Overlord APIMan Guide

Chapter 1. Introduction to APIMan

1.1. What is API Management?

A popular trend in enterprise software development these days is to design applications to be very decoupled and use API's to connect them. This approach provides an excellent way to reuse functionality across various applications and business units. Another great benefit of API usage in enterprises is the ability to create those API's using a variety of disparate technologies.

However, this approach also introduces its own pitfalls and disadvantages. Some of those disadvantages include things like:

- Difficulty discovering or sharing existing API's
- Difficulty sharing common functionality across API implementations
- Tracking of API usage/consumption

API Management is a technology that addresses these and other issues by providing a management layer to track APIs and configure governance policies, as well as a runtime layer that sits between the API and the client. This runtime layer is responsible for applying the policies configured during management.

Therefore an API management system tends to provide the following features:

- Centralized governance policy configuration
- Tracking of API's and consumers of those API's
- Easy sharing and discovery of API's
- Leveraging common policy configuration across different API's

1.2. Project Goals

The goals of the JBoss overlord API management project are to provide an easy to use and powerful management layer as well as a small, fast, low overhead runtime layer to implement standard API management functionality.

1.3. Typical Use Cases

Some common API management use cases include:

1.3.1. Security

APRs will very often have a security requirement such that clients connecting to the API must authenticate in some fashion. Authentication can vary greatly both in the protocols used to authenticate and the identity source used for validation.

It can often be convenient to provide authentication at the API management layer to free up the back end service from having to do this work. This approach also has the side benefit of centralizing configuration of authentication for a wide array of disparate services.

Therefore the API management layer must provide authentication capabilities using a wide range of protocols including BASIC, digest, OAuth, etc.

1.3.2. Throttling

The API management layer is a convenient place to ensure throttling (also known as rate limiting) to your API's. Throttling is a way to prevent individual clients from issuing too many requests to an API. Because all requests to an API go through the API management runtime layer it is an excellent place to do this throttling work.

1.3.3. Metering

There are a number of reasons why an API provider would be interested in the number of requests made to her API. The most common reason for public facing API's is to implement billing based on usage per consumer. Metering is the feature that provides this API management capability.

Chapter 2. Getting Started

Chapter 3. Installation Guide

Chapter 4. User Guide

This section of the project documentation is intended to be used by end users of the system. If you are using JBoss overlord API management to either provide or consume API's then this guide should help you be successful.

4.1. Overview

There are two layers to the API management project. There is a management layer which allows end users to centrally manage their API's. Additionally there is a runtime layer which is responsible for applying those policies to API requests.

This section of the guide will focus mostly on the management layer as it is this layer that the end-user is most concerned with.

4.2. Management Layer

4.2.1. Concepts

The management layer allows end-users to track, configure, and share API's with other users. All of this is accomplished by the end user by logging into the API management user interface.

4.2.2. Data Model

It is perhaps most important to understand the various entities used by the management layer, as well as their relationships with each other.

4.2.2.1. Organizations

The top level container concept within the API management project its called the organization. All other entities are managed within the scope of an organization.

When users log into the API management system they must be affiliated with one or more organization. Users can have different roles within that organization allowing them to perform different actions and manage different entities. Please see the *User Management* section below for more details on this topic.

What an organization actually represents will depend upon who is using API management. When installed within a large enterprise, an organization may represent an internal group within IT (for example the HR group). If installed in the cloud, an organization might represent an external company or organization.

In any case, an organization is required before the end user can create or consume services.

4.2.2.2. Policies

The most important concept in API management is the policy. The policy is the unit of work executed at runtime in order to implement API governance. All other entities within the management layer exist in support of configuring policies and sensibly applying them at runtime.

When a request for a Service is made at runtime, a policy chain is created and applied to the inbound request, prior to proxying that request to the back-end API implementation. This policy chain consists of policies configured in the management layer.

An individual policy consists of a type (e.g. authentication or rate limiting) as well as configuration details specific to the type and instance of that policy. Multiple policies can be configured per service resulting in a policy chain that is applied at runtime.

It is very important to understand that policies can be configured at three different levels within API management. Policies can be configured on a service, on a plan, or on an application. For more details please see the sections below.

4.2.2.3. Plans

A plan is a set of policies that define a level of service. Whenever a service is consumed it must be consumed through a plan. Please see the section on ‘Service Contracts’ for more information.

An organization can have multiple plans associated with it. Typically each plan within an organization consists of the same set of policies but with different configuration details. For example, an organization might have a Gold plan with a rate limiting policy that restricts consumers to 1000 requests per day. The same organization may then have a Silver plan which is also configured with a rate limiting policy, but which restricts consumers to 500 requests per day.

4.2.2.4. Services

A service represents an external API that is being governed by the API management system. A service consists of a set of metadata including name and description as well as an external endpoint defining the API implementation. The external API implementation endpoint includes the type/protocol and the actual endpoint location so that the API can be properly proxied to at runtime.

In addition, policies can be configured on a service. Typically, the policies applied to services are things like authentication. Any policies configured on service will be applied at runtime regardless of the application and service contract. This is why authentication is a common policy to configure at the service level.

Services must be offered through a valid plan configured in the same organization. Service consumers must consume the service through one of those plans. Please see the section on ‘Service Contracts’ for more information.

Only once a service is fully configured, including its policies, implementation, and plans can it be published to the runtime gateway for consumption by applications. Once this is done, the service

cannot be changed. If changes are required, a new version of the service must be created and configured.

4.2.2.5. Applications

An application represents a consumer of an API. Typical API consumers are things like mobile applications and B2B applications. Regardless of the actual application implementation, an application must be added to the API management system so that contracts can be created between it and the services it wishes to consume.

An application consists of basic metadata such as name and description. Policies can also be configured on an application, but are optional.

Finally, service contracts can be created between an application and the service(s) it wishes to consume. Once the service contracts are created, the application can be registered with the runtime gateway. Once this registration is complete, the application can no longer be altered. If changes are required, a new version of the application must be created and configured.

4.2.2.6. Service Contracts

A service contract is simply a link between an application and a service through a plan offered by that service. This is the only way that an application can consume a service. If there are no applications that have created service contracts with a service, that service cannot be accessed through the API management runtime gateway.

When a service contract is created, the system generates a unique API key specific to that contract. All requests made to the service through the API management runtime layer must include this API key. The API key is used to create the runtime policy chain from the policies configured on the service, plan, and application.

4.2.2.7. Policy Chain

A policy chain is an ordered sequence of policies that are applied when a request is made for a service through the API management runtime layer. The order that policies are applied is important and is as follows:

1. Application
2. Plan
3. Service

Within these individual sections, the end user can specify the order of the policies.

When a request for a service is received by the runtime layer the policy chain is applied to the request in the order listed above. If none of the policies fail, the runtime layer will proxy the request to the backend API implementation. Once a response is received from the back end API

implementation, the policy chain is then applied in reverse order to that response. This allows each policy to be applied twice, once to the inbound request and then again to the outbound response.

4.2.3. User Management

The management layer offers user role capabilities at the organization level. Users can be members of organizations and have specific roles within those organizations. The roles themselves are configurable by an administrator, and each role provides the user with a set of permissions that determine what actions the user can take within an organization.

4.2.3.1. New Users

End users must self register with the management UI in order to be given access to an organization or to create their own organization. In some configurations it is possible that end user self registration is unavailable and instead user information is provided by a standard source of identity such as LDAP. In either case, the actions a user can take are determined by that user's role memberships within the context of an organization.

4.2.3.2. Membership

End users can be members of organizations. All memberships in an organization include the specific roles the user is granted. It is typically up to the owner of an organization to grant role memberships to the members of that organization.

4.2.3.3. Roles

Roles determine the capabilities granted a user within the context of the organization. The roles themselves and the capabilities that those roles grant are configured by system administrators. For example, administrators would typically configure the following roles:

- Organization Owner
- Service Developer
- Application Developer

Each of these roles is configured by an administrator to provide a specific set of permissions allowing the user to perform relevant actions appropriate to that role. For example the Application Developer role would grant an end user the ability to manage applications and service contracts for those applications. However that user would not be able to create or manage the organization's services or plans.

4.2.4. Managing Organizations

Before any other actions can be taken an organization must exist. All other operations take place within the context of an organization.

In order to create an organization click the ‘Create a New Organization’ link found on the dashboard page that appears when you first login. Simply provide an organization name and description and then click the ‘Create Organization’ button. If successful you will be taken to the organisation details page.

If you create multiple organizations, you can see the list of those organizations on your home page. For example, you may click the ‘Go to My Organizations’ link from the dashboard page.

4.2.5. Managing Plans

Plans must be managed within the scope of an organization. Once created, plans can be used for any service defined within that same organisation. To see a list of existing plans for an organization, navigate to the ‘Plans’ tab for that organization on its details page.

4.2.5.1. Creating a Plan

Plans can be created easily from the ‘Plans’ tab of the organization details page. Simply click the ‘New Plan’ button and then provide a plan name, version, and description. Once that information is provided, click the ‘Create Plan’ button. If successfully created, you’ll be taken to the plan details page.

4.2.5.2. Plan Policies

If you switch to the ‘Policies’ tab on the plan details page you can configure the list of policies for the plan. Please note that the order of the policies can be changed and is important. The order that the policies appear in the user interface determines the order they will be applied at runtime. You can drag a policy up and down the list to change the order.

To add a policy to the plan click the ‘Add Policy’ button. On the resulting page choose the type of policy you wish to create and then configure the details for that policy. Once you have configured the details click the ‘Add Policy’ button to add the policy to the plan.

4.2.6. Providing Services

A core capability of API management is for end users to create, manage, and configure services they wish to provide. This section explains the steps necessary for end users to provide those services.

4.2.6.1. Creating a Service

First the user must create a service within an organization. If an organization does not yet exist one can easily be created. See the ‘Managing Organizations’ section for details.

From the organization details page, navigate to the ‘Services’ tab and click on the ‘New Service’ button. You will be asked to provide a service name, version number, and description.

If successfully created, you will be taken to the service details page. From here you can configure the details of the service.

4.2.6.2. Service Implementation

Every service must be configured with an API implementation. The implementation indicates the external service that the runtime layer will proxy to if all the policies are successfully applied. Click the ‘Implementation’ tab to configure the API endpoint and API type details on your service.

Do not forget to click the Save button when you are done making changes.

4.2.6.3. Available Plans

Before a service can be consumed by an application, it must make itself available through at least one of the organization’s plans. This is done by navigating to the ‘Plans’ tab on the service details page. The ‘Plans’ tab will list all of the available plans defined by the organization. Simply choose one or more plan from this list.

After you have selected at least one plan, make sure to click the Save button.

4.2.6.4. Managing Policies

Service policies can be added and configured by navigating to the ‘Policies’ tab on the service details page. The ‘Policies’ tab presents a list of all the policies configured for this service. To add another policy to the service click the ‘Add Policy’ button. On the resulting page choose the type of policy you wish to create and then configure the details for that policy. Once you have configured the details click the ‘Add Policy’ button to add the policy to the service.

4.2.6.5. Publishing in the Gateway

After all of the configuration is complete for a service, it is time to publish the service to the runtime gateway. This can be done from the ‘Overview’ tab of the service details page. Simply click the ‘Publish’ button on the ‘Overview’ tab to publish the service to the runtime gateway. If successful, the status of the service will change to “Published” and the ‘Publish’ button will disappear.

It is worth repeating that the ‘Publish’ button will be disabled until the service is fully configured (at which time it transitions to “Ready” status). This includes at least the implementation and one or more available plans. Service policies are optional.

4.2.7. Consuming Services

After the service providers have added a number of services to the API management system, those services can be consumed by applications. This section explains how to consume services.

4.2.7.1. Creating an Application

In order to consume a service you must first create an application. Applications must exist within the context of an organization. If an organization does not yet exist for this purpose, simply create a new organization. See the section above on ‘Managing Organizations’ for more information.

To create a new application click the ‘Create a New Application’ link on the dashboard page. On the resulting page provide an application name, version, and description and then click the ‘Create

Application' button. If the application is successfully created, you will be taken to the application details page.

4.2.7.2. Creating Service Contracts

The primary action taken when configuring an application is the creation of contracts to services. This is what we mean when we say "consuming a service". There are a number of ways to create service contracts. This section will describe the most useful of these options.

From the application details page select the 'Overview' tab. Click on the 'Search for services to consume' link in the 'Things To Do' section. You will be taken to a page that will help you search for and find the service you wish to consume.

Use the controls on this page to search for a service. Once you have found the service you are interested in, click on its name in the search results area. This will take you to the service details page for service consumers. The consumer-oriented service details page presents you with all of the information necessary to make a decision about how to consume the service. It includes a list of all the service versions and a list of all of the available plans the service can be consumed through.

Note that you can click on an individual plan to see the details of the policies that will be enforced should that plan be chosen. Click on the 'Create Contract' button next to the plan you wish to use when consuming this service. You will be taken to the new contract page to confirm that you want to create a service contract to this service through the selected plan. If you are sure this is the service contract you wish to create, click the 'Create Contract' button and then agree to the terms and conditions. If successful, you will be taken to the 'Contracts' tab on the application details page.

From the 'Contracts' tab on the application details page you can see the list of service contracts already created for this application. It is also possible to break service contracts from this same list by clicking an appropriate 'Break Contract' button.

4.2.7.3. Managing Policies

Just like plans and services, applications can have configured policies. The 'Policies' tab will present a list of all the policies configured for this application. To add another policy to the application click the 'Add Policy' button. On the resulting page choose the type of policy you wish to create and then configure the details for that policy. Once you have configured the details click the 'Add Policy' button to add the policy to the application.

4.2.7.4. Registering in the Gateway

After at least one service contract has been created for the application, it is possible to register the application with the runtime gateway. Until the application is registered with the runtime gateway, it is not possible to make requests to back-end services on behalf of that application.

To register the application with the gateway navigate to the 'Overview' tab on the application details page. The status of the application should be "Ready", and the 'Register' button should

be enabled. Click the ‘Register’ button to register the application with the runtime gateway. If successful, the application status will change to “Registered”, and the ‘Register’ button will disappear.

4.2.7.5. Live Service Endpoints

After an application has been registered with the runtime gateway, it is possible to send requests to the back-end services on behalf of that application (through the application’s service contracts). To do this you must know the URL of the managed service. This URL includes the API Key generated for the Service Contract.

To view a list of all of these managed endpoints, navigate to the ‘APIs’ tab on the service detail page. Each service contract is represented in the list of managed endpoints. You can copy one of the endpoints by hovering your mouse over the appropriate item in the list and clicking the ‘Copy’ button. Use the copied URL to issue requests to the service.

4.3. Runtime Layer

4.3.1. Configuration

4.3.2. Invoking Services

Chapter 5. Developer Guide

Bibliography

Books

[walsh-muellner] Norman Walsh & Leonard Muellner. *DocBook - The Definitive Guide*. O'Reilly & Associates. 1999. ISBN 1-56592-580-7.