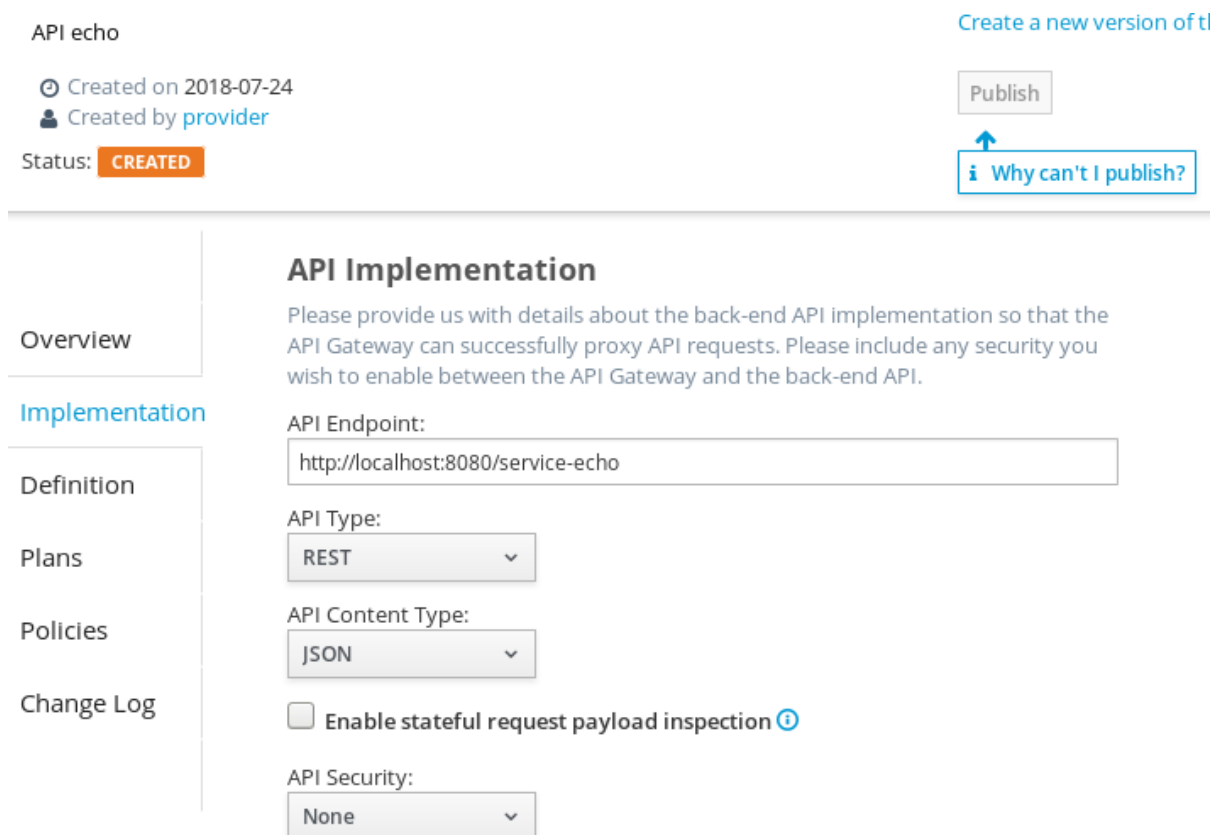
Initial Version 1.0

Description API echo

Create API Cancel

Figure 9. Create a new API

- Select **Implementation** from the menu. Add details for the back-end API. Click **Save**.



API echo [Create a new version of this API](#)

Created on 2018-07-24
Created by provider

Status: **CREATED** [Why can't I publish?](#)

API Implementation

Please provide us with details about the back-end API implementation so that the API Gateway can successfully proxy API requests. Please include any security you wish to enable between the API Gateway and the back-end API.

API Endpoint:

API Type:

API Content Type:

☐ Enable stateful request payload inspection ⓘ

API Security:

Overview
Implementation
Definition
Plans
Policies
Change Log

Figure 10. Create the APIs Endpoint

- Select **Plans** from the menu. Choose the Plan required. Click **Save**.

Public API

Select this option if you wish this API to be accessible directly, without an API Contract. Typically (but not always) this option is used instead of selecting plan(s).

☐ Make this API public

Available Plans

Choose which plans should be presented when Client Apps create a link (Contract) to this API. Note that only plans in a 'Locked' state show up in this list.

<input checked="" type="checkbox"/> gold	1.0
<input type="checkbox"/> Silver	1.0

Save **Cancel**

Figure 11. Choose a Plan

- Select **Policies** from the menu. Choose a **Policy Type**. Add a descriptive **Realm** name. Add an **Identity. Source** and create a new User.

Identity Source

Static

The "static" identity source is typically only useful for testing - you probably don't want to use it in production!

Static Identities

Clear


Remove

user : Add

Figure 12. Add an authentication policy

- Click **Publish**. At the top of the page, to publish the API.

Home » ABC APIs » echo



Version: echo ▼ New Version ⚙️ ▼

Demo

Created on 2018-07-24
Created by provider

Status: PUBLISHED

Link my Client App to this API (New Contract)

Create a new version of this API (New Version)

Retire

Overview

Implementation


Definition

Plans

API Policies

Here is a list of all Policies defined for this API. These Policies will be applied to all invocations of this API by any Client App, regardless of the Plan used in its Contract.

⋮



BASIC Authentication Policy

Policy created by provider on 2018-07-25

Access to the API is protected by BASIC Authentication through the 'Echo' authentication realm.

Remove

Figure 13. Publish the API

4.3. CONFIGURATION FOR THE CONSUMER

- Login to the Consumer account. Select **Create a New Organization**.

Organization Name

AJAX API Consumer

Description

Manage APIs and Client Apps.

Create Organization

Cancel

Figure 14. Create an API consumer organization

- Select **Client App** from the menu. Click **New Client App**.

Organization

AJAX API Consumer ▼

Client App Name

/ eat-echo

Initial Version

1.0

Description

Consume echo.

Create Client App

Cancel

Figure 15. Create a client application

- Select **Search for APIs to consume**. Enter the name of the API. Click **Search**.

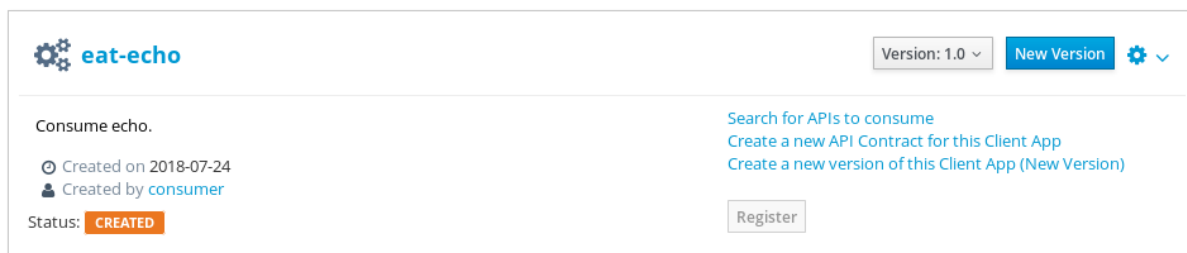


Figure 16. Search for APIs to consume

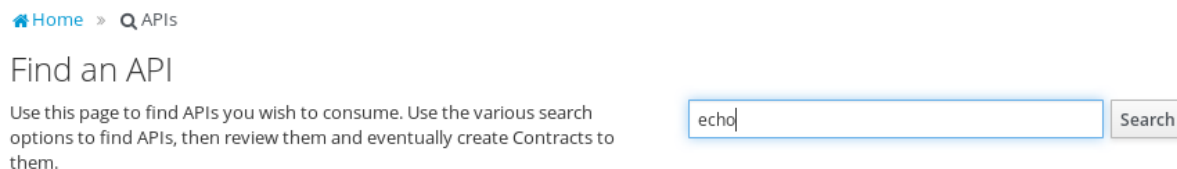


Figure 17. Search for echo

- Select **Contracts** from the menu. Click **Create Contract** to add the Client contract to the API.

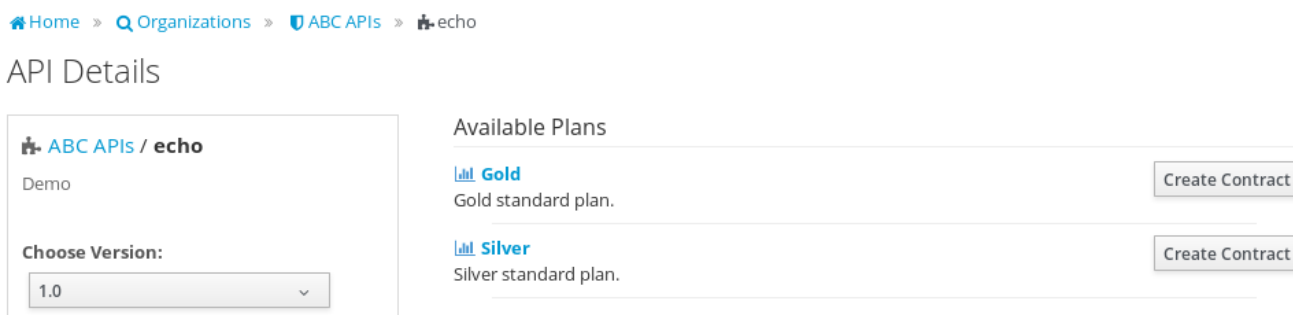


Figure 18. Client application details

- Select **Policies** from the menu. Choose a **Policy Type**.

The figure illustrates a three-step process for creating a new API contract, connected by downward arrows:

- From Client App**: The Client App that will be used as the source of the new API Contract. Choose one of your available Client Apps below, and then choose a Client App version.
AJAX API Consumer / eat-echo ▾ 1.0 ▾
- Using Plan**: Use the drop-down below to choose one of the Plans made available by the selected API.
Gold ▾
- To API**: Use this section to choose what API the Client App will be consuming (aka the "target" of this API Contract).
ABC APIs / echo ⇄ 1.0
(click to change)

At the bottom of the form are two buttons: **Create Contract** and **Cancel**.

Figure 19. Create a new contract

4.3.1. Register the application with the API Gateway

- Select **Contracts** from the menu. Click **Register** the application.

Home > AJAX API Consumer > eat-echo

eat-echo Version: 1.0 [New Version](#)

Consume echo.

Created on 2018-07-24
Created by consumer

Status: **REGISTERED** [Re-Register](#) [Unregister](#)

[Search for APIs to consume](#)
[Create a new API Contract for this Client App](#)
[Create a new version of this Client App \(New Version\)](#)

API Contracts

Here is a list of all APIs that this Client App is currently contracted to utilize. This provides a list of all APIs that Client App can potentially invoke.

Filter by org or API name... [Break All](#) [New Contract](#)

[ABC APIs / echo](#)
API version 1.0 via plan Gold entered into on 2018-07-24
Demo

Figure 20. Register the client application

- Select **APIs** from the menu. Click on the **Information** icon. To display the API key and endpoint.

Client App APIs

Below is a list of all the APIs this Client App consumes. This information is derived from the set of API Contracts the Client App has entered into. Manage these Contracts by switching to the "Contracts" tab.

[Download as JSON](#) [Download as XML](#)

API	Version	Plan
ABC APIs / echo	1.0	Gold

Figure 21. Client applications API

Copy API Endpoint

To successfully invoke the managed API for this API contract, you must provide the appropriate API Key with each request. The API Key can be provided either by sending it as an HTTP Request Header named X-API-Key, or you can send it as a URL query parameter.

As Query Parameter
`https://localhost:8443/apiman-gateway/ABCAPIS/echo/1.0?apikey=4eeb2fa1-b3ee-45f6-af4t`

As HTTP Request Header
`X-API-Key: 4eeb2fa1-b3ee-45f6-af4b-823f771a366a`

[Done](#)

Figure 22. API key and endpoint

4.3.2. Test the rate limiting policy

The **rate limiting policy** is configured to provide authentication and to limit the client application to ten requests per day.

- Copy the displayed endpoint into a browser to test authentication and login.

```
https://localhost:8443/apiman-gateway/ABCAPIs/echo/6.0?apikey=0eeba35e-b1f8-4265-ab47-ed440bac83bf
```

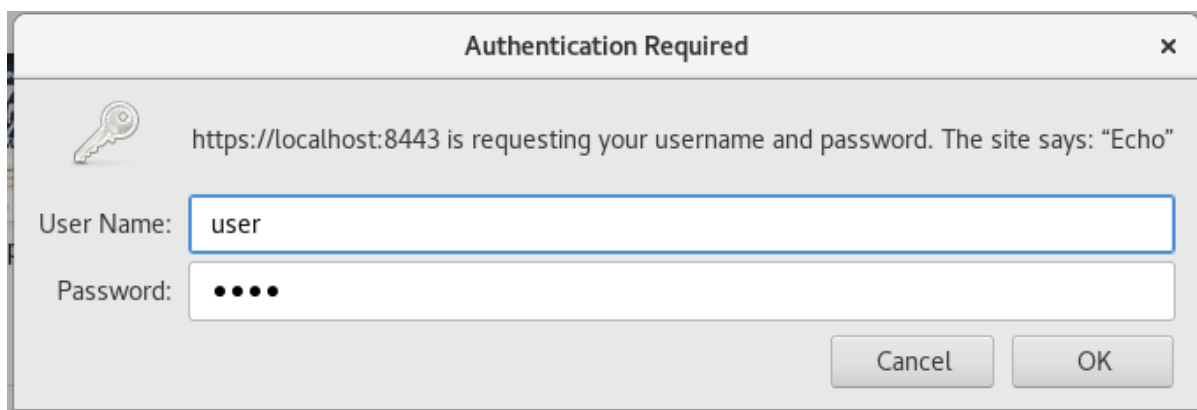


Figure 23. Authentication show this Policy works

5. REFERENCE

Apiman: <http://www.apiman.io/>