



Red Hat 3scale 2-saas

Infrastructure

For Use with Red Hat 3scale 2-saas

Red Hat 3scale 2-saas Infrastructure

For Use with Red Hat 3scale 2-saas

Legal Notice

Copyright © 2018 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux ® is the registered trademark of Linus Torvalds in the United States and other countries.

Java ® is a registered trademark of Oracle and/or its affiliates.

XFS ® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL ® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js ® is an official trademark of Joyent. Red Hat Software Collections is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack ® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide documents deployment and infrastructure management with Red Hat 3scale 2-saas.

Table of Contents

CHAPTER 1. HOW TO DEPLOY A FULL-STACK API SOLUTION WITH FUSE, 3SCALE, AND OPENSIFT	3
1.1. PART 1: FUSE ON OPENSIFT SETUP	4
1.1.1. Step 1	4
1.1.2. Step 2	5
1.1.3. Step 3	5
1.1.4. Step 4	6
1.1.5. Step 5	7
1.1.6. Step 6	8
1.1.7. Step 7	8
1.1.8. Step 8	9
1.1.9. Step 9	9
1.1.10. Step 10	10
1.1.11. Step 11	11
1.2. PART 2: CONFIGURE 3SCALE API MANAGEMENT	11
1.2.1. Step 1	11
1.2.2. Step 2	11
1.2.3. Step 3	12
1.2.4. Step 4	12
1.2.5. Step 5	13
1.3. PART 3: INTEGRATION OF YOUR API SERVICES	14
1.4. PART 4: TESTING THE API AND API MANAGEMENT	14
1.4.1. Step 1	14
1.4.2. Step 2	15
1.4.3. Step 3	15
1.4.4. Step 4	16
1.4.5. Step 5	16

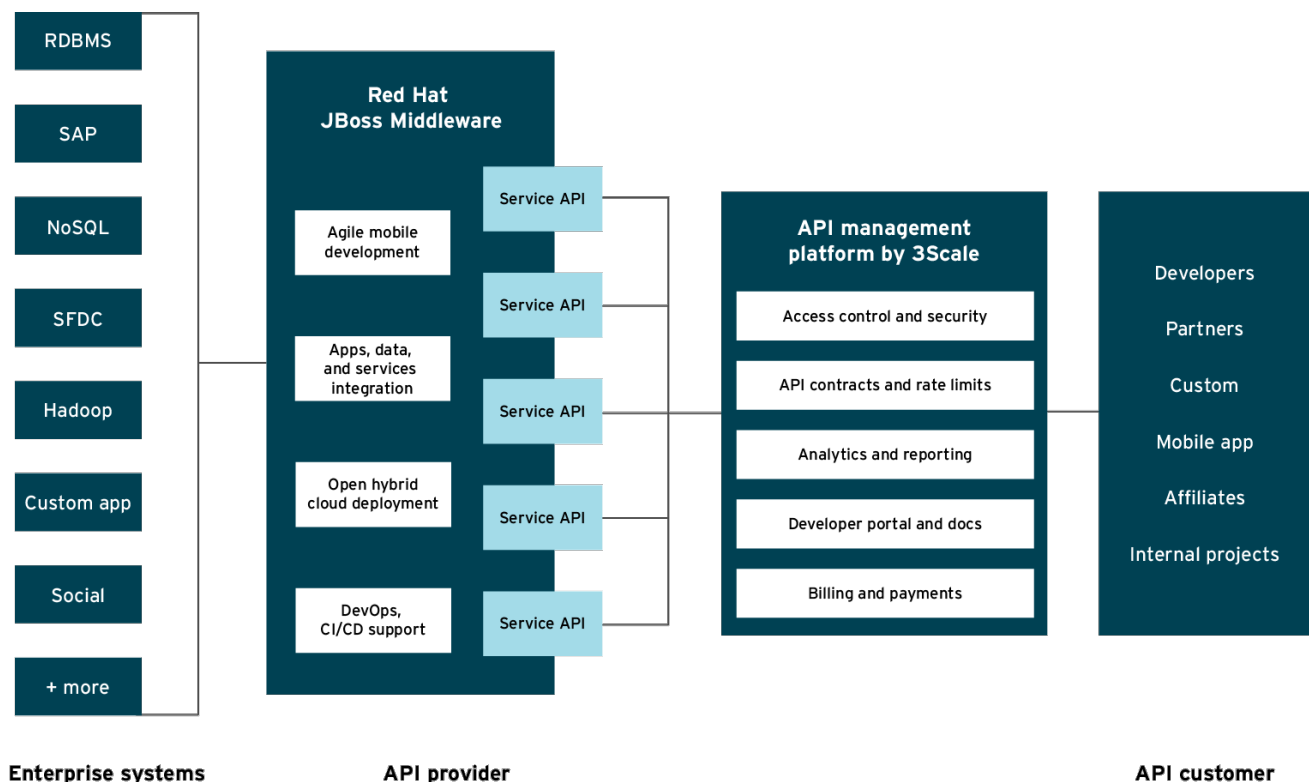
CHAPTER 1. HOW TO DEPLOY A FULL-STACK API SOLUTION WITH FUSE, 3SCALE, AND OPENSIFT

This tutorial describes how to get a full-stack API solution (API design, development, hosting, access control, monetization, etc.) using Red Hat JBoss xPaaS for OpenShift and 3scale API Management Platform - Cloud.

The tutorial is based on a collaboration between Red Hat and 3scale to provide a [full-stack API solution](#). This solution includes design, development, and hosting of your API on the [Red Hat JBoss xPaaS for OpenShift](#), combined with the 3scale API Management Platform for full control, visibility, and monetization features.

The API itself can be deployed on Red Hat JBoss xPaaS for OpenShift, which can be hosted in the cloud as well as on premise (that's the Red Hat part). The API management (the 3scale part) can be hosted on Amazon Web Services (AWS), using 3scale [APIcast](#) or OpenShift. This gives a wide range of different configuration options for maximum deployment flexibility.

The diagram below summarizes the main elements of this joint solution. It shows the whole integration chain including enterprise backend systems, middleware, API management, and API customers.



JB0095

For specific support questions, please [contact support](#).

This tutorial shows three different deployment scenarios step by step:

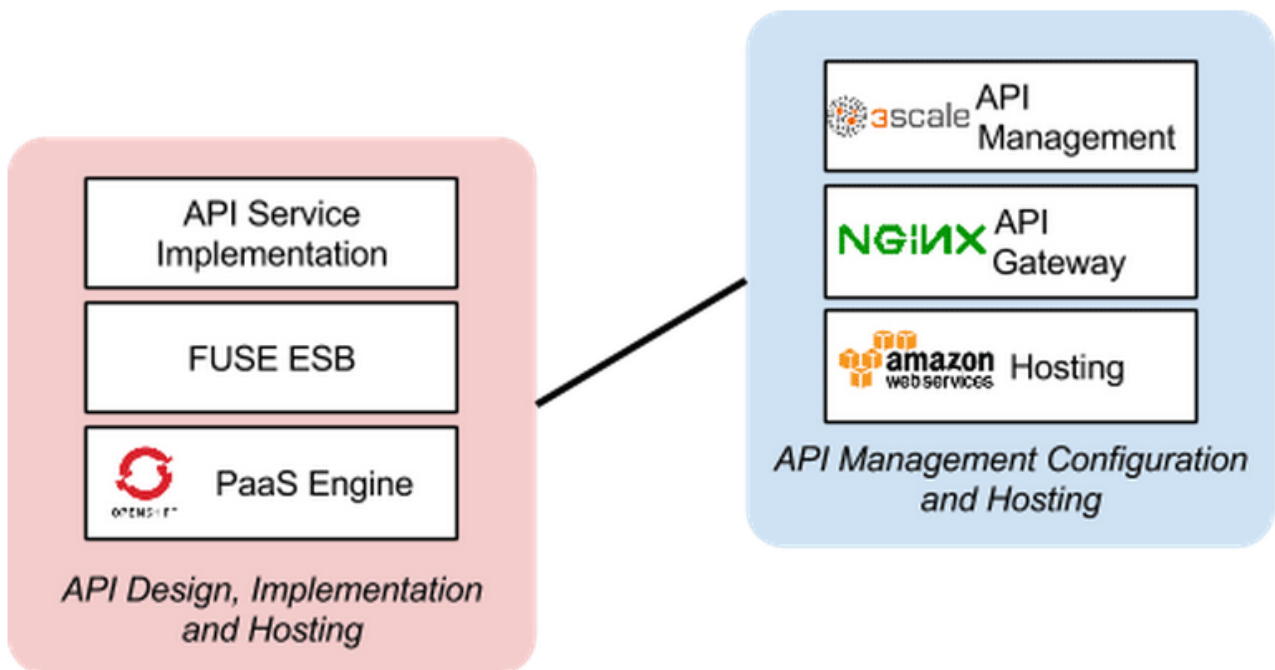
1. Scenario 1 – A [Fuse on OpenShift](#) application containing the API. The API is managed by 3scale with the API gateway hosted on Amazon Web Services (AWS) using the [3scale API](#).
2. Scenario 2 – A Fuse on OpenShift application containing the API. The API is managed by 3scale with the API gateway hosted on [APIcast](#) (3scale's cloud hosted API gateway).

- Scenario 3 – A Fuse on OpenShift application containing the API. The API is managed by 3scale with the API gateway hosted on [OpenShift](#)

This tutorial is split into four parts:

- [Part 1: Fuse on OpenShift](#) setup to design and implement the API
- [Part 2](#): Configuration of 3scale API Management
- [Part 3](#): Integration of your API services
- [Part 4](#): Testing the API and API management

The diagram below shows the roles the various parts play in this configuration.

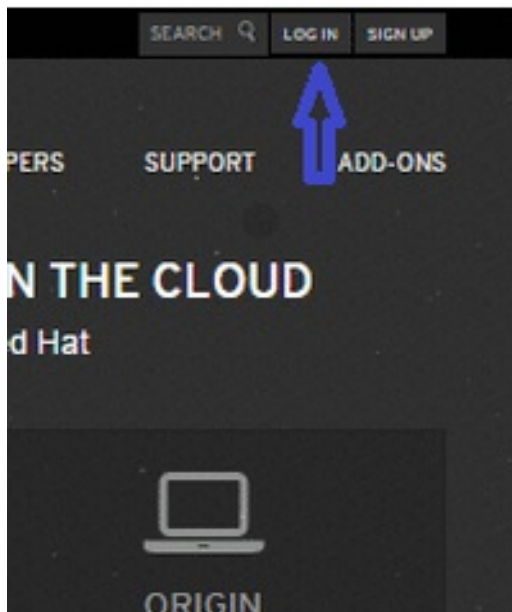


1.1. PART 1: FUSE ON OPENSIFT SETUP

You will create a [Fuse on OpenShift](#) application that contains the API to be managed. You will use the REST quickstart that is included with Fuse 6.1. This requires a medium or large gear, as using the small gear will result in memory errors and/or horrible performance.

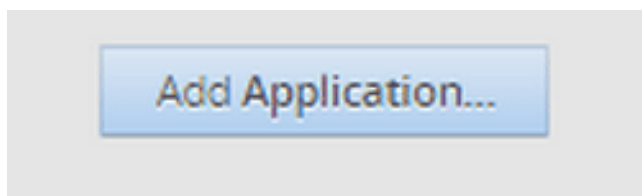
1.1.1. Step 1

Sign in to your OpenShift online account. Sign up for an OpenShift online account if you don't already have one.



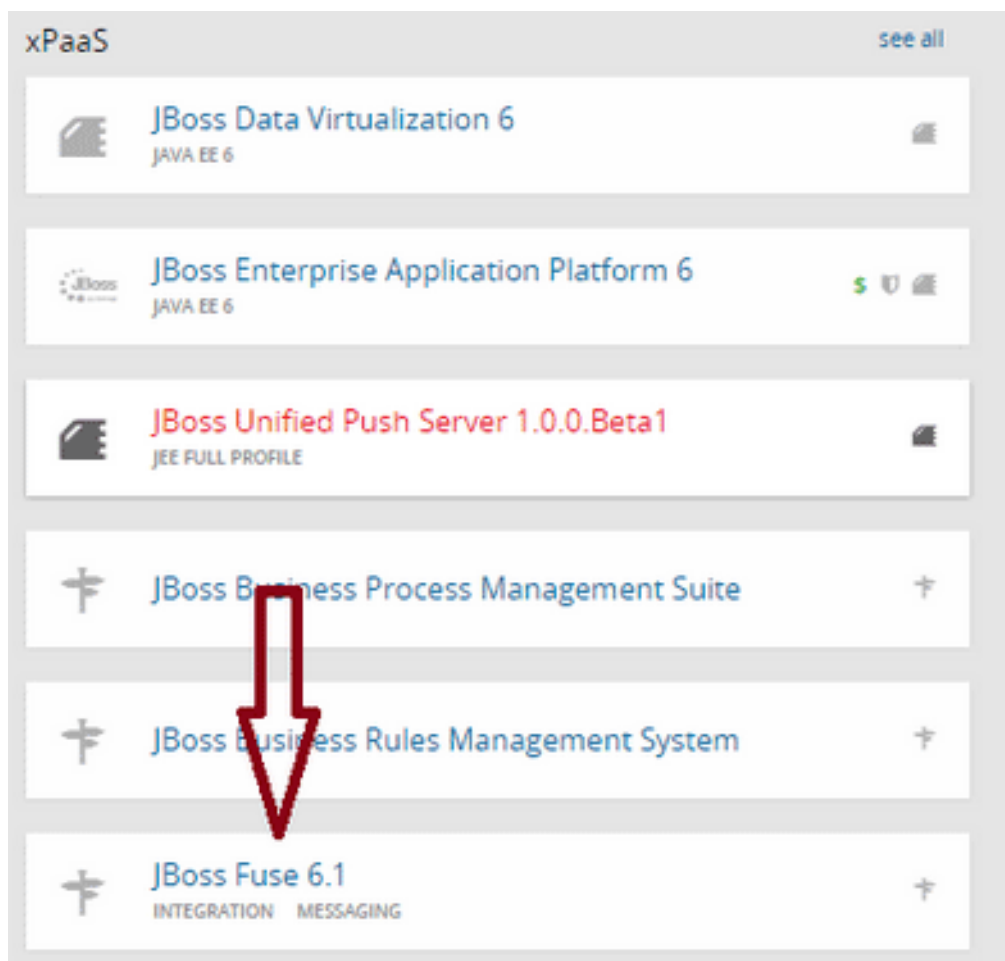
1.1.2. Step 2

Click the "add application" button after signing in.



1.1.3. Step 3

Under xPaaS, select the Fuse type for the application.



1.1.4. Step 4

Now configure the application. Enter the subdomain you'd like your application to show up under, such as "restapitest". This will give a full URL of the form "appname-domain.rhcloud.com" – in the example below "restapitest-ossmentor.rhcloud.com". Change the gear size to medium or large, which is required for the Fuse cartridge. Now click on "create application".

Applications
Settings
Support
Add-ons

1 Choose a type of application
2 **Configure the application**
3 Next steps

Based On
JBoss Fuse 6.1 Quickstart

The JBoss Fuse enterprise service bus is a technology for building and implementing communication between different applications, services and data. It's specifically designed for extensive connectivity. This cartridge is an alpha release of JBoss Fuse 6.1 for OpenShift.

Note: It is recommended that you use a medium sized gear to deploy JBoss Fuse due to memory requirements. Running in a small gear may result in slow interface responsiveness.

[Learn more](#)

☆ OpenShift maintained

Does not receive automatic security updates

Public URL

http://restapitest-ossmentor.rhcloud.com

OpenShift will automatically register this domain name for your application. You can add your own domain name later.

Source Code

Optional URL to a Git repository
Branch/tag

We'll create a Git code repository in the cloud, and populate it with a set of reasonable defaults. If you provide a Git URL, your application will start with an exact copy of the code and configuration provided in this Git repository.

Gears

medium

Gears are the application containers running your code. For most applications, the small gear size provides plenty of resources. If you require more resources, select a different gear size here. You can also [upgrade your plan](#) to get access to more gear sizes.

Cartridges

manifest.yml

Applications are composed of cartridges - each of which exposes a service or capability to your code. All applications must have a web cartridge.

Downloaded cartridges do not receive updates automatically.

Scaling

No scaling

OpenShift automatically routes web requests to your web gear. If you allow your application to scale, we'll set up a load balancer and allocate more gears to handle traffic as you need it.

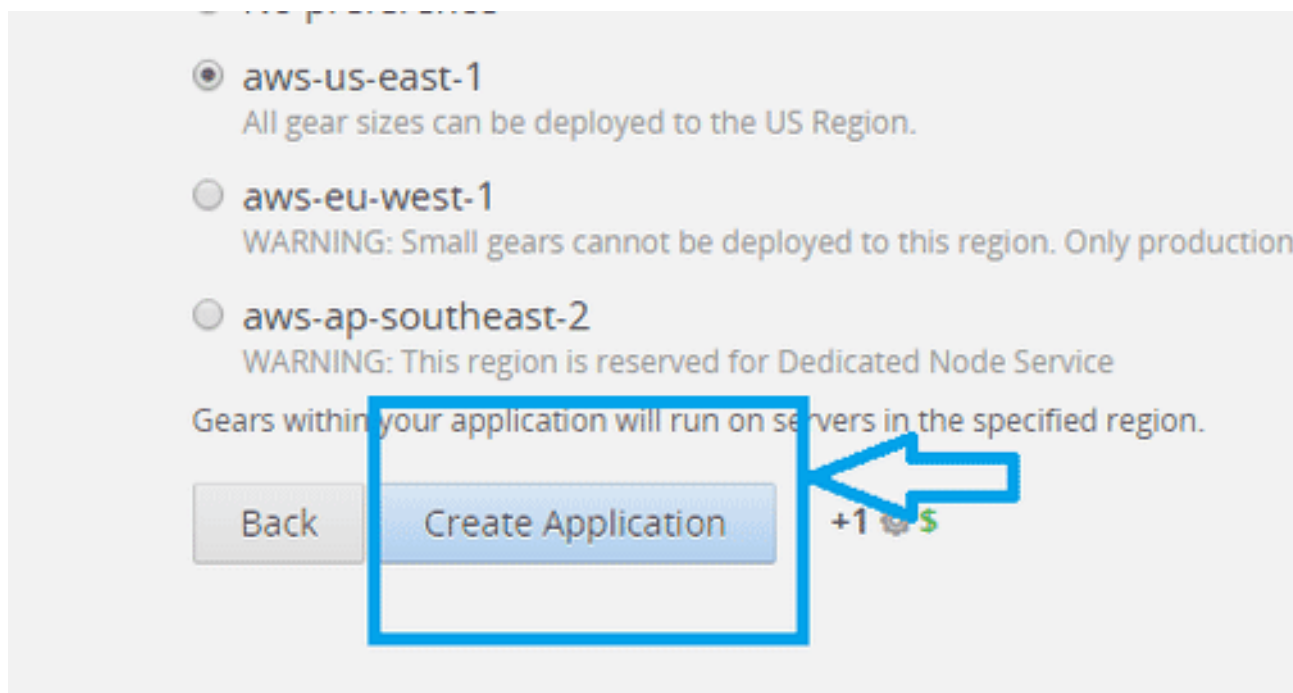
Region

☐ No preference
☒ **aws-us-east-1**
All gear sizes can be deployed to the US Region.
☐ aws-eu-west-1
WARNING: Small gears cannot be deployed to this region. Only production gears can be deployed to the EU Region (small,highcpu, medium, and large).
☐ aws-ap-southeast-2
WARNING: This region is reserved for Dedicated Node Service
Gears within your application will run on servers in the specified region.

Back
Create Application
+1 \$

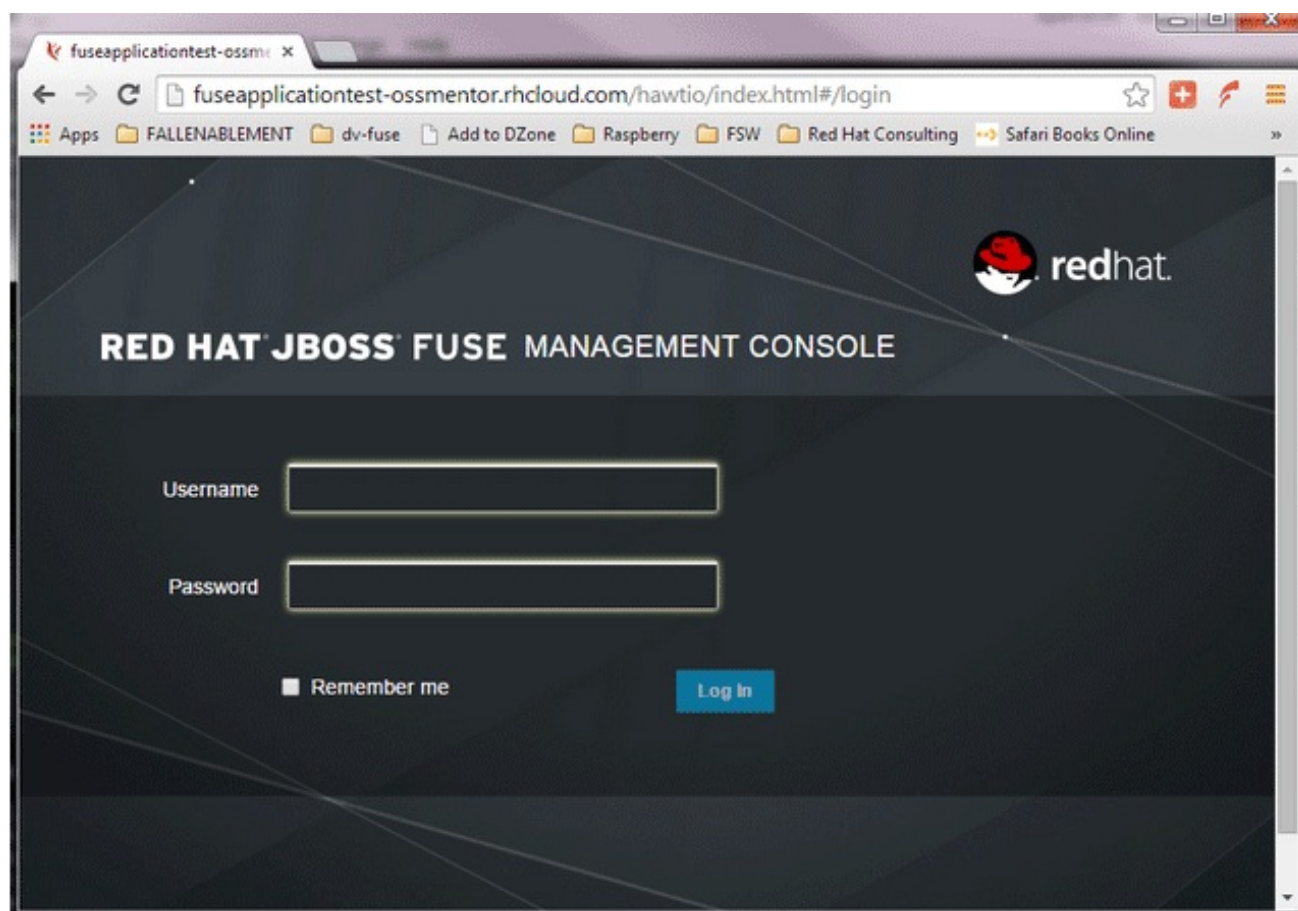
1.1.5. Step 5

Click "create application".



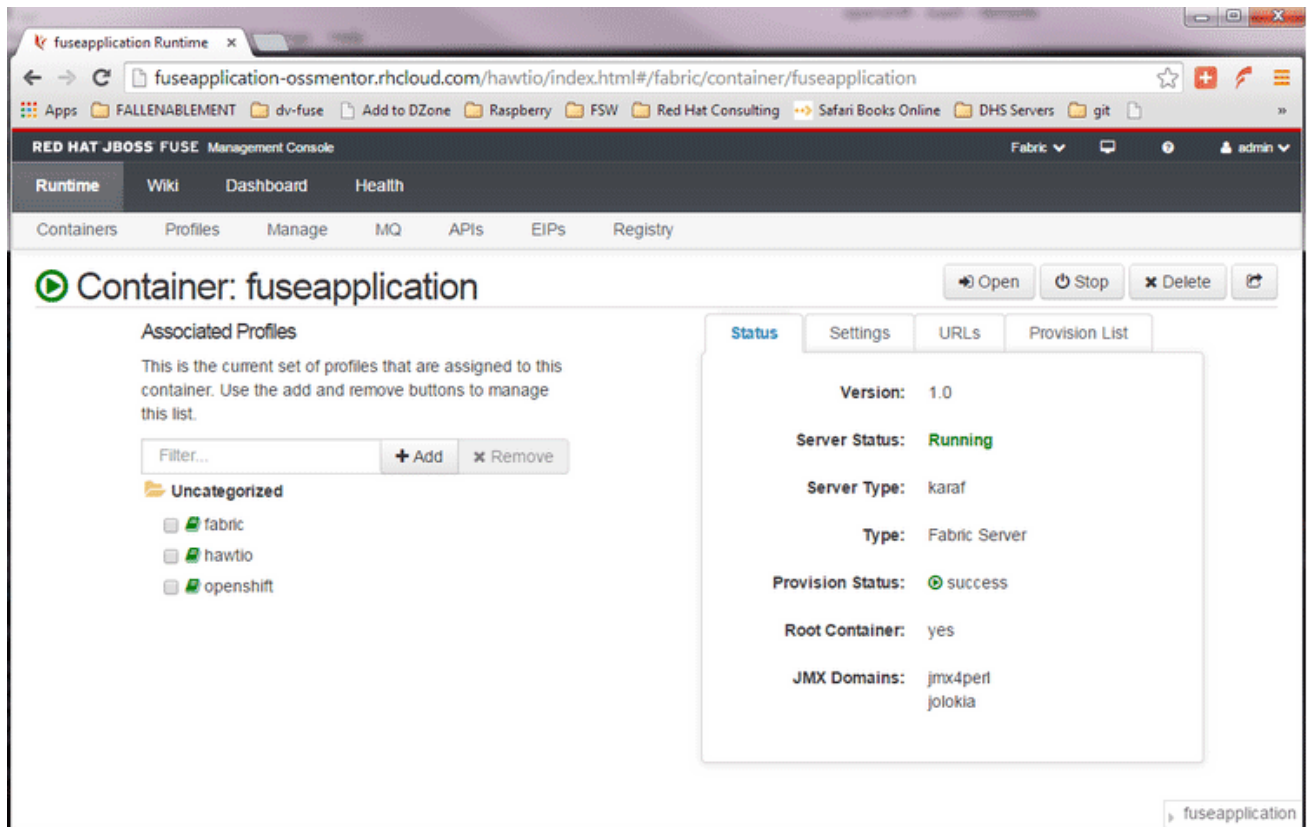
1.1.6. Step 6

Browse the application hawtio console and sign in.



1.1.7. Step 7

After signing in, click on the "runtime" tab and the container, and add the REST API example.

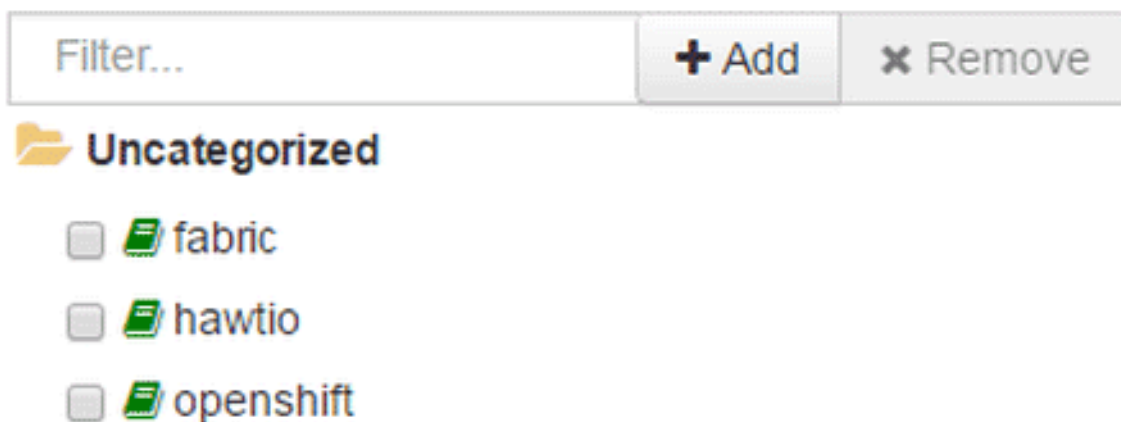


1.1.8. Step 8

Click on the "add a profile" button.

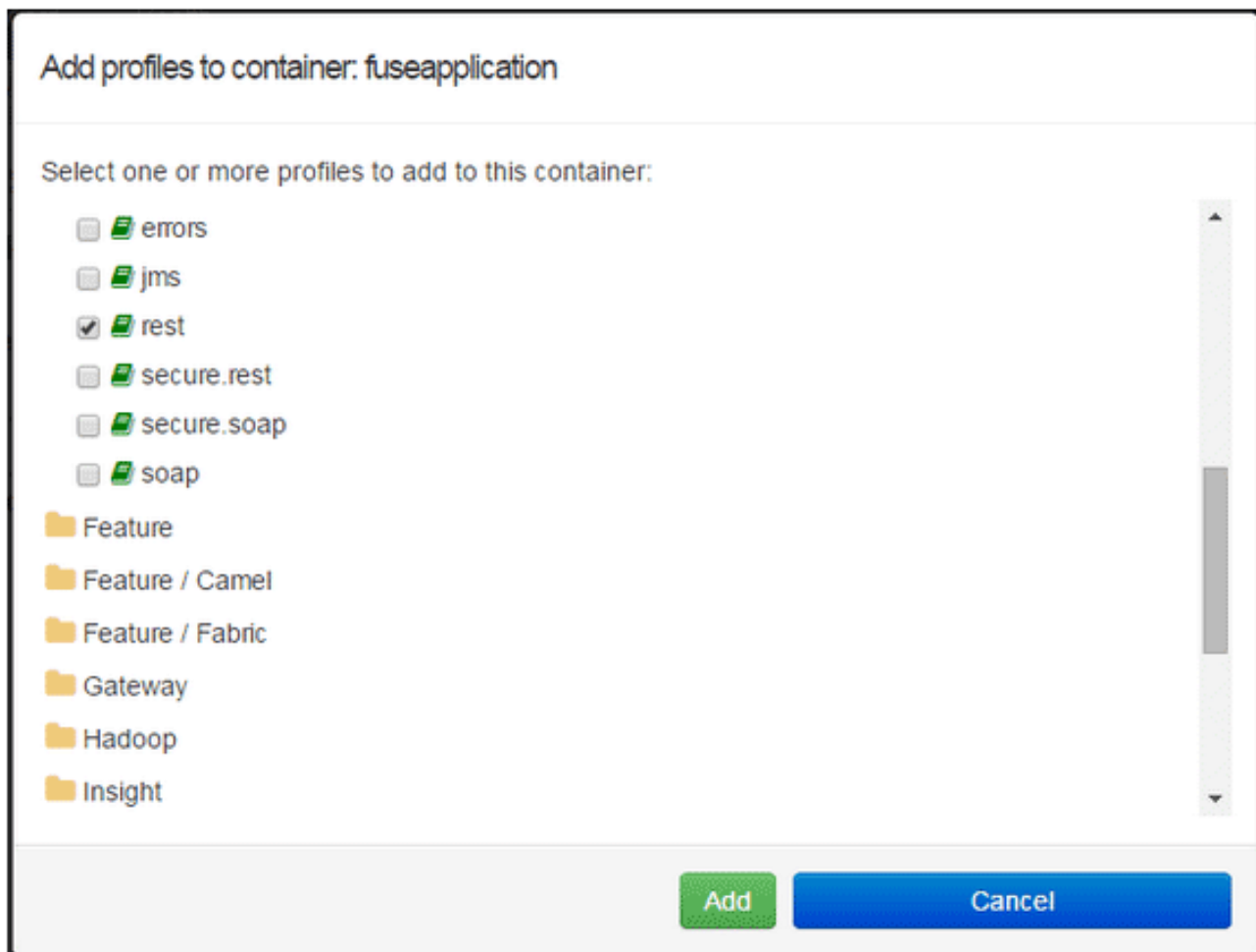
Associated Profiles

This is the current set of profiles that are assigned to this container. Use the add and remove buttons to manage this list.



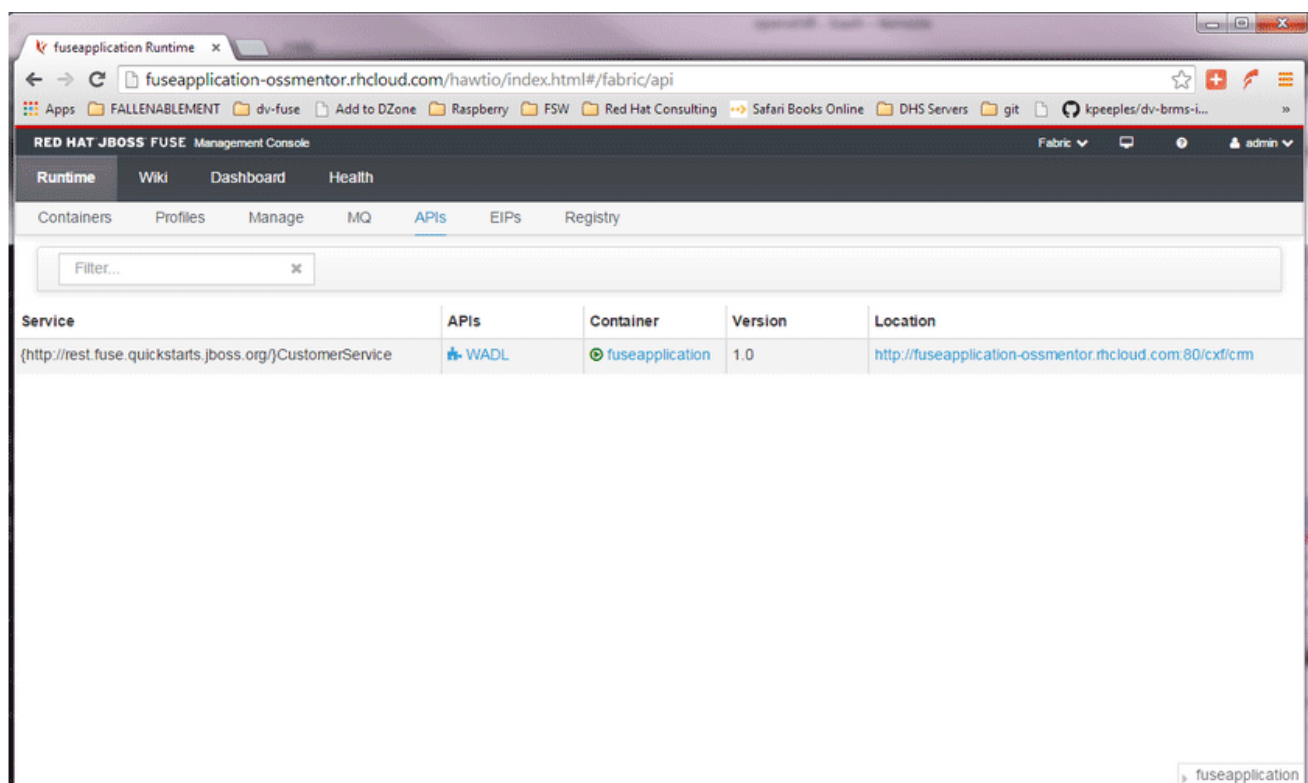
1.1.9. Step 9

Scroll down to examples/quickstarts and click the "REST" checkbox, then "add". The REST profile should show up on the container associated profile page.



1.1.10. Step 10

Click on the runtime/APIs tab to verify the REST API profile.



1.1.11. Step 11

Verify the REST API is working. Browse to customer 123, which will return the ID and name in XML format.



1.2. PART 2: CONFIGURE 3SCALE API MANAGEMENT

To protect the API that you just created in [Part 1](#) using 3scale API Management, you first must conduct the according configuration, which is then later deployed according to one of the three scenarios presented.


Once you have your API set up on OpenShift, you can start setting it up on 3scale to provide the management layer for access control and usage monitoring.

1.2.1. Step 1

Log in to your 3scale account. You can sign up for a 3scale account at www.3scale.net if you don't already have one. When you log in to your account for the first time, follow the wizard to learn the basics about integrating your API with 3scale.

1.2.2. Step 2

In **[your_API_name] > Integration > Configuration**, you can enter the public URL for the Fuse application on OpenShift that you just created, e.g. "restapitest-ossmentor.rhcloud.com" and click on **Test**. This will test your setup against the 3scale API Gateway in the staging environment. The staging API gateway allows you to test your 3scale setup before deploying your proxy configuration to AWS.

Staging: 3scale-hosted to configure & test your integration [documentation](#)[deployed](#) | [deployment history](#)


API ?

Private Base URL* [Use Echo API](#)

Private address of your API that will be called by the API gateway.

API GATEWAY ?

Public Base URL*

Public address of your API gateway in the staging environment. You can use this address to call the API for testing purposes.

▶ **MAPPING RULES**

▶ **AUTHENTICATION SETTINGS**

CLIENT ?

API test GET request

Optional GET request to a API gateway endpoint. We will use this call to validate your API gateway setup using credentials of the first live application. You can try it yourself by copying the following command into your shell:

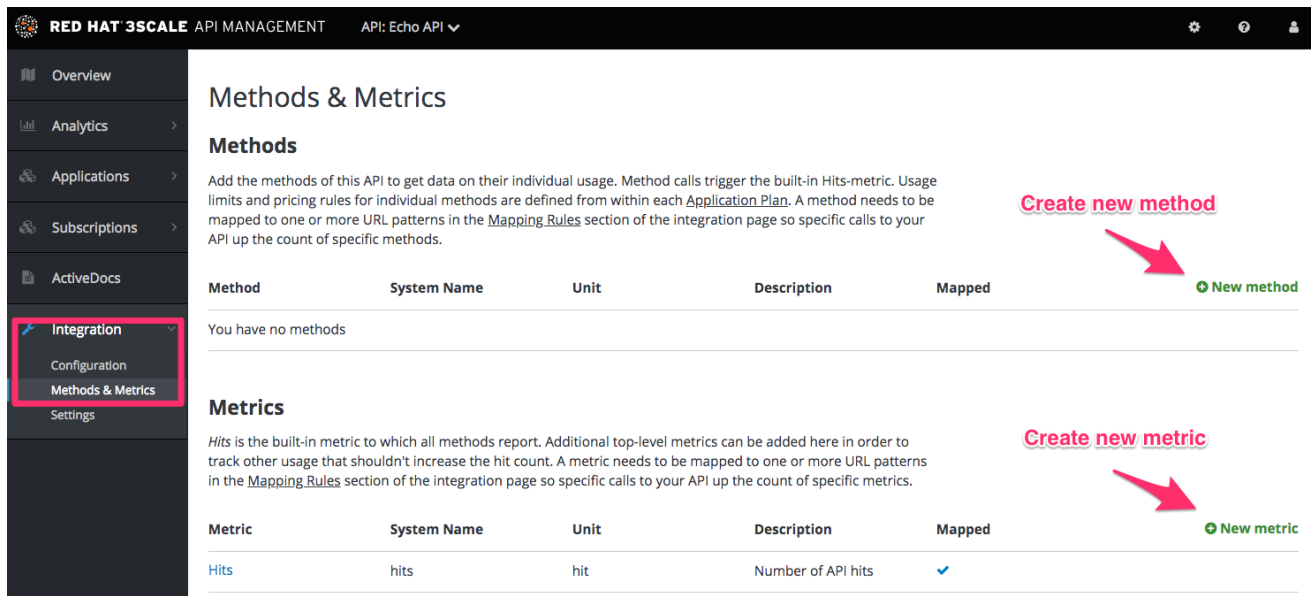
```
curl "https://api-2445581450779.staging.apicast.io:443/v1/word/good.json?user_key=44e72dedd214c812990c1b3ab12f5ba3"
```

Connection between client, gateway & API is working correctly as [reflected in the analytics section.](#)

[Update & Test Staging Configuration](#)

1.2.3. Step 3

The next step is to set up the API methods that you want to monitor and rate limit. To do that go to **[your_API_name] > Integration > Methods & Metrics** and click on 'New method'.



RED HAT 3SCALE API MANAGEMENT API: Echo API

Methods & Metrics

Methods

Add the methods of this API to get data on their individual usage. Method calls trigger the built-in Hits-metric. Usage limits and pricing rules for individual methods are defined from within each [Application Plan](#). A method needs to be mapped to one or more URL patterns in the [Mapping Rules](#) section of the integration page so specific calls to your API up the count of specific methods.

Method	System Name	Unit	Description	Mapped
You have no methods				

[Create new method](#)

[New method](#)

Metrics

Hits is the built-in metric to which all methods report. Additional top-level metrics can be added here in order to track other usage that shouldn't increase the hit count. A metric needs to be mapped to one or more URL patterns in the [Mapping Rules](#) section of the integration page so specific calls to your API up the count of specific metrics.

Metric	System Name	Unit	Description	Mapped
Hits	hits	hit	Number of API hits	✓

[Create new metric](#)

[New metric](#)

For more details on creating methods, visit our [API definition tutorial](#).

1.2.4. Step 4

Once you have all of the methods that you want to monitor and control set up under the application plan, you'll need to map these to actual HTTP methods on endpoints of your API. Go back to the integration page and expand the "mapping rules" section.

▼ MAPPING RULES ?

Verb	Pattern		+	Metric or Method (Define)
GET	/	1		hits

[Add Mapping Rule](#)

Create mapping rules for each of the methods you created under the application plan.

Rule	Pattern	+/-	Create Proxy Rule
POST	/setAB	1	<div>✓ hits</div> <div>getHelloMethodSystemName</div>

Once you have done that, your mapping rules will look something like this:

▼ MAPPING RULES ?

Verb	Pattern		+	Metric or Method (Define)
GET	/v1/words/{word}.json	1		get_word
GET	/v1/sentences/{sentence}.json	1		get_sente
POST	/v1/words/{word}.json	1		set_word

[Add Mapping Rule](#)

For more details on mapping rules, visit our [tutorial about mapping rules](#).

1.2.5. Step 5

Once you've clicked "update and test" to save and test your configuration, you are ready to download the set of configuration files that will allow you to configure your API gateway on AWS. For the API gateway, you should use a high-performance, open-source proxy called [nginx](#). You will find the necessary configuration files for nginx on the same integration page by scrolling down to the "production" section.

Production: On-premises Gateway

To deploy an on-premises API gateway, add the Public Base URL of your API, download the Nginx Config files and [follow the documentation](#) to install in your servers.



API


Private Base URL



API GATEWAY

Public Base URL

Public address of your API gateway in the production environment. This is used to customize the server_name directive in the Nginx Config file which will otherwise be set to the variable \$hostname.

[Update Production Configuration](#) [Download the Nginx Config files](#)

The next section will now take you through various hosting scenarios.

1.3. PART 3: INTEGRATION OF YOUR API SERVICES

There are several ways in which you can integrate your API services in 3scale. Choose the [deployment option](#) that best fits your needs.

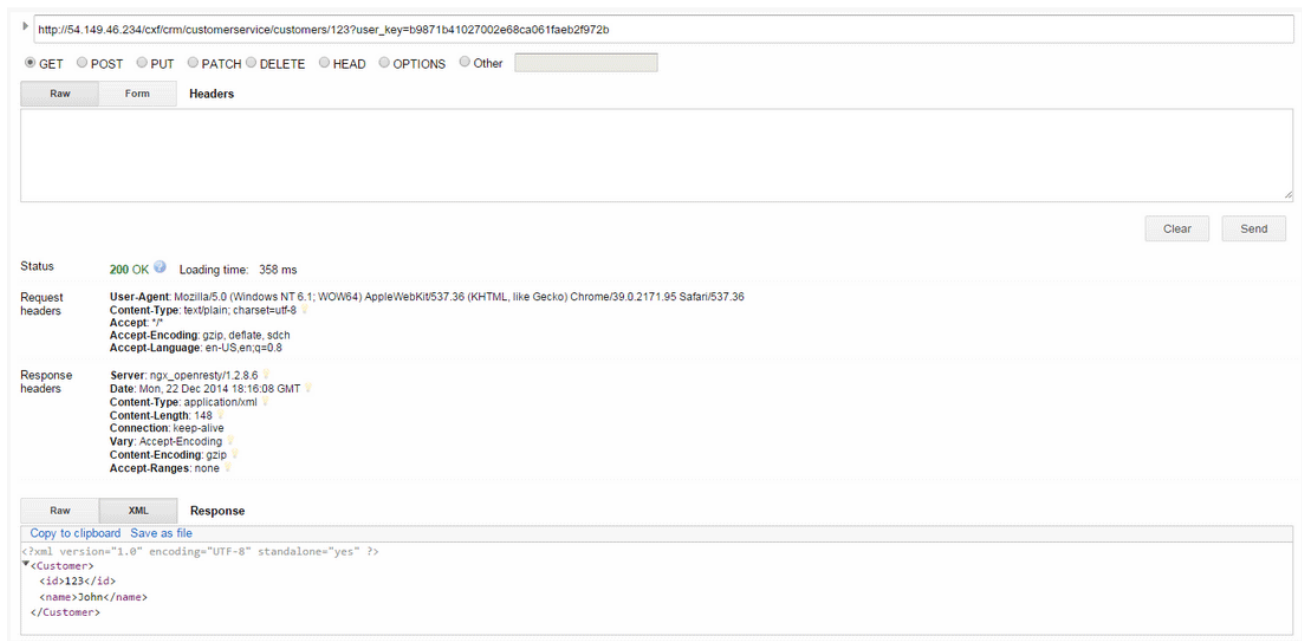
1.4. PART 4: TESTING THE API AND API MANAGEMENT

Testing the correct functioning of the API and the API Management is independent from the chosen scenario. You can use your favorite REST client and run the following commands.

1.4.1. Step 1

Retrieve the customer instance with id 123.

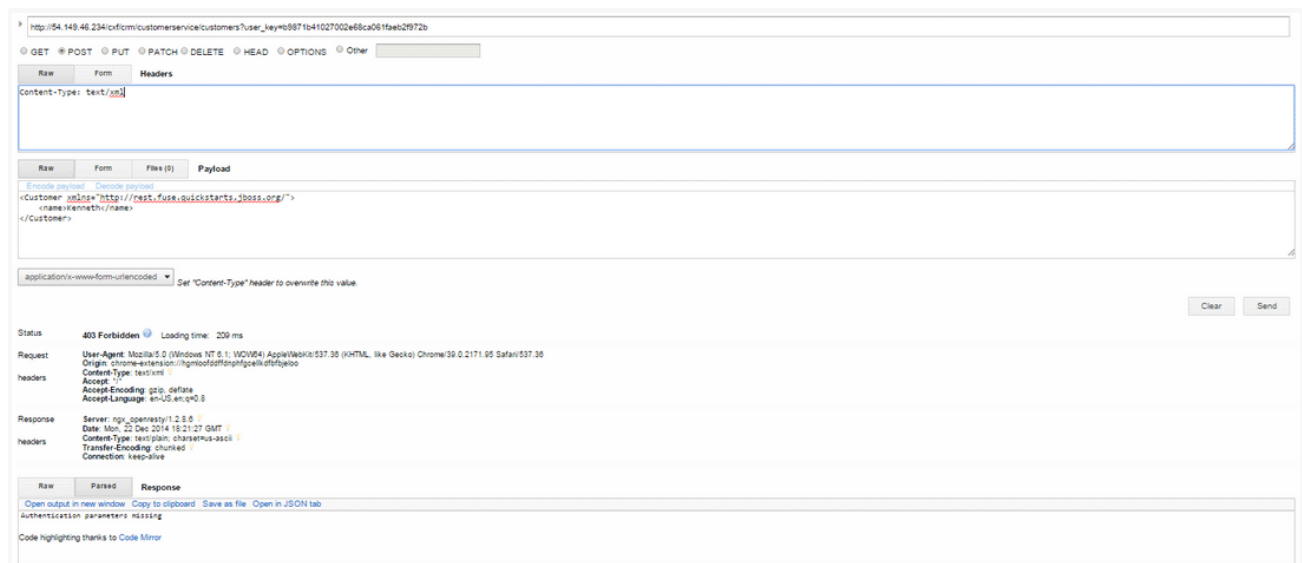
```
http://54.149.46.234/cxf/crm/customerservice/customers/123?  
user_key=b9871b41027002e68ca061faeb2f972b
```



1.4.2. Step 2

Create a customer.

```
http://54.149.46.234/cxf/crm/customerservice/customers?
user_key=b9871b41027002e68ca061faeb2f972b
```



1.4.3. Step 3

Update the customer instance with id 123.

```
http://54.149.46.234/cxf/crm/customerservice/customers?
user_key=b9871b41027002e68ca061faeb2f972b
```

http://54.149.46.234/cxf/crm/customerservice/customers?user_key=b9871b41027002e68ca061faeb2f972b

GET POST PUT PATCH DELETE HEAD OPTIONS Other

Raw Form Headers

Content-Type: text/xml

Raw Form Files (0) Payload

Encode payload Decode payload

```
<Customer xmlns="http://rest.fuse.quickstarts.jboss.org/">
  <name/ary:/name>
    <id>123/</id>
  </Customer>
```

application/x-www-form-urlencoded Set "Content-Type" header to overwrite this value

Clear Send

Status: 403 Forbidden Loading time: 200 ms

Request: User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36
Origin: chrome-extension://hgmpoofdfphtgpeklctftrfjeko
Content-Type: text/xml
headers: Accept: */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8

Response: Server: ngi_nginx/1.2.8.0
Date: Mon, 22 Dec 2014 18:24:03 GMT
Content-Type: text/plain; charset=us-ascii
Transfer-Encoding: chunked
headers: Connection: keep-alive

Raw Parsed Response

Open output in new window Copy to clipboard Save as file Open in JSON tab

Authentication parameters missing

Code highlighting thanks to Code Mirror

1.4.4. Step 4

Delete the customer instance with id 123.

```
http://54.149.46.234/cxf/crm/customerservice/customers/123?
user_key=b9871b41027002e68ca061faeb2f972b
```

http://54.149.46.234/cxf/crm/customerservice/customers/123?user_key=b9871b41027002e68ca061faeb2f972b

GET POST PUT PATCH DELETE HEAD OPTIONS Other

Raw Form Headers

Raw Form Files (0) Payload

Encode payload Decode payload

```
<Customer xmlns="http://rest.fuse.quickstarts.jboss.org/">
  <name/ary:/name>
    <id>123/</id>
  </Customer>
```

application/x-www-form-urlencoded Set "Content-Type" header to overwrite this value

Clear Send

Status: 403 Forbidden Loading time: 211 ms

Request: User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/39.0.2171.95 Safari/537.36
Origin: chrome-extension://hgmpoofdfphtgpeklctftrfjeko
Content-Type: application/x-www-form-urlencoded
headers: Accept: */*
Accept-Encoding: gzip, deflate, sdch
Accept-Language: en-US,en;q=0.8

Response: Server: ngi_nginx/1.2.8.0
Date: Mon, 22 Dec 2014 18:25:03 GMT
Content-Type: text/plain; charset=us-ascii
Transfer-Encoding: chunked
headers: Connection: keep-alive

Raw Parsed Response

Open output in new window Copy to clipboard Save as file Open in JSON tab

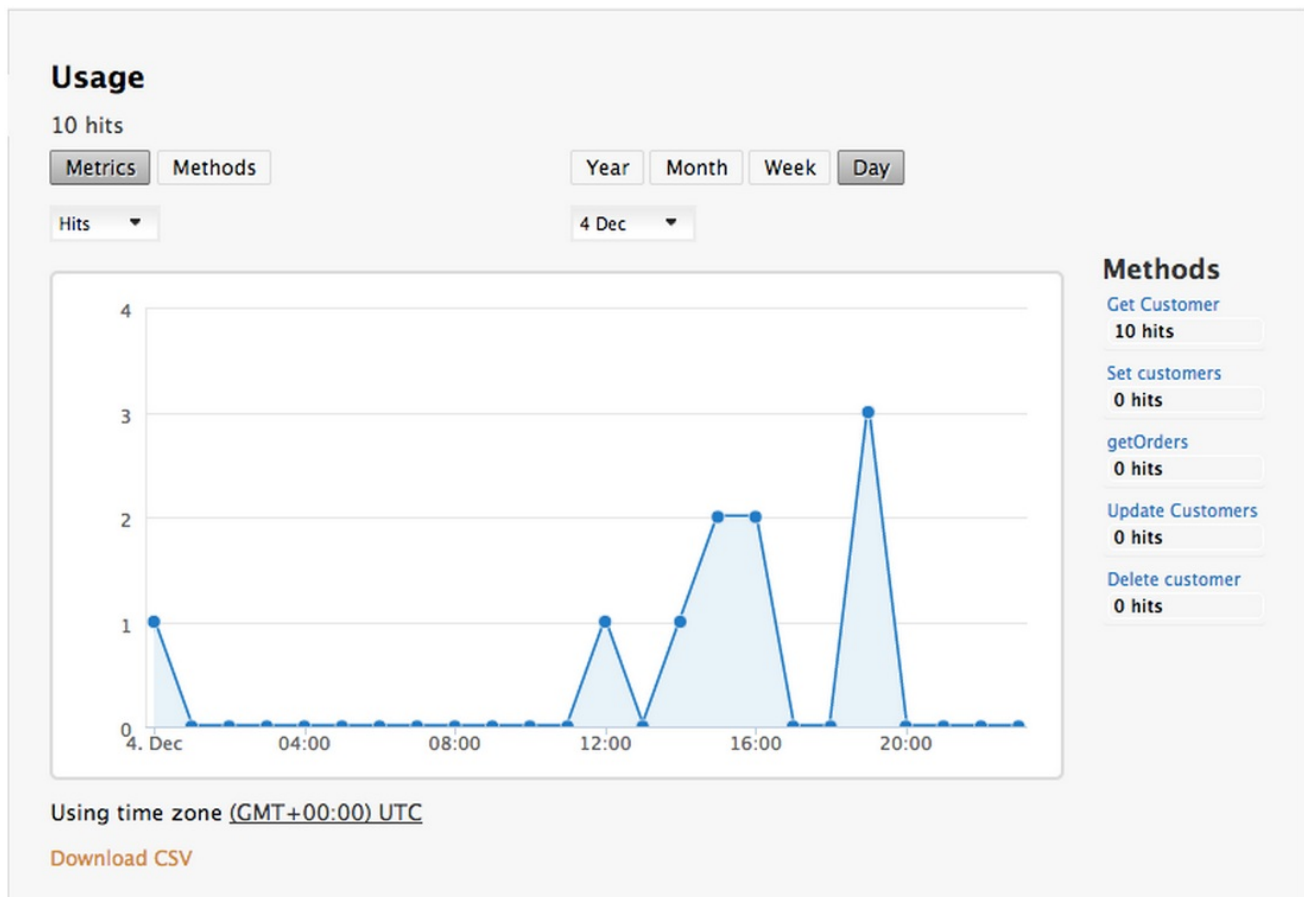
Authentication parameters missing

Code highlighting thanks to Code Mirror

1.4.5. Step 5

Check the API Management analytics of your API.

If you now log back in to your 3scale account and go to Monitoring > Usage, you can see the various hits of the API endpoints represented as graphs.



This is just one element of API Management that brings you full visibility and control over your API. Other features include:

1. Access control
2. Usage policies and rate limits
3. Reporting
4. API documentation and developer portals
5. Monetization and billing

For more details about the specific API Management features and their benefits, please refer to the [3scale API Management Platform product description](#).

For more details about the specific Red Hat JBoss Fuse product features and their benefits, please refer to the [JBoss Fuse Overview](#).

For more details about running Red Hat JBoss Fuse on OpenShift, please refer to the [Getting Started with JBoss Fuse on OpenShift](#).