WSO2 API Manager

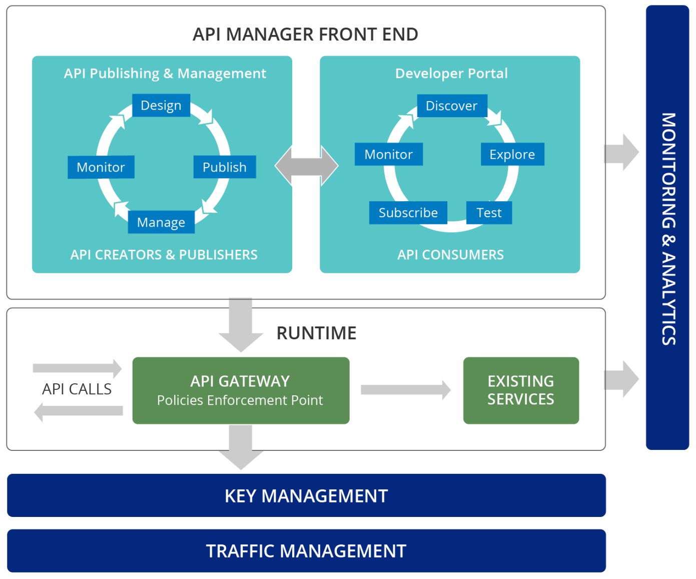
Documentation

Version 3.0.0

Table of Contents

WSO2 API Manager Documentation

Welcome to WSO2 API Manager Documentation! WSO2 API Manager (APIM) is a fully open source, complete solution for creating, publishing and managing all aspects of an API and its lifecycle, and is ready for massively scalable deployments.



Get started with WSO2 API Manager

If you are new to using WSO2 API Manager, follow the steps below to get started:

	Get familiar with WSO2 API Manager Understand the basics of the API Manager, including the business cases it solves, its features, and its architecture.
	Quick Start Guide Download, install, and get up and running quickly.

Deep dive into WSO2 API Manager

	Tutorials		**Deep Dive**		**Admin Guide**
	WSO2 APIs		**Analytics**		**Reference Guide**

To download a PDF of this document or a selected part of it, click here (only generate one PDF at a time). You can also use this link to export to HTML or XML.

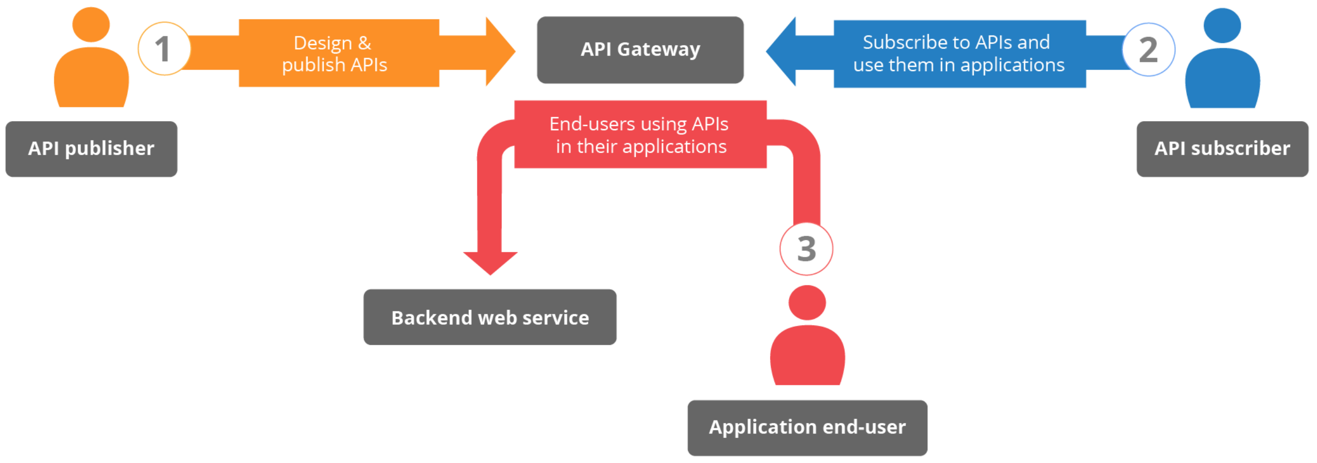
Introduction

The topics in this section introduce WSO2 API Manager, the business cases it solves, its features, and architecture.

- Overview
- Architecture

Overview

WSO2 API Manager is a 100% open source enterprise-class solution that supports API publishing, lifecycle management, application development, access control, rate limiting and analytics in one cleanly integrated system.



It has the following capabilities:

Design, compose and publish APIs - Allows you to create APIs and publish them through the API Publisher. Role-based access control clearly separates the API developers (creators) from API publishers (who hold the responsibility to make an API publicly available).

Enable API discovery - Gives easy access to your APIs by providing a store-like experience. APIs can be documented, tagged, categorized, and tested from the API Store itself.

Perform QoS functions - The high performance gateway acts as a strong proxy for your backend services and performs quality of service functions such as security, rate limiting and analytics.

Gain insight into API usage - Monitors API and application usage, at operation and business levels, which can then be analyzed within the product itself. Also monitors system behavior.

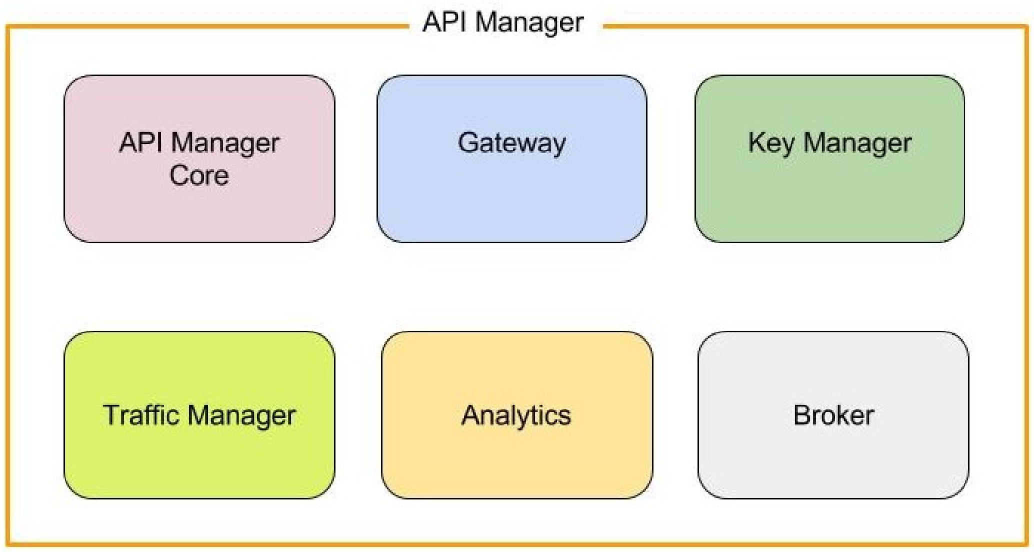
Secure APIs - Leverages the OAuth specification and supports common grant types such as client credentials, code, password, implicit, Security Assertion Markup Language (SAML) and Integrated Windows Authentication (IWA) allowing APIs to be used by different types of applications.

Architecture

WSO2 API Manager 3.0.0 is the next-generation API Management solution, which is rearchitected from the ground up with the latest technologies and patterns. It is built on top of Carbon Kernel 5.0.0, the core of the next generation WSO2 Carbon Platform, which comes with a completely redesigned and streamlined architecture. The next generation API Manager's extendible, componentized architecture allows you to easily customize and extend the solution according to the given requirement.

An API Manager 3.0.0 deployment comprises of the following high-level components

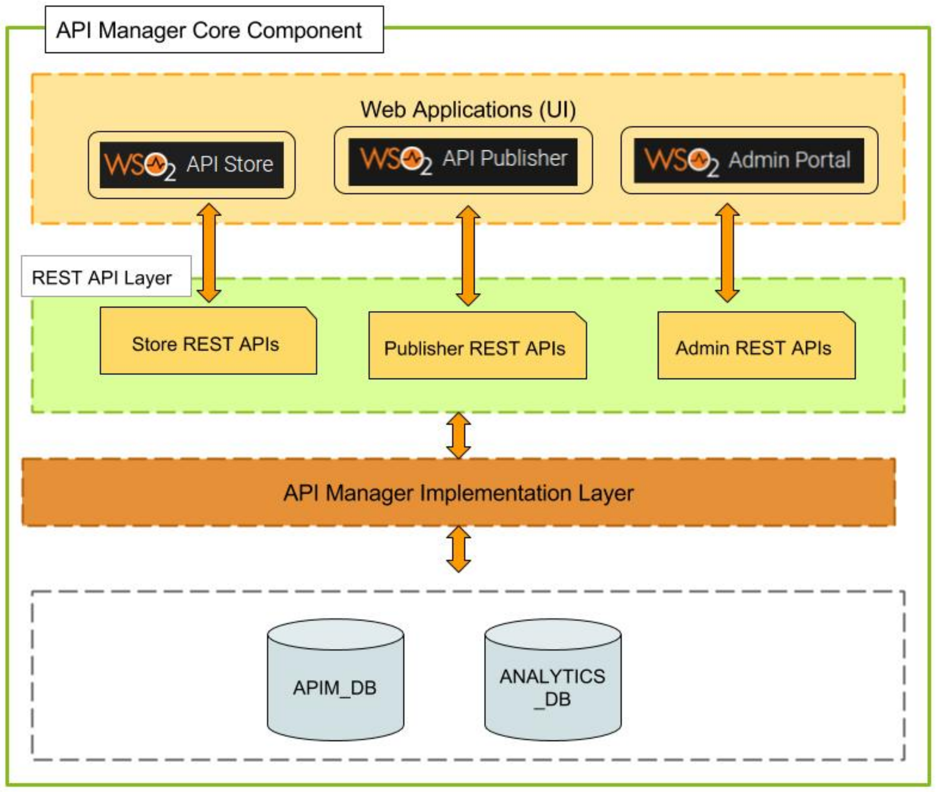
- API Manager Core
- Gateway
- Key Manager
- Traffic Manager
- Analytics
- Broker



Let's take a look at each component and go through the functionalities and how they are integrated, incorporated and perform to build the API Management solution.

API Manager Core

The API Manager Core component is the most significant component, where all API Management related sub-components reside and where the business logic is implemented. The following diagram depicts the aggregation of the sub-components of the API Manager Core.

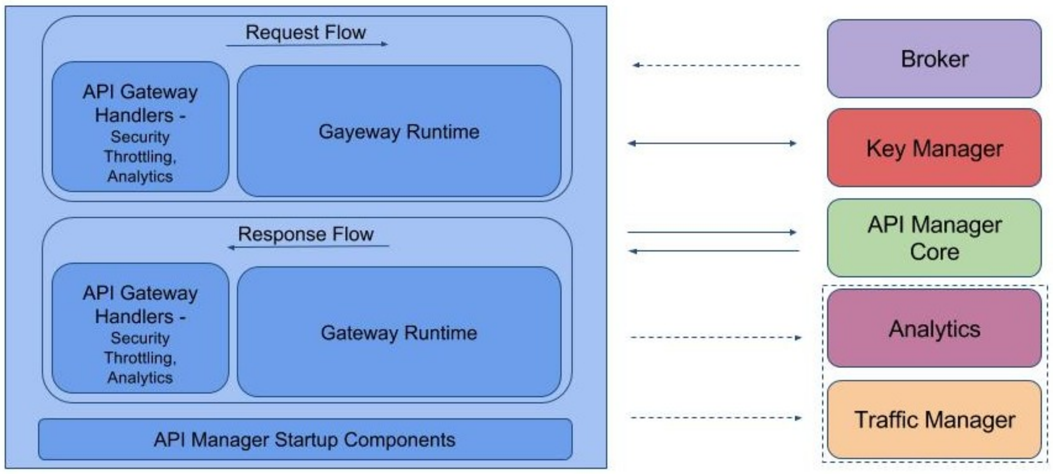


- The top layer or the web interface of the component is where the users interact with the API Management solution and consume the core API Management features shifted with the product. This layer comprises of a set of web applications developed on top of the WSO2 Unified UI Framework. The web interface consists of the following web applications:
    - **API Publisher** - Provides an end-user, collaborative web interface for API providers to compose, publish and manage APIs and share documentation.
    - **API Store** - Provides an end-user, collaborative web interface for consumers to self-register, discover API functionality, subscribe to APIs, evaluate them and interact with API publishers.
    - **API Admin Portal** - Provides an end-user, collaborative web interface for API Manager administrator users to manage the deployment, users, configurations, view statistics, etc.
- The web applications consume the RESTful API layer underneath, which is written using swagger 2.0.0. The REST API layer provides a distinct set of REST APIs for the API Publisher, API Store and Admin portal.
- The business logic is implemented in the API Manager implementation layer. All the API Management related backend service implementations reside in this layer and they are exposed as microservices.
- The bottom most layer is the data layer where the API management related data is persisted. The data layer includes the API Manager database, which is dedicated to persisting API related data, and the Analytics database, which is dedicated to persisting API analytics data.

Gateway

Gateway is the runtime backend component, which acts as an API proxy that intercepts API requests, handles message mediation, applies policies such as throttling and security, and publishes API statistics. The next-generation WSO2 Gateway powered by WSO2 Carbon Kernel C5, performs as the API Gateway, together with a set of gateway handlers.

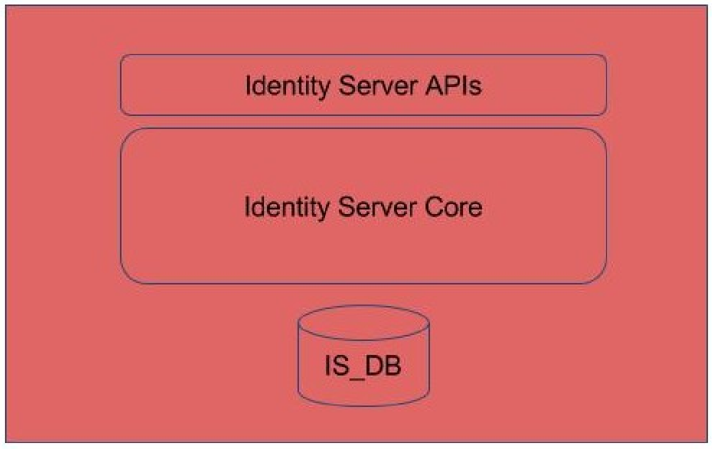
The diagram below depicts how the message flow is handled in the Gateway component. By default, the gateway is attached with an extendible Security handler, Throttling handler and an Analytics handler to manage message inflow and outflow. Simply, a request received at the gateway and the response returning from the gateway pass through the set of handlers where the security validations, throttling requests based on the throttling policy and publishing API runtime statistics take place.


Request Flow
API Gateway Handlers - Security Throttling, Analytics
Gayeway Runtime
Response Flow
API Gateway Handlers - Security Throttling, Analytics
Gateway Runtime
API Manager Startup Components
Broker
Key Manager
API Manager Core
Analytics
Traffic Manager

Key Manager

The Key Manager component handles security and key related operations like user authentications, API key generation and key validations. Key management operations are based on OAuth 2.0.0 protocol specifications.

When a user tries to sign in to the API Store, API Publisher or API Admin portal, OAuth tokens are requested from the key manager and tokens are generated with corresponding user permissions. This particular token is used to handle user authentications, back-end REST API invocations, and to access control in the UI. Similarly, during API key generation, the relevant services in the key manager component are invoked and requested tokens are generated. When API requests are received at the gateway, it contacts the key management component and verifies the validity of the token and executes other security checks. These use cases are described in detail in the tutorials section.



Traffic Manager

The Traffic Manager helps users to regulate API traffic, make APIs and applications available to consumers at different service levels, and secure APIs against security attacks. The Traffic Manager features a dynamic throttling engine to process throttling policies in real-time, including rate limiting of API requests. For more information.

Analytics

Additionally, monitoring and analytics are provided by the analytics component, WSO2 API Manager Analytics. This component provides a host of statistical graphs, an alerting mechanism on pre-determined events and a log analyzer.

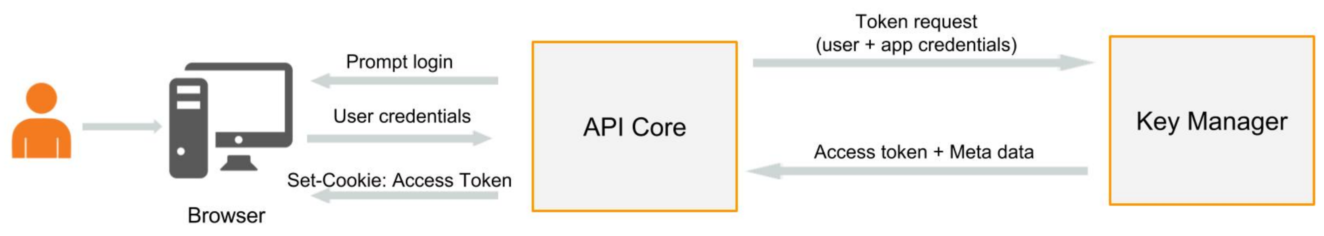
API Management User Scenarios

Let's see how these main components collaboratively interact with each in different user scenarios.

User Login

User login to in API Store, API Publisher and API Admin Portal is handled using OAuth 2.0.0 protocol. Prior to user login, OAuth Dynamic Client Registration (DCR) for API Publisher, API Store and API Admin Portal web applications take place (during server startup). Created OAuth application credentials (application consumer key and consumer secret) credentials are maintained within core component.

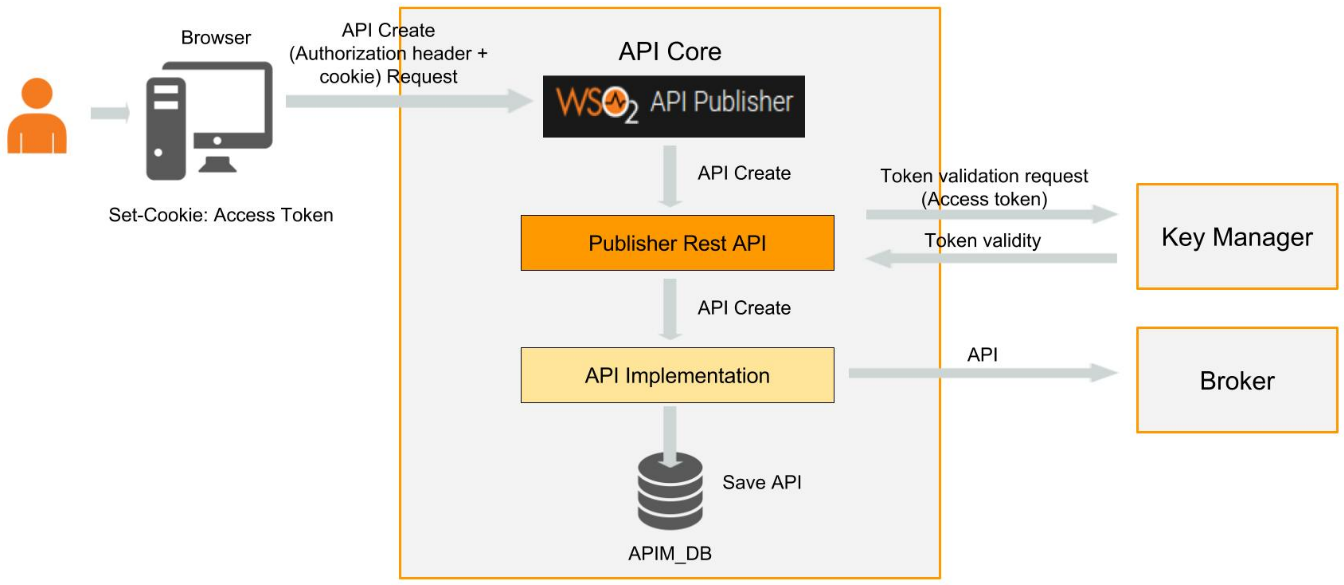
Following diagram illustrate the flow of user login in API Manager web applications.



1. Login page (API Publisher, API Store or API Admin Portal) prompts in the browser.
2. The user provides login credentials (username and password).
3. API Core sends the token generate request with user credentials and OAuth application credentials of the web application.
4. Key Manager receives the token generate request and validate user credentials and application credentials received.
5. If the credentials are valid, then the access token is generated according to user permissions and provided OAuth scopes (OAuth scopes controls the access to REST API resources)
6. The access token is returned to API Core and it is set as a cookie in web browser.

Create an API

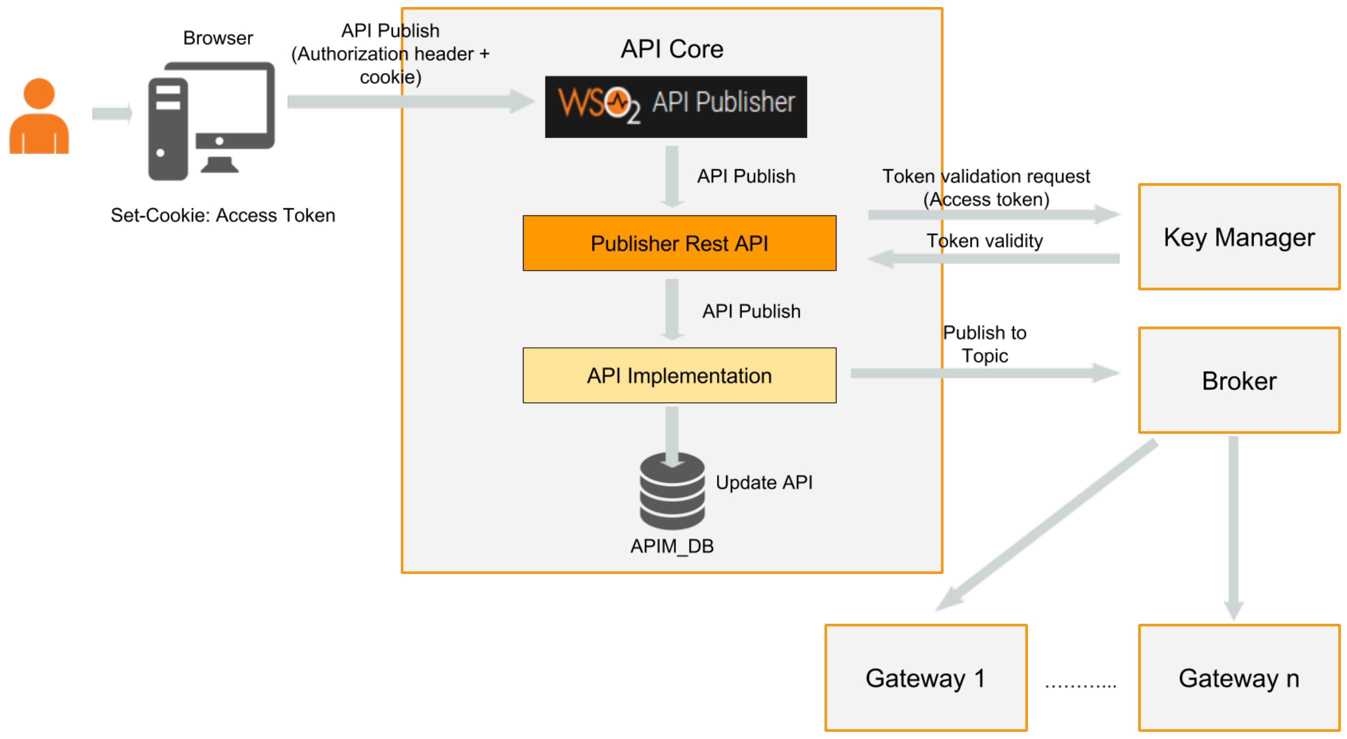
API Publisher is used by API providers to create APIs. Following diagram depicts the component interaction in creating an API in API Publisher UI.



1. API creators Login to API Publisher
2. User permissions are checked to enable/disable API create functionality
3. If the user is permitted, the user creates an API in API Publisher.
4. API create request is sent to relevant Publisher Rest API with the Authorization header, which was defined using access token cookie extracted from the browser.
5. REST API access tokens are maintained in a Cache. But if the key cannot be found in the key cache, it will contact Key Manager services to validate the key.
6. If the token is valid, the API will be created and persisted. Meanwhile, it will be published to Broker topic.

Publish an API

When the APIs are created, API publishers will review and publish them. Below is the component interaction in publishing an API.



1. An API publisher login to API Publisher
2. User permissions are checked to enable/disable API publish functionality.
3. If the user is permitted, the user creates an API in API Publisher.
4. API publish request is sent to relevant Publisher Rest API with the Authorization header, which was defined using access token cookie extracted from the browser.
5. REST API access tokens are maintained in a Cache. But if the key cannot be found in the key cache, it will contact Key Manager services to validate the key.
6. If the token is valid, the API is published and it's life-cycle status is changed to 'Published'.
7. When the API is published, it will be published to broker topic with published APIs.
8. Gateways who have been subscribed to broker topic keep listening and once an API is published, they get instantly notified.

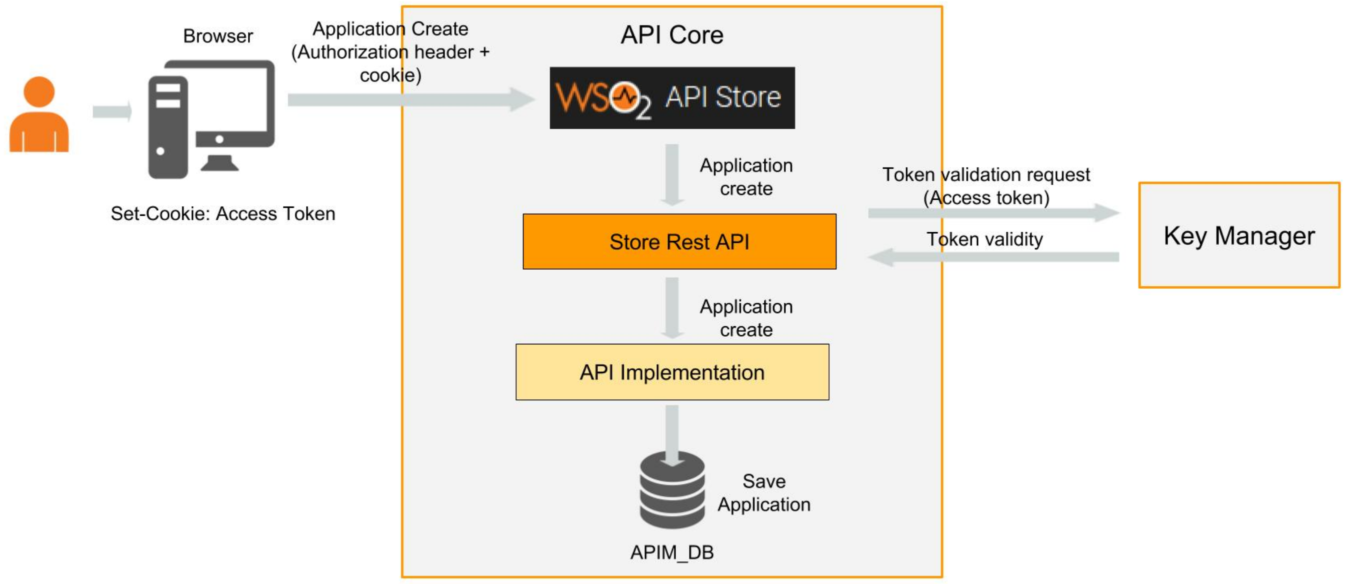
Create an Application

An application is a logical collection of APIs and it is primarily used to decouple the consumer from the APIs. It allows you to :

- Generate and use a single key for multiple APIs
- Subscribe multiple times to a single API with different SLA levels

An Application is required to make a subscription to an API. Applications are available at different SLA levels, and have application-level throttling tiers engaged in them. A throttling tier determines the maximum number of calls you can make to an API during a given period of time.

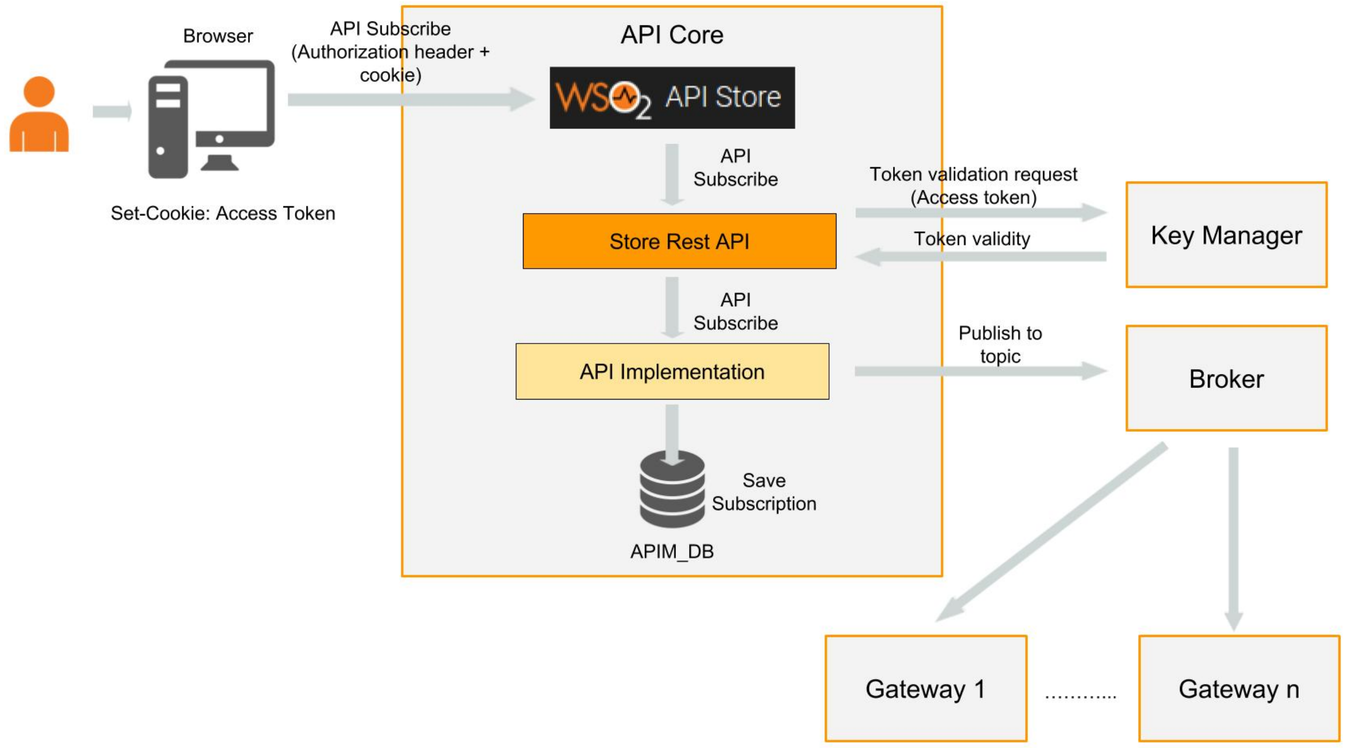
Following is how the application creation happens in API Manager.



1. API Consumer login to API Store
2. User permissions are checked to enable/disable application creation functionality.
3. If the user is permitted, the user creates an application specifying the application throttling tiers etc.
4. Application creation request is sent to relevant REST API with authorization header defined with access token.
5. REST API access tokens are maintained in a Cache. But if the key cannot be found in the key cache, it will contact Key Manager services to validate the key.
6. If the token is valid, the application will be created and persisted in API Manager Core.

Subscribe to an API

Published APIs are available in API Store, where the API consumers can register themselves, explore, subscribe and consume the APIs. In API Manager, it has this concept of subscribing to an API or APIs via the applications. Following steps illustrate how the key components in API Manager 3.0.0 interact during an API subscription.



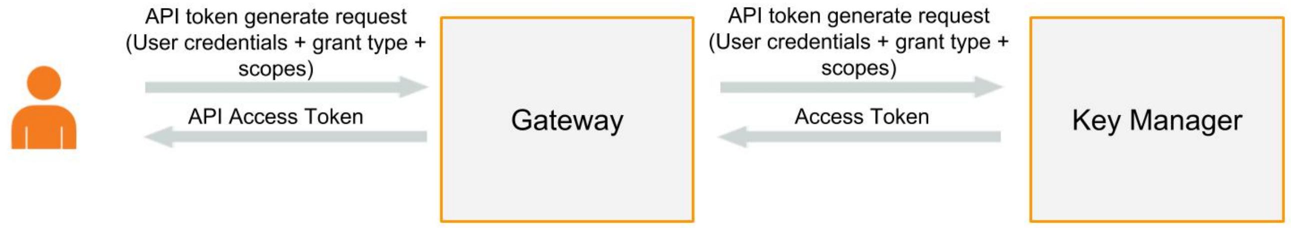
1. An API consumer logs in to API Store.
2. User permissions, API visibility and subscription availability are checked to enable/disable subscription functionality and subscription capability of a given API.
3. API consumer selects the API which he/she wants to subscribe and make subscription under a preferred application.
4. API subscription request with API details and application details is sent to relevant Store Rest API with the Authorization header, which was defined using access token cookie extracted from the browser.
5. REST API access tokens are maintained in a Cache. But if the key cannot be found in the key cache, it will contact Key Manager services to validate the key.
6. If the token is valid, the API subscription is persisted in API Core.
7. If the subscription is successful, it is published to API subscription topic in Broker.
8. Broker broadcasts the API subscription to all the Gateways who have been subscribed to that particular broker topic. Then the Gateways will know who are subscribed to a given API and who should be allowed to pass through.

Generate API Access Token

An **access token** is a simple string that is passed as an HTTP header of a request (For example, "`Authorization : Bearer NtBQkXoKElu0H1a1fQ0DWfo6IX4a`.Access tokens authenticate API users and applications, and ensure better security (e.g., prevent **DoS attacks**). If a token that is passed with a request is invalid, the request is discarded in the first stage of processing.

As mentioned earlier, API Manager uses OAuth 2.0.0 protocol specification for token management in Key Manager. API Gateway provides a Token API which can be used to generate, renew or revoke access tokens. API consumers need those access tokens to invoke APIs subscribed under an application so that the access tokens are passed in the HTTP header when invoking APIs.

Following is the component interaction during an API access token generation time.

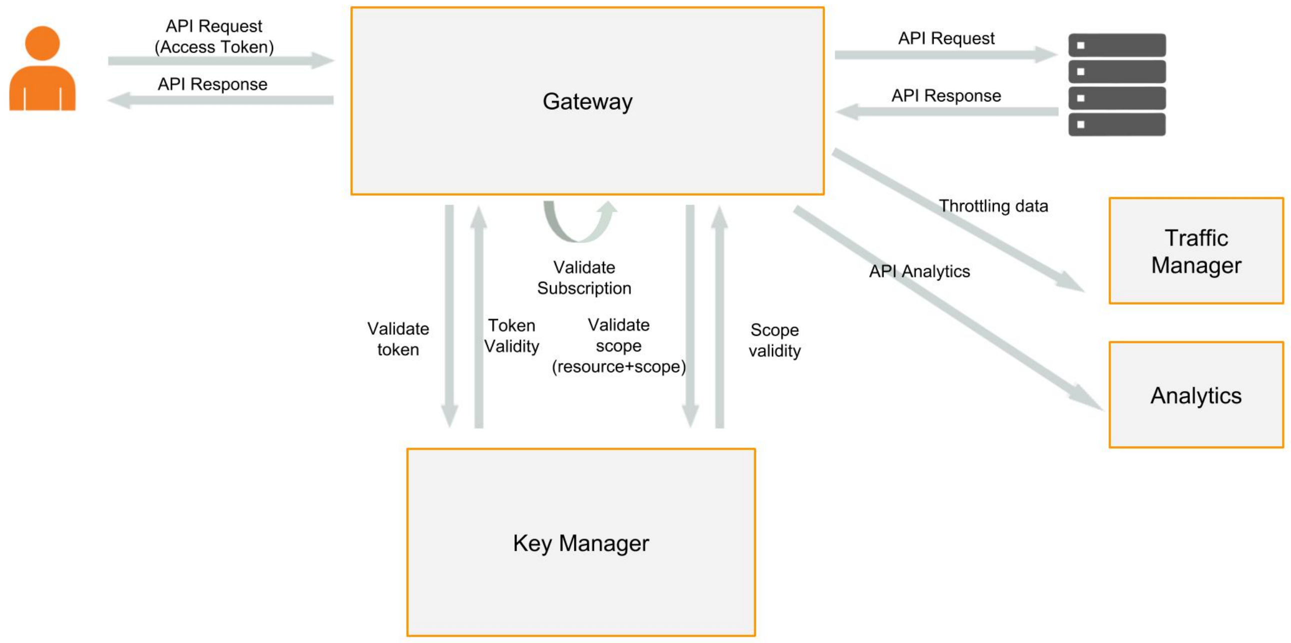

API token generate request (User credentials + grant type + scopes)
API Access Token
Gateway
API token generate request (User credentials + grant type + scopes)
Access Token
Key Manager

1. API token generation request is sent to Gateway token API with details like user credentials, OAuth grant type, scopes etc. This step can be performed as an authenticated user in API Store web application or directly calling token generation API.
2. Gateway token API calls Key Manager token generation services to authenticate the user and generate access token based on the given grant type and OAuth scopes. During this step, OAuth service provider will be created in key manager component againts the subscription.
3. The access token is returned to API consumer.

Revoking and refreshing token will follow the same flow as in token generation.

Invoking an API

After generating an access token, the consumers can simply invoke the API/APIs subscribed under the application. Below diagram depicts how the components are correlated to manage API throttling, collect statistics and enforce security.



1. API invocation request is sent with API access token embedded the Authorization header
2. Gateway receives the request and contacts Key Manager to authenticate user and validate the token. This is handled by gateway security handler. If the user is not authenticated or the tokens are not valid, the API request processing is stopped error responses are retured to the user.
3. If the token is valid, the gateway validates the API subscription. API Gateway retrieves API subscriptions from API Core(During server startup) and via broker topics. If the subscription is invalid, API request processing is

stopped relevant error responses are returned to the user.

4. Then the token scopes are validated against the API resources. If the scope validation fails, it will stop processing the request.
5. The request is validate againts the defined throttling policies. This is managed by Throttling handler in Gateway and Traffic Manager.
6. After validating the request, it will pass through gateway mediation flow and finally, it will be forwarded to the actual backend server.
7. The response from actual backend will go through the set of handlers and the response is sent to the user
8. All API statistics will be published to Analytics component via Gateway Analytics handler. The Analytics component is responsible for generating statistics based on the collested raw data.

Installation Guide

Follow the instructions below to download, install and set up the WSO2 API Manager.

Prerequisites

- Download and install JDK 8 update 40 or later
  You must set your `JAVA_HOME` environment variable to point to the directory where the Java Development Kit (JDK) is installed on the computer. Environment variables are global system variables accessible by all the processes running under the operating system.

  Click here to expand instructions for setting the environment variable in Linux, OSX and Windows.
  - Linux or OSX
  - **Windows**

  These instructions set the `JAVA_HOME` environment variable in Linux and OSX.

  1. In your home directory, open the BASHRC file (.bash_profile file on Mac) using editors such as vi, emacs, pico, or mcedit.
  2. Assuming you have **JDK 1.8.0_111** in your system, add the following two lines at the bottom of the file, replacing `/usr/java/jdk1.8.0_111` with the actual directory where the JDK is installed.

     ```
     On Linux:
     export JAVA_HOME=/usr/java/jdk1.8.0_111
     export PATH=${JAVA_HOME}/bin:${PATH}

     On OS X:
     export
     JAVA_HOME=/Library/Java/JavaVirtualMachines/jdk1.8.0_111/Content
     s/Home
     ```

  3. Save the file.

     If you do not know how to work with text editors in a Linux SSH session, run the following command: `cat >> .bashrc.` Paste the string from the clipboard and press "Ctrl+D."

  4. To verify that the `JAVA_HOME` variable is set correctly, execute the following command:

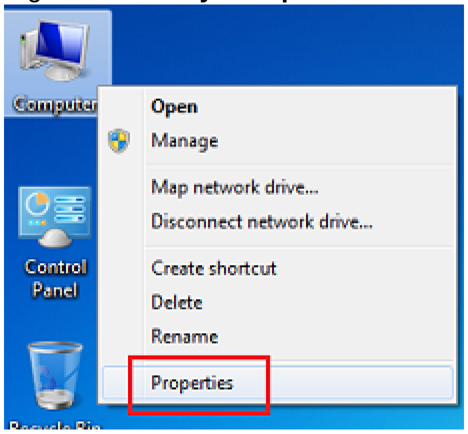
     ```
     On Linux:
     echo $JAVA_HOME

     On OS X:
     which java

     If the above command gives you a path like /usr/bin/java, then it
     is a symbolic link to the real location. To get the real
     location, run the following:
     ls -l `which java`
     ```

  5. The system returns the JDK installation path.

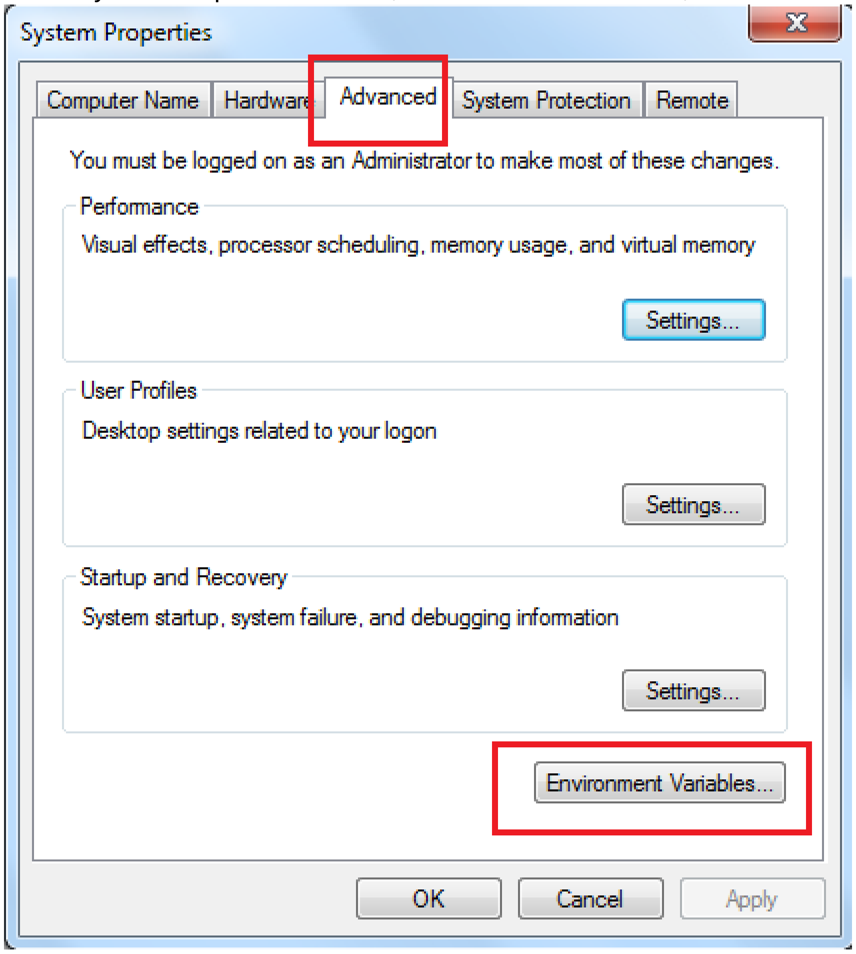
Typically, the JDK is installed in a directory under `C:/Program Files/Java, such as C:/Program Files/Java/jdk1.8.0_111`. If you have multiple versions installed, choose the latest one, which you can find by sorting by date. The following are some options to configure this.

Setting up JAVA_HOME using the system properties

1. Right-click the **My Computer** icon on the desktop and click **Properties**.



2. In the System Properties window, click the **Advanced** tab, and then click **Environment Variables**.



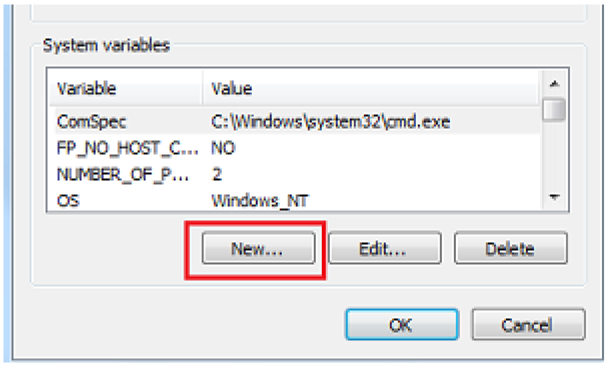
You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory usage, and virtual memory

Desktop settings related to your logon

System startup, system failure, and debugging information

3. Click **New** under **System variables** (for all users) or under **User variables** (just for the user who is currently logged in).



Variable	Value
ComSpec	C:\Windows\system32\cmd.exe
FP_NO_HOST_C...	NO
NUMBER_OF_P...	2
OS	Windows_NT

4. Enter the following information:
    - In the **Variable name** field, enter: `JAVA_HOME`
    - In the **Variable value** field, enter the installation path of the Java Development Kit, such as: `c:/Program Files/Java jdk1.6.0_27`

The `JAVA_HOME` variable is now set and will apply to any subsequent command prompt windows you open. If you have existing command prompt windows running, you must close and reopen them for the `JAVA_HOME` variable to take effect, or manually set the `JAVA_HOME` variable in those command prompt windows as described in the next section. To verify that the `JAVA_HOME` variable is set correctly, open a command window (from the **Start** menu, click **Run**, and then type `CMD` and click **Enter**) and execute the following command:

```
set JAVA_HOME
```

The system returns the JDK installation path. You are now ready to run the product.

Setting JAVA_HOME temporarily using the Windows command prompt (CMD)

You can temporarily set the `JAVA_HOME` environment variable within a Windows command prompt window (CMD). This is useful when you have an existing command prompt window running and you do not want to restart it.

1. In the command prompt window, enter the following command where `<JDK_INSTALLATION_PATH>` is the JDK installation directory and press **Enter.**

   ```
   set JAVA_HOME=<JDK_INSTALLATION_PATH>
   ```

   For example: `set JAVA_HOME=c:/Program Files/java/jdk1.6.0_27`

   The JAVA_HOME variable is now set for the current CMD session only.

2. To verify that the `JAVA_HOME` variable is set correctly, execute the following command:

   ```
   set JAVA_HOME
   ```

3. The system returns the JDK installation path.

- Download and install Apache ActiveMQ 5.14.0
- Download WSO2 Identity Server 5.3.0
- Download WSO2 APIM Data Analytics Server 3.0.0 M3
- Download WSO2 API Manager 3.0.0 M3
- Download WSO2 API Manager Gateway 3.0.0 M3

Setting up the environment

Start the servers given below in the following order.

Use the default port offset for all the servers.

1. Run the following command to start ActiveMQ:

   ```
   ./bin/activemq start
   ```

2. Start WSO2 Identity Server using one of the commands given below:
   - On Windows: `<IS_HOME>/bin/wso2server.bat --run`
   - On Linux/OSX: `sh <IS_HOME>/bin/wso2server.sh`
3. Start WSO2 APIM Data Analytics Server using one of the commands given below:
   - On Windows: `<API-M_DAS_HOME>/bin/worker.bat --run`
   - On Linux/OSX: `sh <API-M_DAS_HOME>/bin/worker.sh`
4. Start WSO2 API Manager using one of the commands given below:
   - On Windows: `<API-M_HOME>/bin/carbon.bat --run`
   - On Linux/OSX: `sh <API-M_HOME>/bin/carbon.sh`

   Make sure you start the Gateway at the end, after starting all the other servers.

5. Run the following command from `<GATEWAY_HOME>` to start the Gateway:

   ```
   bin/ballerina run service org/wso2/carbon/apimgt/gateway
   ```

Quick Start Guide

WSO2 API Manager is a complete solution for designing and publishing APIs, creating and managing a developer community, and for securing and routing API traffic in a scalable way. It leverages proven components from the WSO2 platform to secure, integrate and manage APIs. In addition, it integrates with the WSO2 analytics platform an d provides out of the box reports and alerts, giving you instant insights into the APIs behavior.

The quick start helps you get started with the WSO2 API Manager.

Accessing the API Publisher | Accessing the API Store

Before you begin:

Make sure you have followed the installation guide and set up the API Manager.

Accessing the API Publisher

WSO2 API Manager provides a simple web interface called **WSO2 API Publisher** for API development and management. It is a structured GUI designed for API creators to design, develop and manage APIs.

Once the server has started, you can access the API Publisher by typing its URL in a web browser. The URL is in the following format: `https://<Server Host>:9292/publisher`

You can use this URL to access the API Publisher on this computer from any other computer connected to the Internet or LAN. When accessing the API Publisher from the same server where it is installed, you can type `localh ost` instead of the IP address as follows: `https://localhost:9292/publisher`

To log into the API Publisher:

1. Open the API Publisher (`https://<hostname>:9292/publisher`) and sign in with **`admin/admin`** crede ntials.
2. Once you are authenticated, you are directed back to the API Publisher API listing page, where you can create APIs.

Accessing the API Store

WSO2 API Manager provides a simple web interface called **WSO2 API Store** for consuming the managed APIs by application developers. It is a structured GUI designed for API developers to browse and subscribe APIs in to their applications for consuming at run time.

To log into the API Store:

1. Open the API Store (`https://<hostname>:9292/store`) and sign in with **`admin/admin`** credentials.
2. Once you are authenticated, you are directed back to the API Store API listing page, where you can consume APIs.

Tutorials

This section covers the following use cases of the product:

- Create and Publish an API
- Subscribe to an API
- Invoke an API

Create and Publish an API

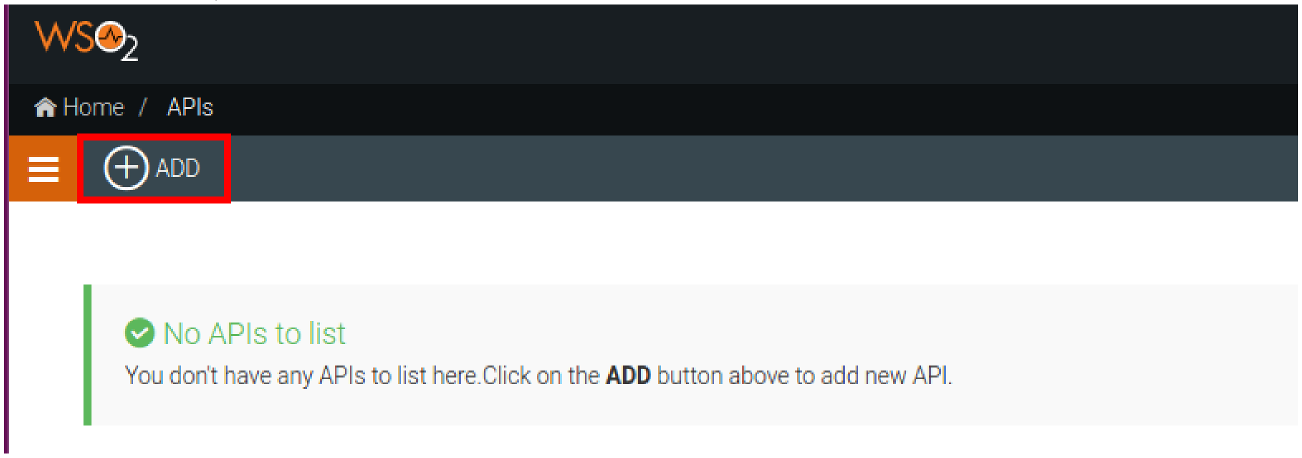
Before you begin:

Make sure you have followed the installation guide and set up the API Manager.

API creation is the process of linking an existing backend API implementation to the API Publisher so that you can manage and monitor the API's lifecycle, documentation, security, community and subscriptions.

The steps below demonstrate how to create a new API.

1. Sign in to the WSO2 API Publisher (`https://<hostname>:9292/publisher`)
2. In the **APIS** menu, click **ADD**.



No APIs to list

You don't have any APIs to list here.Click on the **ADD** button above to add new API.

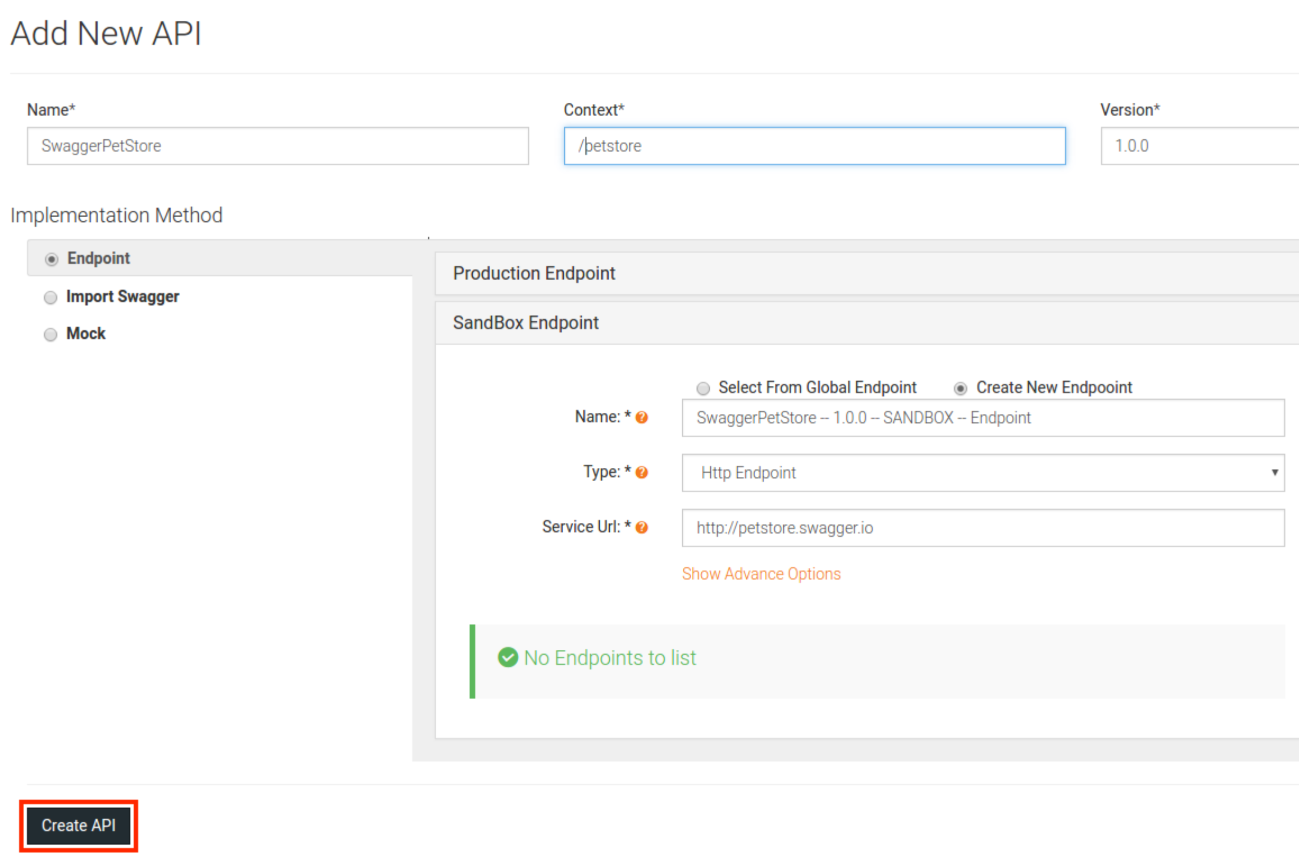
3. Give the information in the table below and click **Create API**.

Field	Sample value
Name	SwaggerPetstore
Context	/petstore
Version	1.0.0
Endpoint	`http://petstore.swagger.io`

Always use "/" before the API context you create as shown in the above example.

Make sure you enter both the Production and Sandbox endpoints to avoid errors when restarting the

Gateway.



Add New API

Name*: SwaggerPetStore
Context*: /petstore
Version*: 1.0.0

Implementation Method

(● Endpoint) Import Swagger / Mock

Production Endpoint

SandBox Endpoint

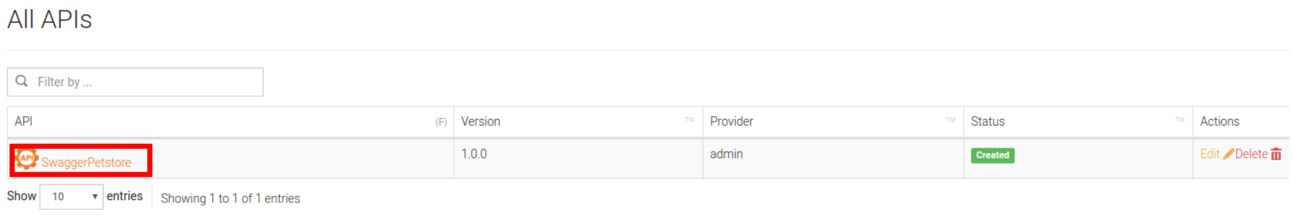
Select From Global Endpoint [✓ Create New Endpoint]

Name: * SwaggerPetStore -- 1.0.0 -- SANDBOX -- Endpoint
Type: * Http Endpoint
Service Url: * http://petstore.swagger.io

Show Advance Options

No Endpoints to list

4. Select the added API from the API list.



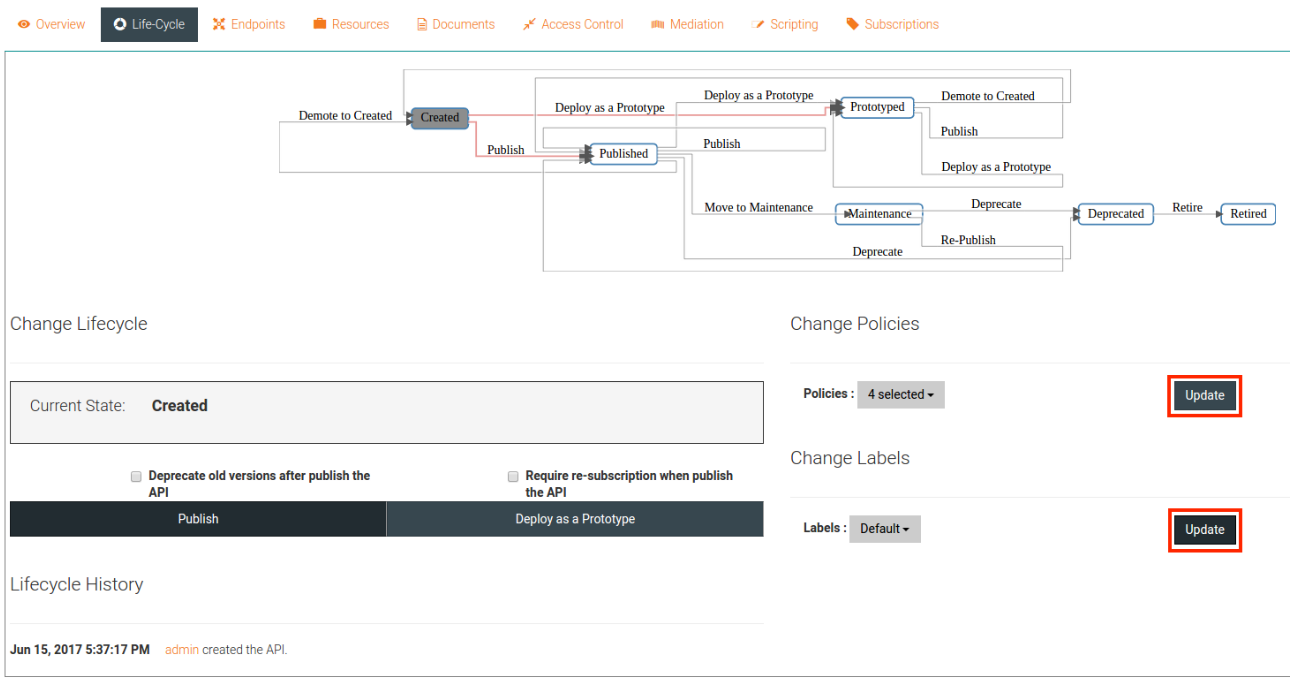
All APIs

API	Version	Provider	Status	Actions
SwaggerPetstore	1.0.0	admin	Created	Edit Delete

Show 10 entries Showing 1 to 1 of 1 entries

The API can be available at different levels of service. They allow you to limit the number of successful hits to an API during a given period of time.

5. In the **Lifecycle** tab, select all the subscription policies available and click **Update**.
6. Under the **Change Labels** area, select **Default** and click **Update**.



Change Lifecycle

Current State: **Created**

☐ **Deprecate old versions after publish the API**

☐ **Require re-subscription when publish the API**

Change Policies

Policies : 4 selected

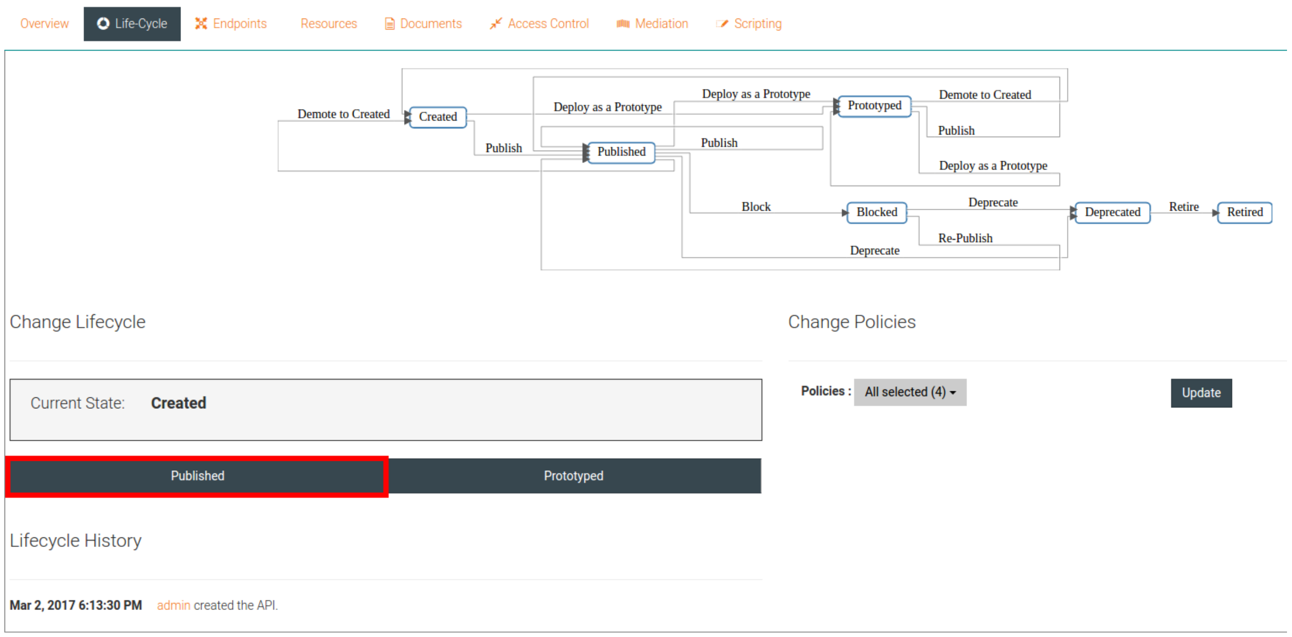
Change Labels

Labels : Default

Lifecycle History

Jun 15, 2017 5:37:17 PM admin created the API.

7. Click **Published**.



Change Lifecycle

Current State: **Created**

Change Policies

Policies : All selected (4)

Lifecycle History

Mar 2, 2017 6:13:30 PM admin created the API.

This publishes the API that you just created to the API Store so that subscribers can use it. You have successfully created an API.

Subscribe to an API

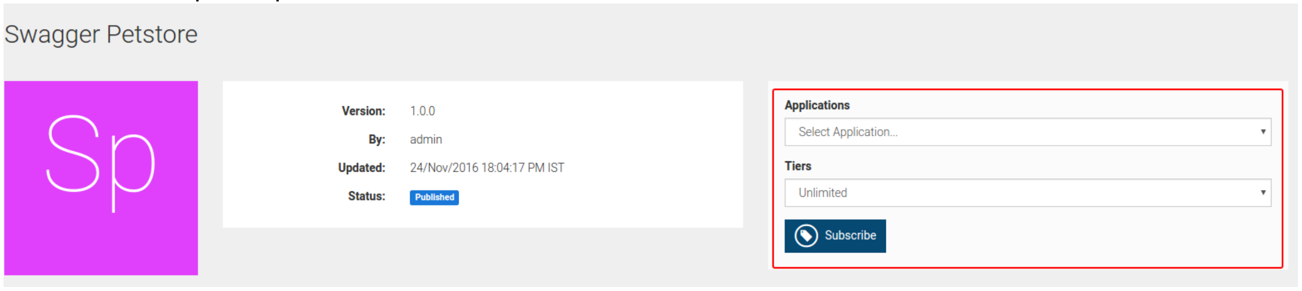
Before you begin:

- Make sure you have followed the installation guide and set up the API Manager.
- Make sure you have created and published an API.

You **subscribe** to a published API before using it in your applications. Subscription enables you to receive access tokens and be authenticated to invoke the API.

The examples here use the Swagger Pet store REST API, which is created in the section Create and Publish an API.

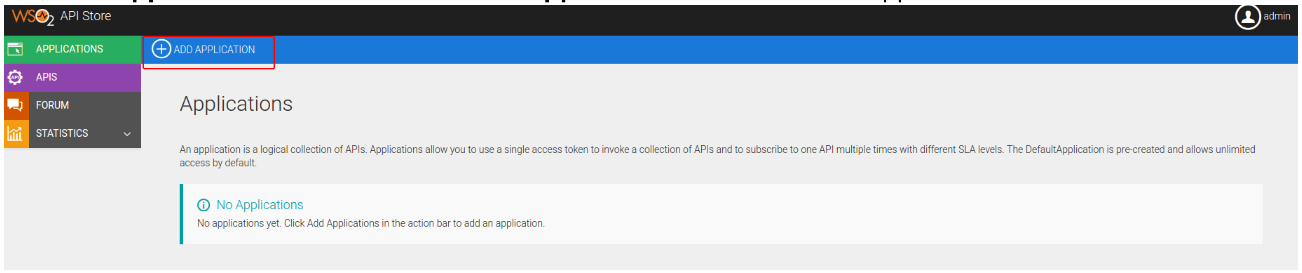
1. Sign in to the WSO2 API Store (`https://<hostname>:9292/store`) and click on an API (e.g., Swagger Petstore) to open it.
2. Note the subscription options for the REST API.


Swagger Petstore

Version: 1.0.0
By: admin
Updated: 24/Nov/2016 18:04:17 PM IST
Status: Published

Applications: Select Application...
Tiers: Unlimited

3. Click the **Applications** menu and click **Add Application** to create a new application.

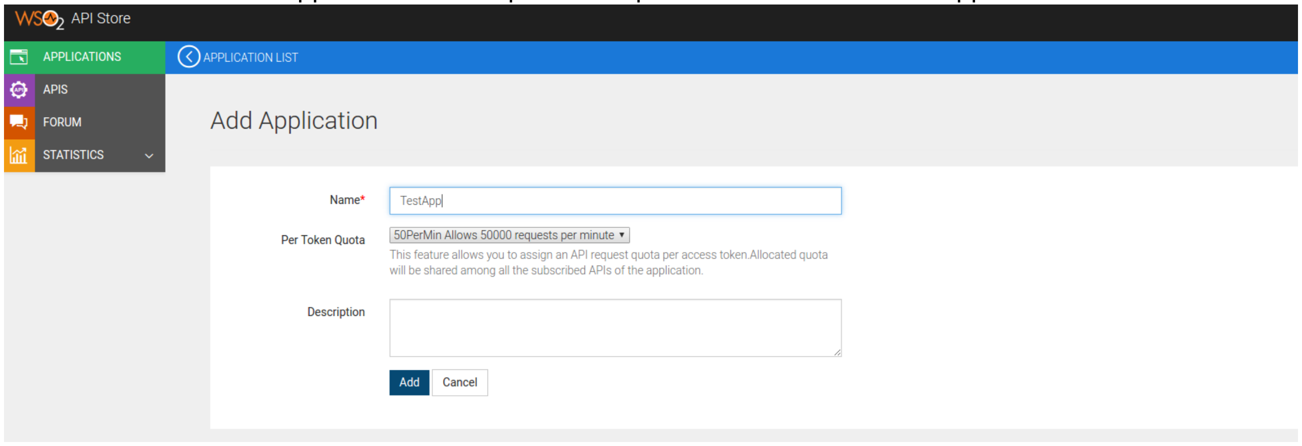


Applications

An application is a logical collection of APIs. Applications allow you to use a single access token to invoke a collection of APIs and to subscribe to one API multiple times with different SLA levels. The DefaultApplication is pre-created and allows unlimited access by default.

No Applications
No applications yet. Click Add Applications in the action bar to add an application.

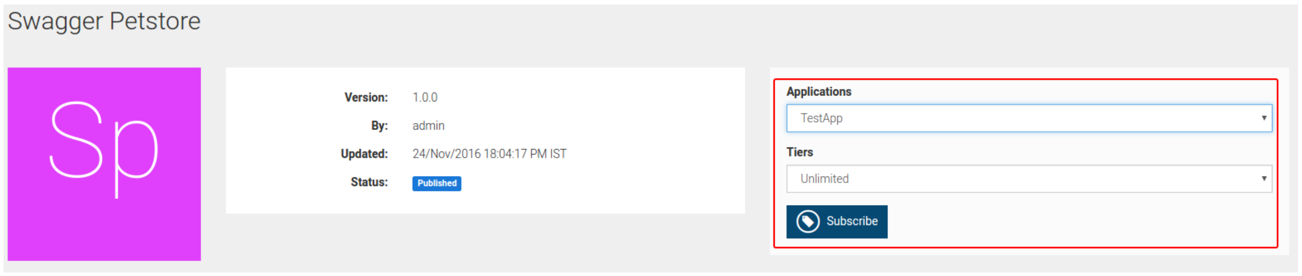
4. Enter the name as TestApp and select the per token quota as 50PerMin for the application and click **Add**.



Add Application

Name*: TestApp
Per Token Quota: 50PerMin Allows 50000 requests per minute
This feature allows you to assign an API request quota per access token.Allocated quota will be shared among all the subscribed APIs of the application.
Description:

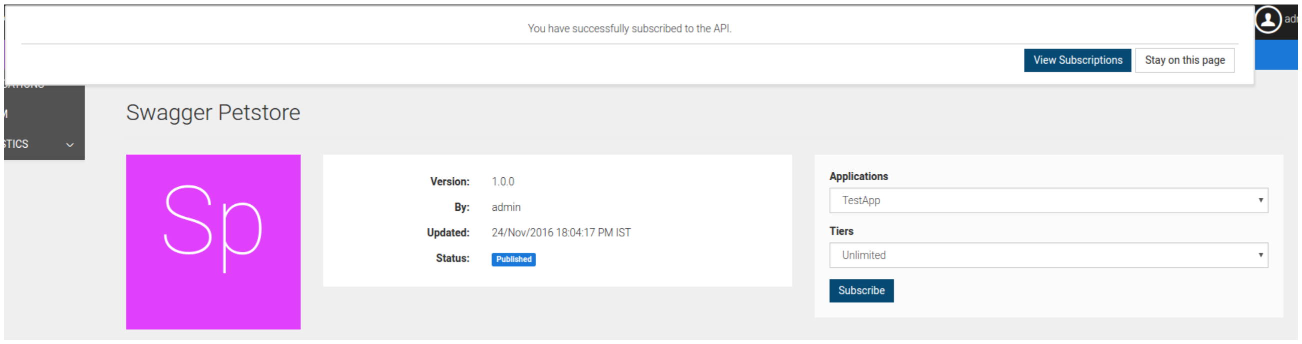
5. Click **APIs** and click on the Swagger Pet store API to view the API's subscription options.
6. Select the application that you just created, a tier, and click **Subscribe**.


Swagger Petstore

Version: 1.0.0
By: admin
Updated: 24/Nov/2016 18:04:17 PM IST
Status: Published

Applications: TestApp
Tiers: Unlimited

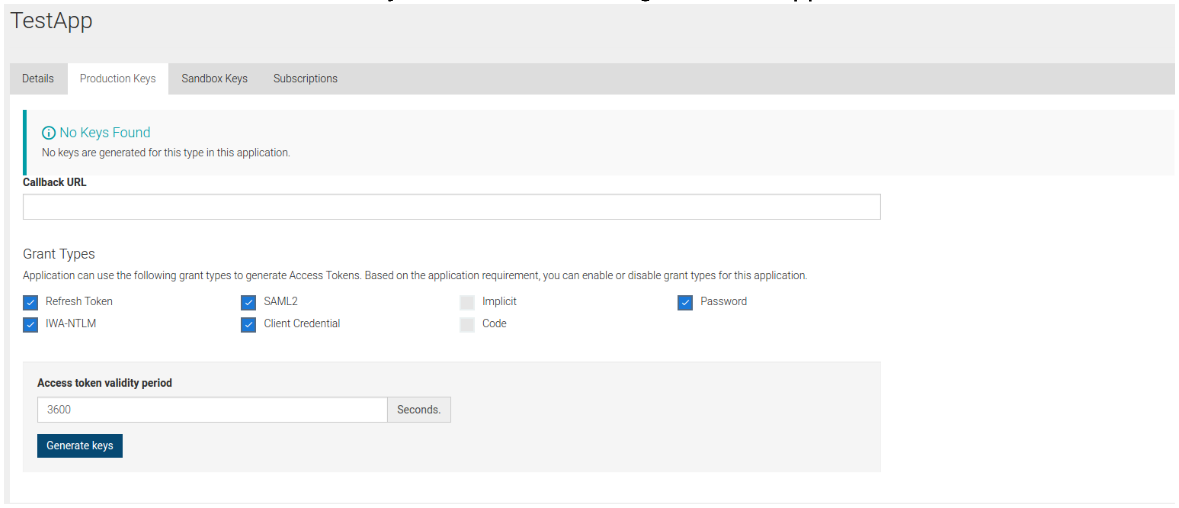
7. Click the **View Subscriptions** button when prompted.
   The **Subscriptions** tab opens.


You have successfully subscribed to the API.

Version: 1.0.0
By: admin
Updated: 24/Nov/2016 18:04:17 PM IST
Status: Published

Applications: TestApp
Tiers: Unlimited

8. Click the **Production Keys** tab and click **Generate Keys** to create an application access token. You can use this token to invoke all APIs that you subscribe to using the same application.



No Keys Found
No keys are generated for this type in this application.

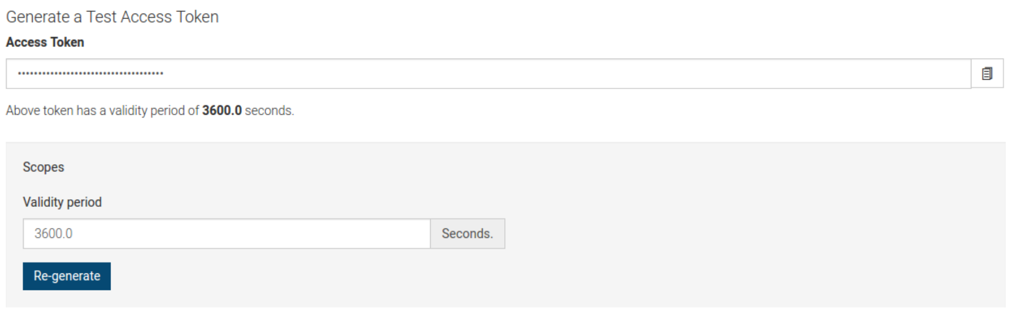
Callback URL

Grant Types
Application can use the following grant types to generate Access Tokens. Based on the application requirement, you can enable or disable grant types for this application.

✓ Refresh Token	✓ SAML2	Implicit	✓ Password
✓ IWA-NTLM	✓ Client Credential	Code	

Access token validity period: 3600 Seconds.

9. Keep a copy of the generated access token for future reference. You use this token later to invoke APIs.



Access Token: ••••••••••••••••••••••••••••••••••••••

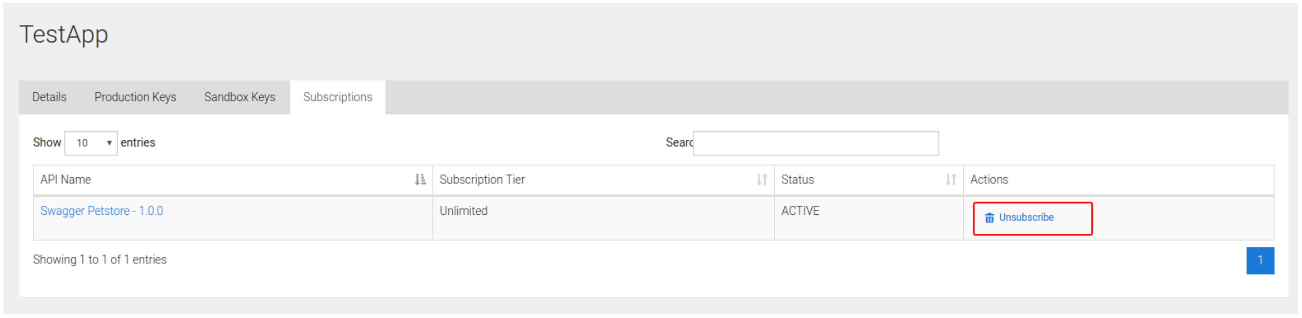
Above token has a validity period of **3600.0** seconds.

Scopes

Validity period: 3600.0 Seconds.

You have successfully subscribed to an API and got the access token as well.

10. **To unsubscribe from an API**, click the **Applications** menu and click **View** next to the application used for the subscription. Go to the **Subscriptions** tab, locate the API, and click the **Unsubscribe** link associated with it.



API Name	Subscription Tier	Status	Actions
Swagger Petstore - 1.0.0	Unlimited	ACTIVE	Unsubscribe

Showing 1 to 1 of 1 entries

11. Restart the Gateway using the following command:

```
bin/ballerina run service org/wso2/carbon/apimgt/gateway
```

Invoke an API

Before you begin:

- Make sure you have followed the installation guide and set up the API Manager.
- Make sure you have created, published and subscribed to an API.

Once a new API is published, the Gateway needs to be restarted for the APIs to be deployed. To see how to start the Gateway, see step 5 of the installation guide.

1. Generate an access token for the application that is subscribed to the API that you wish to invoke.

   Make sure you back up the access token.

2. Invoke the API using the following CURL command.

   ```
   curl -k -X GET --header 'Accept: application/json' --header
   'Authorization: Bearer {access_token}'
   'https://localhost:9092/{api_context}'
   ```

   For example,

   `curl -k -X GET --header 'Accept: application/json' --header 'Authorization: Bearer 6bba7fa7-3005-39f8-8495-98f78eae338b' 'https://localhost:9092/test'`