

# Quick Start Guide

WSO2 API Manager is a complete solution for publishing APIs, creating and managing a developer community, and for routing API traffic in a scalable manner. It leverages the integration, security and governance components from the WSO2 Enterprise Service Bus, WSO2 Identity Server, and WSO2 Governance Registry. In addition, as it is powered by WSO2 Business Activity Monitor (BAM), WSO2 API Manager is ready for massively scalable deployments immediately.

## ✔ Before you begin,

1. Install [Oracle Java SE Development Kit \(JDK\)](#) version 1.6.24 or later or 1.7.\* and set the JAVA\_HOME environment variable.
2. [Download](#) WSO2 API Manager.
3. Start the API Manager by going to <APIM\_HOME>/bin using the command line and executing `wso2server.bat` (for Windows) or `wso2server.sh` (for Linux).

Let's go through the use cases of the API Manager:

- Invoking your first API
- Understanding the API Manager concepts
- Deep diving into the API Manager



# Invoking your first API

- 1 Open the API Publisher (<https://<hostname>:9443/publisher>) and log in with admin/admin credentials.
- 2 Click the **Deploy Sample API** button. It deploys a sample API called WeatherAPI into the API Manager.

WSO2 API PUBLISHER

APIs

Browse

Add

All Statistics

My APIs

Subscriptions

Statistics

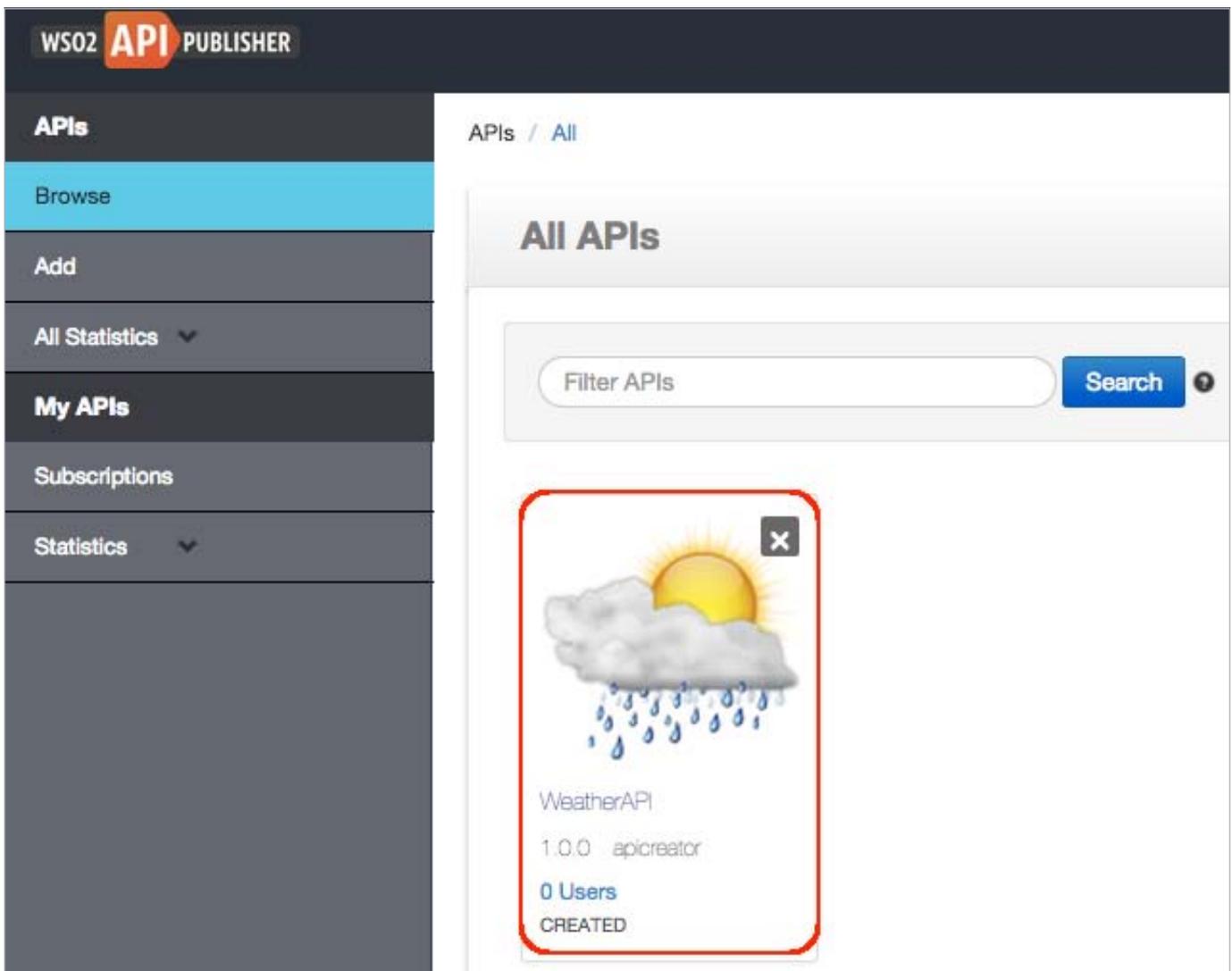
APIs / All

All APIs

No APIs created yet. Click the button below to get started.

New API... Deploy Sample API

3 Click WeatherAPI to open it.



Let's publish this API.

- 4 Go to the **Lifecycle** tab and note that the **State** is PUBLISHED. The API is already published to the API Store.

The screenshot shows the WS02 API PUBLISHER interface. On the left, there is a sidebar with 'APIs', 'Browse', and 'All Statistics'. The main content area shows the 'WeatherAPI - 1.0.0' management page. The 'Lifecycle' tab is selected and highlighted with a red box. Below the tabs, the 'State' is set to 'PUBLISHED', also highlighted with a red box. There is a checked checkbox for 'Propagate Changes to API Gateway' and 'Update' and 'Reset' buttons at the bottom.

- 5 Log in to the API Store (<https://<hostname>:9443/store>) with **admin/admin** credentials and note that WeatherAPI is visible under the **APIs** menu. Click it to open the API.

The screenshot shows the WS02 API STORE interface. The top navigation bar includes 'APIs', 'Prototyped APIs', 'My Applications', 'My Subscriptions', 'Forum', and 'Statistics'. The 'APIs' menu item is highlighted with a red box. Below the navigation, there is a search bar and a 'Recently Added' section. The 'APIs' section displays a card for 'WeatherAPI-1.0.0' by 'apicreator', which is highlighted with a red box. The card features a weather icon and a five-star rating.

- 6 The subscription options are on the right-hand side of the page. Select the default application and an available tier, and click **Subscribe**.

The screenshot shows the WSO2 API Store interface. At the top, there is a navigation bar with 'API STORE' and several menu items: 'APIs', 'Prototyped APIs', 'My Applications', 'My Subscriptions', 'Forum', 'Statistics', 'Themes', and 'TestUser'. Below the navigation bar is a search bar labeled 'Search API'. The main content area displays details for 'WeatherAPI - 1.0.0' by 'apicreator'. The details include a weather icon, a rating of 'Your rating: N/A' with five stars, a version of '1.0.0', a status of 'PUBLISHED', and an update date of '15/May/2015 15:26:35 PM IST'. On the right side, there are two dropdown menus: 'Applications' (set to 'DefaultApplication') and 'Tiers' (set to 'Unlimited'). Below these is a 'Subscribe' button. A red box highlights the 'Applications' and 'Tiers' dropdowns and the 'Subscribe' button. At the bottom, there are tabs for 'Overview', 'Documentation', 'API Console', and 'Throttling Info'.

- 7 When the subscription is successful, go to the **My Subscriptions** page and click the **Generate keys** button to generate an access token to invoke the API.

The screenshot shows the 'My Subscriptions' page in the WSO2 API Store. The navigation bar at the top has 'My Subscriptions' highlighted with a red box. Below the navigation bar is a search bar labeled 'Search API'. The main content area is titled 'Subscriptions' and contains the following text: 'Create access tokens to applications. Because an application is a logical collection of APIs, you can use a single access token to invoke multiple APIs and to subscribe to one API multiple times with different SLA levels.' Below this text is a section titled 'Applications With Subscriptions' with a dropdown menu set to 'DefaultApplication' and a 'Show Keys' checkbox. Underneath, there are two sections: 'Keys - Production' and 'Keys - Sandbox'. The 'Keys - Production' section contains the text 'Production keys are not yet generated for this application.' and a 'Generate keys' button highlighted with a red box. To the right of this section is an 'Allowed Domains' section with a text input field containing 'ALL' and a 'Token Validity' section with a text input field containing '3600' and a 'Seconds' label.

- Click the **APIs** menu in the API Store again and then click the API to open it. When the API opens, click its **API Console** tab.

**WeatherAPI - 1.0.0**

admin



**Rating:** Your rating: N/A  
★★★★★

**Version:** 1.0.0

**Status:** PUBLISHED

**Updated:** 15/May/2015 16:38:57 PM IST

**Applications**  
Select Application...

**Tiers**  
Unlimited  
Allows unlimited requests

[Subscribe](#)

Overview | Documentation | **API Console** | Throttling Info

Try: DefaultApplication On Production Environment.

Set Request Header: Authorization : Bearer 52fea1671e65033b465f57531a2b80

**WeatherAPI** [Swagger \( /swagger.json \)](#)

default [Show/Hide](#) [List Operations](#) [Expand Operations](#)

**GET /**

**OPTIONS /**

- Expand the GET method, give the parameter value as "London", and click **Try it out**.

**GET /**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
q	London	Name of the City	query	string

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200			

[Try it out!](#)

- 10 Note the response for the API invocation. It returns the weather in London.

### Response Body

```
{
  "coord": {
    "lon": -81.23,
    "lat": 42.98
  },
  "sys": {
    "message": 0.013,
    "country": "CA",
    "sunrise": 1431684065,
    "sunset": 1431736888
  },
  "weather": [
    {
      "id": 804,
      "main": "Clouds",
      "description": "overcast clouds",
      "icon": "04d"
    }
  ],
  "base": "stations",
  "main": {
    "temp": 285.05
```

You have deployed a sample API, published it to the API Store, subscribed to it, and invoked the API using our integrated API Console.



# Understanding the API Manager concepts

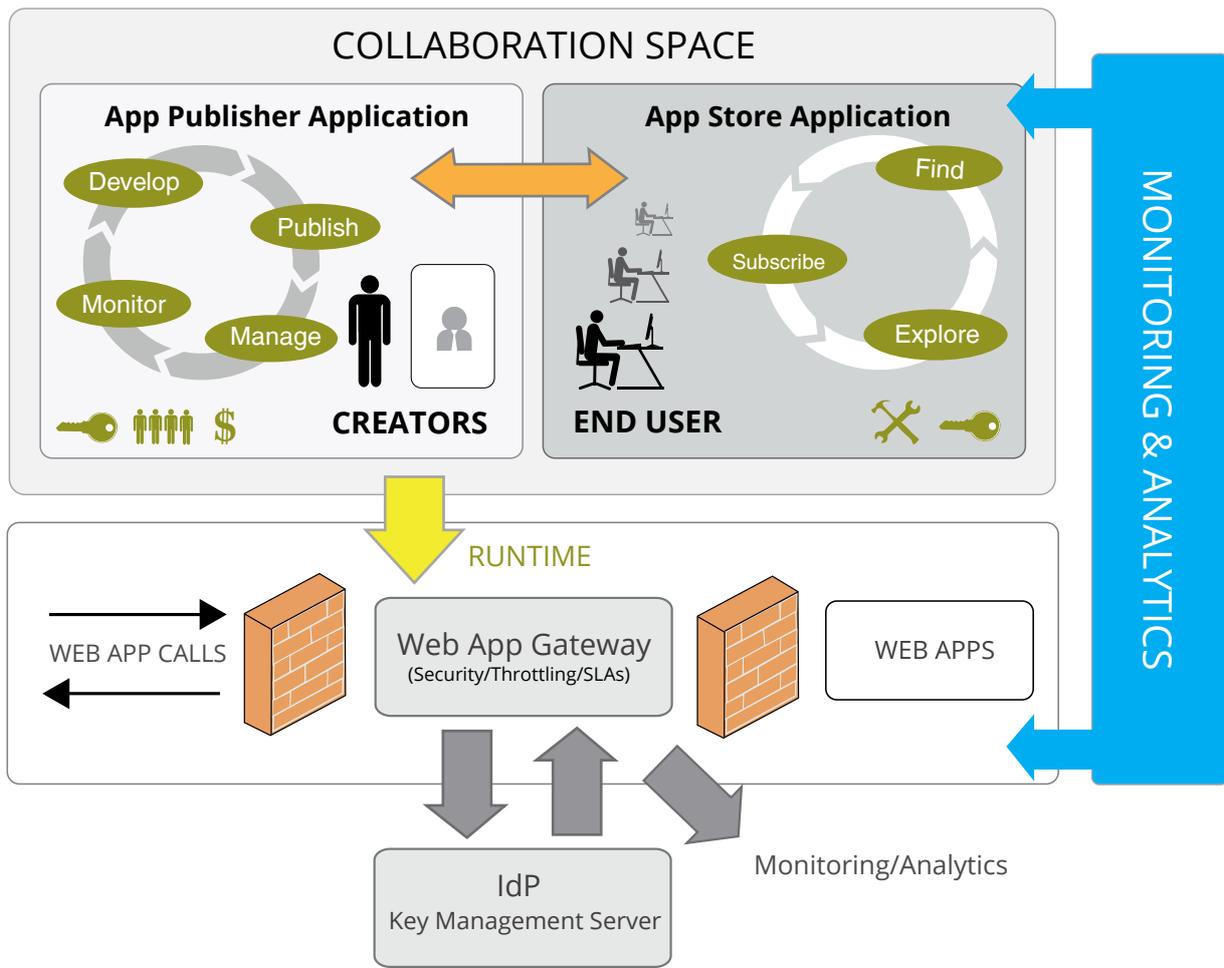
Before we look into the API management activities in detail, let's take a look at the basic API management concepts.

[\[ Components \]](#) [\[ Users and roles \]](#) [\[ API lifecycle \]](#) [\[ Applications \]](#) [\[ Throttling tiers \]](#) [\[ API keys \]](#)  
[\[ Application access tokens \]](#) [\[ Application user access token \]](#) [\[ API resources \]](#)

## Components

The API Manager comprises the following components:

- **API Gateway:** Secures, protects, manages, and scales API calls. It is a simple API proxy that intercepts API requests and applies policies such as throttling and security checks. It is also instrumental in gathering API usage statistics. The Web interface can be accessed via `https://<Server Host>:9443/carbon`.
- **Key Manager:** Handles all security and key-related operations. API gateway connects with the Key Manager to check the validity of subscriptions, OAuth tokens, and API invocations. The Key Manager also provides a token API to generate OAuth tokens that can be accessed via the Gateway.
- **API Publisher:** Enables API providers to publish APIs, share documentation, provision API keys, and gather feedback on features, quality, and usage. You access the Web interface via `https://<Server Host>:9443/publisher`.
- **API Store:** Enables API consumers to self register, discover and subscribe to APIs, evaluate them, and interact with API publishers. You access the Web interface via `https://<Server Host>:9443/store`.



## Users and roles

The API manager offers three distinct community roles that are applicable to most enterprises:

- **Creator:** A creator is a person in a technical role who understands the technical aspects of the API (interfaces, documentation, versions, how it is exposed by the Gateway, etc.) and uses the API publisher to provision APIs into the API Store. The creator uses the API Store to consult ratings and feedback provided by API users. Creators can add APIs to the store but cannot manage their life cycle (e.g., make them visible to the outside world).
- **Publisher:** A publisher manages a set of APIs across the enterprise or business unit and controls the API life cycle and monetization aspects. The publisher is also interested in usage patterns for APIs and has access to all API statistics.
- **Consumer:** A consumer uses the API Store to discover APIs, see the documentation and forums, and rate/comment on the APIs. Consumers subscribe to APIs to obtain API keys.

## API lifecycle

An API is the published interface, while the service is the implementation running in the backend. APIs have their own life cycles that are independent of the backend services they rely on. This life cycle is exposed in the API Publisher Web interface and is managed by the publisher role.

The following stages are available in the default API life cycle:

- **Created:** AAPI metadata is added to the API Store, but it is not visible to subscribers yet, nor deployed to the API Gateway.
- **Prototyped:** The API is deployed and published in the API Store as a prototype. A prototyped API is usually a mock implementation made public in order to get feedback about its usability. Users can try out a prototyped API without subscribing to it.
- **Published:** The API is visible in the API Store and available for subscription.
- **Deprecated:** The API is still deployed in the API Gateway (i.e., available at runtime to existing users) but not visible to subscribers. You can deprecate an API automatically when a new version of it is published.
- **Retired:** The API is unpublished from the API Gateway and deleted from the Store.
- **Blocked:** Access to the API is temporarily blocked. Runtime calls are blocked, and the API is not shown in the API Store anymore.

You can manage the API and service life cycles in the same governance registry/repository and automatically link them. This feature is available in WSO2 Governance Registry (version 4.5 onwards).

## Applications

An application is primarily used to decouple the consumer from the APIs. It allows you to do the following:

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

You create an application to subscribe to an API. The API Manager comes with a default application, and you can also create as many applications as you like.

## Throttling tiers

Throttling tiers are associated with an API at subscription time. They define the throttling limits enforced by the API Gateway, e.g., 10 TPS (transactions per second). You define the list of tiers that are available for a given API at the publisher level. The API Manager comes with three predefined tiers (Gold/Silver/Bronze) and a special tier called Unlimited, which you can disable by editing the <TierManagement> element of the <APIM\_HOME>/repository/conf/api-manager.xml file.

## API keys

The API Manager supports two scenarios for authentication:

- An access token is used to identify and authenticate a whole application
- An access token is used to identify the final user of an application (for example, the final user of a mobile application deployed on many different devices)

## Application access tokens

Application access tokens are generated by the API consumer and must be passed in the incoming API requests. The API Manager uses the OAuth2 standard to provide key management. An API key is a simple string that you pass with an HTTP header (e.g., “Authorization: Bearer NtBQkXoKElu0H1a1fQ0DWfo6IX4a”), and it works equally well for SOAP and REST calls.

Application access tokens are generated at the application level and valid for all APIs that you associate with the application. These Tokens have a fixed expiration time, which is set to 60 minutes by default. You can change this to a longer time, even for several weeks. Consumers can regenerate the access token directly from the API Store. To change the default expiration time, you open the <APIM\_HOME>/repository/conf/identity.xml file and change the value of the element <ApplicationAccessTokenDefaultValidityPeriod>. If you set a negative value, the token never expires.

## Application user access tokens

You generate access tokens on demand using the Token API. In case a token expires, you use the Token API to refresh it.

Application user access tokens have a fixed expiration time, which is 60 minutes by default. You can update it to a longer time by editing the <ApplicationAccessTokenDefaultValidityPeriod> element in the <APIM\_HOME>/repository/conf/identity.xml file.

The token API takes the following parameters to generate the access token:

- Grant Type
- Username
- Password
- Scope

To generate a new access token, you issue a Token API call with the above parameters where grant\_type=password. The Token API then returns two tokens: an access token and a refresh token. The access token is saved in a session on the client side (the application itself does not need to manage users and passwords). On the API Gateway side, the access token is validated for each API call. When the token expires, you refresh the token by issuing a token API call with the above parameters where grant\_type=refresh\_token and passing the refresh token as a parameter.

## API resources

An API is made up of one or more resources. Each resource handles a particular type of request and is analogous to a method (function) in a larger API. API resources accept the following optional attributes:

- **verbs:** Specifies the HTTP verbs a particular resource accepts. Allowed values are GET, POST, PUT, OPTIONS, DELETE. You can give multiple values at once.
- **uri-template:** A URI template as defined in <http://tools.ietf.org/html/rfc6570> (e.g., /phoneverify/<phoneNumber>)

- **url-mapping:** A URL mapping defined as per the servlet specification (extension mappings, path mappings, and exact mappings)
- **Throttling tiers:** Limits the number of hits to a resource during a given period of time.
- **Auth-Type:** Specifies the Resource level authentication along the HTTP verbs. Auth-type can be None, Application, or Application User.
  - None: Can access the particular API resource without any access tokens
  - Application: An application access token is required to access the API resource
  - Application User: A user access token is required to access the API resource



# Deep diving into the API Manager

Let's take a look at the typical API management activities in detail:

- Creating users and roles
- Creating an API from scratch
- Adding API documentation
- Adding interactive documentation
- Versioning the API
- Publishing the API
- Subscribing to the API
- Invoking the API
- Monitoring APIs and viewing statistics

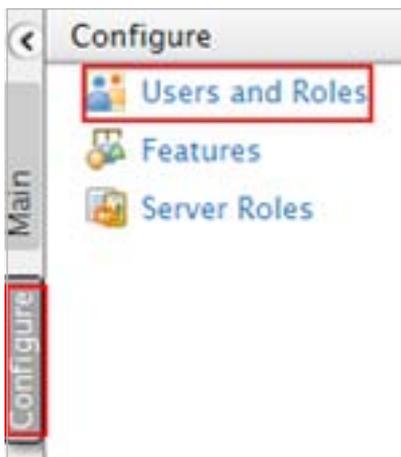
## Creating users and roles

In [Users and roles](#), we introduced a set of users who are commonly found in many enterprises. Let's see how you can log in to the Management Console as an admin and create these roles.

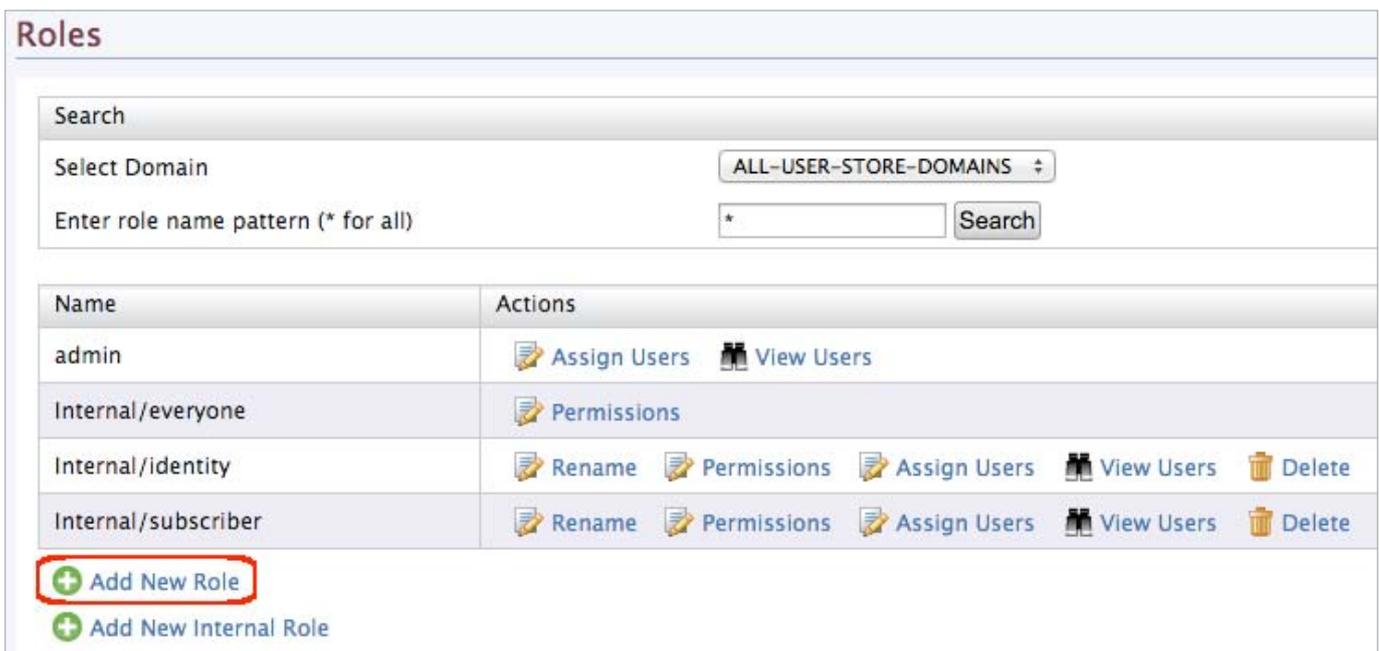
1

Log in to the Management Console ( <https://<hostname>:9443/carbon> ) of the API Manager using admin/admin credentials.

- 2 Select the **Users and Roles** menu under the **Configure** menu.



- 3 Click the **Roles** link and then click **Add New Role**.

A screenshot of the 'Roles' management page. The page has a search bar at the top with a 'Select Domain' dropdown set to 'ALL-USER-STORE-DOMAINS' and an 'Enter role name pattern (\* for all)' input field containing an asterisk. Below the search bar is a table with two columns: 'Name' and 'Actions'. The table lists several roles: 'admin', 'Internal/everyone', 'Internal/identity', and 'Internal/subscriber'. Each role has specific actions available, such as 'Assign Users', 'View Users', 'Permissions', 'Rename', and 'Delete'. At the bottom of the page, there are two buttons: 'Add New Role' (highlighted with a red box) and 'Add New Internal Role'.

- 4 Give the role name as creator and click **Next**.

**Add Role**

**Step 1 : Enter role details**

Enter role details

Domain

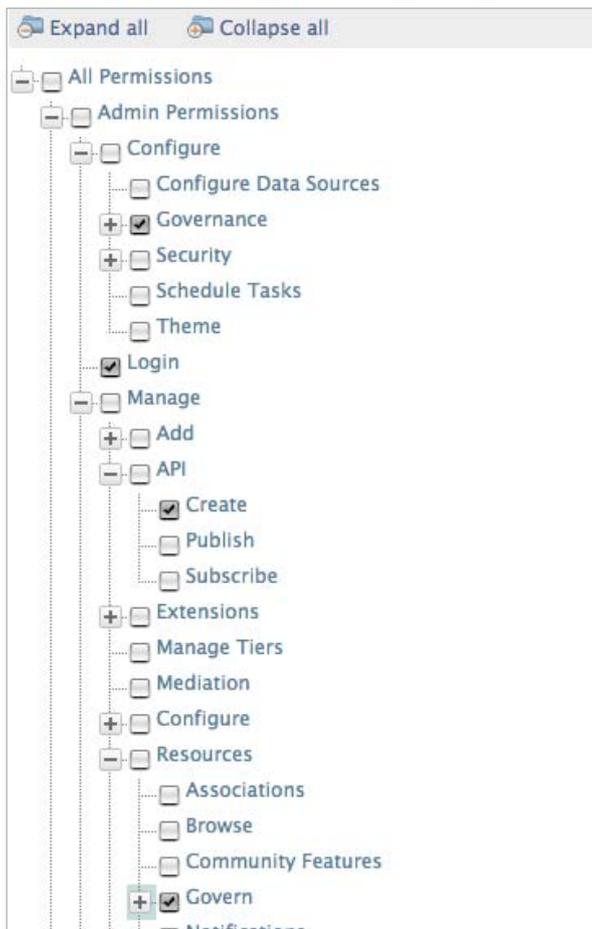
Role Name\*

**Next >**

- 5 A list of permissions opens. Select the following and click **Finish**.

- Configure > Governance and all underlying permissions.
- Login
- Manage > API > Create
- Manage > Resources > Govern and all underlying permissions

### Step 2 : Select permissions to add to Role



6 Similarly, create the publisher role with the following permissions.

- Login
- Manage > API > Publish

7 Note that the API Manager comes with the subscriber role available by default. It has the following permissions:

- Login
- Manage > API > Subscribe

8 The roles you added (creator, internal/subscriber, and publisher) are now displayed under Roles.

**Roles**

Search

Select Domain ALL-USER-STORE-DOMAINS ▾

Enter role name pattern (\* for all)  Search

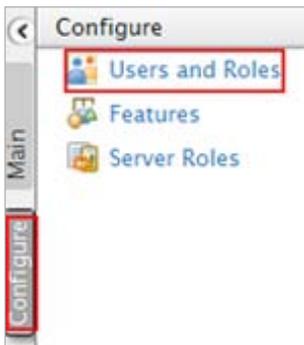
Name	Actions
admin	Assign Users  View Users
<b>creator</b>	Rename  Permissions  Assign Users  View Users  Delete
Internal/everyone	Permissions
Internal/identity	Rename  Permissions  Assign Users  View Users  Delete
<b>Internal/subscriber</b>	Rename  Permissions  Assign Users  View Users  Delete
<b>publisher</b>	Rename  Permissions  Assign Users  View Users  Delete

Add New Role

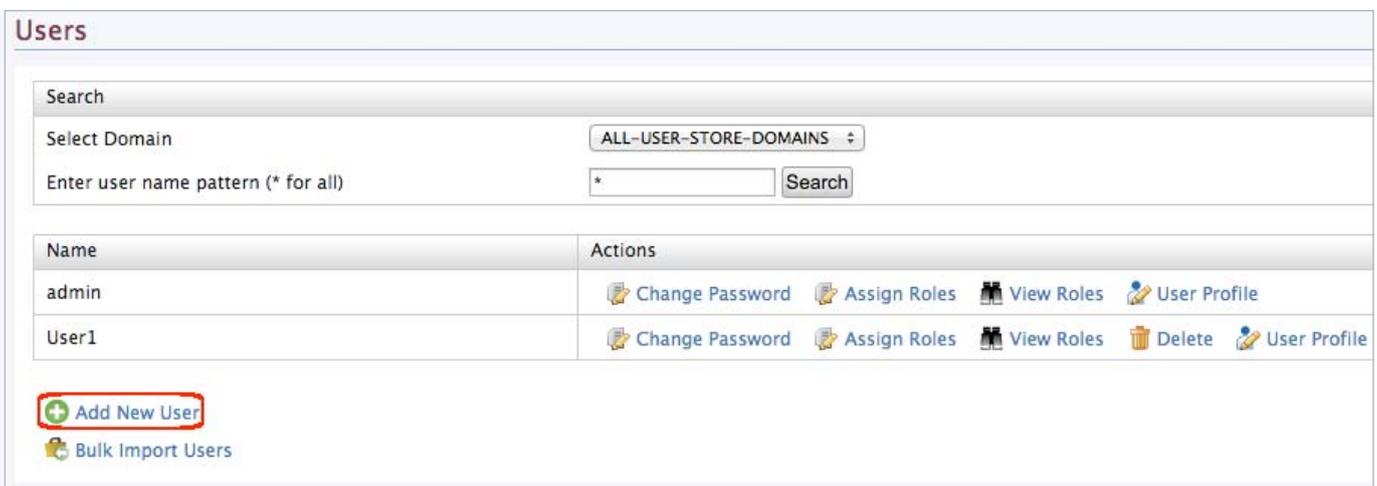
Add New Internal Role

Let's create users for each of the roles.

- 9 Click the **Users and Roles** menu under the **Configure** menu again.



- 10 Click the **Users** link and then click **Add New User**.



- 11 Give the username/password and click **Next**. For example, let's create a new user by the name apipublisher.

 A screenshot of the 'Add User' form, specifically 'Step 1 : Enter user name'. The form has a title 'Add User' and a subtitle 'Step 1 : Enter user name'. Below the subtitle is a section titled 'Enter user name' containing four input fields: 'Domain' (a dropdown menu set to 'PRIMARY'), 'User Name\*' (a text input containing 'apipublisher'), 'Password\*' (a password input with dots), and 'Password Repeat\*' (a password input with dots). At the bottom of the form, there are three buttons: 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a red box.

- 12 Select the role you want to assign to the user (e.g., publisher) and click **Finish**.

**Add User**

**Step 2 : Select roles of the user**

Enter role name pattern (\* for all)

**Users of Role**

Select all on this page | Unselect all on this page

- admin
- creator
- publisher
- Internal/everyone
- Internal/subscriber
- Internal/identity

- 13 Similarly, create a new user by the name apicreator and assign the creator role.

## Creating an API from scratch

Let's create an API from scratch.

- 1 Log in to the API Publisher (<https://<hostname>:9443/publisher>) as apicreator.

- 2 Select the option to design a new API and click **Start Creating**.

**WSO2 API PUBLISHER**

**APIs**

Browse

**Add**

All Statistics ▾

**My APIs**

Subscriptions

Statistics ▾

**Tier Permissions**

Tier Permissions

APIs / Add New API

**Lets get started!**

- I have an Existing API**  
 Use an existing API endpoint to create a managed API. If you have API definition you can import it to speed up API creation.
- I have a SOAP Endpoint**  
 Use an existing SOAP endpoint to create a managed API. Import WSDL of the SOAP service.
- Design new API**  
 Design and prototype a new API with API Manager.

3 Give the information in the table below and click **Implement** to move on to the next page.

Field	Sample Value
Name	PhoneVerification
Context	/phoneverify
Version	1.0.0
Visibility	Public
API Definition	<ul style="list-style-type: none"> <li>• URL pattern: CheckPhoneNumber</li> <li>• Request types: GET, POST</li> </ul>

1 Design
2 Implement
3 Manage

### Design API

---

#### General Details

Name:  ⓘ

Context:  ⓘ

Version:   
E.g.: v1.0.0 ,v1.0 , 1.0.0, 1.0

Visibility:  ⓘ

Thumbnail Image:  No file chosen  • Dimensions (max): 100 x 100 pixels

Description:

Tags:  ⓘ  
Type a tag and Enter

---

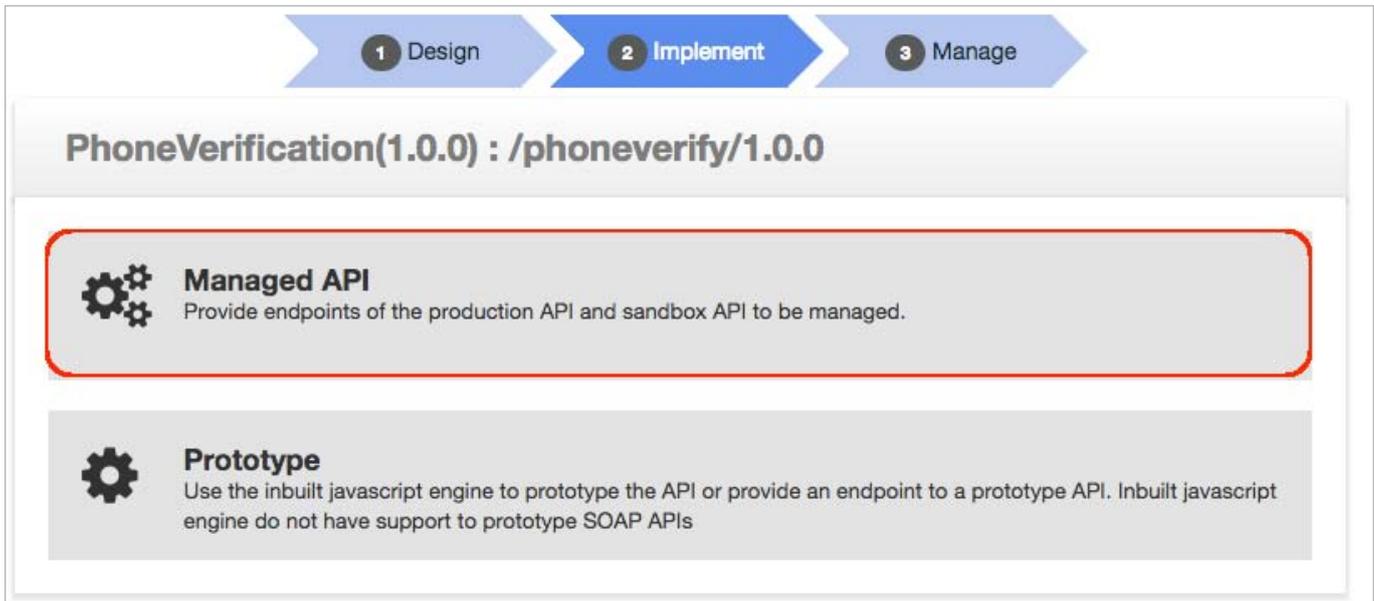
#### API Definition

URL Pattern

GET  
  POST  
  PUT  
  DELETE  
 more

---

4 Select the **Managed API** option.



1 Design 2 **Implement** 3 Manage

**PhoneVerification(1.0.0) : /phoneverify/1.0.0**

 **Managed API**  
Provide endpoints of the production API and sandbox API to be managed.

 **Prototype**  
Use the inbuilt javascript engine to prototype the API or provide an endpoint to a prototype API. Inbuilt javascript engine do not have support to prototype SOAP APIs

5 Give the following information in the **Implement** tab that opens and click **Manage** once you are done.

Field	Sample Value
Endpoint type	HTTP
Production endpoint	<p>In this guide, we work with a service exposed by the Cdyne services provider. We use their phone validation service, which has SOAP and REST interfaces. Endpoint is <a href="http://ws.cdyne.com/phoneverify/phoneverify.asmx">http://ws.cdyne.com/phoneverify/phoneverify.asmx</a>.</p> <p>This sample service has two operations: CheckPhoneNumber and CheckPhoneNumbers. Let's use CheckPhoneNumber here.</p>

**1 Design**   **2 Implement**   **3 Manage**

**PhoneVerification : /phoneverify/1.0.0**

Implementation Method    Backend Endpoint    Specify Inline

**Endpoints**

Endpoint Type:\*   HTTP Endpoint ⓘ

Production Endpoint:   m/phoneverify/phoneverify.asmx   Advanced Options   Test  
E.g.: http://appserver/resource

Sandbox Endpoint:   m/phoneverify/phoneverify.asmx   Advanced Options   Test  
E.g.: http://appserver/resource

[Show More Options](#)

**Save**   **Deploy Prototype**   **Manage**   **Cancel**

**6** Click **Manage** to go to the Manage tab and provide the following information. Leave default values for the rest of the parameters in the UI.

Field	Value	Description
Tier Availability	<Select all available tiers>	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.

1 Design 2 Implement 3 Manage

## Manage API: PhoneVerification : /phoneverify/1.0.0/1.0.0

### Configurations

Make this default version  ?  
No default version defined for the current API

Tier Availability:\* 4 selected ?

Transports:\*  HTTPS  HTTP ?

Sequences:  Check to select a custom sequence to be executed in the message flow

Response Caching: Disabled ?

[Gateway Environments >](#)

[Business Information >](#)

[Resources](#)

7 Once you are done, click **Save**.

## Adding API documentation

1 After saving the API, click its thumbnail in the API Publisher to open it.

2 Click on the API's **Docs** tab and click the **Add New Document** link.

### PhoneVerification - 1.0.0 [Edit](#)

[Overview](#) [Versions](#) Docs [Users](#)

+ Add New Document

Name	Type	Modified On	Actions
No documentation associated with the API			

3 The document options appear. Note that you can create documentation inline, via a URL, or as a file. For inline documentation, you can edit the content directly from the API publisher interface. You get several documents types:

- How To
- Samples and SDK
- Public forum / Support forum (external link only)
- API message formats
- Other

4 Create a 'How To' named PhoneVerification, specifying in-line content as the source and optionally entering a summary. When you have finished, click **Add Document.**"

**PhoneVerification - 1.0.0** [Edit](#)

[Overview](#) [Versions](#) **[Docs](#)** [Users](#)

**+ Add New Document**

Name\*

Summary

Type

- How To
- Samples & SDK
- Public Forum
- Support Forum
- Other (specify)

Source

- In-line <sup>?</sup>
- URL <sup>?</sup>
- File <sup>?</sup>

**Add Document** Cancel

5 Once the document is added, click **Edit Content** to open an embedded editor.

**PhoneVerification - 1.0.0** [Edit](#)

[Overview](#) [Versions](#) **[Docs](#)** [Users](#)

**+ Add New Document**

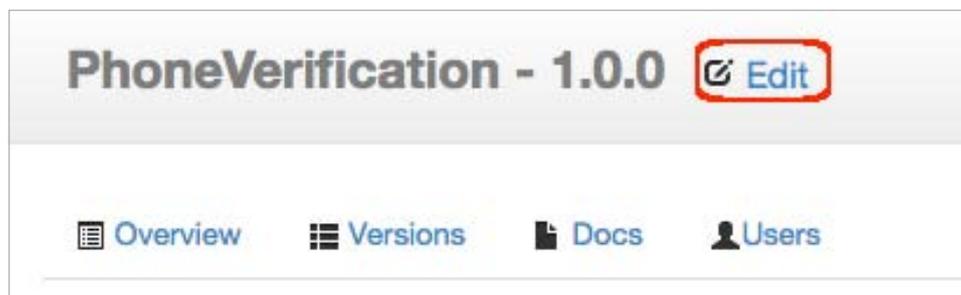
Name	Type	Modified On	Actions
PhoneVerification	How To	5/19/2015, 2:27:47 PM	<b><a href="#">Edit Content</a></b> <a href="#">Update</a> <a href="#">Delete</a>



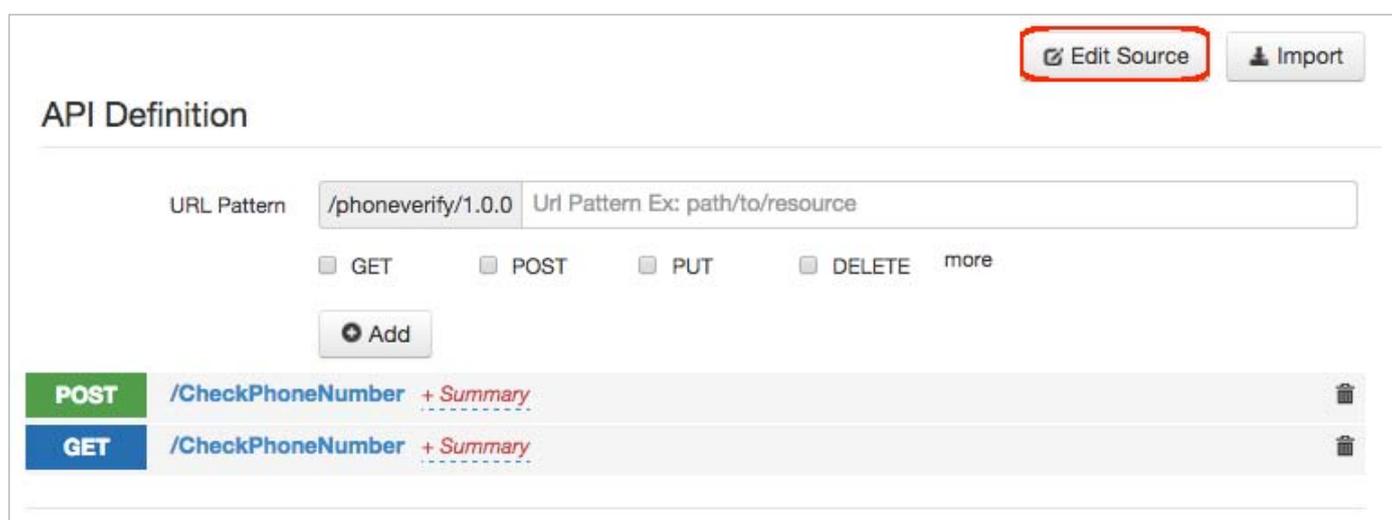
Also, see the [Swagger 2.0 specification](#).

1 Open the API Publisher (<https://<hostname>:9443/publisher>) and log in as apicreator.

2 Click the PhoneVerification API to open it and then click the **Edit** right next to the API's name. This opens the API in its edit mode.



3 Click the **Edit Source** button near the **Resources** section.



4 The JSON code of the API opens in a separate page. Expand its GET method, add the following parameters to it, and click **Save**.

parameters:

- name: PhoneNumber  
paramType: query  
required: true  
type: string  
description: Give the phone number to be validated  
in: query
- name: LicenseKey  
paramType: query  
required: true  
type: string  
description: Give the license key as 0 for testing purpose  
in: query

```

1 - paths:
2 -   /CheckPhoneNumber:
3 -     post:
4 -       x-auth-type: "Application & Application User"
5 -       x-throttling-tier: Unlimited
6 -       responses:
7 -         "200": {}
8 -     get:
9 -       x-auth-type: "Application & Application User"
10 -      x-throttling-tier: Unlimited
11 -      responses:
12 -        "200": {}
13 -      parameters:
14 -        - name: PhoneNumber
15 -          paramType: query
16 -          required: true
17 -          type: string
18 -          description: Give the phone number to be validated
19 -          in: query
20 -        - name: LicenseKey
21 -          paramType: query
22 -          required: true
23 -          type: string
24 -          description: Give the license key as 0 for testing purpose
25 -          in: query
26 - swagger: "2.0"
27 - info:|
28 -   title: PhoneVerification
29 -   version: 1.0.0

```

5 Back in the API Publisher, note that the changes you did appear in the API Console’s UI. You can add more parameters and edit the summary/descriptions using the API Publisher UI as well. Once done, click **Save**.

API Definition

URL Pattern  Url Pattern Ex: path/to/resource

GET  POST  PUT  DELETE [more](#)

POST	/CheckPhoneNumber	<a href="#">+ Summary</a>	<input type="button" value="trash"/>
GET	/CheckPhoneNumber	<a href="#">+ Summary</a>	<input type="button" value="trash"/>

Description :

[+ Add Implementation Notes](#)

Response Content Type : [application/json](#)

Parameters :

Parameter Name	Description	Parameter Type	Data Type	Required
PhoneNumber	Give the phone number to be validated	query	string	True
LicenseKey	Give the license key as 0 for testing purpose	query	string	True

## Versioning the API

Let's create a new version of this API.

- 1 Log in to the API Publisher as apicreator if you are not logged in already.
- 2 Click the PhoneVerification API, and then click **Create New Version** on its Overview tab.

WSO2 API PUBLISHER

APIs / All / PhoneVerification-1.0.0

### PhoneVerification - 1.0.0 [Edit](#)

**Overview** Versions Docs Users

0 Users  
CREATED 1.0.0 Docs

Visibility	Public
Context	/phoneverify/1.0.0
Production URL	http://ws.cdyne.com/phoneverify/phone...
Date Last Updated	5/19/2015, 3:19:29 PM
Tier Availability	Bronze, Unlimited, Gold, Silver
Default API Version	None
Published Environments	Production and Sandbox

**Create New Version**

- 3 Give a new version number (e.g., 2.0.0) and click **Done**.

To Version

E.g., v1.0.1

Default Version  No default version defined for the current API

**Done** Cancel

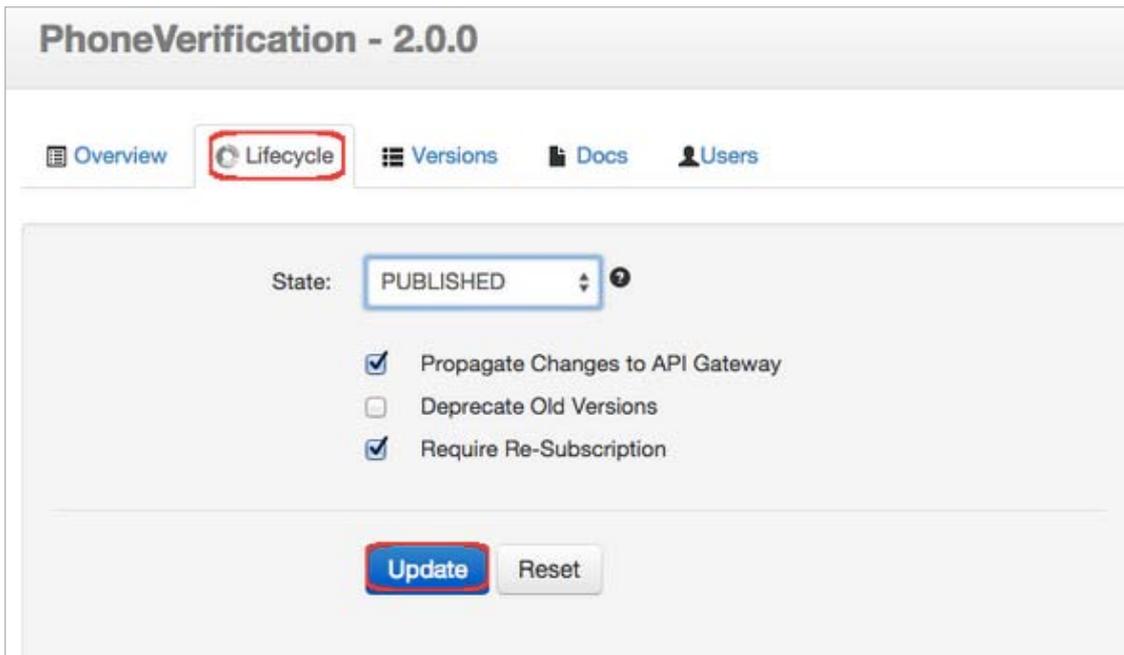
- 4 Note that the new version of the API is created in the API Publisher.

## Publishing the API

- 1 Log in to the API Publisher as the apipublisher user that you created earlier in this guide and click the PhoneVerification API version 2.0.0.

The screenshot displays the WSO2 API Publisher interface. The left sidebar has a dark grey background with white text for navigation: 'APIs', 'Browse' (highlighted in light blue), 'Add', 'All Statistics' (with a dropdown arrow), 'My APIs', 'Subscriptions', and 'Statistics' (with a dropdown arrow). The main content area has a white background and is titled 'APIs / All'. Below the title is a search bar with the text 'Filter APIs' and a blue 'Search' button. Two API cards are shown in a grid. Each card has a blue square icon with three white gears and a small 'x' in the top right corner. The first card is for 'PhoneVerification 1.0.0' and the second is for 'PhoneVerification 2.0.0'. The '2.0.0' version is circled in red. Both cards indicate '0 Users CREATED'.

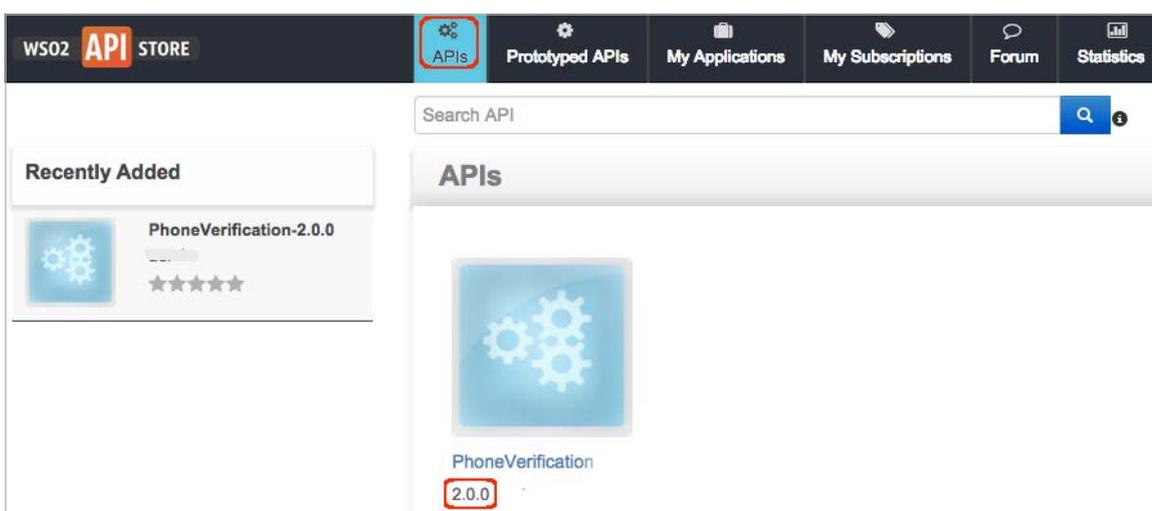
- 2 The API opens. Go to its **Lifecycle** tab, select the state as PUBLISHED from the drop-down list, and click **Update**.



The three check boxes mean the following:

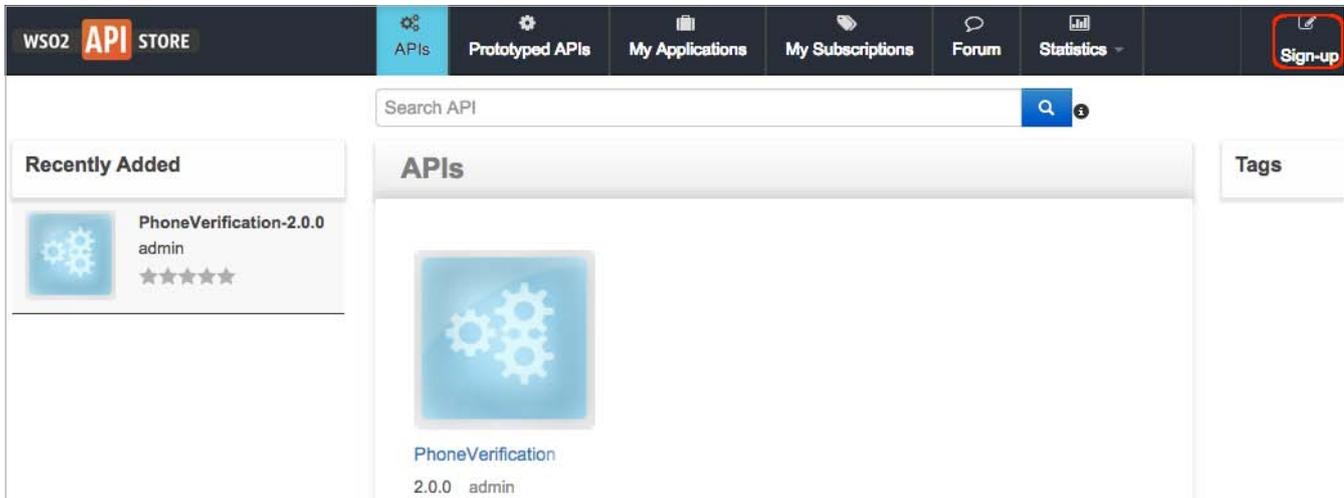
- **Propagate Changes to API Gateway:** Used to define an API proxy in the API Gateway runtime component, allowing the API to be exposed to the consumers via the API Gateway. If this option is left unselected, the API metadata will not change, and you will have to manually configure the API Gateway according to the information published in the API Store.
- **Deprecate Old Versions:** If selected, any prior versions of the API that are published will be set to the DEPRECATED state automatically.
- **Require Re-Subscription:** Invalidates current user subscriptions, forcing users to subscribe again.

- 3 Go to the API Store (<https://<hostname>:9443/store>) using your browser and note that the PhoneVerification 2.0.0 API is visible under the **APIs** menu.



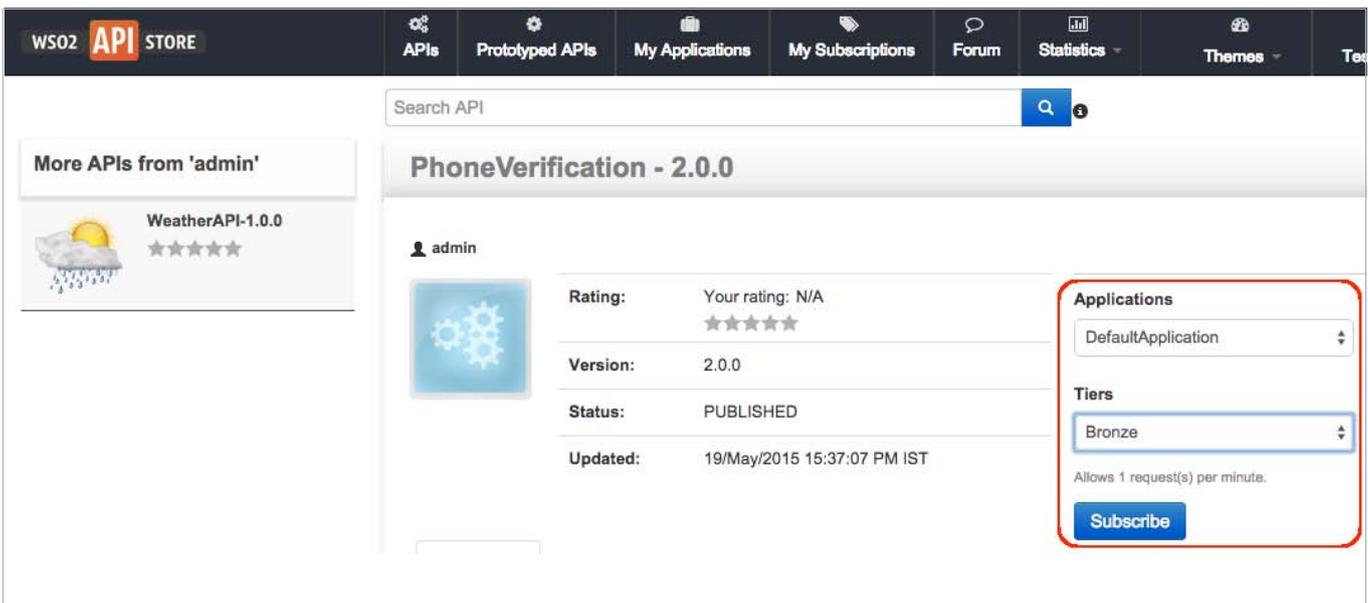
## Subscribing to the API

- 1 Go to the API Store (<https://<hostname>:9443/store>) and create an account using the **Sign-up** link.



- 2 After signing up, log in to the API Store and click the PhoneVerification 2.0.0 API that you published earlier.

- 3 Note that you can now see the subscription options on the right-hand side of the UI. Select the default application, select Bronze tier, and click **Subscribe**.



- 4 Once the subscription is successful, go to the **My Subscriptions** page.
- 5 In the **My Subscriptions** page, click the **Generate** buttons to generate access tokens that you need to invoke the API.

APIs Prototyped APIs My Applications **My Subscriptions** Forum Statistics

Search API

## Subscriptions

Create access tokens to applications. Because an application is a logical collection of APIs, you can use a single access token to invoke multiple APIs and to subscribe to one API multiple times with different SLA levels.

### Applications With Subscriptions

DefaultApplication  Show Keys

#### Keys - Production

Production keys are not yet generated for this application.

**Generate**

#### Allowed Domains

ALL

The domains from which requests are allowed to the APIs. Leave empty or enter "ALL" to allow all domains.

Token Validity: 3600 Seconds

#### Keys - Sandbox

**Tip:** You can set a token validity period in the given text box. By default, it is set to one hour. If you set a minus value (e.g., -1), the token will never expire.

You are now successfully subscribed to an API. Let's invoke it.

## Invoking the API

- 1 Click the **APIs** menu in the API Store and then click on the API that you want to invoke. When the API opens, go to its **API Console** tab.

The screenshot shows the API Store interface. At the top, there is a navigation bar with icons for APIs, Prototyped APIs, My Applications, My Subscriptions, Forum, Statistics, Tools, Themes, and admin. Below the navigation bar is a search bar labeled 'Search API'. The main content area displays the details for the 'PhoneVerification - 1.0.0' API. The details include a rating of 'N/A', version '1.0.0', status 'PUBLISHED', and an update date of '08/Dec/2014 16:50:11 PM IST'. There are also dropdown menus for 'Applications' (Set to 'Select Application...') and 'Tiers' (Set to 'Unlimited'). A 'Subscribe' button is visible. Below the details, there are tabs for 'Overview', 'Documentation', 'API Console' (which is highlighted with a red box), and 'Throttling Info'. The 'API Console' tab shows a 'Try' section with a dropdown for 'DefaultApplication' and a 'Production' environment. Below this is a 'Set Request Header' section with 'Authorization : Bearer' and a token 'b690ca26e6375b44b5bde78a44fbcf2'. The main part of the console shows the 'PhoneVerification' resource with a 'Swagger Resource Listing (/api-docs)' link. Underneath, there is a table of operations for 'checkphonenumber' with columns for 'Show/Hide', 'List Operations', 'Expand Operations', and 'Raw'. Two operations are listed: a GET method for '/CheckPhoneNumber' and a POST method for '/CheckPhoneNumber'. The GET method is highlighted with a blue box.

- 2 Expand the GET method of the resource CheckPhoneNumber. Note the parameters that you added when creating the interactive documentation now appear with their descriptions, so that as a subscriber, you know how to invoke this API.

The screenshot shows the 'GET /CheckPhoneNumber' method expanded in the API Console. Below the method name, there is a 'Parameters' section with a table of parameters. The table has columns for 'Parameter', 'Value', 'Description', 'Parameter Type', and 'Data Type'. Two parameters are listed: 'PhoneNumber' and 'LicenseKey'. Both parameters are marked as '(required)' and have input fields. The 'PhoneNumber' parameter has a description 'Give the phone number to be validated' and is of type 'query' with data type 'string'. The 'LicenseKey' parameter has a description 'Give the license key as 0 for testing purpose' and is of type 'query' with data type 'string'. A red box highlights the parameter table.

Parameter	Value	Description	Parameter Type	Data Type
PhoneNumber	(required)	Give the phone number to be validated	query	string
LicenseKey	(required)	Give the license key as 0 for testing purpose	query	string

- 3 Give sample values for the PhoneNumber and LicenseKey and click **Try it out** to invoke the API.

**GET /CheckPhoneNumber**

**Parameters**

Parameter	Value	Description	Parameter Type	Data Type
PhoneNumber	18006785432	Give the phone number to be validated	query	string
LicenseKey	0	Give the license key as 0 for testing purpose	query	string

**Response Messages**

HTTP Status Code	Reason	Response Model	Headers
200			

Try it out!

**POST /CheckPhoneNumber**

- 4 Note the response for the API invocation. Because we used a valid phone number in this example, the response is valid.

**Response Body**

```
<?xml version="1.0" encoding="utf-8"?>
<PhoneReturn xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://ws.cdyne.com/PhoneVerify/query"> <Company>Toll Free</Company>
<Valid>true</Valid>
  <Use>Assigned to a code holder for normal use.</Use>
  <State>TF</State>
  <RC />
  <OCN />
  <OriginalNumber>18006785432</OriginalNumber>
  <CleanNumber>8006785432</CleanNumber>
  <SwitchName />
  <SwitchType />
  <Country>United States</Country>
  <CLLI />
  <PrefixType>Landline</PrefixType>
  <LATA />
  <sms>Landline</sms>
  <Email />
  <AssignDate />
  <TelecomCity />
  <TelecomCounty />
</PhoneReturn>
```

**Response Code**

200

**Response Headers**

```
Pragma: no-cache
Content-Type: text/xml; charset=utf-8
Cache-Control: no-cache
Expires: -1
```

You have invoked an API using the API Console.

## Monitoring APIs and viewing statistics

Both the API publisher and store provide several statistical dashboards. Some of them are as follows:

- Number of subscriptions per API (across all versions of an API)
- Number of API calls being made per API (across all versions of an API)
- The subscribers who did the last 10 API invocations and the APIs/versions they invoked
- Usage of an API and from which resource path (per API version)
- Number of times a user has accessed an API
- The number of API invocations that failed to reach the endpoint per API per user
- API usage per application
- Users who make the most API invocations, per application
- API usage from resource path, per application

The statistics in these dashboards are based on data from WSO2 Business Activity Monitor (BAM). The steps below explain how to configure WSO2 BAM 2.5.0 with the API Manager.

✔ If you are on **Windows**, note the following:

- If you installed the JDK in Program Files in the Windows environment, avoid the space by using PROGRA~1 when specifying environment variables for JAVA\_HOME and PATH. Otherwise, the server throws an exception.
- Install Cygwin (<http://www.cygwin.com>). WSO2 BAM depends on Apache Hadoop, which requires Cygwin in order to run on Windows. Install at least the basic net (OpenSSH, tcp\_wrapper packages) and security-related Cygwin packages. After installing Cygwin, update the PATH variable with C:/cygwin/bin. If you already have WSO2 BAM running, you must restart it now.

Steps below explain how to configure [WSO2 BAM 2.5.0](#) with the API Manager. Let's do the configurations first.

- 1 Apply an offset of 3 to the default BAM port by editing the <BAM\_HOME>/repository/conf/carbon.xml file. This makes the BAM server run on port 9446 instead of the default port 9443 and avoids port conflicts when multiple WSO2 products run on the same host.

```
<Offset>3</Offset>
```

- 2 Download MySQL from <https://www.mysql.com/> and install it in your server.

- 3 Create a MySQL database (e.g., TestStatsDB) to save the statistical data collected by the BAM. You do not need to create any tables in it.

```
mysql -u <username> -p <password> -h <host_name or IP>;
CREATE DATABASE TestStatsDB;
```

- 4 Save the [MySQL connector JAR](#) inside both <APIM\_HOME>/repository/components/lib and <BAM\_HOME>/repository/components/lib folders.

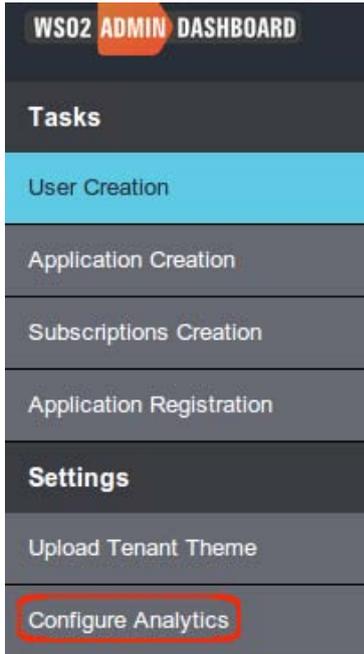
- 5 Give the datasource definition under the <datasource> element in the <BAM\_HOME>/repository/conf/datasources/master-datasources.xml file. For example,

```
<datasource>
  <name>WSO2AM_STATS_DB</name>
  <description>The datasource used for getting statistics to API Manager</description>
  <jndiConfig>
    <name>jdbc/WSO2AM_STATS_DB</name>
  </jndiConfig>
  <definition type="RDBMS">
    <configuration>
      <url>jdbc:mysql://localhost:3306/TestStatsDB</url>
      <username>db_username</username>
      <password>db_password</password>
      <driverClassName>com.mysql.jdbc.Driver</driverClassName>
      <maxActive>50</maxActive>
      <maxWait>60000</maxWait>
      <testOnBorrow>>true</testOnBorrow>
      <validationQuery>SELECT 1</validationQuery>
      <validationInterval>30000</validationInterval>
    </configuration>
  </definition>
</datasource>
```

- 6 Start the BAM server.

- 7 Start the API Manager and log in to its Admin Dashboard Web application (<https://<Server Host>:9443/admin-dashboard>) with **admin/admin** credentials.

- 8 Click the **Configure Analytics** menu.



- 9 Select the **Enable** check box to enable statistical data publishing and add the following:

- Add a URL group as tcp://<BAM server IP>:7614 and click **Add URL Group**.
- Fill the details under **Statistics Summary Database** according to the information you added to the master-datasources.xml file in step 5.

### Configure Analytics

Enable API usage publishing and Statistics aggregation

Enable

#### Event Receiver Configurations

URL Group:\* Username:\* Password:\*

admin  [Add URL Group](#)

Event Receiver Group	Username	Actions
tcp://localhost:7614	admin	<a href="#">Delete</a>

#### Data Analyzer Configurations

URL:\* Username:\* Password:\*

https://localhost:9446 admin

#### Statistics Summary Datasource

URL:\* JDBC Driver Class:\* Username:\* Password:\*

jdbc:mysql://localhost:3306/TestStatsDB com.mysql.jdbc.Driver root

Show More Options

[Save](#)

- 10 Click **Save**. BAM deploys the Analytics toolbox, which describes the information collected, how to analyze the data, and the location of the database where the analyzed data is stored.
- 11 Invoke several APIs to generate some statistical data and wait a few seconds.
- 12 Connect to the API Publisher as a creator or publisher and click the statistical dashboards available under the **All Statistics** and **Statistics** menus. For example,

The screenshot shows the WSO2 API Publisher interface. On the left, a navigation menu is visible with the following sections: 'APIs', 'Browse', 'Add', 'All Statistics' (highlighted with a red box), 'API Subscriptions', 'API Usage', 'API Response Times', 'API Last Access Times' (highlighted with a red box), 'API Usage by Resource Path', 'API Usage by Destination', 'API Usage Comparison', 'Faulty Invocations', 'My APIs', 'Subscriptions', 'Statistics' (highlighted with a red box), and 'Tier Permissions'. The main content area displays the 'API Last Access Times (Across All Versions)' dashboard. At the top of the dashboard, there are filters for 'Hour', 'Day', 'Week', and 'Month', and a date range '2015-04-18 14:13 to 2015-05-18 14:13'. Below the filters, there is a 'Show 10 entries' dropdown and a search box. The dashboard contains a table with the following data:

API	VERSION	SUBSCRIBER	ACCESS TIME (GMT+05:30)
[Redacted]	1.0.0	[Redacted]	5/18/2015, 2:12:00 PM
[Redacted]	1.0.0	[Redacted]	5/8/2015, 3:37:00 PM

Below the table, it says 'Showing 1 to 2 of 2 entries'.

The **All Statistics** menu is available for both API creators and publishers. It shows statistics of all APIs. The **Statistics** menu is available for API creators to see statistics of only the APIs created by them.

This concludes the API Manager quick start. You have set up the API Manager and gone through the basic use cases of the product. For more advanced use cases, please see the [User Guide](#) and the [Admin Guide](#) of the API Manager documentation.