

## TUTORIAL:

# DEPLOYING A 3SCALE API GATEWAY ON RED HAT OPENSHIFT

This tutorial describes how to deploy a dockerized version of the 3scale API Gateway 1.0 (APIcast) that is packaged for easy installation and operation on OpenShift V3.

In the tutorial we use the OpenShift Origin VM—a Virtual Machine image you can download and run locally, so that you can follow this tutorial without running a full OpenShift deployment.

3scale offers multiple options for implementing traffic management

- **Deploy an API gateway**, which is a customized reverse proxy between the client application and your API backend. The gateway then handles authorization of incoming calls and traffic reporting
- **Integrate using a software plugin** embedded in your API application code. Native-language plugins manage request authorization via 3scale's Service Management API.

# Tutorial Prerequisites

To follow the tutorial steps below, you'll first need to address the following prerequisites:

## 3scale Account Configuration

You should have a 3scale API provider account (you can easily [create one here](#)) and have an API configured in the 3scale Admin Portal.

These steps will show you how to find the following necessary information:

- Your 3scale admin URL
- The API provider key for your 3scale account
- Your user key

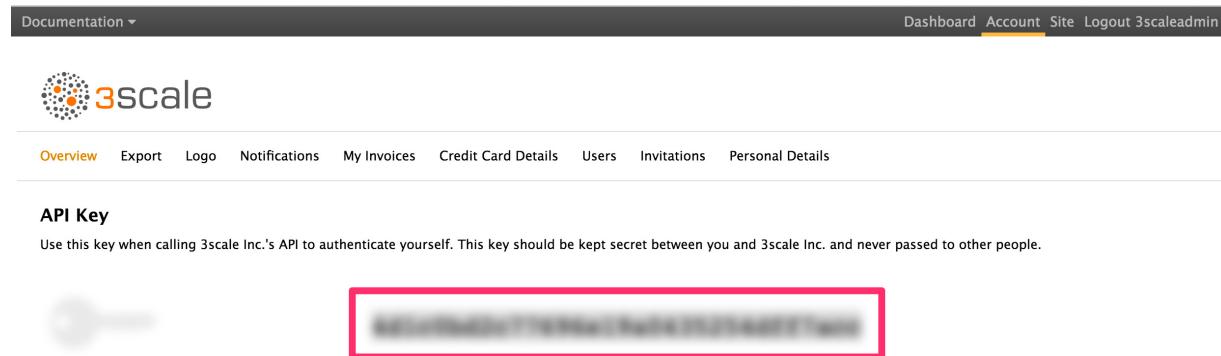
If you already have this information handy and your API is already configured in 3scale, feel free to skip to the **Set up OpenShift** section.

*If you do not have an API running and configured in your 3scale account, please follow the instructions in our [Quickstart](#) to do so.*

Log in to your 3scale Admin Portal with the URL provided to you. It will look something like this:

<https://MYDOMAIN-admin.3scale.net/>

Note down your 3scale Admin Portal URL, in this tutorial we will refer to it as  
**THREESCALE\_ADMIN\_URL**



The screenshot shows the 3scale Admin Portal dashboard. At the top, there is a navigation bar with links for Documentation, Dashboard, Account (which is highlighted in orange), Site, Logout, and 3scaleadmin. Below the navigation bar is the 3scale logo. The main content area has a dark header with the text "API Key". Below the header, there is a note: "Use this key when calling 3scale Inc.'s API to authenticate yourself. This key should be kept secret between you and 3scale Inc. and never passed to other people." A large, red rectangular box highlights a blurred-out API key value.

From the **Overview** sub-tab note down your 3scale Provider Key, referred to in the UI as **API Key**.

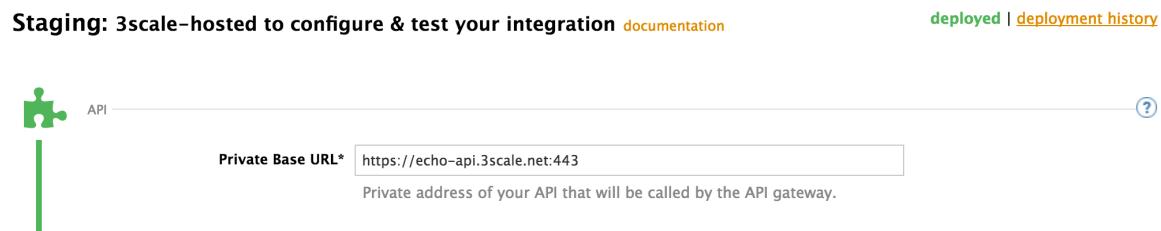
Later in this tutorial we refer to this as your **THREESCALE\_PROVIDER\_KEY**

Keep your 3scale Provider Key private. Do not share it with anyone and do not put into code repositories or into any document that may reveal it to others.

In the 3scale Admin Portal you should either have an API of your own running and configured or use the Echo API that was set up by the onboarding wizard. This tutorial will use the Echo API as an example throughout.

Navigate to the **Dashboard > API** tab. If you have more than one API in your account, select the API you want to manage with the API gateway. Select the Integration link at top-left.

If you're setting this up for the first time, you'll need to test to confirm that your private (unmanaged) API is working before proceeding. If you've already configured your API and sent test traffic, feel free to skip this step.



In the screenshot above, the 3scale provided Echo API is used as an example. You can use this or configure the API to refer to your real API's Private Base URL

Test your private (unmanaged) API is working using this **curl** command:

```
curl "https://echo-api.3scale.net:443/"
```

You should get a response similar to this:

```
{
  "method": "GET",
  "path": "/",
  "args": "",
  "body": "",
  "headers": {
    "HTTP_VERSION": "HTTP/1.1",
    "HTTP_HOST": "echo-api.3scale.net",
    "HTTP_USER_AGENT": "curl/7.43.0",
    "HTTP_ACCEPT": "*/*",
    "HTTP_X_FORWARDED_FOR": "10.1.0.154",
    "HTTP_CONNECTION": "close"
  }
}
```

Once you've confirmed that your API is working, scroll down to the bottom of the **Staging** section where you can see a sample `curl` command:

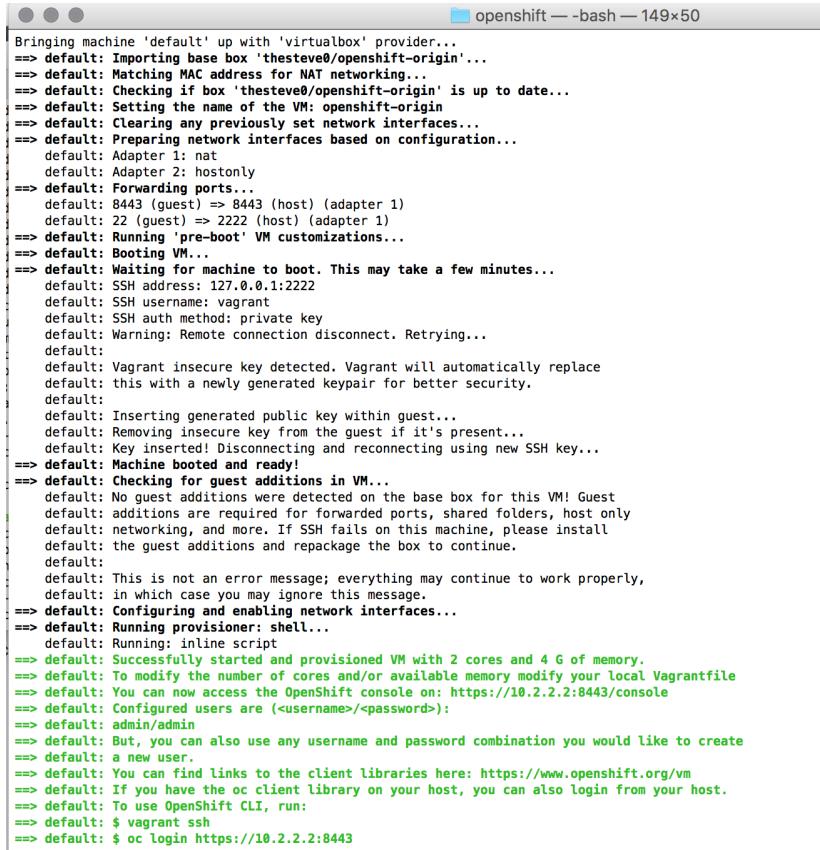
```
curl "https://XXXXXX.staging.apicast.io:443/?user_key=[MY_USER_KEY]"
```

Note down your `MY_USER_KEY` to use later to authenticate to your managed API

## Set up OpenShift

1. Download required tools and files
  - a. Go to <https://www.openshift.org/vm/> and click on the **Downloads** menu item
  - b. Download and install the required tools `vagrant` and `VirtualBox`
  - c. Download and install the OpenShift Client tools for your operating system and check they can be executed by typing `oc version` at a terminal prompt.
2. Start the OpenShift Origin VM
  - a. Start a command prompt, create a new directory and change into it.
  - b. At the terminal prompt type:

```
vagrant init thesteve0/openshift-origin
```



```
Bringing machine 'default' up with 'virtualbox' provider...
==> default: Importing base box 'theSteve0/openshift-origin'...
==> default: Matching MAC address for NAT networking...
==> default: Checking if box 'theSteve0/openshift-origin' is up to date...
==> default: Setting the name of the VM: openshift-origin
==> default: Clearing any previously set network interfaces...
==> default: Preparing network interfaces based on configuration...
    default: Adapter 1: nat
    default: Adapter 2: hostonly
==> default: Forwarding ports...
    default: 8443 (guest) => 8443 (host) (adapter 1)
    default: 22 (guest) => 2222 (host) (adapter 1)
==> default: Running 'pre-boot' VM customizations...
==> default: Booting VM...
==> default: Waiting for machine to boot. This may take a few minutes...
    default: SSH address: 127.0.0.1:2222
    default: SSH username: vagrant
    default: SSH auth method: private key
    default: Warning: Remote connection disconnect. Retrying...
    default:
    default: Vagrant insecure key detected. Vagrant will automatically replace
    default: this with a newly generated keypair for better security.
    default:
    default: Inserting generated public key within guest...
    default: Removing insecure key from the guest if it's present...
    default: Key inserted! Disconnecting and reconnecting using new SSH key...
==> default: Machine booted and ready!
==> default: Checking for guest additions in VM...
    default: No guest additions were detected on the base box for this VM! Guest
    default: additions are required for forwarded ports, shared folders, host only
    default: networking, and more. If SSH fails on this machine, please install
    default: the guest additions and repackage the box to continue.
    default:
    default: This is not an error message; everything may continue to work properly,
    default: in which case you may ignore this message.
==> default: Configuring and enabling network interfaces...
==> default: Running provisioner: shell...
    default: Running: inline script
==> default: Successfully started and provisioned VM with 2 cores and 4 G of memory.
==> default: To modify the number of cores and/or available memory modify your local Vagrantfile
==> default: You can now access the OpenShift console on: https://10.2.2.2:8443/console
==> default: Configured users are (<username>/<password>):
==> default: admin/admin
==> default: But, you can also use any username and password combination you would like to create
==> default: a new user.
==> default: You can find links to the client libraries here: https://www.openshift.org/vm
==> default: If you have the oc client library on your host, you can also login from your host.
==> default: To use OpenShift CLI, run:
==> default: $ vagrant ssh
==> default: $ oc login https://10.2.2.2:8443
```

That will create a file called `Vagrantfile` in the current directory.

- c. Start the VM using the command `vagrant up`. Downloading the image and starting the VM may take several minutes (it may take a bit longer if you're running Windows).

Upon starting the VM, you'll see output which includes instructions for logging in to your OpenShift Origin VM. :

```
oc login https://[OPENSHIFT-VM-IP]:8443
```

Your `OPENSHIFT-VM-IP` will be unique.

Note down your `OPENSHIFT-VM-IP` as you'll need to use it repeatedly in this tutorial.

3. Configure a host for the OpenShift Origin VM on your local machine. This will later allow you to expose the managed API using a hostname (this tutorial uses the example `gateway.openshift.demo`).

You'll need to add an entry for the Origin VM to your local hosts file.

If your host PC is linux-based, this file is `/etc/hosts`.

If you are running Windows PC, this hosts file will typically be: `C:\Windows\System32\drivers\etc\hosts`

Add a new line like this:

```
OPENSHIFT-VM-IP gateway.openshift.demo
```

Example:

```
10.2.2.2 gateway.openshift.demo
```

## Tutorial Steps

### Create your 3scale API Gateway using a template

1. Login into OpenShift using the `oc` command from the OpenShift Client tools you downloaded and installed in step 1. The default login credentials are **username: admin** and **password: admin**

```
oc login https://OPENSHIFT-VM-IP:8443
```

You may get a security warning and asked whether you wish to continue with an insecure selection. Choose **yes** to proceed.

The response should indicate **Login successful.**

2. Create your project. This example sets the display name as gateway

```
oc new-project "3scalegateway" --display-name="gateway" --description="3scale gateway demo"
```

The response should look like this: :

```
Now using project "3scalegateway" on server "https://[OPENSHIFT-VM-IP]:8443".
```

3. Create an application template for your 3scale API Gateway.

```
oc create -f https://raw.githubusercontent.com/3scale/docker-gateway/master/3scale-gateway-openshift-template.yml -n openshift
```

The response should look like this:

```
template "3scale-gateway" created
```

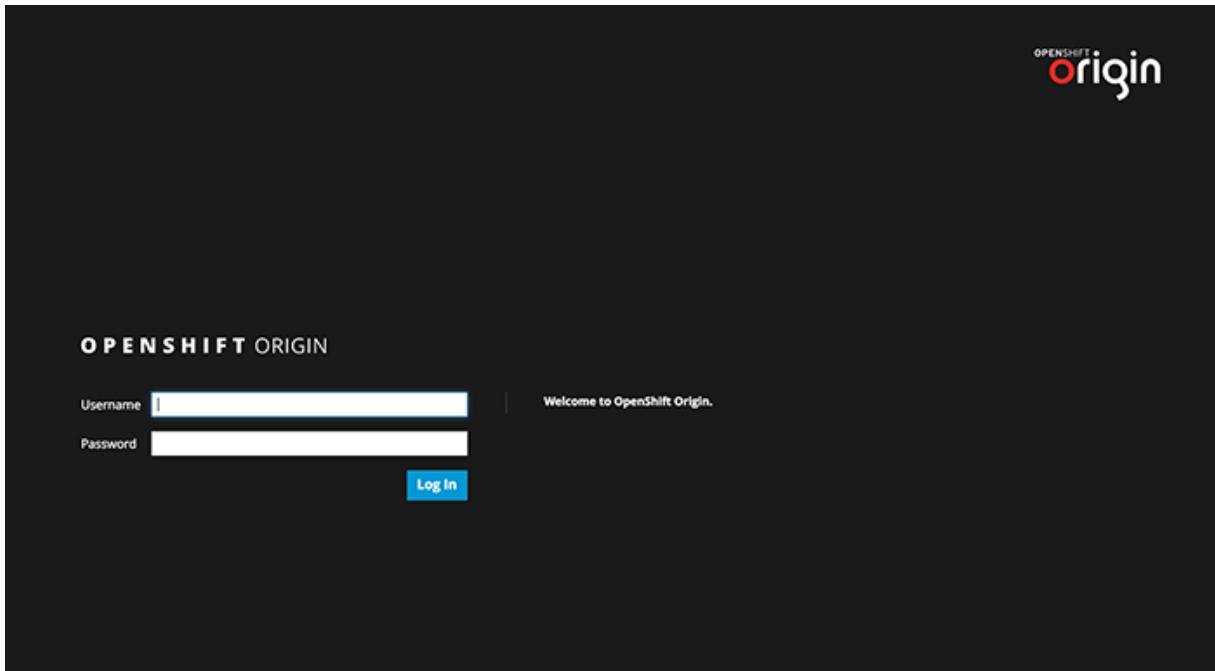
You may receive an error warning that the template already exists if you've previously attempted this step. You can continue, or login to the UI as described below and delete any projects from previous attempts and then return to this point.

## Deploying the 3scale API Gateway

1. Open the web console for your OpenShift Origin VM in your browser: [https://\[OPENSHIFT-VM-IP\]:8443/console/](https://[OPENSHIFT-VM-IP]:8443/console/)

You may receive a warning about an untrusted web-site. As this is the Origin VM running on your local machine there are no real risks. Select **Accept the Risks** in the browser and proceed to view the Web console.

You should see the login screen:



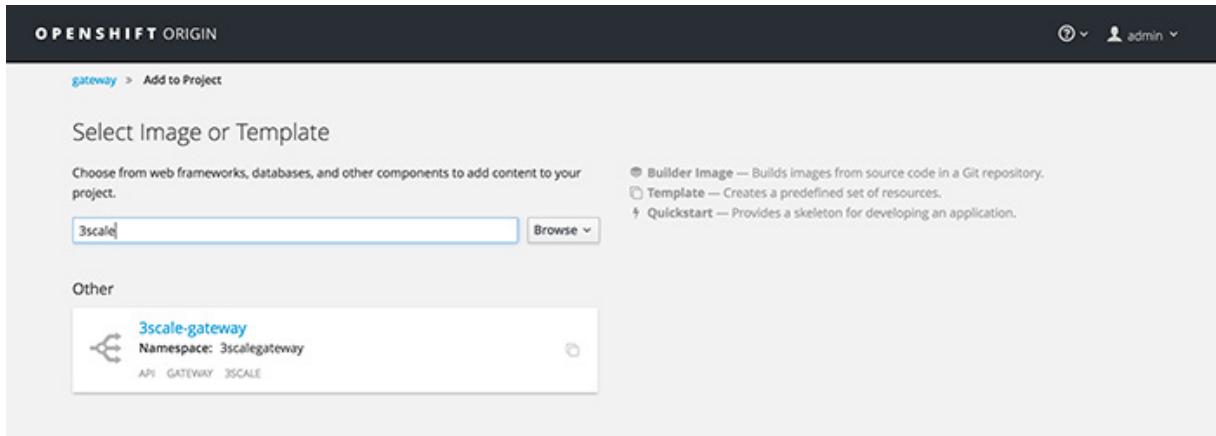
2. Login using the default credentials of **u: admin** and **p: admin**

Your list of projects will now include the gateway project you created from the command line above.

3. Select the gateway project and then click the **Add to Project** button.

The image shows the OpenShift Origin project list interface. The top navigation bar includes the 'OPENSHIFT ORIGIN' logo, a help icon, and a user dropdown set to 'admin'. The main title 'Projects' is centered above a list of projects. A 'New Project' button is located in the top right corner of the list area. The project list contains five entries: 'default', 'gateway', 'openshift', 'openshift-infra', and 'sample'. Each project entry is a card with the project name in blue, a description below it, and a trash icon in the top right corner. At the bottom of the list area, a note states: 'A project admin can add you as an admin to a project by running the command `oc policy add-role-to-user admin admin -n <projectname>`'.

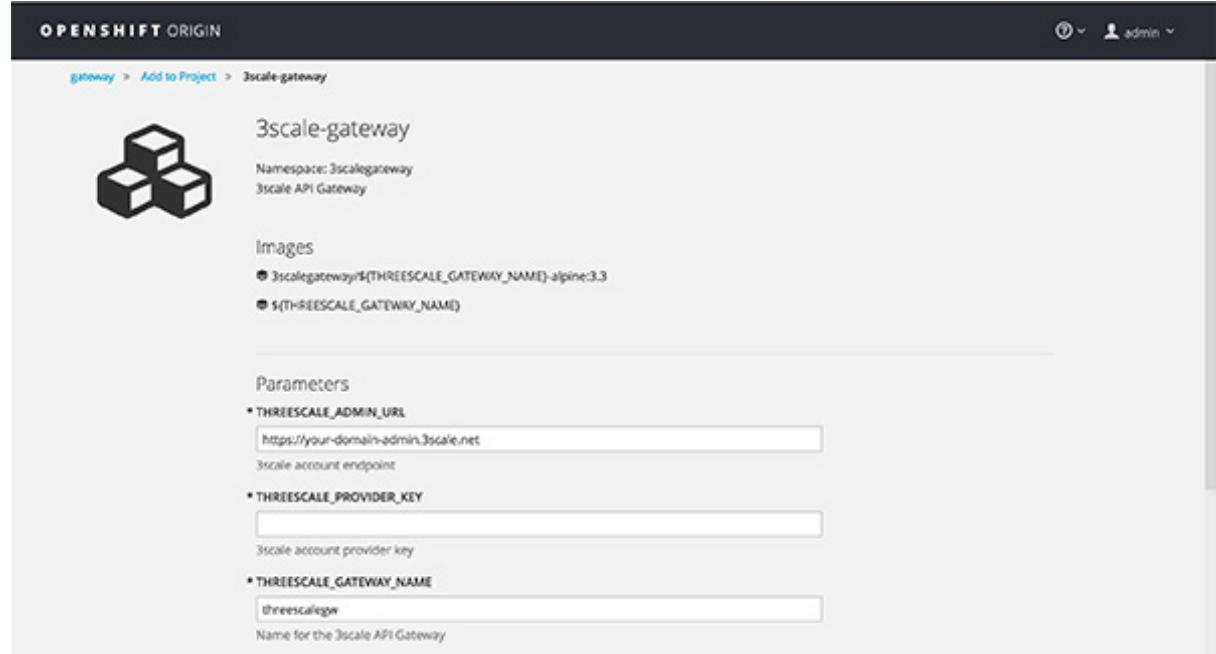
4. Search for 3scale and select the **3scale-gateway** template.
5. You'll be prompted to add your 3scale account details at this step. You'll need to include: **THREESCALE\_ADMIN\_URL** (including "https://") and **THREESCALE\_PROVIDER\_KEY**



The screenshot shows the 'Add to Project' screen in OpenShift Origin. The search bar contains '3scale' and the results list '3scale-gateway' under the 'Other' category. The '3scale-gateway' entry includes a icon, the name, the namespace '3scalegateway', and labels 'API', 'GATEWAY', and '3SCALE'.

When you've added this information, scroll to the bottom of the page and click **Create**

6. You will be shown the **Application** created screen:
7. Click on **gateway** (top left in the breadcrumbs) and you will be shown the **Overview** tab.



The screenshot shows the 'Overview' tab for the '3scale-gateway' application. It displays the application icon (three cubes), name, namespace ('3scalegateway'), and description ('3scale API Gateway'). The 'Images' section shows two available images: '3scalegateway/\${THREESCALE\_GATEWAY\_NAME}-alpine:3.3' and '5(\${THREESCALE\_GATEWAY\_NAME})'. The 'Parameters' section contains three required parameters: **THREESCALE\_ADMIN\_URL** (set to 'https://your-domain-admin.3scale.net'), **THREESCALE\_PROVIDER\_KEY** (empty), and **THREESCALE\_GATEWAY\_NAME** (set to 'threescalegw').

Application created. [Continue to overview](#).

Manage your app

The web console is convenient, but if you need deeper control you may want to try our command line tools.

Command line tools

Download and install the `oc` command line tool. After that, you can start by logging in, switching to this particular project, and displaying an overview of it, by doing:

```
oc login https://10.2.2.2:8443
oc project 3scalegateway
oc status
```

For more information about the command line tools, check the [CLI Reference](#) and [Basic CLI Operations](#).

OpenShift has now downloaded the code for the API gateway and has started a build to construct it.

8. When the build completes, the UI will refresh and show two instances of the API gateway (2 "pods") that have been started by OpenShift, as defined in the template.

Overview

Filter by label Add

SERVICE threescalegw 8080/TCP → 8080 [Create Route](#)

Build threescalegw #1 is running. A new deployment will be created automatically once the build completes. [View Log](#)

There are no pods or deployments for this service.

Each instance of the 3scale API Gateway, upon starting, downloads the required configuration files and code from 3scale using the configuration settings you provided on the **Integration** tab of your 3scale Admin Portal.

OpenShift will maintain two gateway instances and monitor the health of both; an unhealthy API gateway will automatically be replaced with a new one.

9. In order to allow your API gateways to receive traffic, you'll need to create a route. Start by clicking on **Create route**.

Project gateway

Overview

SERVICE threescalegw

DEPLOYMENT: THRESCALEGW, #1

CONTAINER: DOCKER-GATEWAY

Image: 3scalegateway/threescalegw (9a305f6) 2.6 MiB

Build: threescalegw, #1

Source: Added initial support for Openshift v3 (fad320d)

Ports: 8080/TCP

8080/TCP → 8080 Create Route

2 pods

8 minutes ago Image change

Details

Select an object to see more details.

A pod contains one or more Docker containers that run together on a node, containing your application code.

A service groups pods and provides a common DNS name and an optional, load-balanced IP address to access them.

A deployment is an update to your application, triggered by a changed image or configuration.

In the **Configure a host for the Origin Virtual Machine** section, enter the hostname you set in your `/etc/hosts` file in the Hostname field (e.g. `gateway.openshift.demo`) and then click the **Create** button.

OPENSHIFT ORIGIN

3scalegateway > Create Route

Create Route

Routing is a way to make your application publicly visible. Create a route to expose service threescalegw.

\* Name: threescalegw

A unique name for the route within the project.

Hostname: gateway.openshift.demo

Public hostname for the route. If not specified, a hostname is generated.

Path: /

Path that the router watches to route traffic to the service.

Target Port: 8080 → 8080 (TCP)

Target port for traffic.

Show options for secured routes

Create Cancel

10. Your API gateways are now ready to receive traffic. OpenShift takes care of load-balancing incoming requests to the route across the two running instances of the API gateway.

Test that the API gateway authorizes a valid call to your API, by executing a `curl` command with the valid `MY_USER_KEY` that you noted down earlier:

```
curl "http://gateway.openshift.demo/?user_key= MY_USER_KEY"
```

**11.** Test it does not authorize an invalid call to your API

```
curl "http://gateway.openshift.demo/?user_key=[bad key value]"
```

You can access see the logs of API gateway traffic by clicking **Browse > Pods** and then select one of the pods and then selecting **Logs**.

## Success!!

Your API is now protected by two instances of the 3scale running on RedHat OpenShift, following all the configuration that you set-up in the 3scale Admin Portal.

If you wish to shut down the OpenShift Origin VM to save resources, you can do so by opening VirtualBox and right-clicking on the **origin vm** then selecting **Close > ACPI Shutdown**. You can then close VirtualBox.

## Next Steps

Now that you have an API Gateway up and running on your local machine you can:

- Explore how to configure access policies for your API, and engage developers with a Developer Portal by following the [Quickstart](#)
- Whenever you make changes to your API definition in the 3scale Admin Portal — in particular the 3scale metrics/methods and mapping rules — you should create a new deployment in OpenShift. This will start new instances that will download and run your new API definition. Then OpenShift will shut down gracefully the previous instances.
- Run Openshift Enterprise V3 on your dedicated datacenter or on your favorite cloud platform and then follow the same instructions to open up your API to the world.