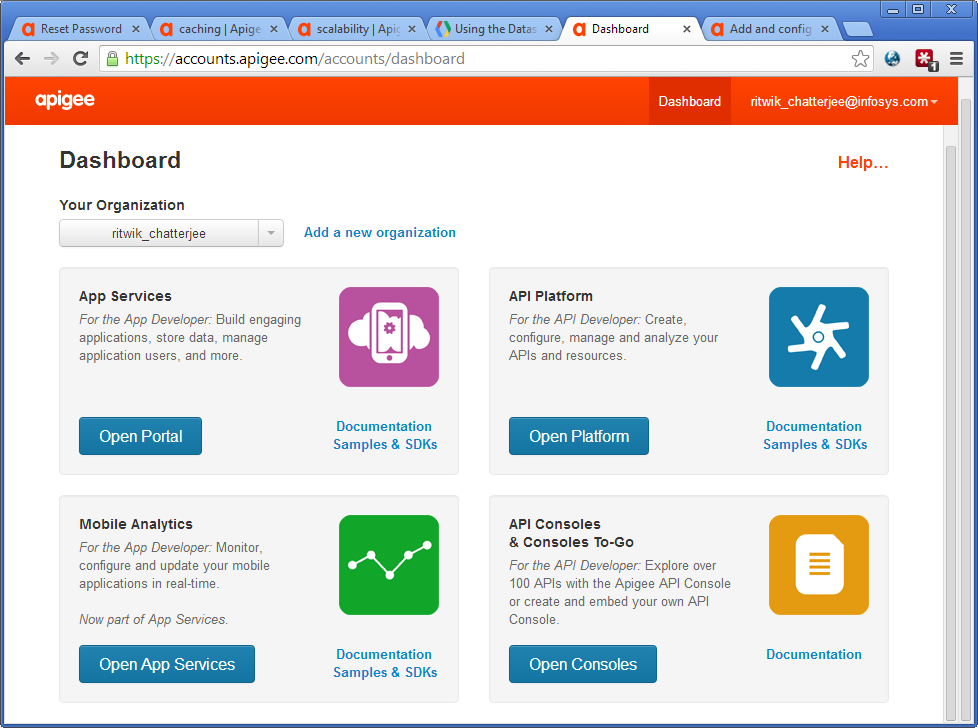
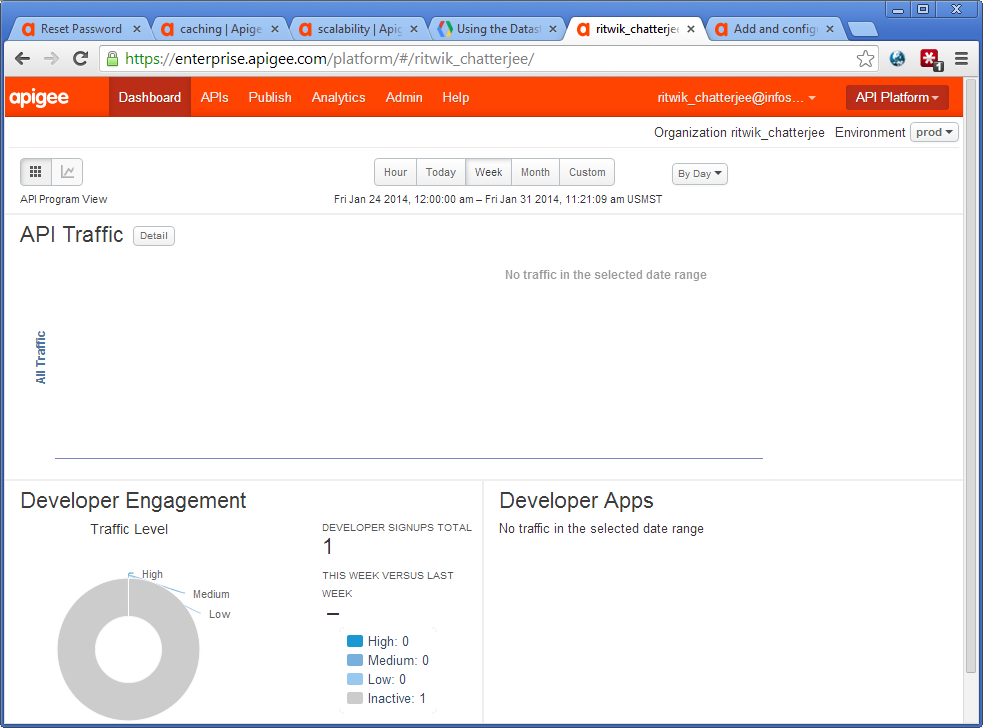
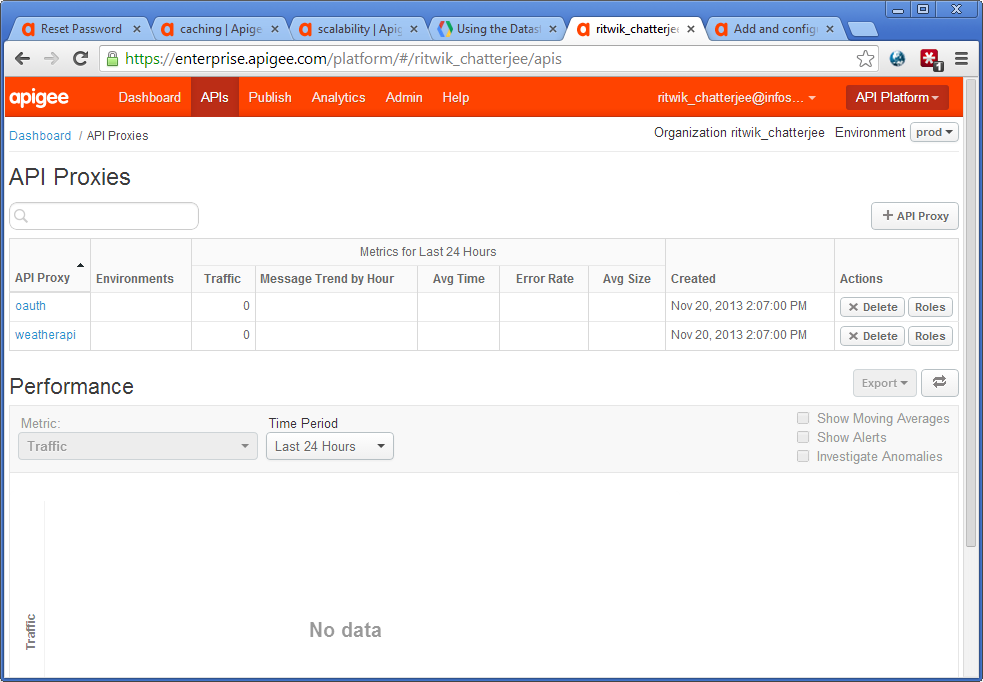
Apigee Training Guide

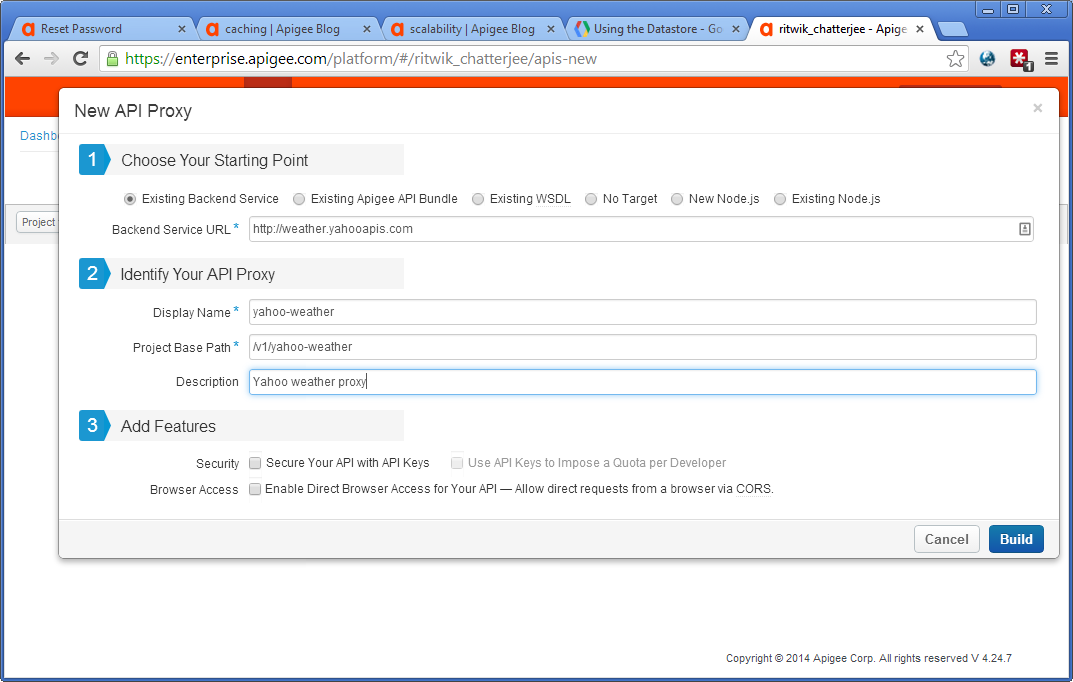
1. User Dashboard

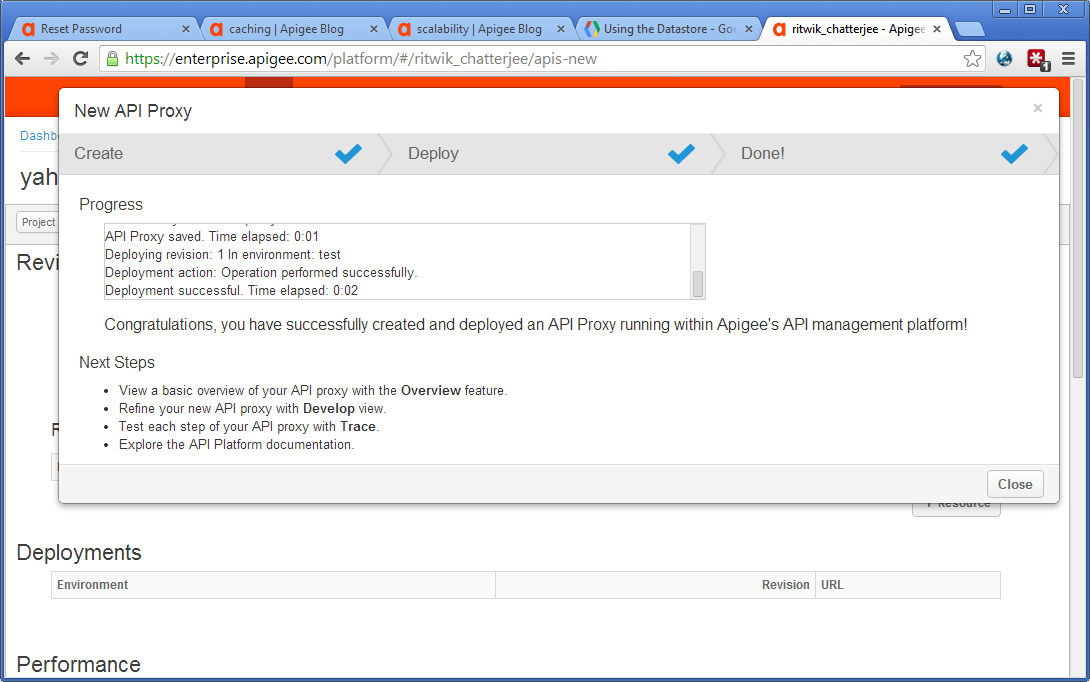


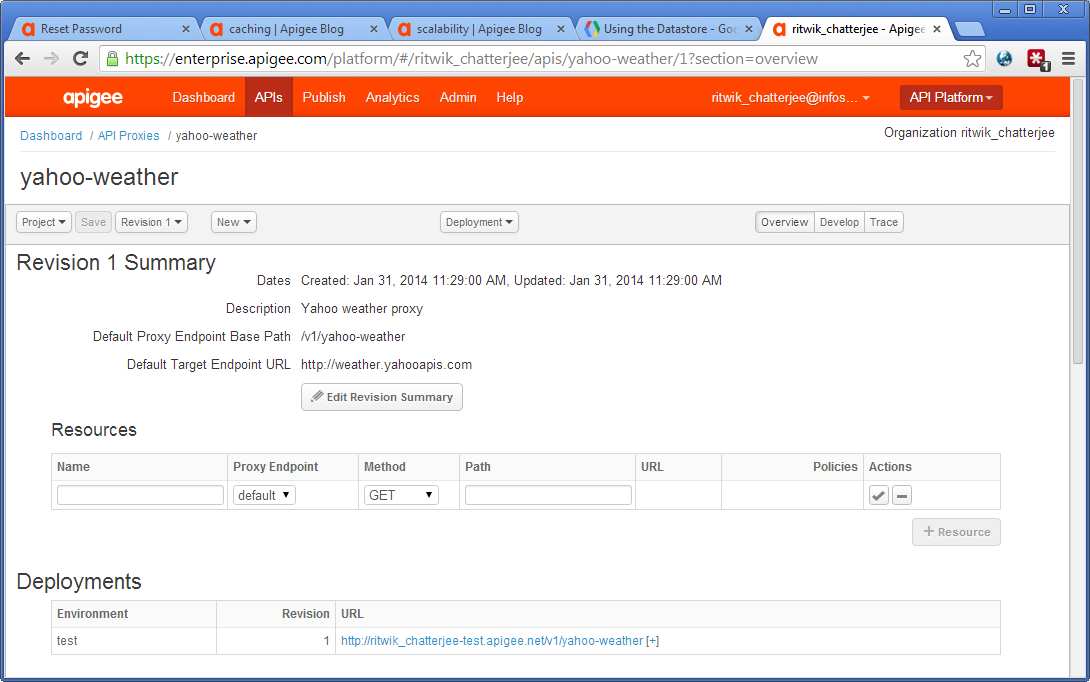
1. API Platform landing page

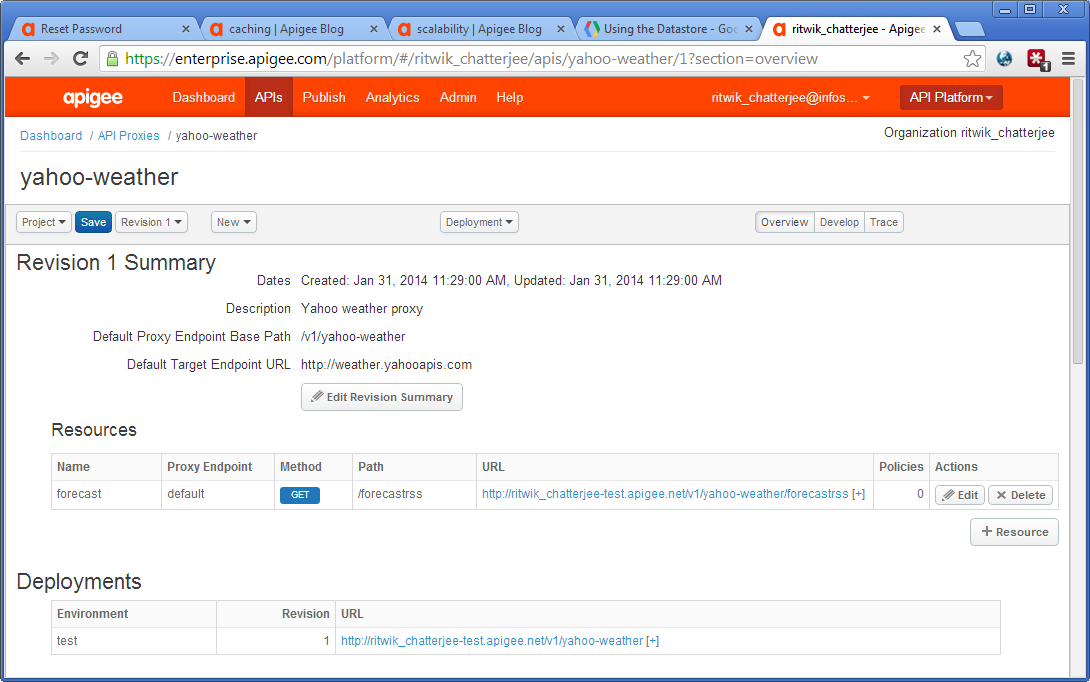


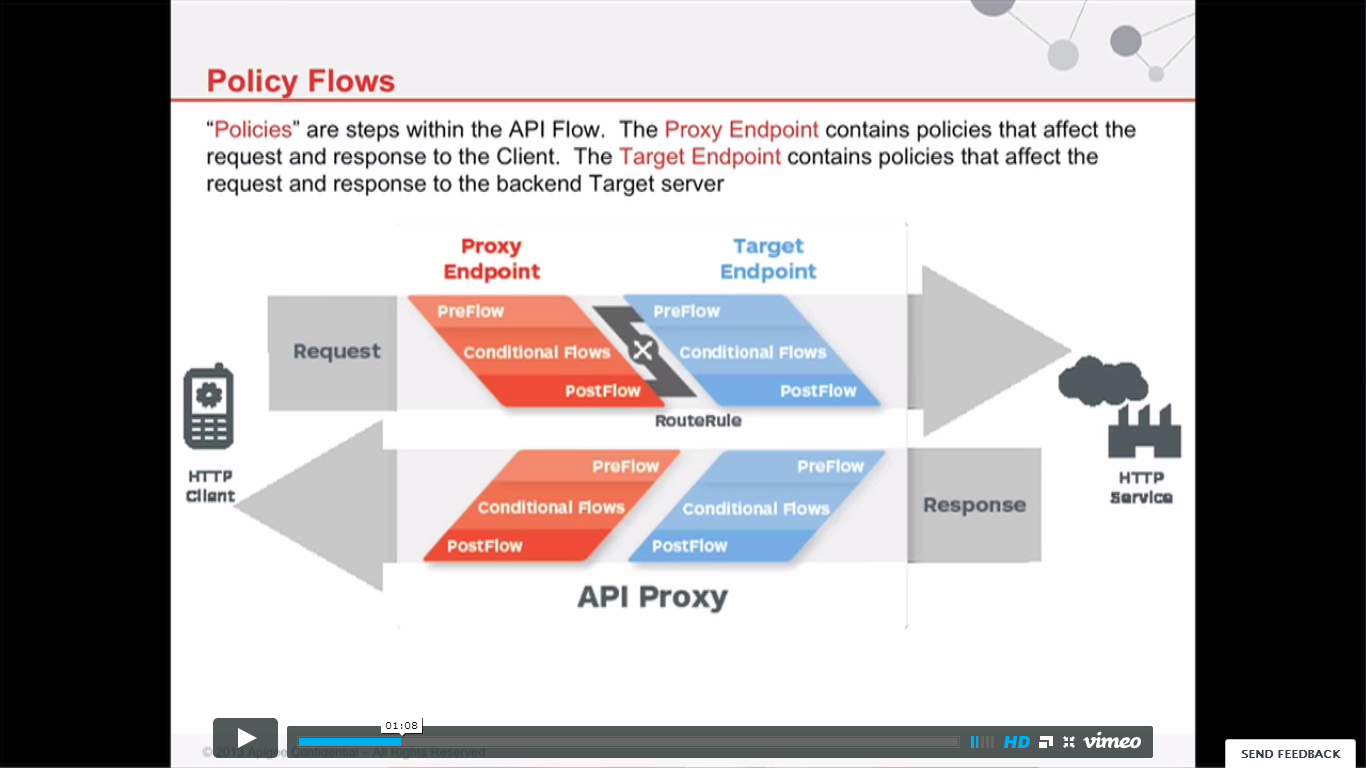


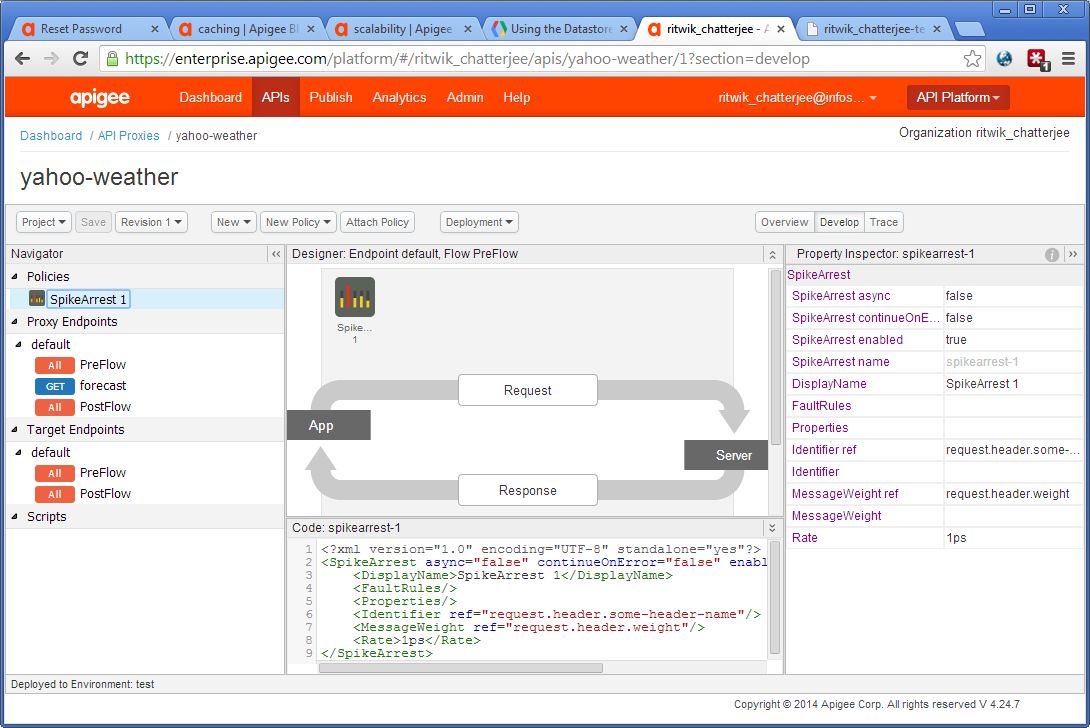


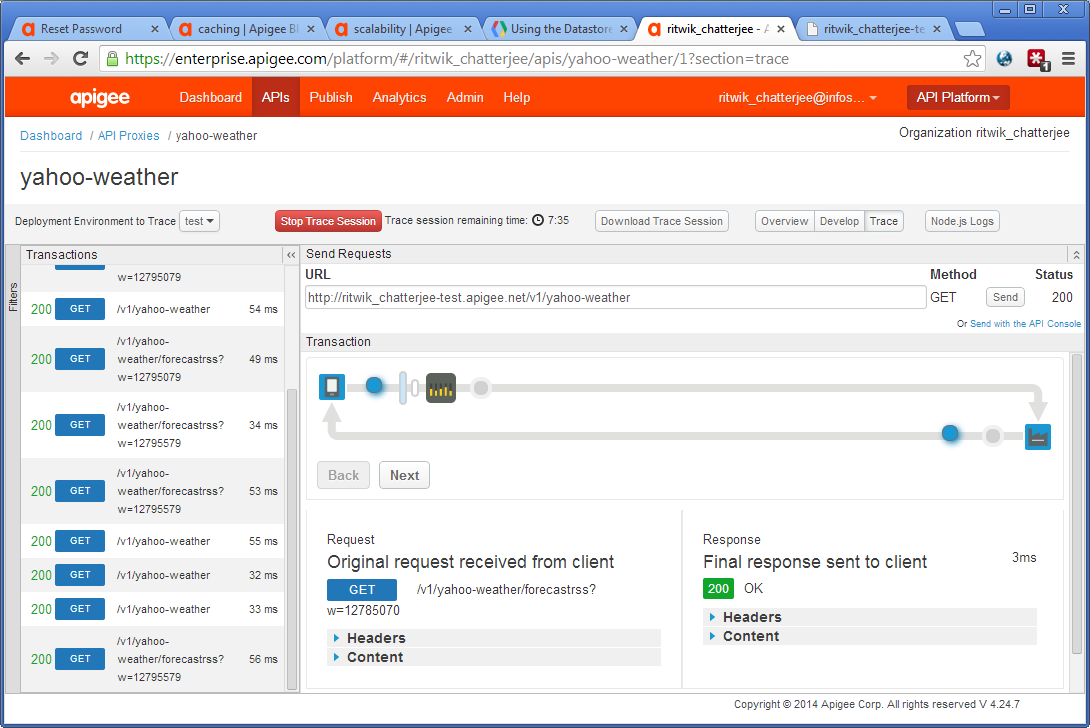






x





Step 1: Add an API and define API resources

When you add an API to Apigee Edge, you configure an API proxy for one or more existing services. Under the hood, Apigee Edge generates a new API, with its own unique network address and a processing pipeline that handles request and response messages.  The API is exposed over the Internet and can be invoked over HTTP by client apps.

The API created can be as simple or as detailed as you like. An API can be a simple 'passthrough', exposing a single API method and funneling any type of request to a particular backend service. It can also  be extremely granular, specifying responses based on the HTTP verb of the request, the URI requested, the content of the request or response, and so on.

Add the Yahoo Weather API

In this tutorial, you will create a proxy for the Yahoo Weather API. The Yahoo Weather API returns XML-formatted weather reports based on an identifier called a WOEID (where on Earth ID). The WOEID for Palo Alto, CA is 12797282. You can call this API directly. In a Web browser, enter:

<http://weather.yahooapis.com/forecastrss?w=12797282>

The result is an XML-formatted weather report.

Now you will go through the steps required to send the same request, and receive the same response, via an API proxy on Apigee Edge.

You must first add the Yahoo Weather API:

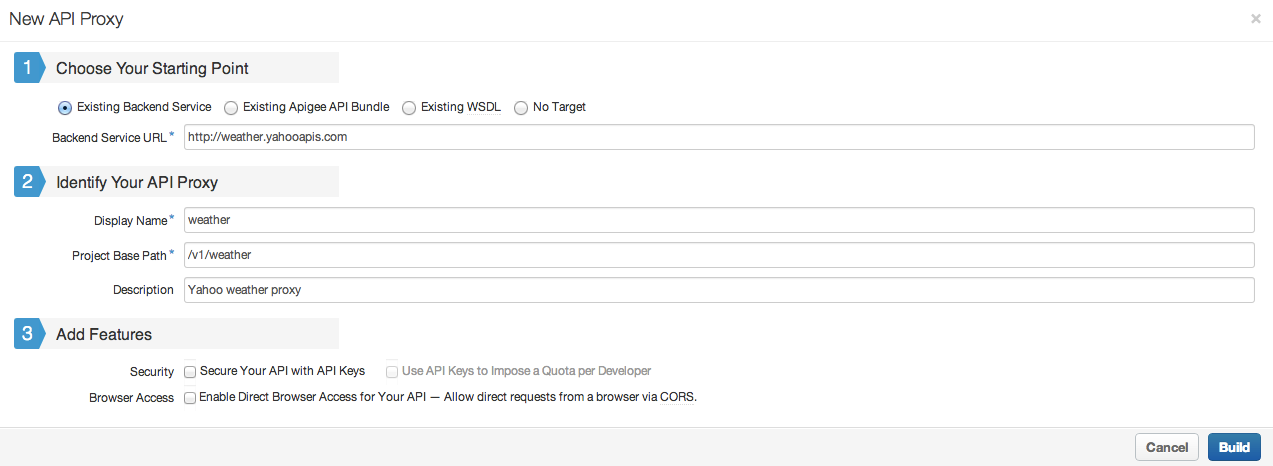
1. Login to [https://enterprise.apigee.com](https://enterprise.apigee.com/). (You can obtain a free account at<https://accounts.apigee.com/accounts/sign_up>.)
2. If you land on the Dashboard page, select the Open Platform button in the API Platform box.
3. In the API Platform UI, select the APIs tab.  
     
   APIs tab_0.png  
     
   You will see two existing API Proxies: oauth and weatherapi. Ignore those for now. They are for use in the tutorial [Secure calls to your API through OAuth 2.0: Client Credentials](http://apigee.com/docs/gateway-services/content/secure-calls-your-api-through-oauth-20-client-credentials).
4. Click the add (+) API Proxy button.

http://apigee.com/docs/sites/docs/files/API%20Proxy%20button.png

1. In the "Choose Your Starting Point" section of the New API Proxy page, select Existing Backend Service and enter *http://weather.yahooapis.com* as the Backend Service URL.  
   The Backend Service URL defines the target URL that Apigee Edge invokes on behalf of apps. You can add any URL that is accessible over the open Internet.
2. In the "Identify Your API Proxy" section, enter *weather* for the Display Name and *Yahoo weather proxy* as the description.
3. The Project Base Path is automatically set to */v1/weather*.  
   The Project Base Path is embedded in the proxy URL that apps call to invoke your API. Apigee Edge uses the proxy URL to match and route incoming request messages to the proper API proxy.  
     
   The base of the API URL is automatically generated based on your organization and environment. Your*organization* is typically synonymous with the API project name you provided when you signed up for an Apigee account. Although you can join more than one organization, most users will have an account in only one organization.  
     
   An *environment* provides a runtime execution context for APIs. By default, Apigee organizations are provisioned with two environments: 'test' and 'prod'.

For example, if your account is in the organization called apifactory, and your API is configured in the environment test, then the URL that you share with developers would be:  
  
http://apifactory-test.apigee.net/v1/weather.

1. Leave the "Add Features" section as is.



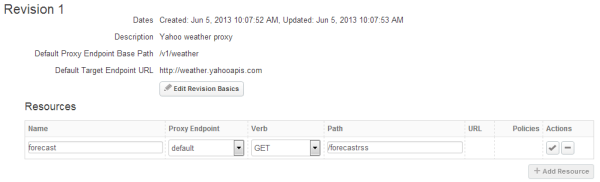
1. Click Build.  
   In response, you should see an acknowledgement that your new API proxy was successfully created and deployed. Apigee Edge generates and deploys a set of configuration files that govern the behavior of the processing pipeline for this API proxy as Revision 1. Your new API is automatically deployed into your test environment and listed in the API summary page.
2. Click Close in the acknowledgement to display the details page for the API proxy.

Define API resources

Now you're ready to define your first resource. Defining API resources is optional. However, by defining API resources, you gain more granular visibility and control over the API.  
  
An API resource is a URI. By defining specific resources, you gain the ability to apply policies to specific URIs, as well as operational visibility into the performance or consumption of those URIs. You can even further refine resource definitions by specifying the HTTP verb used against the URI.

In this example, you define a resource that represents a weather forecast, along with the verb GET.

1. In the main menu of the management UI, click APIs to display the API Proxies page. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).
2. Click *weather* in the API Proxies table.
3. In the Resources section of the weather details page, click the **+ Resource** button.  
   http://apigee.com/docs/sites/docs/files/add_resource_button.png
4. Enter *forecast* for the Name.
5. Click the Proxy Endpoint menu and choose default.
6. Click the Verb menu and choose GET.
7. Enter /forecastrss as the resource Path. By defining this URI, you can monitor and manage the resource separately from other resources.



1. Click the check icon in the Actions column.

The resource is added to the API proxy.

1. Click Save.

Call the Yahoo Weather API by using your new Apigee API proxy

Now that you have a proxy for the Yahoo Weather API, you can invoke it indirectly through Apigee Edge.

In your Web browser, enter the following, substituting your Apigee organization name for {org\_name}.

http://{org-name}-test.apigee.net/v1/weather/forecastrss?w=12797282

For example:

http://myorg-test.apigee.net/v1/weather/forecastrss?w=12797282

Look for the following content in the response:

<title>Yahoo! Weather - Palo Alto, CA</title>

<link>http://us.rd.yahoo.com/dailynews/rss/weather/Palo\_Alto\_\_CA/\*http://weather.yahoo.com/forecast/USCA1093\_f.html</link>

<description>Yahoo! Weather for Palo Alto, CA</description>

<language>en-us</language>

You have added an API to Apigee Edge. You now have the ability modify the characteristics and behavior of the API. In the next step you'll add policies to configure your new API.

Step 2: Add policies to your API

To design and customize the behavior of your APIs, you apply a set of policies. Each policy performs one of several API management tasks. For example, you can add policies for response caching, rate limiting, XML-to-JSON transformation, and so on.

You can add policies using the API Proxy Editor. The API Proxy Editor lets you see the structure of your API proxy and configure its flow. The editor presents a visual representation of your proxy's message flows as well as an editable display of the XML that defines the proxy.

When you add a policy, you specify the *Flow* that defines when the policy is executed. Each request and response path in a ProxyEndpoint and TargetEndpoint defines the following Flow types:

* **PreFlow:**Always executes before any other Flows. The processing steps defined in the PreFlow are applied to every message that passes through an Endpoint.
* **Conditional Flows:** Execute only when a conditional statement defined for the Flow evaluates to true. An Endpoint may define any number of conditional Flows, but only the first conditional Flow whose condition evaluates to true will execute.
* **PostFlow**: Always executes after all other Flows. As with PreFlow, the processing defined for PostFlow is applied to every message.

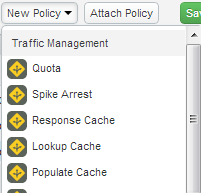
Set a rate limit for your API

To help you learn how to configure APIs, you're going to add the Spike Arrest policy to protect your backend from high traffic. The Spike Arrest policy prevents traffic spikes (or bursts) that can be caused by an increase in usage, buggy clients, or malicious attacks. When the number of requests exceeds the rate limit, the API returns an HTTP 500 error for a request.

1. In the main menu of the management UI, click APIs to display the API Proxies page. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).
2. Click *weather* in the API Proxies table.
3. On the weather detail page, click the Develop button to open the API Proxy Editor.

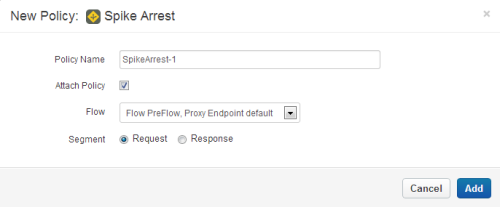
http://apigee.com/docs/sites/docs/files/ovDevTrace_v24.png

1. In the API Proxy Editor, click New Policy. This displays a categorized list of all the policies you can create.



1. Select Spike Arrest in the Traffic Management category.

The New Policy dialog appears showing:



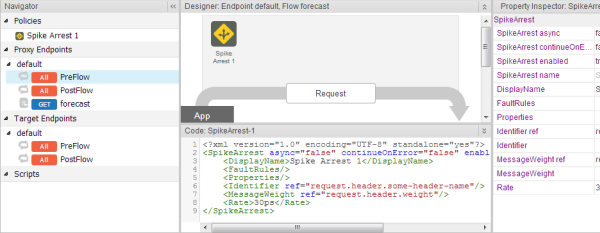
* + The Policy name assigned to the new policy.
  + An Attach Policy checkbox. Checking this box attaches the policy to the Flow.
  + A Flow drop-down menu.
  + Segment request and response radio buttons.

1. Click Add to accept the defaults.

The new policy is attached to the PreFlow flow of the request message. This means that Apigee Edge will include the Spike Arrest policy as part of its processing of messages sent from a client app to your API. Because you accepted the default selections for the Flow and Segment options, the Spike Arrest policy is applied to request messages at the ProxyEndpoint.

Learn about flows and endpoints in [Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies). What’s important to understand at this point is that the Spike Arrest policy will be processed first by your API proxy.

1. Ensure that the PreFlow flow is selected in the left side of the API Proxy Editor.
2. Select the Spike Arrest policy and examine what’s been added:



* + The policy is added to the list of policies in the **Navigator view** in the upper left of the API Proxy Editor.
  + The policy is added to the **Designer view** in the top center of the API Proxy Editor, which is a visual representation of your proxy's message flows. Note that the icon appears only when you select PreFlow under Proxy Endpoints in the left navigation area.
  + The XML for the policy is displayed in the **Code view** in the bottom center of the API Proxy Editor.
  + The XML element and attribute values for the policy are displayed in the **Property Inspector**at the right in the API Proxy Editor**.**

1. In the XML for the policy, change the value of the <Rate> element to *75ps*. You can specify the rate as an integer value per minute (pm) or per second (ps).

This limits traffic to a maximum of 75 messages per second. Notice that the Rate value in the Property Inspector also changes to 75ps. You can also change the Rate value in the Property Inspector and it will be reflected in the XML view.

1. Click Save to save the current revision with your changes.

Learn more

[Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies)

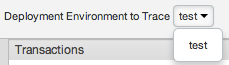
[**Back to top**](http://apigee.com/docs/api-services/content/add-and-configure-your-first-api#top)

Step 3: Execute and Trace API calls

The Trace feature lets you review data on how a message changes as policies are executed. You can see how each policy is performing and how long it takes to execute. By examining the data, you can see exactly where things are breaking down. Trace shows you data from a number of calls taken from live data. However, since the weather API is not live you'll send calls manually so you can see how policies are being executed.

1. In the management UI, click the APIs tab. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).

http://apigee.com/docs/sites/docs/files/APIs%20tab_2.png

1. Click *weather* in the API Proxies table.
2. On the weather detail page, click the *Trace* button to access the Trace page.  
   http://apigee.com/docs/sites/docs/files/ovDevTrace_v24.png
3. On the Trace page, choose the *test* environment. This will let you set up a trace session on the test environment.  
   
4. Click Start Trace Session.

http://apigee.com/docs/sites/docs/files/Start%20Trace%20Session.png

This runs a trace session for 10 minutes. While the trace session is running, messages are captured from live traffic. However, since your API has not be exposed to developers yet, you won't have any traffic.

1. Request your API.  
     
   You can request your API directly from a Web browser by entering the following URL:

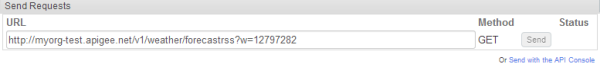
http://{org-name}-{env}.apigee.net/v1/weather/forecastrss?w=12797282

where:

* + {org-name} is the name of your organization
  + {env} is the environment, which is either test or prod. For this tutorial, you deployed the API Proxy to the test environment.

For example:

http://myorg-test.apigee.net/v1/weather/forecastrss?w=12797282  
  
Or, request the URL from the Trace page by entering the URL and clicking Send in the Send Requests section.

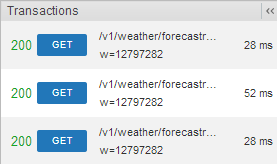


You can click Send multiple times to generate as much traffic as you wish.

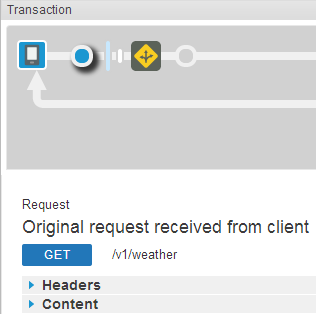
You can also use the Apigee Console to send a message directly to your API. The API Console lets you examine the request and response objects, set request headers, and set body parameters. To send the request from the console:

* + Click Send with the Apigee Console, just below the Send button in the Send Requests area of the Trace page.  
    The console appears in another browser window.
  + Enter the URL of your API.
  + Click Send.  
    This sends a call to your API. You can click send multiple times to generate as much traffic as you wish.
  + Switch back to the trace session.

The processing time for each message appears in the Transactions section of the Trace page.



1. Click the first message in the Transactions section.  
   The response and request flows of your API call are shown graphically in the Transaction section (below the Send Requests section). Below the graphic you can see the results of the first step in the request flow.



1. Click the Next button on the far-right of the window to see how the Spike Arrest policy executed.
2. Click Next again to see the response from the backend service.  
   At each step you can expand the results to get more detail. In this case, expand the Content section so you can see the forecast data that was sent back from the Yahoo weather service.
3. In the main menu of the management UI, click APIs to display the API Proxies page.
4. Click *weather* in the API Proxies table.
5. On the weather detail page, click the Develop button to open the API Proxy Editor.
6. In the Spike Arrest policy, change the value of the <Rate> element to *1ps*, and click the Save button to save the current revision with your changes.
7. Start a new trace session, and make several fast requests to your API, so you make more than one request in a second.  
     
   You will see an HTTP 500 error in the Transactions section of the Trace page when you exceed one request per second. ​If you can't send requests quickly enough to get a 500 response, change the rate to*5pm* (5 requests per minute, which Edge enforces at 10 milisecond intervals) and send more requests.

## Step 4: Convert XML to JSON

The response from your weather API contains XML data. This can be a problem for developers whose apps want to access the backend service through your API, but only accept JSON responses from their RESTful API calls. You can however add the XML to JSON policy to your API to convert response data from XML to JSON.

With this policy, the payload of an XML message is parsed and converted into JSON, and the content-type is changed to application/json. The policy only works when the source content-type is application/xml. See[Convert XML to JSON](http://apigee.com/docs/api-services/content/convert-xml-json) for details.

To add an XML to JSON policy:

1. In the management UI, click the APIs tab. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).
2. Click weather in the API Proxies table.
3. On the weather detail page, click the Develop button to open the API Proxy Editor.
4. In the API Proxy Editor, click New policy, and select XML to JSON in the Mediation category.
5. In the New Policy dialog select the following:
   * Keep the default values for Policy Name and Attach Policy.
   * In the Flow drop-down menu select Flow PostFlow, Proxy Endpoint default.
   * The Response segment radio button.
6. Click Add.

The new policy is attached to the PostFlow flow of the response message. Because you selected Flow PostFlow, Proxy Endpoint default, and chose the Response segment, the XML to JSON policy is applied to response messages at the ProxyEndpoint.  
  
If the XML to JSON policy does not appear in the Designer area of the screen, select PostFlow under Proxy Endpoints in the left navigation area. To see the Spike Arrest policy, select PreFlow under Proxy Endpoints.

1. Click Save.
2. Request the URL of the API in a browser, from the Trace page, or from the Apigee Console to see that the response is now formatted as JSON:

http://{org-name}-test.apigee.net/v1/weather/forecastrss?w=12797282

### Learn more

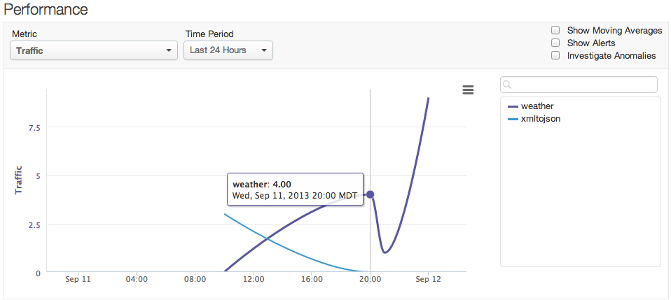
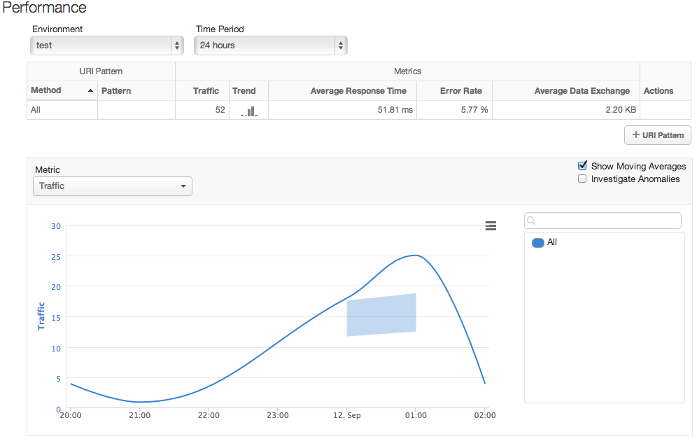
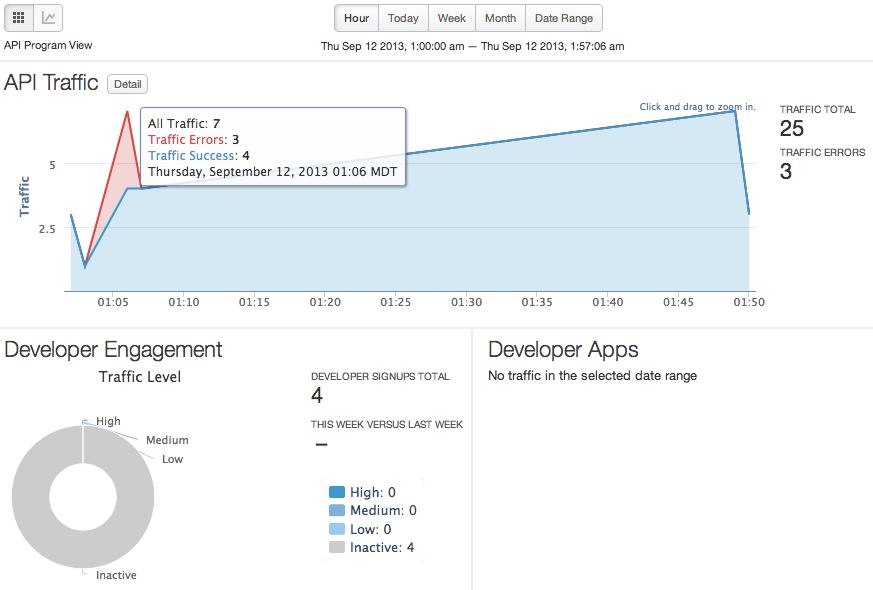
* [Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies)
* [Overview of API publishing](http://apigee.com/docs/api-services/content/overview-api-publishing)

[**Back to top**](http://apigee.com/docs/api-services/content/add-and-configure-your-first-api#top)

## Step 5: Monitor your API's performance

Apigee Edge collects information as data passes through it. The data includes API call information (URL, IP, and user ID) latency, errors, and so on. This data is gathered into trend charts and tables throughout the management UI. You can use this data to both monitor the overall health of your API program and the health of individual APIs. See [Visualize analytics data](http://apigee.com/docs/analytics-services/content/visualize-analytics-data) for more information.

Now that you've deployed your API and made requests, you can use the data charts to see metrics.

1. First, look at the performance of all your APIs. In the management UI, click APIs. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).
2. If not already selected, select test in the Environment drop-down list at the top of the page.
3. Scroll down to the Performance section to the chart displaying metrics for all of your APIs. There may be a few minutes' lag time between when you send requests and when the metrics are available in the charts.  
   If you were to make requests over several hours, your chart might look something like this:  
     
     
     
     
   You can show or hide metrics for APIs and policies by clicking their names in the right pane of the chart. You can select different time periods, different types of metrics, and you can move your mouse pointer over the chart to see details about specific data points.  
     
   You probably won't have much data yet. But select a Time Period of one hour to see information about the weather API you just created.
4. Now look at metrics for only your weather API. Click weather in the API Proxies list, click Overview, then scroll down to the Performance section, where you can view metrics for only the weather API and its resources. Again, over several hours of sending requests, your chart might look something like this:  
     
     
   
5. For a comprehensive, multi-dimensional view of your entire API program, click Dashboard in the management UI menu. The dashboard provides views of traffic, partner and developer engagement, and top performing APIs, products, apps, and developers. You can view different date ranges and drill down into API traffic details.  
     
   Analytics charts can also be exported to various file formats.  
     
   

Secure calls to your API through API key validation

It’s important to protect your API from unauthorized access. One way to do that is to validate API keys (also called "public keys", "consumer keys" or "app keys").

The tutorial [Add and configure your first API](http://apigee.com/docs/api-services/content/add-and-configure-your-first-api) shows how to create your first API proxy. This tutorial shows you how to secure an API using the VerifyAPIKey policy.

In this tutorial, you create a new version of the API proxy to the Yahoo weather service that adds the VerifyAPIKey policy automatically to the API proxy when you create it. The advantage of creating a new version of the API proxy is that Apigee Edge creates all of the policies necessary to enforce API keys for you. If you have an existing API proxy and want to add API key validation, you can add those policies manually at any time.

Before you start, here is some information about API products, API keys, and apps.

**The Relationship between organization, API products, apps, and API keys**

Apigee Edge generates an API key for an app that is registered in an organization and is associated with an API product.

* An **organization** represents a container that holds all the objects in an Apigee account. The name of your organization is typically the API project name you provided when you signed up for an Apigee account.
* An **API product** is a bundle of API proxies combined with a service plan that sets limits on access to those APIs. API products are the central mechanism that Apigee Edge uses for authorization and access control to your APIs.
* An **app** is a client-side app that a developer registers with an API product. Registering the app with the API product generates the API key.
* An **API key** is a string with authorization information that a client-side app uses to access the resources exposed by the API product. The API key is generated when a registered app is associated with an API product.

**Developers must provide this key in requests to your API’s resources.**

You enforce API key validation by applying the VerifyAPIKey policy (see [Enforce access control using VerifyAPIKey](http://apigee.com/docs/api-services/content/enforce-access-control-using-verifyapikey)). At runtime, that policy verifies the following:

* That the API key is valid
* That it hasn’t been revoked
* That it matches the API key for the API product that exposes the requested resources

When an app makes a call to your API, the app must supply the correct key. If the key is valid, the call can access the resources exposed in the API product. If the key is invalid, it results in an authorization failure.

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## What you'll do in this tutorial

This tutorial shows you how to use the Apigee Edge management UI to create an API proxy and secure it using the VerifyAPIKey policy. In this tutorial, you add the VerifyAPIKey policy to the API proxy when you initially create the proxy. You can also add the VerifyAPIKey policy to an existing proxy.

**Video:** Check out this short video to learn how to add the VerifyAPIKey policy to an existing proxy.

[Show/Hide Video](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#demo1)

For more information on API keys, see [Secure APIs with API keys](http://apigee.com/docs/api-platform/content/secure-apis-api-keys).

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

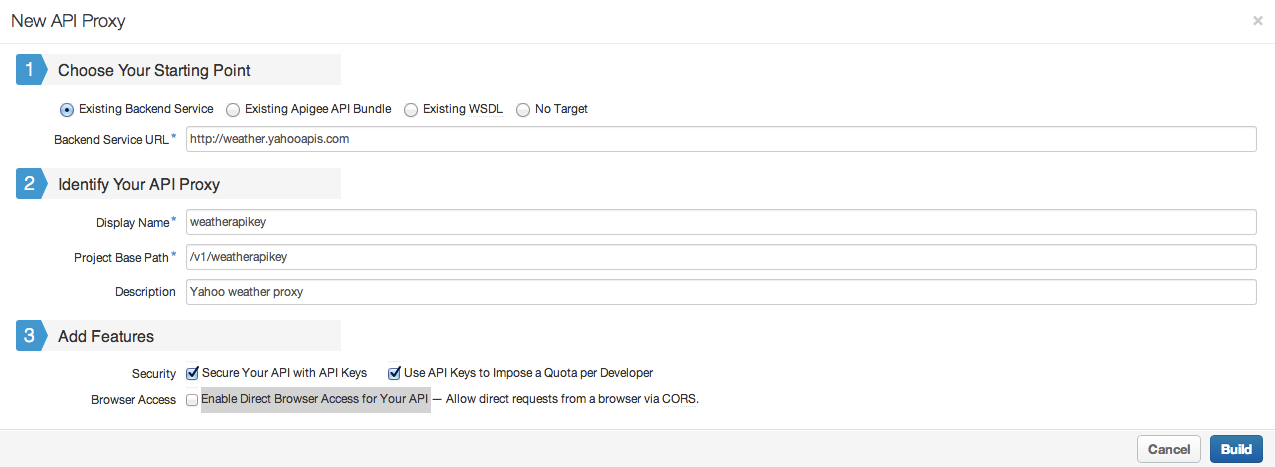
## Step 1: Create an API proxy

Configure an API proxy for the Yahoo! Weather API. This repeats some of the same actions as in the tutorial[Add and configure your first API](http://apigee.com/docs/api-services/content/add-and-configure-your-first-api). However, you’ll also add security to your API proxy in this step.

1. Login to [https://enterprise.apigee.com](https://enterprise.apigee.com/).
2. In the API Platform UI, select the APIs tab.
3. Click (+) API Proxy.
4. Make the following selections and entries in the New API Proxy page.

| **Field** | **Value** | **Description** |
| --- | --- | --- |
| **Choose your starting point** |  |  |
| Starting point radio button | Existing Backend Service | The API proxy will target an existing service accessible over the Internet. |
| Backend Service URL | http://weather.yahooapis.com | The URL that the API Platform invokes on behalf of apps that call your API through the API proxy URL. This is the URL for the Yahoo! Weather API. |
| **Identify your API proxy** |  |  |
| Display Name | weatherapikey | The name displayed for your API |
| Project Name | /v1/weatherapikey | The base path that the API platform uses to construct the API proxy URL. Apps will call this URL to invoke your API. [1](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#footnote1) |
| Description | Yahoo weather proxy | The description of the API. |
| **Add Features** |  |  |
| Secure Your API with API Keys checkbox | checked | This adds API key-based authentication to your API proxy. When this checkbox is checked, it displays the Impose Quota per Developer checkbox. |
| Use API Keys to Impose a Quota per Developer checkbox | checked | This adds a policy that limits the number of request messages that can be submitted to your API proxy by an individual app over an interval of time. |
| Enable Direct Browser Access for Your API  checkbox | unchecked | If checked, this enables CORS (Cross-origin resource sharing), allowing a browser to make direct requests to another domain. |

1 The API proxy URL is of the form http(s)://{org}-{env}.apigee.net/{base\_path}, where {org} is your organization, {env} is the environment, and {base\_path} is the root under which all services are provided by the API. You specify the base path (in this tutorial, /v1/weatherapikey) in the dialog. Apigee Edge then constructs the API proxy URL. So if your organization is myorg, and the environment is test, the constructed API proxy URL ishttp://myorg-test.apigee.net/v1/weatherapikey. Apps will call this URL to invoke the API.

The filled-in page should look like this:  


1. Click Build to save the entries you made in the page.  
   In response, you should see an acknowledgement that your new API proxy was successfully created and deployed in the 'test' environment.
2. Click Close.

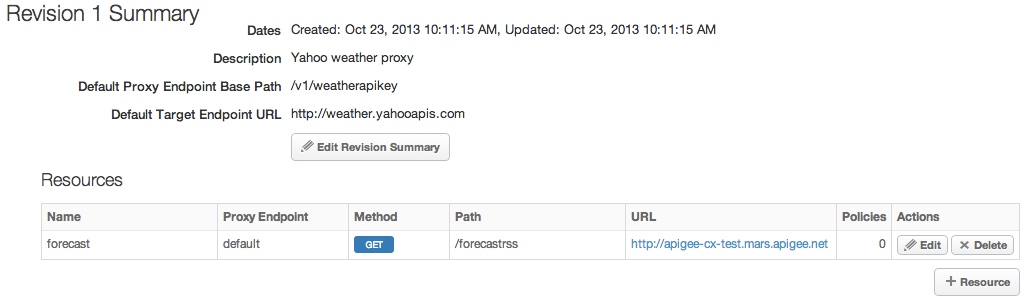
Learn more:

* [Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies)
* [Configure and deploy an API proxy](http://apigee.com/docs/api-platform/content/configure-and-deploy-api-proxy) (creating an API proxy programmatically)

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 2: Define API resources

1. In the management UI, select the APIs tab to display the API Proxies page.
2. Click weatherappikey in the API Proxies table.
3. In the Resources section, click the (+) Add Resource button.
4. Enter the following values:
   * Name: forecast
   * Proxy Endpoint: default
   * Verb: GET
   * Path: /forecastrss



1. Click the check icon in the Actions columns to validate the resource.
2. Click Save.

The resource is added to the API proxy.

Learn more:

* [Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies)
* [Configure and deploy an API proxy](http://apigee.com/docs/api-platform/content/configure-and-deploy-api-proxy) (creating an API proxy programmatically)

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 3: Examine generated policies

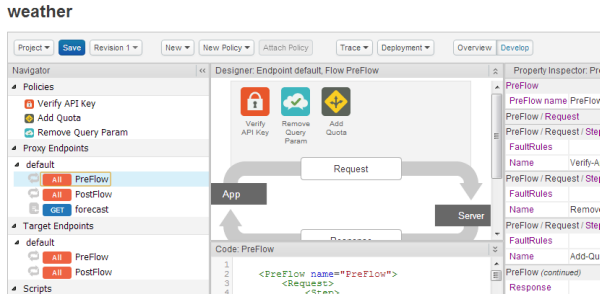
1. Open the API Proxy Editor by clicking the Develop button in the details page for the weatherapikey API proxy.
2. On the left side of the API Proxy Editor, select PreFlow under Proxy Endpoints > default to show the generated policies in the Policy Designer.  
     
   For more information on Flows and Endpoints, see [Understanding APIs and API proxies](http://apigee.com/docs/api-services/content/understanding-apis-and-api-proxies).

Notice that three policies have already been added to your API proxy’s request message flow:

| **Name** | **Policy Type** | **Description** |
| --- | --- | --- |
| Verify API Key | VerifyAPIKey | Verifies the API key for an API product, returns an error if it is invalid, and if it is valid, looks up the attributes from the API product. |
| Remove Query Param | AssignMessage | Modifies request messages in the API proxy flow. Here, the policy is used to prevent the API key from being sent to the backend URL. |
| Add Quota | Quota | Enforces a limit on the number of API calls made by apps over an interval of time. |

Also notice what’s been added for these policies to the API Proxy Editor:

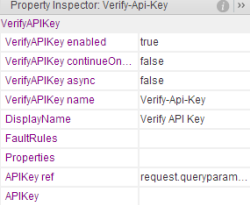
* The policies are added to various views in the Proxy Editor.
* The policies are added to the list of policies in the **Navigator view**.
* The policies are added to the **Designer** view, which is a visual representation of your proxy's message flows.
* The XML for each policy is displayed in the **Code view**.
* The XML element and attribute values for each policy are displayed in the **Property Inspector**.



### The VerifyAPIKey policy

The VerifyAPIKey policy validates API keys. Typically, you want API key validation to happen as soon as a request is received by your API proxy. The PreFlow is always the first flow to execute at an endpoint, and the ProxyEndpoint is the first endpoint in the request pipeline. Because the VerifyAPIKey policy is the first policy in the PreFlow, the API key will be validated by the policy as soon as it’s received by the API proxy. With the policy in place, this means that a call to the API won't work unless it includes the appropriate API key.

Click the Verify API Key policy icon, and examine the XML for the policy in the Code view. You can also view the values for the policy’s XML elements and attributes in the Property Inspector. Here’s what you should see in the Property Inspector:



The XML for the policy should look something like this:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<VerifyAPIKey async="false" continueOnError="false" enabled="true" name="Verify-Api-Key">

<DisplayName>Verify API Key</DisplayName>

<APIKey ref="request.queryparam.apikey"/>

</VerifyAPIKey>

Note especially the ref parameter of the <APIKey> element, which identifies where the policy should check for the API key. API keys can be located in a query parameter, a form parameter, or an HTTP header. Apigee Edge provides a message variable for each type of location.

| **Request variable** | **Where expected key location is provided** |
| --- | --- |
| request.queryparam.{queryparam\_name} | query parameter |
| request.header.{header\_name} | header |
| request.formparam.{formparam\_name} | form parameter |

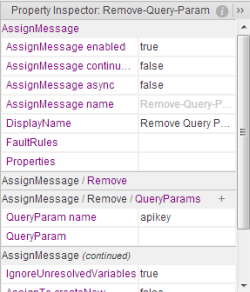
In this case, the key is expected in a query parameter named apikey.

<apikey ref="request.queryparam.apikey" />

### The AssignMessage policy

The next policy to be enforced in the PreFlow is an AssignMessage policy named Remove Query Param. It strips the verified API key from the outbound request to the backend service. Only the Apigee Edge security layer needs to be aware of the API key.

Examine the XML for the AssignMessage policy. Here’s what you should see in the Property Inspector:



The XML for the policy should look like this:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<AssignMessage async="false" continueOnError="false" enabled="true" name="Remove-Query-Param">

<DisplayName>Remove Query Param</DisplayName>

<Remove>

<QueryParams>

<QueryParam name="apikey"/>

</QueryParams>

</Remove>

<IgnoreUnresolvedVariables>true</IgnoreUnresolvedVariables>

<AssignTo createNew="false" transport="http" type="request"/>

</AssignMessage>

The AssignMessage policy defines a set of elements that perform actions such as populating or modifying HTTP headers, query parameters, and XML or JSON payload content. For example, the AssignMessage policy above uses a <Remove> element to remove the query parameter named apikey from the HTTP request message attached to the flow.

An AssignMessage policy is named that way because the policy requires a message to be 'assigned' as a variable. By assigning a message to a variable, you make the message available to the message flow. Other policies in the flow can then access the variable's content.

For example, the following creates an HTTP message and assigns it to a variable named Request.

<AssignTo createNew="true" transport="http" type="request">Request</AssignTo>

The type value indicates the type of the message: request or response.

If the createNew parameter is set to false, as it is in the following <AssignTo> element, no variable is created.

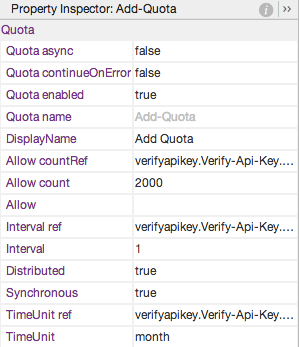
<AssignTo createNew="false" transport="http" type="request"></AssignTo>

However, in this case, you can use the AssignMessage policy to modify the request or response message where the policy is attached.

### The Quota policy

The final policy in the PreFlow is a Quota policy named Add Quota. It enforces a limit on the number of API calls made by apps over an interval of time. The limit is set in the API product. For example, you may want to limit apps to 1 request per minute, or to 10,000 requests per month.

Examine the XML for the Quota. Here’s what you should see in the Property Inspector:



The XML for the policy should look like this:

<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<Quota async="false" continueOnError="false" enabled="true" name="Add-Quota">

<DisplayName>Add Quota</DisplayName>

<Allow countRef="verifyapikey.Verify-Api-Key.apiproduct.developer.quota.limit" count="2000"/>

<Interval ref="verifyapikey.Verify-Api-Key.apiproduct.developer.quota.interval">1</Interval>

<Distributed>true</Distributed>

<Synchronous>true</Synchronous>

<TimeUnit ref="verifyapikey.Verify-Api-Key.apiproduct.developer.quota.timeunit">month</TimeUnit>

</Quota>

The Quota policy sets values for the maximum count of messages for the quota (2000), the time interval (1), and the time unit (month). The policy also references variables that are populated by Apigee Edge when the VerifyAPIKey policy is enforced. These values take precedence over the values set in the policy. For example, the following specifies a maximum message count for the quota based on the limit set in the API product that is used in validating the request:

<Allow countRef="verifyapikey.Verify-Api-Key.apiproduct.developer.quota.limit" count="5000"/>

Learn more:

* [Enforce access control using VerifyAPIKey](http://apigee.com/docs/api-services/content/enforce-access-control-using-verifyapikey)
* [Generate or modify messages using AssignMessage](http://apigee.com/docs/api-services/content/generate-or-modify-messages-using-assignmessage)
* [Rate limit API traffic using Quota](http://apigee.com/docs/api-services/content/rate-limit-api-traffic-using-quota)

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 4: Add an API product

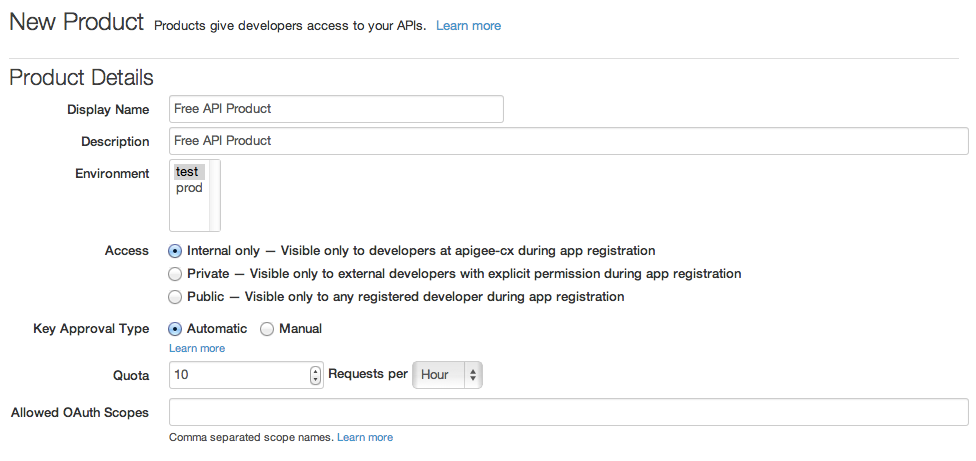
An API product is a collection of API resources combined with a service plan and presented to developers as a bundle. The product can also include some metadata specific to your business for monitoring or analytics.

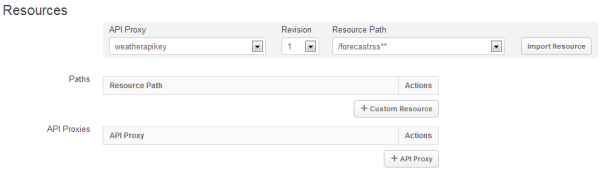
API products are also the central mechanism for authorization and access control to your APIs. API keys are provisioned for API products. An API key is generated when a registered app is associated with an API product. After you create an API product you associate it with an app to generate the API key.

1. In the management UI, click the Publish tab, then Products.
2. Click (+) Product.
3. Enter or select the following in the Add Product dialog fields:

| **Field** | **Value** |
| --- | --- |
| Display Name | Free API Product |
| Description | Free API Product |
| Environment | test |
| Access | Internal only |
| Key approval type | Automatic (the API key is automatically approved after it has been generated) |
| Quota | 10 requests per hour (limits the number of requests to 10 per hour) |
| Allowed Oauth Scopes | Read |

The filled-in dialog should look like this:



1. In the Resources section, choose:
   * **API Proxy**: weatherapikey
   * **Revision**: 1
   * **Resource Path**: /forecastrss\*\*  
       
     
2. Click Import Resource. The resource path is added, and the weatherapikey API proxy is automatically added to the API Proxies for Product section.
3. Click Save.  
   Your new product is listed on the Products page.

Learn more: [Creating API products](http://apigee.com/docs/gateway-services/content/creating-api-products)

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 5: Add a developer

Before an API key can be generated, you need to add an app and associate it with an API product. However, you can't create an app without a developer, so you need to have at least one developer registered in your organization.

If you have a public API product that developers can sign up for through a portal, developers can register themselves. However, for the purposes of the tutorial, you don’t have a developer portal. You add a developer directly in the management UI.

1. In the management UI, click the Publish tab, then Developers.
2. Click (+) Developer.
3. In the New Developer dialog, enter the following values:
   * First Name: Jane
   * Last Name: Tutorial
   * Email address: janetutorial@example.com
4. Click Save.  
   The new developer appears in the list of developers on the Developers page.

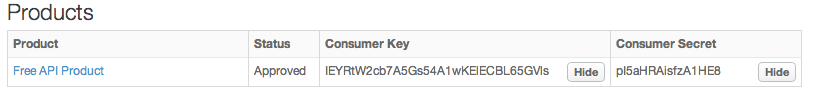
Learn more: [Give developers access to your API](http://apigee.com/docs/api-platform/content/give-developers-access-your-api)

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 6: Create an app and register the app with the API product

Now that you have an API product and a developer you can create an app and register the app with the API product. You register the app to generate an API key for your API products. You can then distribute the key to app developers so they can access the features in the API products from the app.

You create and register an app in the management UI.

1. In the API Platform UI, click the Publish tab, then Developer Apps.
2. Click (+) Developer App.
3. In the Add an App dialog, enter the following values:
   * Display Name: Weather API Key App
   * Developer: Jane Tutorial (janetutorial@example.com)
   * Callback URL: login.weatherapp.com
4. Associate the app with a product: In the Products section, click + Product, select Free API Product, and click the check mark icon to accept the changes.
5. Click Save.  
   The new app appears in the list  of apps on the Developer Apps page.
6. Select Weather API Key App in the list of apps. This opens the details page for the app.
7. Because you selected Key Approval Type: Automatic in a previous step, the API key is automatically approved and you can view it immediately. (If you had selected Approval Type: Manual, you would need to click Approve in the Actions column for Free API Product to approve the API key.)
8. Click Weather API Key App in the Developer Apps list to view the app details.
9. In the Products section, next to the entry for Free API Product, click Show in the Consumer Key and Consumer Secret columns to display the generated keys.  
   

The Consumer Key is another name for the API key and is the only key an app needs to access theforecast resource through your API proxy . The Consumer Secret is needed (along with the Consumer Key) in securing an API through OAUth 2.0. See, for example, [Secure APIs with OAuth 2.0: client credentials](http://apigee.com/docs/api-platform/content/secure-apis-oauth-20-client-credentials).

Learn more: [Give developers access to your API](http://apigee.com/docs/api-platform/content/give-developers-access-your-api).

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 7: Test the VerifyAPIKey policy

To ensure that the VerifyAPIKey policy is working correctly, you need to test it by making a call to the API proxy with a valid API key. You can request the API proxy in a browser, from the Trace page of the API proxy, from the Apigee Console, or by using cURL. The URL to request is in the form:

http://{org-name}-test.apigee.net/v1/weatherapikey/forecastrss?w=12797282&apikey=IEYRtW2cb7A5Gs54A1wKElECBL65GVls

Substitute your Apigee organization name for {org-name} and the correct Consumer Key value.

If you omit the API key from the request, or specify an invalid key value, you will receive an error response in the form:

{"fault":{"faultstring":"Failed to resolve API Key variable null","detail":{"errorcode":"steps.oauth.v2.FailedToResolveAPIKey"}}}

**To use a browser:**

1. Enter the URL in a browser window.

**To use the Trace page:**

1. In management UI, click the APIs tab. If the API Platform page is not open, click [here](https://enterprise.apigee.com/).
2. Click weatherapikey in the API Proxies table.
3. On the weatherapikey detail page, click the Trace button to access the Trace page.
4. On the Trace page, choose the test environment.
5. Click Start Trace Session.
6. Request your API.  
   You can then use the Trace page to examine each part of the request and response.

**To use the Apigee Console:**

1. From the Trace page, click Send with the Apigee Console, just below the Send button in the Send Requests area of the Trace page.  
   The console appears in another browser window.
2. Enter the URL of your API.
3. Click Send.
4. Switch back to the Trace page to see the trace.
5. Enter the HTTP verb and URL for the weather forecast resource, and provide the API key. (You can also enter the URL for the API desired resource, without the GET verb, in your favorite browser.)

**To use cURL:**

1. Install cURL.  
   The [cURL](http://curl.haxx.se/) tool is a convenient way to submit URL-based commands from a command line.
2. Use cURL to request the URL:  
     
   curl "http://myorg-test.apigee.net/v1/weatherapikey/forecastrss?w=12797282&apikey=bagW9lvG8dX4ooUC5wcqW4PP1KZX5jPQ"

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Step 8: Send the API key as a header

The default configuration of the VerifyAPIKey policy in your API proxy specifies to send the API key as a query parameter. In this step, you modify the policy to send the API key as a header instead.

1. In the management UI, select the APIs tab to display the API Proxies page.
2. Click weatherappikey in the API Proxies table.
3. On the weatherapikey detail page, click the Develop button to open the API Proxy Editor.
4. On the left side of the API Proxy Editor, select PreFlow under Proxy Endpoints > default to show the policies in the Policy Designer.
5. Select the Verify API Key policy in the request.
6. Change the ref value of the <APIKey> element to request.header.apikey.  
     
   <APIKey ref="request.header.apikey"/>  
     
   This value specifies that the API key is passed in a header named apikey.
7. Click  Save.
8. Click the Trace button to access the Trace page.
9. Open the Apigee Console by clicking Send with the Apigee Console, just below the Send button in the Send Requests area of the Trace page.
10. Enter the URL of your API, but omit the apikey query parameter.
11. Click Headers, just below the URL.
12. Add a header named apikey, with a value of the API key.
13. Click Send.

[**Back to top**](http://apigee.com/docs/api-services/content/secure-calls-your-api-through-api-key-validation#top)

## Appendix: Retrieving and revoking the API key

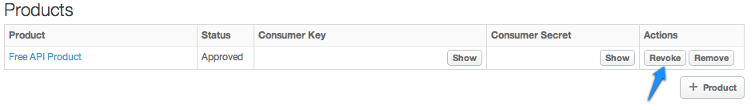
As an admin for your organization, you can retrieve the API key for an app. You can also revoke an API key for an app. You might need to revoke an API key if you find a serious error in an API resource. When you revoking an API key, it blocks access to the associated API product, and all apps that are assigned that key can no longer access the resources exposed by that API product. If there are problems with those resources, revoking the key can prevent errors in your developers’ applications.

**To retrieve an API key:**

1. In the management UI, click the Publish tab, then Developer Apps.
2. In the Developer Apps page, select the app (for example, Weather API Key App).  
   This displays the details page for the app. The keys appear when you click Show in the Consumer Key and Consumer Secret columns.

**To revoke an API key:**

1. In the management UI, click the Publish tab, then Developer Apps.
2. On the Developer Apps page, select the app (for example, Weather API Key App).
3. Click Edit to modify the developer app, and in the Products section of the page, click Revoke in the Actions column.



After you revoke the API key, 'Revoke' changes to 'Approve' in the Actions column. If you click Approve, it makes the API key available again to the app.

If you click Remove in the Actions column, it removes the API product from the app.

Secure calls to your API through OAuth 2.0: Client Credentials

## Step 1: Deploy the API Proxies

