API Management

Introduction

Apiman is the JBoss open source API management solution developed by Red Hat. It mainly consists of two components: API Manager and API Gateway.

The **API Manager** is a component that allows API creation and administrative activities, including:

- API/service providers use a web UI to define service contracts for their APIs;
- apply these contracts across multiple APIs;
- control role-based user access and API versioning;
- contracts govern access to services and limits on the rate at which consumers can access services;
- the web UI enables API consumers to easily locate and access APIs.

The **API Gateway** is the runtime where API implementations are deployed and exposed to consumers. It works the way that the consumer of the service accesses the service through an URL that designates the API Gateway as a proxy for the service. Once policies defined to govern access to the service, the API Gateway then proxies requests to the service's backend API implementation.

Implementation

Java v 1.7 or newer, git and maven should be installed on the system before apiman implementation.

Implementing the solution on a production server, don't forget to change the OOTB default admin username and/or password. Apiman is configured by default to use JBoss KeyCloak http://keycloak.jboss.org/ for password security. The H2 database is used by default and should be reconfigured to a production database. Note: apiman includes DDLs for both MySQL and PostgreSQL.

For the purpose of demo, the default configuration will be used.

First, a client app server is needed on which apiman will be installed and run. An open source JBoss WildFly server release 8.2 http://www.wildfly.org/ will be used in the example.

To install WildFly, simply download http://download.jboss.org/wildfly/10.1.0.Final/wildfly-10.1.0.Final.zip and unzip the file into the directory in which sever will be run.
Then, download the apiman 1.0 WildFly overlay zip file inside the directory that was created when you un-zipped the WildFly download. The apiman 1.0 WildFly overlay zip file is available here: http://downloads.jboss.org/apiman/1.3.1.Final/apiman-distro-wildfly10-1.3.1.Final-overlay.zip

To do so, use following commands:

```
mkdir ~/apiman-1.3.1.Final
cd ~/apiman-1.3.1.Final
unzip wildfly-10.1.0.Final.zip
unzip -o apiman-distro-wildfly10-1.3.1.Final-overlay.zip -d wildfly-10.1.0.Final
```

To start the server, execute these commands:

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

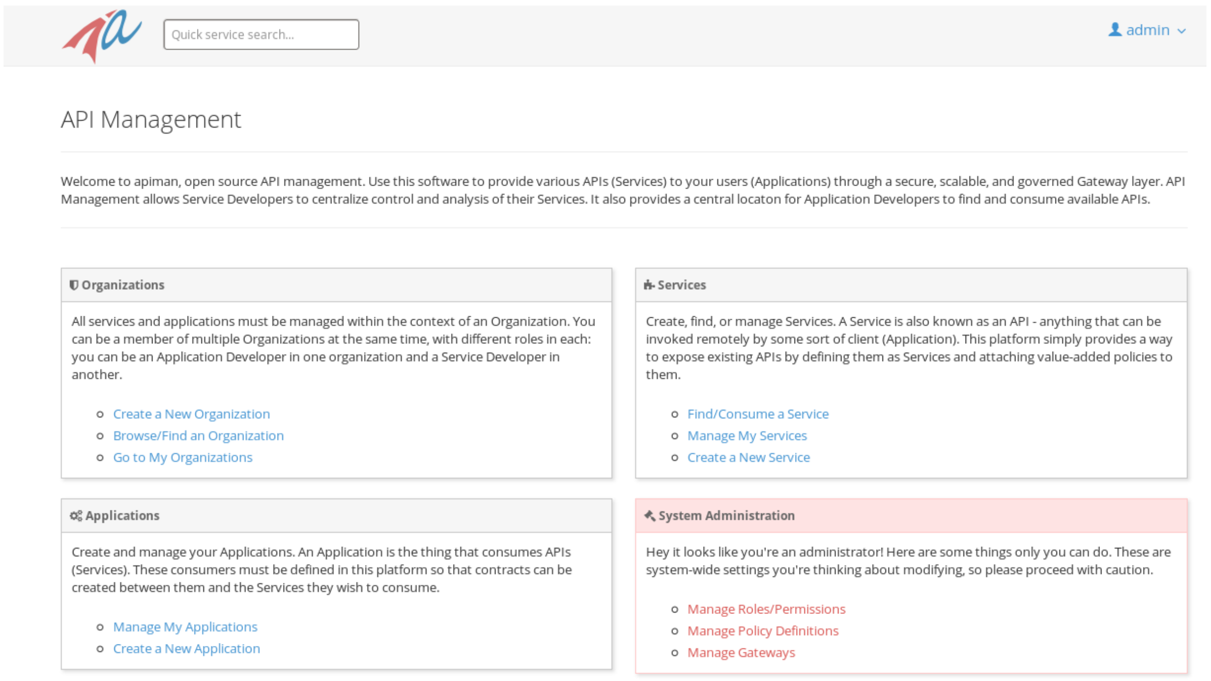
The server will write logging messages to the screen. When some messages similar to this appear, that means the server is up and running with apiman installed:

```
13:57:03,229 INFO  [org.jboss.as.server] (ServerService Thread Pool -- 29) JBAS018559: Deployed "apiman-ds.xml" (runtime-name : "apiman-ds.xml")
13:57:03,261 INFO  [org.jboss.as] (Controller Boot Thread) JBAS015961: Http management interface listening on <a href="http://127.0.0.1:9990/management">http://127.0.0.1:9990/management</a>
13:57:03,262 INFO  [org.jboss.as] (Controller Boot Thread) JBAS015951: Admin console listening on <a href="http://127.0.0.1:9990">http://127.0.0.1:9990</a>
13:57:03,262 INFO  [org.jboss.as] (Controller Boot Thread) JBAS015874: WildFly 8.2.0.Final
```

```
"Tweek" started in 5518ms - Started 754 of 858 services (171 services are lazy, passive or on-demand)
```

> *Note:To access apiman's API Manager UI, go to: http://localhost:8080/apiman-manager, and log in. The username "admin" and a password "admin123!" will be used.*

You should see a screen similar to this:



API Management

Welcome to apiman, open source API management. Use this software to provide various APIs (Services) to your users (Applications) through a secure, scalable, and governed Gateway layer. API Management allows Service Developers to centralize control and analysis of their Services. It also provides a central locaton for Application Developers to find and consume available APIs.

Organizations

All services and applications must be managed within the context of an Organization. You can be a member of multiple Organizations at the same time, with different roles in each: you can be an Application Developer in one organization and a Service Developer in another.

- Create a New Organization
- Browse/Find an Organization
- Go to My Organizations

Services

Create, find, or manage Services. A Service is also known as an API - anything that can be invoked remotely by some sort of client (Application). This platform simply provides a way to expose existing APIs by defining them as Services and attaching value-added policies to them.

- Find/Consume a Service
- Manage My Services
- Create a New Service

Applications

Create and manage your Applications. An Application is the thing that consumes APIs (Services). These consumers must be defined in this platform so that contracts can be created between them and the Services they wish to consume.

- Manage My Applications
- Create a New Application

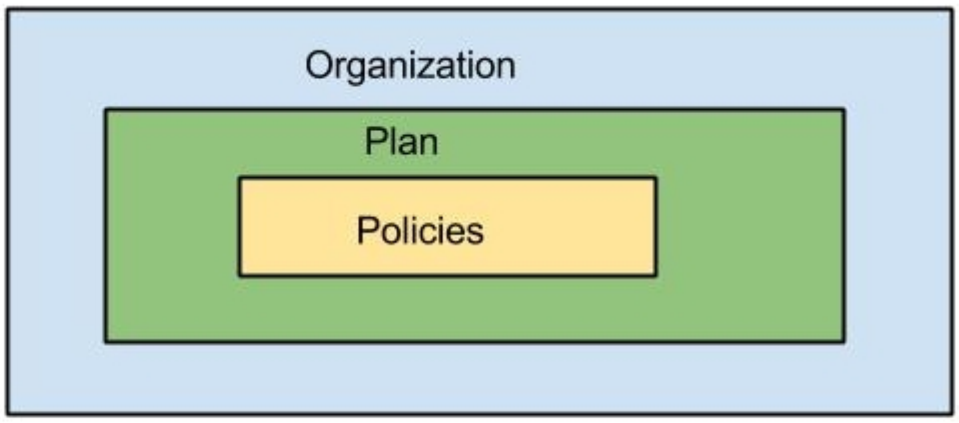
System Administration

Hey it looks like you're an administrator! Here are some things only you can do. These are system-wide settings you're thinking about modifying, so please proceed with caution.

- Manage Roles/Permissions
- Manage Policy Definitions
- Manage Gateways

Before using apiman, you should understand how apiman defines APIs and the meta data, on which they depend, are organized.

Apiman uses a hierarchical data model that consists of three elements: Policies, Plans, and Organizations:



Policies are the key construct for specifying the required behaviors of an API. They are the basis on which the higher level elements of the data model are built. Apiman uses them to enforce API consumer behavior by mandating security or entitlement. They can be applied to either client app level, individual APIs or Plans, giving the API provider flexibility on creating common policies across their organization.

There are several out-of-the-box policies, such as Authorization Policy, Basic Authentication Policy, Quota Policy and Rate Limiting Policy. But API providers can also create their own using plugins and some additional policies such as XML to JSON conversion.

Plans are set of policies, in which quotas, authorization, caching, whitelist and so on can be defined.

Organizations stand on a top level of apiman data model. An organization acts as a container for all other elements and its construct allows different departments or even multiple API providers to coexist in the same Apiman instance.

Example API deployment

Download the source code of example API from git:

```
git clone git@github.com:apiman/apiman-quickstarts.git
```

That API performs to echo back in responses the meta data in the REST (http://en.wikipedia.org/wiki/Representational_state_transfer) requests that it receives.

Maven is used to build the API. To build the API into a deployable .war file, navigate to the directory into which you downloaded the API example:

```
cd apiman-quickstarts/echo-api
mvn package
```

After package was successfully built in the output will be:

```
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
```

And also there will be a full path to .war file:

```
/jboss/local/redhat_git/apiman-quickstarts/echo-api/target/apiman-quickstarts-echo-
api-1.0.1-SNAPSHOT.war
```

To deploy the API the .war file should be copied to WildFly server's deployment directory. After that the WildFly server generates an output similar to this:

```
16:54:44,313 INFO  [org.jboss.as.server.deployment] (MSC service thread 1-7)
JBAS015876: Starting deployment of "apiman-quickstarts-echo-api-1.0.1-
SNAPSHOT.war" (runtime-name: "apiman-quickstarts-echo-api-1.0.1-SNAPSHOT.war")
16:54:44,397 INFO  [org.wildfly.extension.undertow] (MSC service thread 1-16)
JBAS017534: Registered web context: /apiman-echo
16:54:44,455 INFO  [org.jboss.as.server] (DeploymentScanner-threads - 1) JBAS018559:
Deployed "apiman-quickstarts-echo-api-1.0.1-SNAPSHOT.war" (runtime-name : "apiman-
quickstarts-echo-api-1.0.1-SNAPSHOT.war")
```

This output indicates that the URL of the deployed example API is:

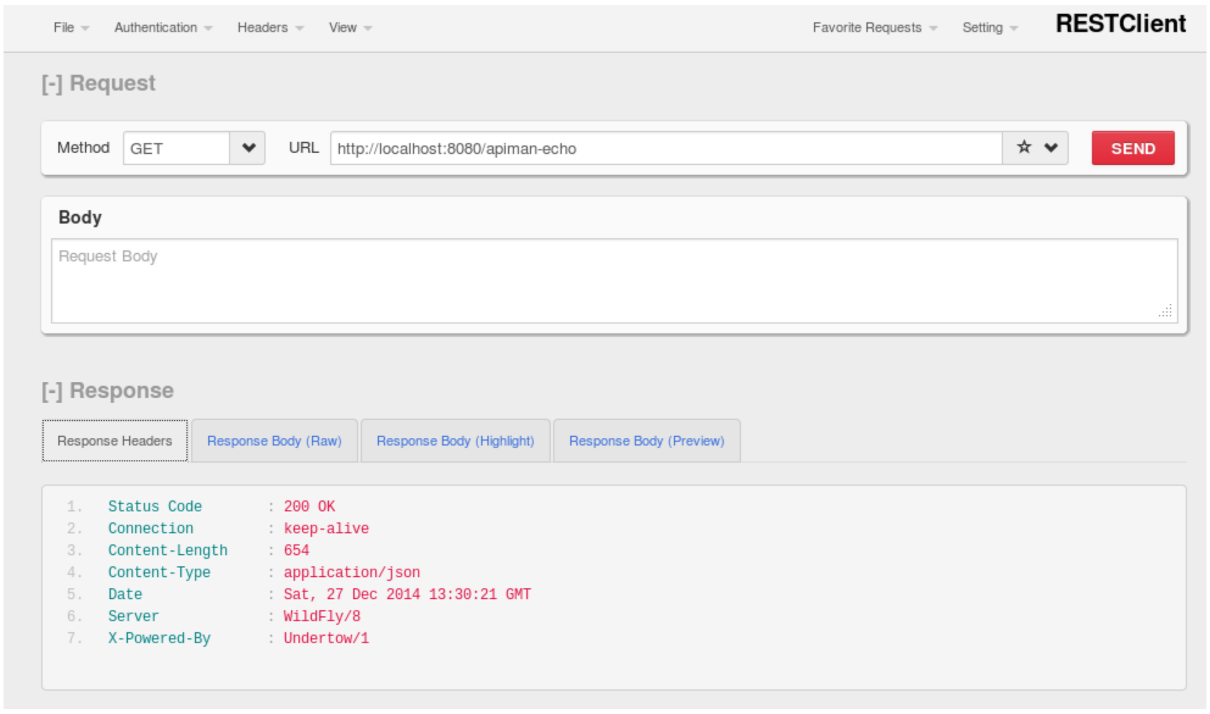
```
[a href="http://localhost:8080/apiman-echo" style="text-decoration: none;"]http://localhost:8080/apiman-echo
```

The deployed example API has that URL in case it’s accessed directly, without going through the API Gateway. The URL to access the API through the API Gateway at runtime will be different.

Accessing and Configuring the Example API via Client App

The Mozilla Firefox add-on (https://addons.mozilla.org/en-US/firefox/addon/restclient/versions/2.0.3) will be used to access API via Client App.

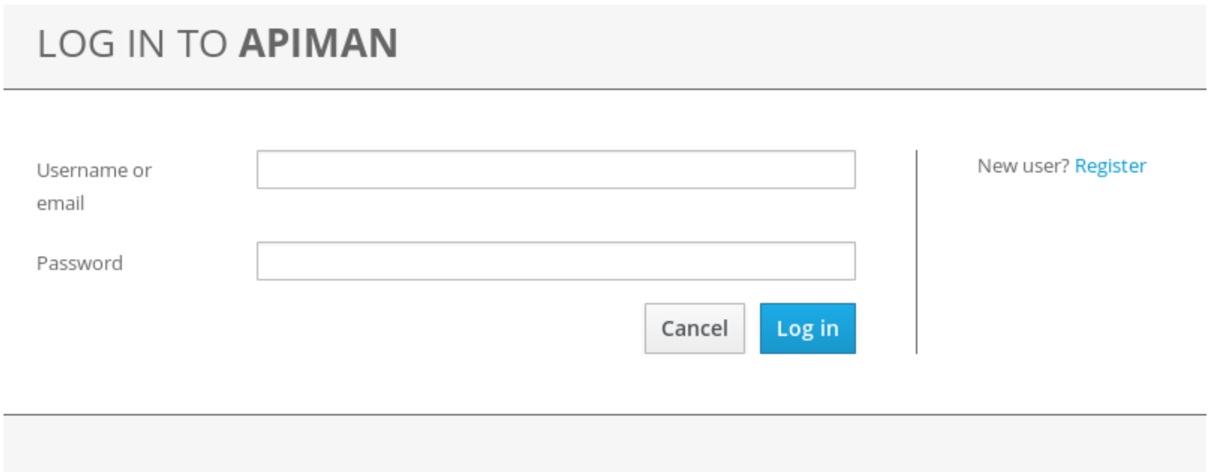
After installation the client into Firefox, the deployed API is accessed by the URL *http://localhost:8080/apiman-echo*. By executing GET command, there will be output similar to this:



```
Status Code      : 200 OK
Connection       : keep-alive
Content-Length   : 654
Content-Type     : application/json
Date             : Sat, 27 Dec 2014 13:30:21 GMT
Server           : WildFly/8
X-Powered-By     : Undertow/1
```

API Provider and Consumer Users Creation

To create API Provider user logout from admin account in API Manager UI:



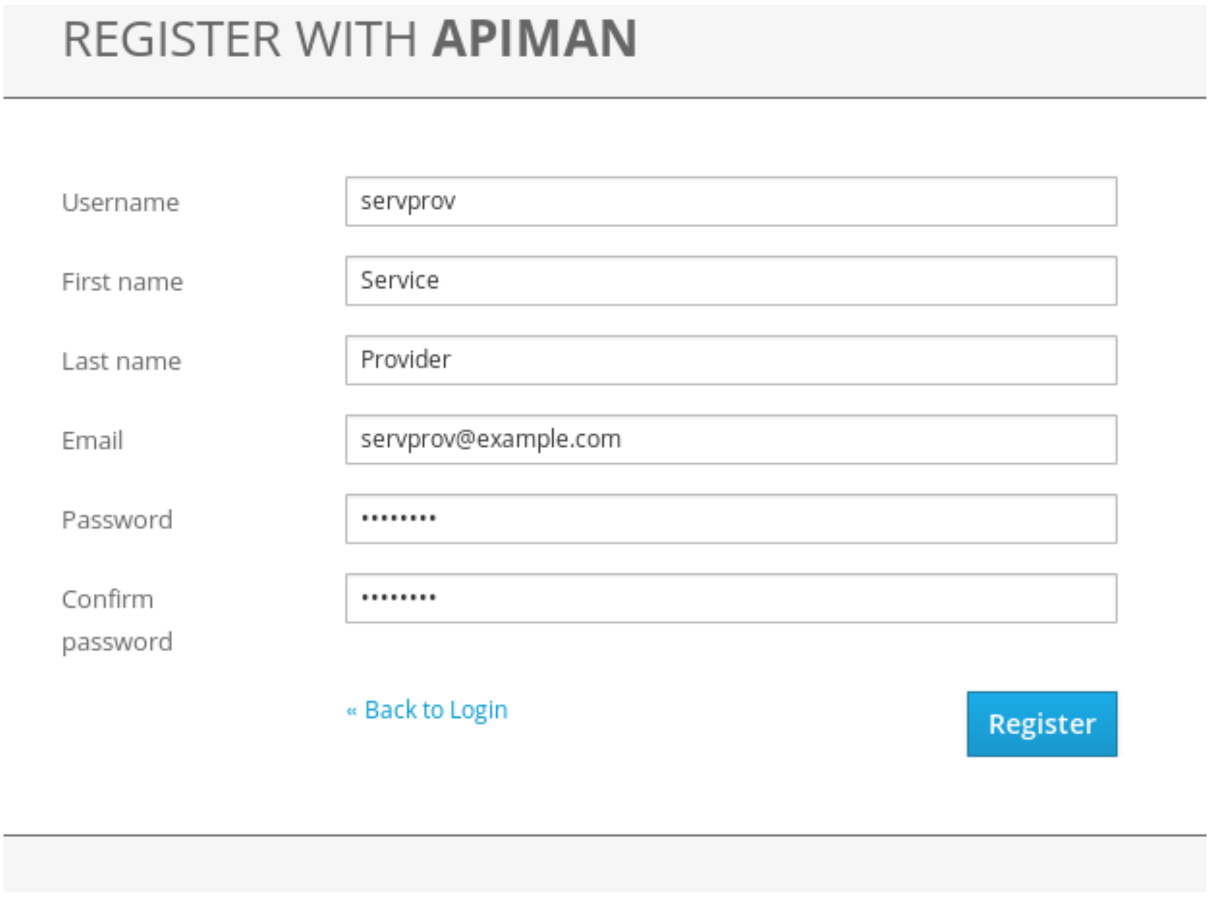
LOG IN TO **APIMAN**

Username or email	
Password	

Cancel · Log in

New user? Register

Select “New user? Register” and create a new API Provider user:

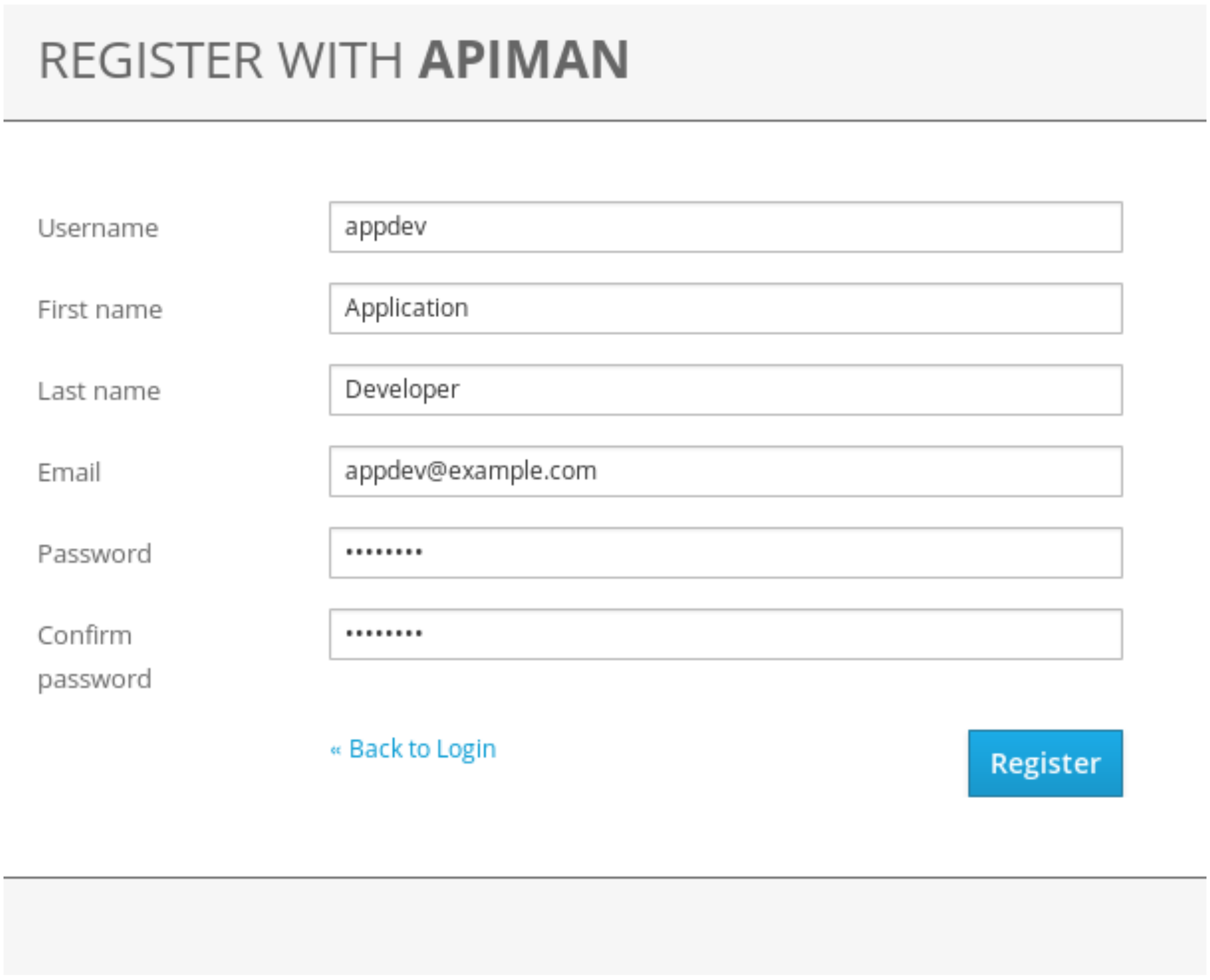


REGISTER WITH **APIMAN**

Username	servprov
First name	Service
Last name	Provider
Email	servprov@example.com
Password	•••••••••
Confirm password	•••••••••

« Back to Login · Register

Then, logout and repeat the new user registration for Consumer:

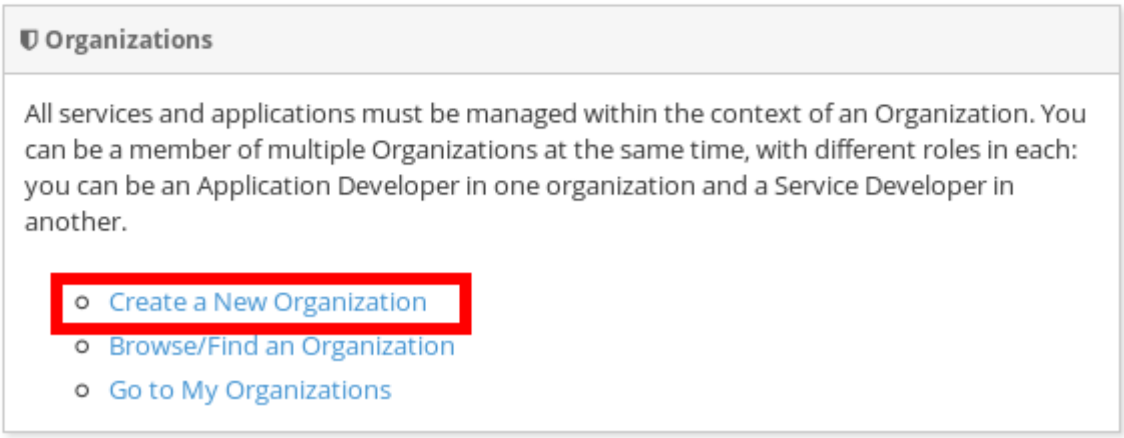


REGISTER WITH **APIMAN**

Username	appdev
First name	Application
Last name	Developer
Email	appdev@example.com
Password	•••••••••
Confirm password	•••••••••

Organization creation

To create the API producer organization, log back into the API Manager UI as the servprov user and select “Create a new Organization”:

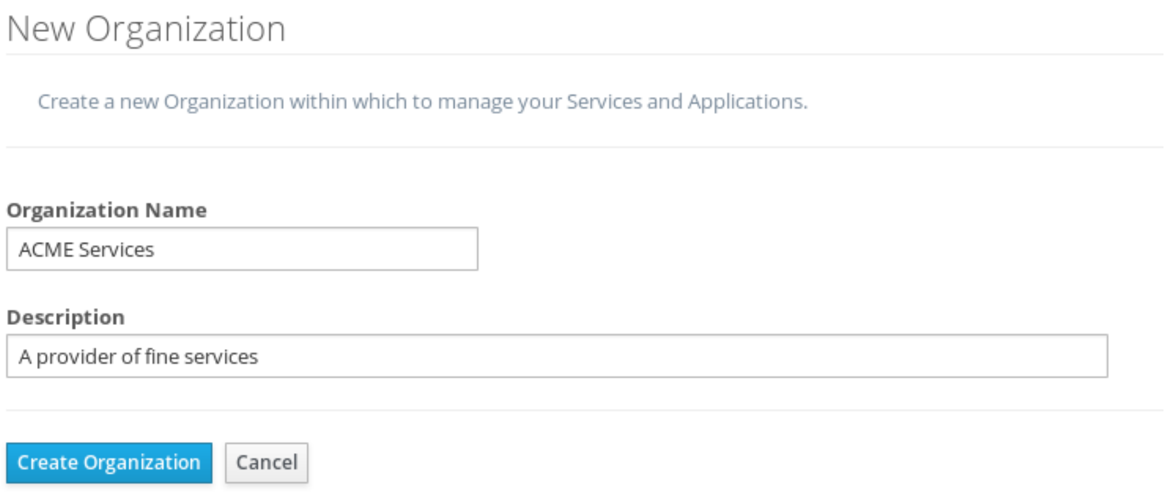


Organizations

All services and applications must be managed within the context of an Organization. You can be a member of multiple Organizations at the same time, with different roles in each: you can be an Application Developer in one organization and a Service Developer in another.

- Create a New Organization
- Browse/Find an Organization
- Go to My Organizations

Provide a name and description for New Organisation and press “Create Organization”:



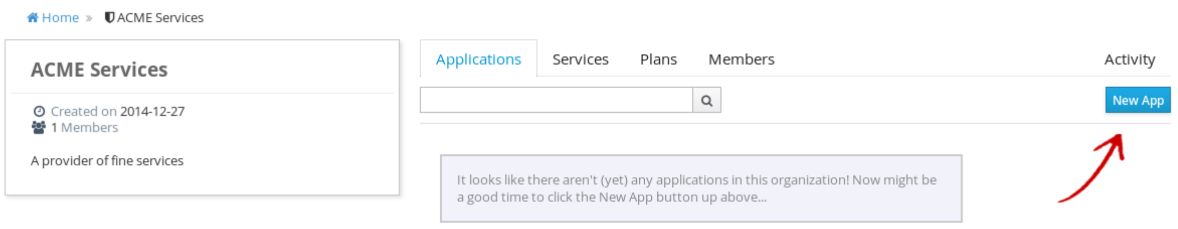
New Organization

Create a new Organization within which to manage your Services and Applications.

Organization Name: ACME Services

Description: A provider of fine services

After creation procedure is done, the Organization configuration is accessible:



ACME Services

Created on 2014-12-27
1 Members

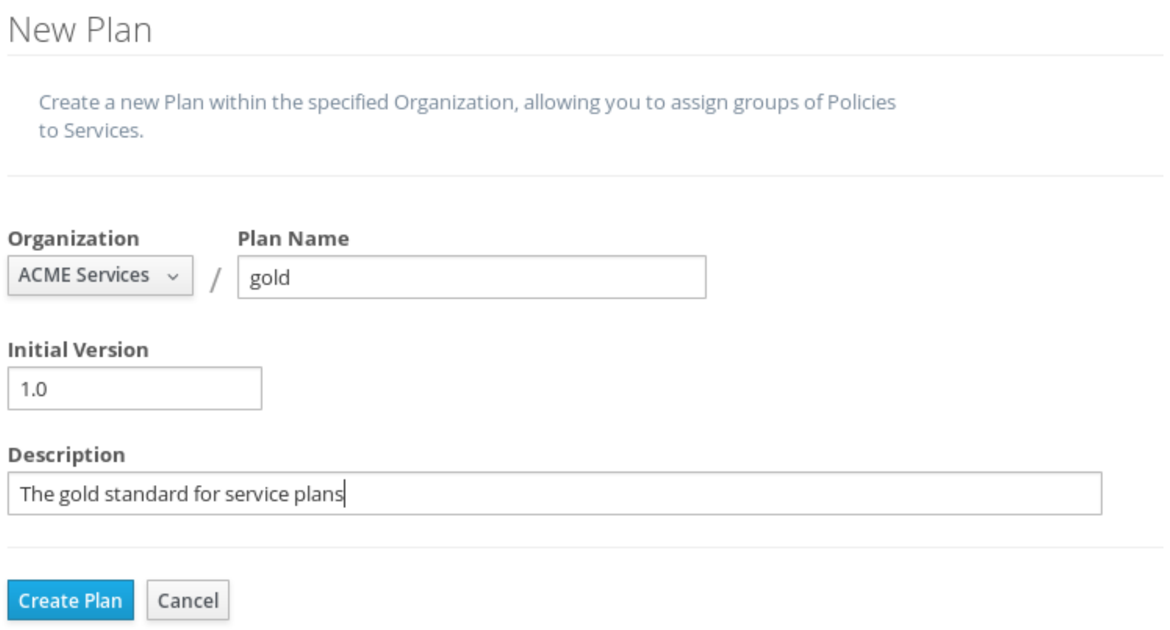
A provider of fine services

It looks like there aren't (yet) any applications in this organization! Now might be a good time to click the New App button up above...

Note: in a production environment, users would request membership in an organization. The approval process for accepting new members into an organization would follow the organization’s workflow, but this would be handled outside of the API Manager. For the purposes of demonstration, that step will be skipped.

API, Policies and Plans Configuration

To configure the API, a plan, which contains policies, must be created. To do so, select the "Plans" tab and fill in the fields:



New Plan

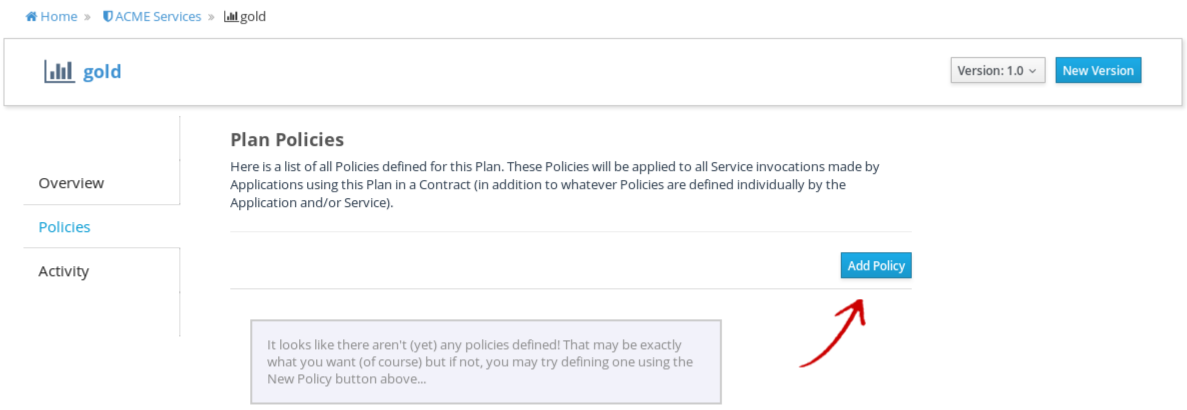
Create a new Plan within the specified Organization, allowing you to assign groups of Policies to Services.

Organization: ACME Services / Plan Name: gold

Initial Version: 1.0

Description: The gold standard for service plans

Once the plan is created, the policies can be added:

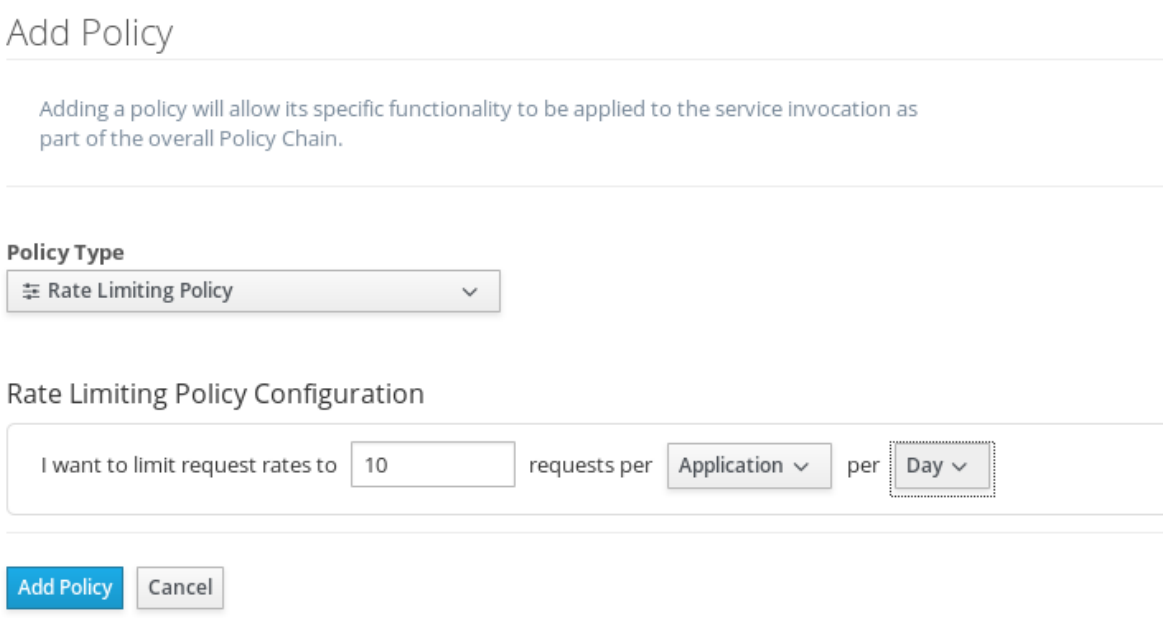


Plan Policies

Here is a list of all Policies defined for this Plan. These Policies will be applied to all Service invocations made by Applications using this Plan in a Contract (in addition to whatever Policies are defined individually by the Application and/or Service).

It looks like there aren't (yet) any policies defined! That may be exactly what you want (of course) but if not, you may try defining one using the New Policy button above...

Apiman provides several OOTB policies. To demonstrate a policy being applied, a Rate Limiting Policy should be selected and its limit set to a very low level. If example API receives more than 10 requests in a day, the policy must block all subsequent requests:



Add Policy

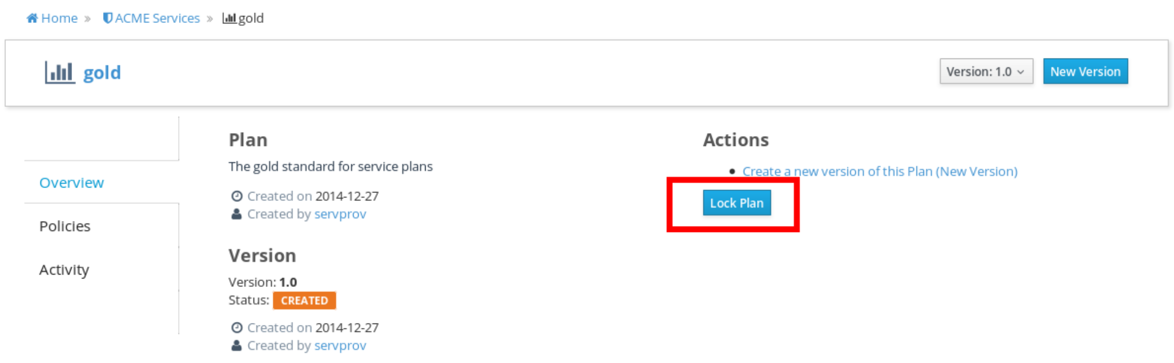
Adding a policy will allow its specific functionality to be applied to the service invocation as part of the overall Policy Chain.

Policy Type: Rate Limiting Policy

Rate Limiting Policy Configuration

I want to limit request rates to 10 requests per Application per Day

After the policy is created and added to plan, the last one must be locked:



Home » ACME Services » gold

gold

Plan

The gold standard for service plans

Created on 2014-12-27
Created by servprov

Version

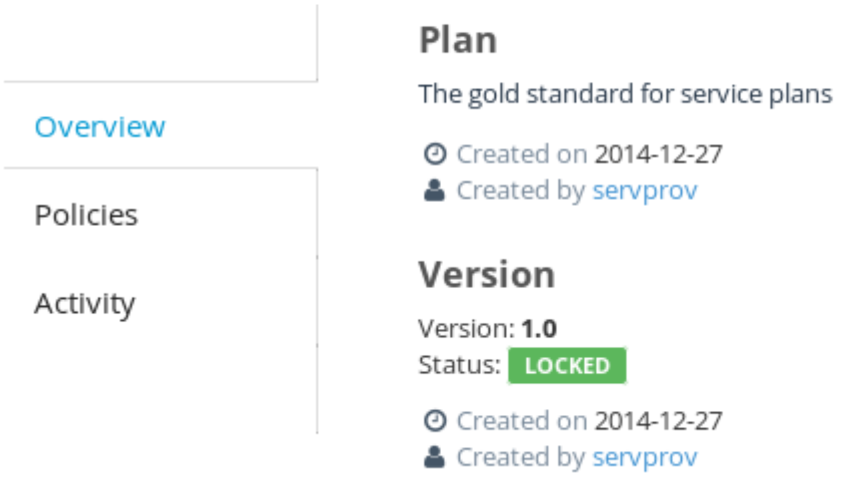
Version: 1.0
Status: CREATED

Created on 2014-12-27
Created by servprov

Actions

- Create a new version of this Plan (New Version)

The result of the locked plan:



Plan

The gold standard for service plans

Created on 2014-12-27
Created by servprov

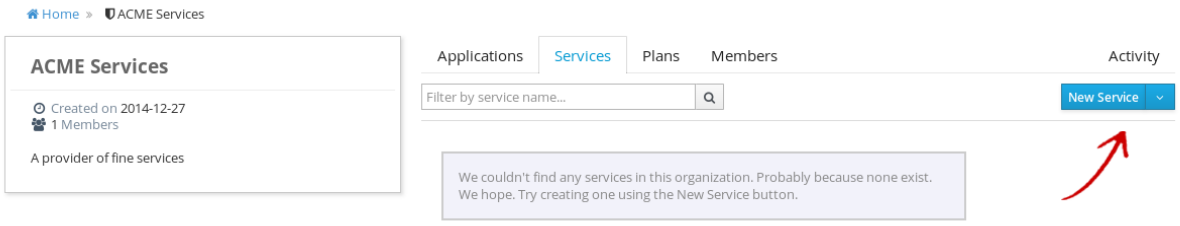
Version

Version: 1.0
Status: LOCKED

Created on 2014-12-27
Created by servprov

To continue demonstration create a new plan (i.e. “silver”), similar to “gold” plan, and set limit to less than 10.

After both plans are created, the API must be defined:



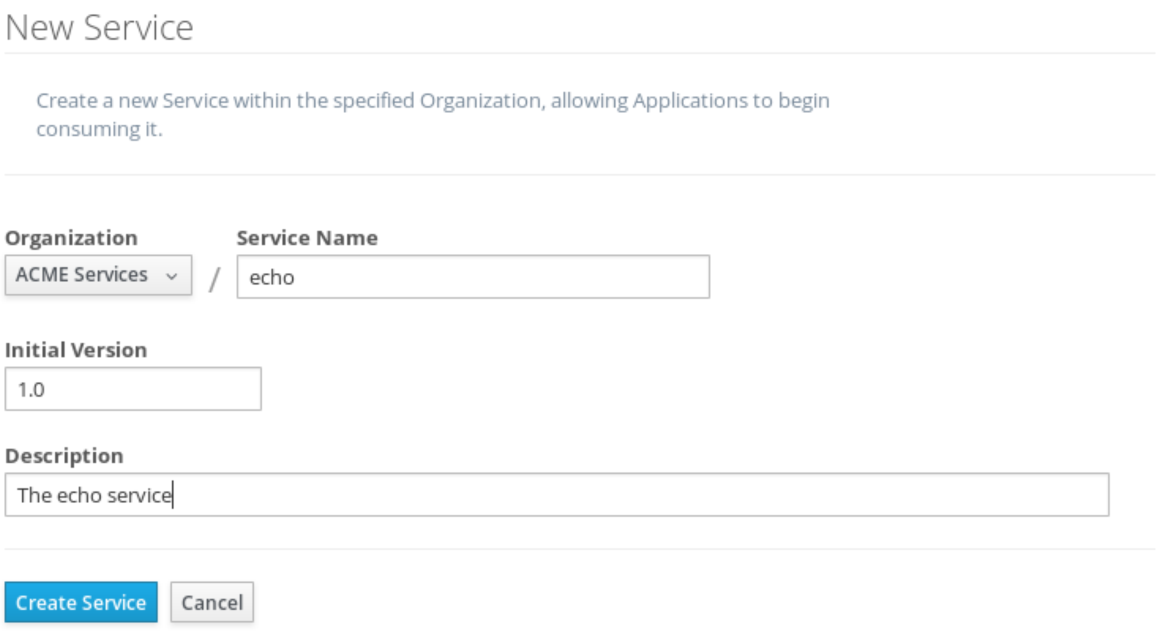
ACME Services

Created on 2014-12-27
1 Members

A provider of fine services

We couldn't find any services in this organization. Probably because none exist. We hope. Try creating one using the New Service button.

Provide the API an appropriate name, so that providers and consumers alike will be able to run a query in the API Manager to find it:



New Service

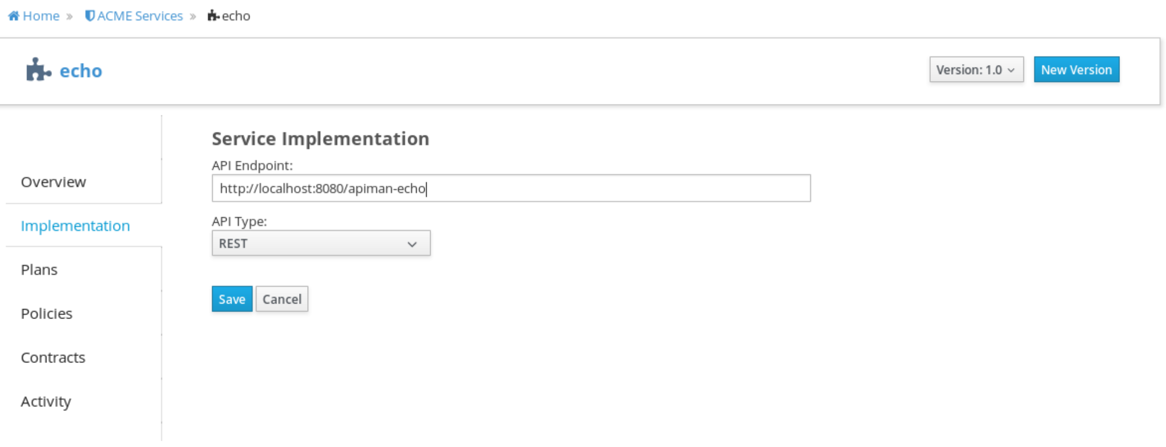
Create a new Service within the specified Organization, allowing Applications to begin consuming it.

Organization: ACME Services / Service Name: echo

Initial Version: 1.0

Description: The echo service

After the API is defined, its implementation must be defined as well:

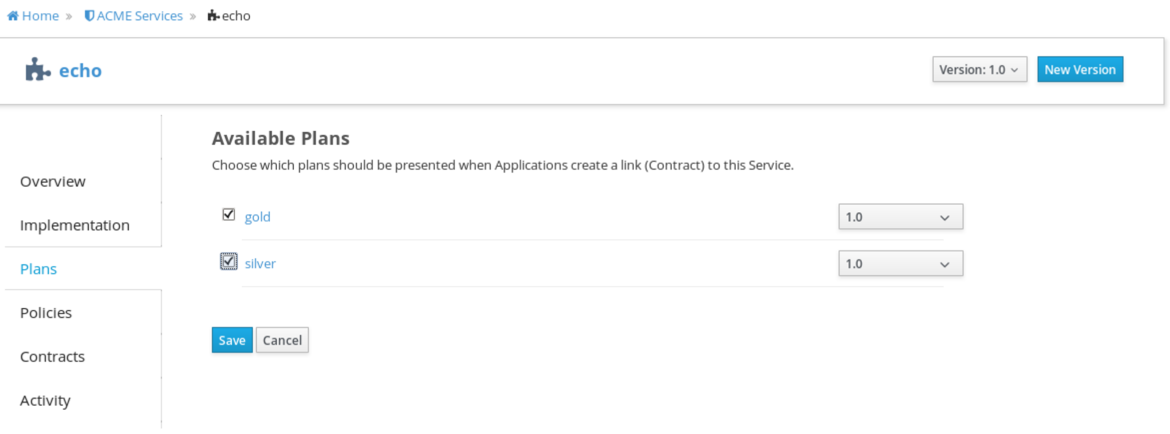


Service Implementation

API Endpoint: http://localhost:8080/apiman-echo

API Type: REST

The plans tab shows which plans are available to be applied to the API:

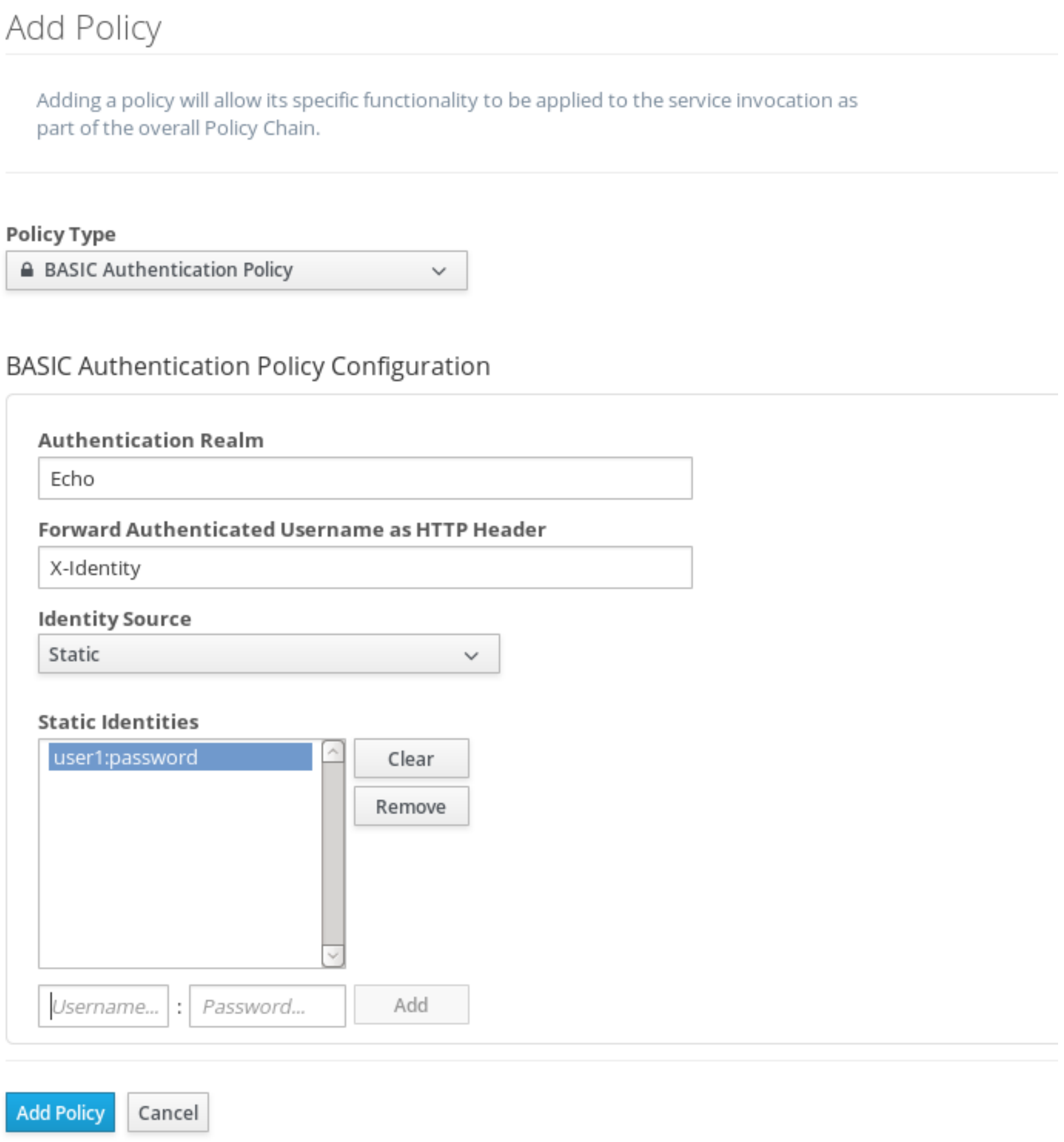


Available Plans

Choose which plans should be presented when Applications create a link (Contract) to this Service.

✓	gold	1.0
✓	silver	1.0

In “Policies” tab define a simple authentication policy. Remember the username and password that are defined here as it will be needed later to demonstrate sending requests to the API:



Add Policy

Adding a policy will allow its specific functionality to be applied to the service invocation as part of the overall Policy Chain.

Policy Type: BASIC Authentication Policy

BASIC Authentication Policy Configuration

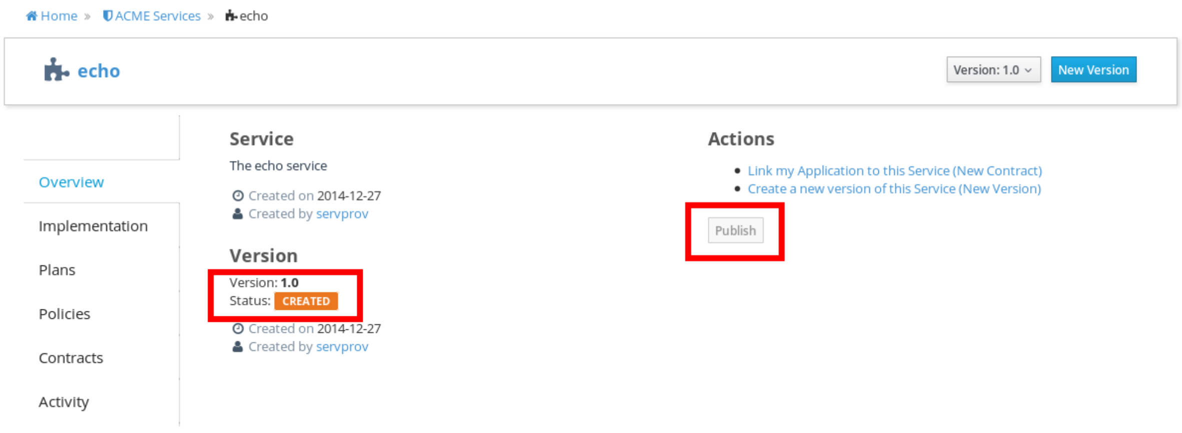
Authentication Realm: Echo

Forward Authenticated Username as HTTP Header: X-Identity

Identity Source: Static

Static Identities: user1:password

After the authentication policy is added, publish the API to the API Gateway:


Home » ACME Services » echo

echo

Version: 1.0
New Version

Overview
Implementation
Plans
Policies
Contracts
Activity

Service

The echo service

Created on 2014-12-27
Created by servprov

Version

Version: 1.0
Status: CREATED

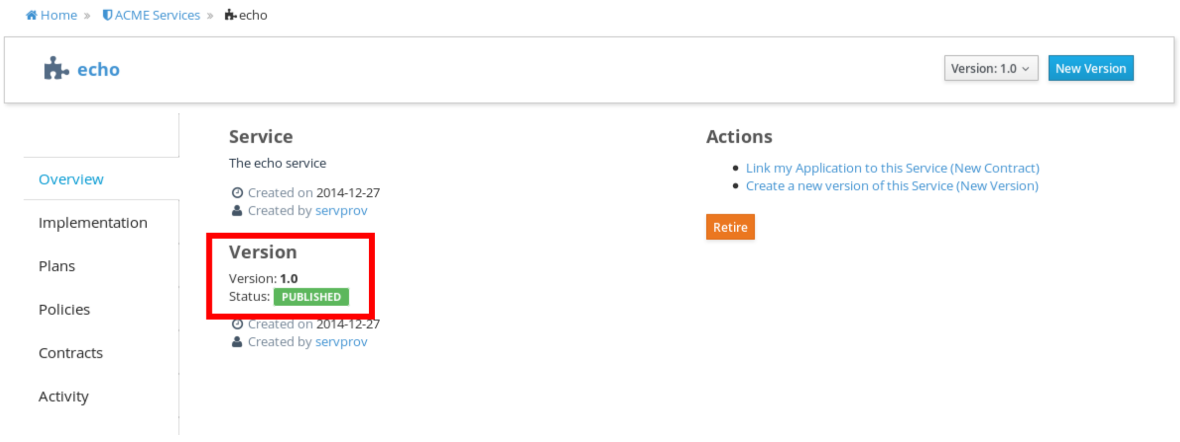
Created on 2014-12-27
Created by servprov

Actions

- Link my Application to this Service (New Contract)
- Create a new version of this Service (New Version)

Publish

The result is a published API:


Home » ACME Services » echo

echo

Version: 1.0
New Version

Overview
Implementation
Plans
Policies
Contracts
Activity

Service

The echo service

Created on 2014-12-27
Created by servprov

Version

Version: 1.0
Status: PUBLISHED

Created on 2014-12-27
Created by servprov

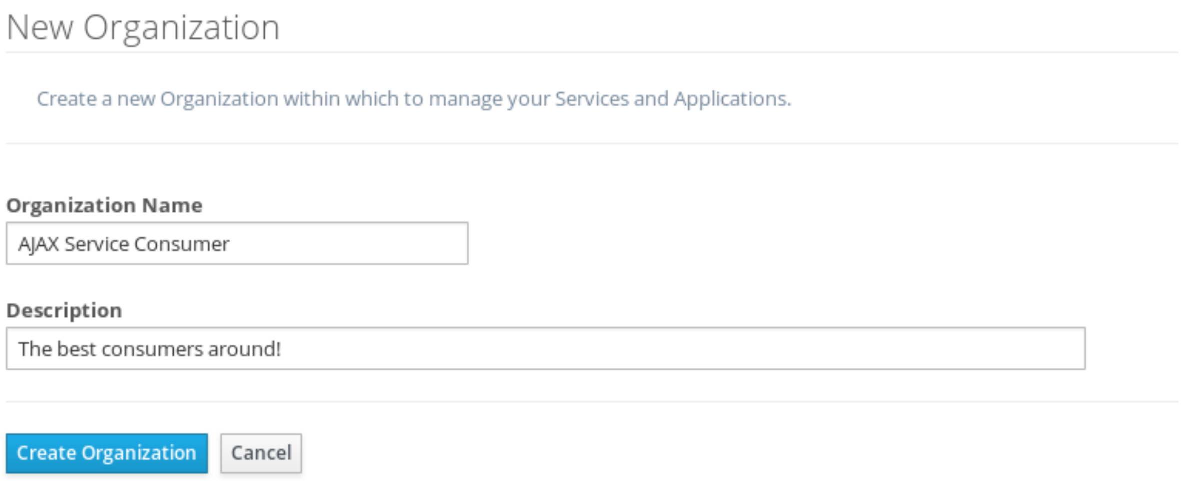
Actions

- Link my Application to this Service (New Contract)
- Create a new version of this Service (New Version)

Retire

The API Consumer Organization

Log in to the API Manager UI as the “appdev” user and create the organization:



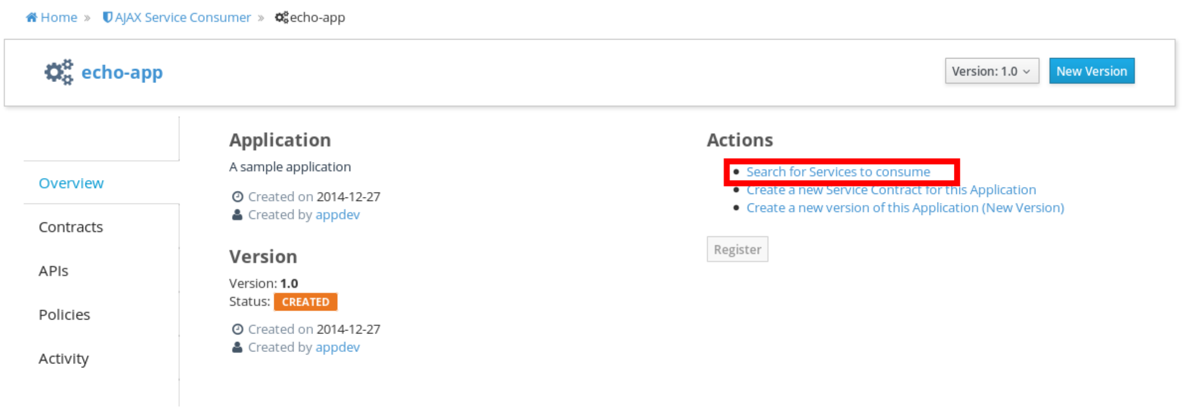
New Organization

Create a new Organization within which to manage your Services and Applications.

Organization Name: AJAX Service Consumer

Description: The best consumers around!

Unlike the previous section for API provider, the first step will be a new client app creation and then search for the API to be used by the client app:



Home » AJAX Service Consumer » echo-app

echo-app

Application

A sample application

Created on 2014-12-27

Created by appdev

Version

Version: 1.0

Status: CREATED

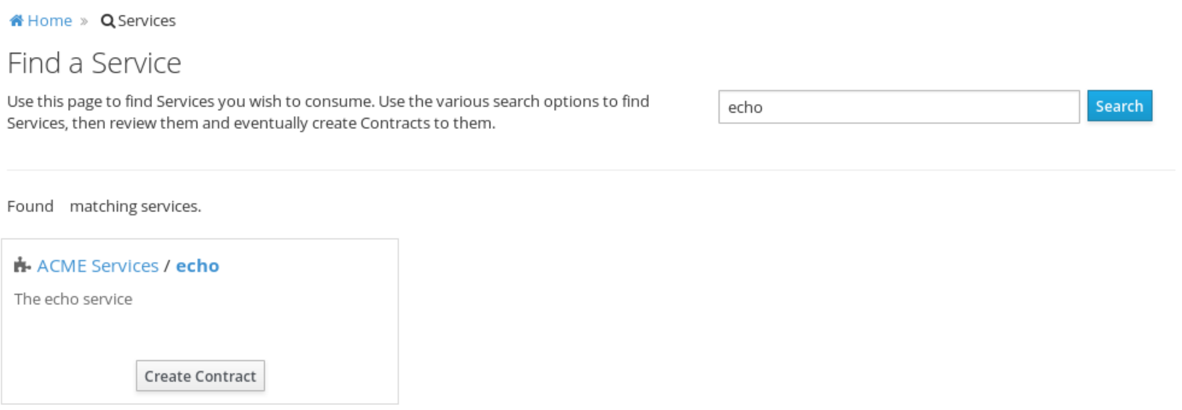
Created on 2014-12-27

Created by appdev

Actions

- Search for Services to consume
- Create a new Service Contract for this Application
- Create a new version of this Application (New Version)

Search for the API:



Home » Services

Find a Service

Use this page to find Services you wish to consume. Use the various search options to find Services, then review them and eventually create Contracts to them.

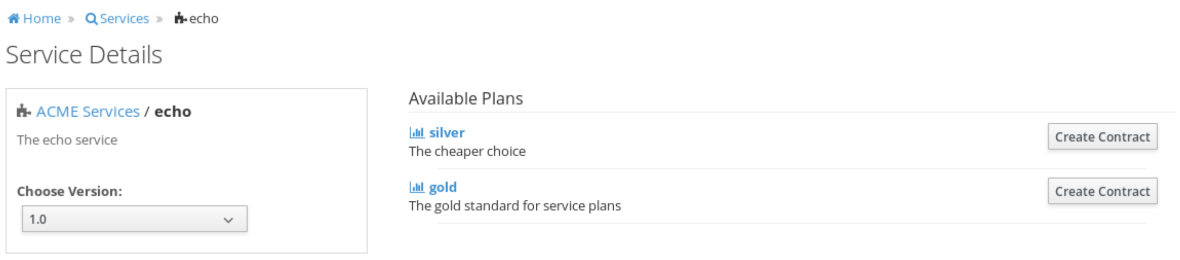
echo

Found matching services.

ACME Services / echo

The echo service

Select the API name, and then specify the plan to be used (“gold”):



Home » Services » echo

Service Details

ACME Services / **echo**

The echo service

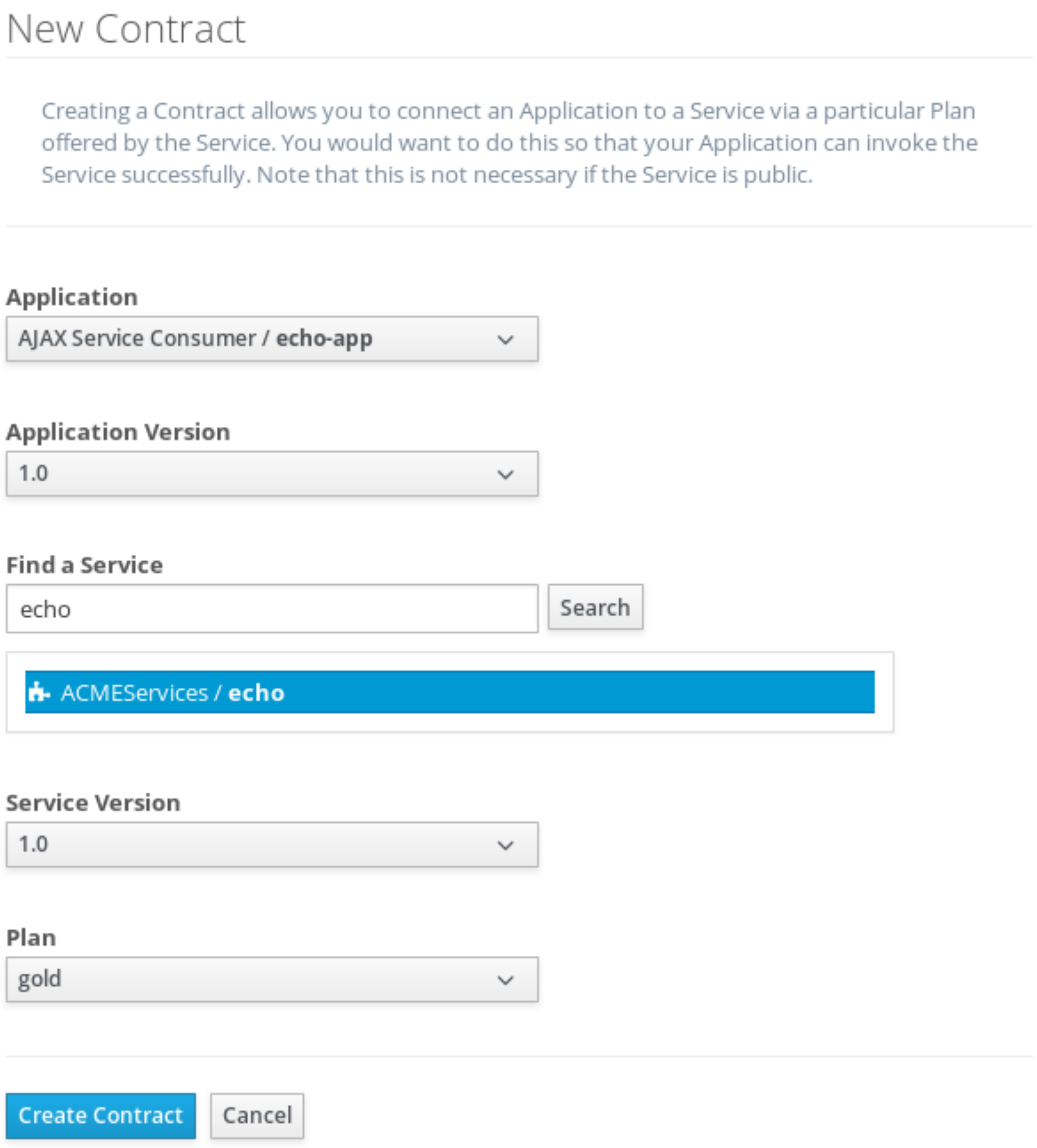
Choose Version: 1.0

Available Plans

silver
The cheaper choice

gold
The gold standard for service plans

Next, select “create contract” for the plan:



New Contract

Creating a Contract allows you to connect an Application to a Service via a particular Plan offered by the Service. You would want to do this so that your Application can invoke the Service successfully. Note that this is not necessary if the Service is public.

Application: AJAX Service Consumer / **echo-app**

Application Version: 1.0

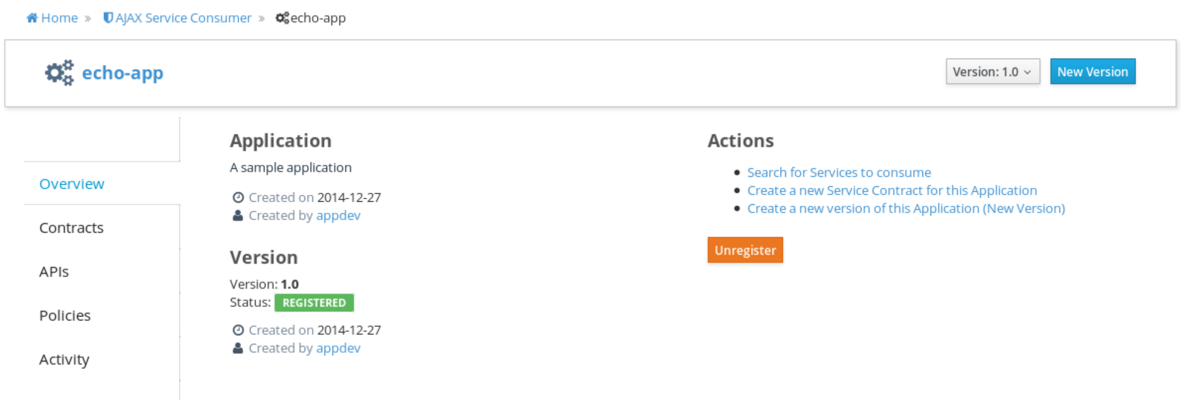
Find a Service: echo

ACMEServices / **echo**

Service Version: 1.0

Plan: gold

Agree to the contract terms and register the client app with the API Gateway so that the gateway can act as a proxy for the API:



Application

A sample application

Created on 2014-12-27
Created by appdev

Version

Version: 1.0
Status: REGISTERED

Created on 2014-12-27
Created by appdev

Actions

- Search for Services to consume
- Create a new Service Contract for this Application
- Create a new version of this Application (New Version)