

# Microsoft

## OPENSHIFT on Azure



Khaled Elbedri

Technology Solutions Professional - Global

Black Belt, Open Source Software

9/27/2017



## TABLE OF CONTENTS

Objectives and pre-requisites .....	2
Introduction to openshift.....	3
EXERCISE-1: Deploy Openshift on azure.....	4
EXERCISE-2: Create and manage projects.....	14
EXERCISE-3: Create and manage Applications .....	17
EXERCISE-4: Configuring automated builds .....	22
EXERCISE-5: Continuous deployment.....	24
EXERCISE-6: Mini project: JBOSS EAP application .....	28
EXERCISE-7: Monitoring oepnshift with azure oms.....	36
EXERCISE-8: Red Hat Cloud Forms on Azure .....	41
End the lab .....	55
References .....	55
Useful links .....	55
Redhat and Microsoft partnership.....	55

## OBJECTIVES AND PRE-REQUISITES

This document describes the steps necessary to deploy and manage Red Hat OpenShift container platform, and CloudForms on Azure. The lab is based on OpenShift version 3.5 and CloudForms version 4.1.

You will need an active Red Hat subscription and a valid Azure account to perform the lab instructions.

- Create your [free azure account](https://azure.microsoft.com/en-us/free/) (<https://azure.microsoft.com/en-us/free/>), today.
- Request a free 30 days [evaluation](#) from RedHat (<https://access.redhat.com/products/red-hat-openshift-container-platform/evaluation>), (<https://access.redhat.com/solutions/411973>)
- You need to have an account on [GitHub](#), If you don't have one create a free account (<https://github.com/>).

**NB:** As an alternative to OpenShift enterprise, you can deploy the community version OpenShift.org. And as an alternative to CloudForms, you can deploy the community version ManageIQ. The community edition represents a test bed incubator and upstream to the enterprise one. All OpenShift container platform features are also available in OpenShift.org.

To perform the lab using OpenShift.org, follow the steps at <https://github.com/Microsoft/openshift-origin> and skip Exercise-1.

**NB:** If you want to use your corporate Azure account, request a resource group with owner access control from your Azure admin.

**NB:** Even though, you can bring your own Red Hat subscription for OpenShift, the deployment template in this lab will provision the master and compute nodes from RHEL images, pay as you go. You will incur extra charges not covered by the Azure free tier. Amount varies depending on the number and size of the vm nodes.

The Lab exercises cover:

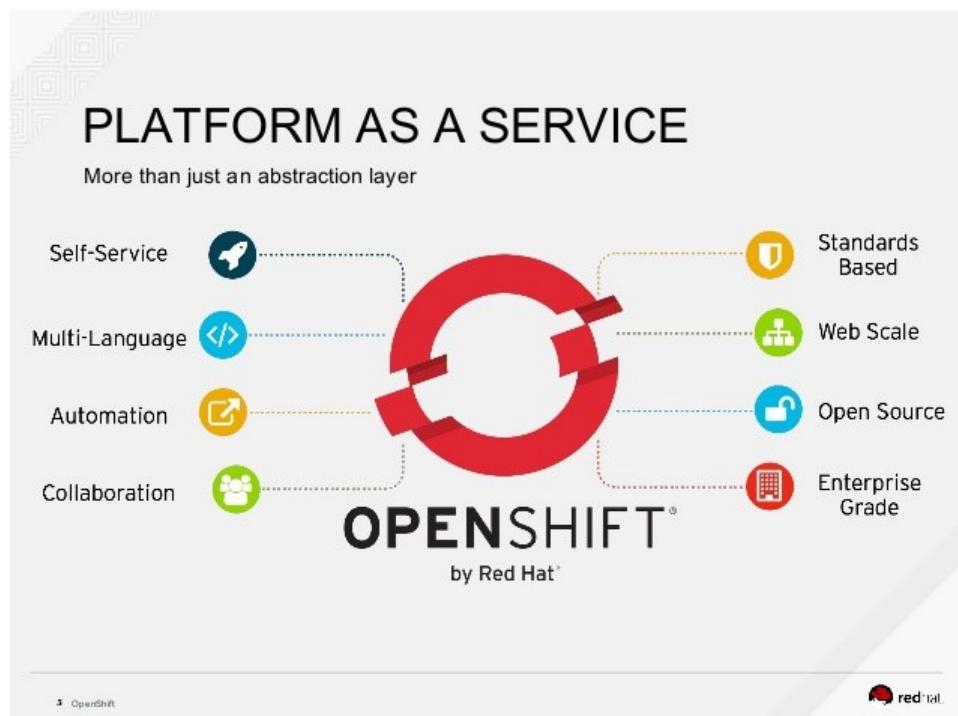
- o Introduction to OpenShift3
- o Deployment of OpenShift on Azure
- o Creating and managing OpenShift projects on azure
- o Creating and managing OpenShift applications on Azure
- o Automating builds with Linux Containers on Azure.
- o Mini project: Jboss EAP application
- o Integration of OpenShift with Azure OMS
- o Installation and introduction to CloudForms

## INTRODUCTION TO OPENSHIFT

OpenShift containers platform is Red Hat's Platform-as-a-Service (PaaS) on top of Kubernetes. It allows developers to quickly develop, host, and scale applications in a cloud environment.

Microsoft and Red Hat have signed a partnership that includes support to run Red Hat OpenShift on Microsoft Azure and Azure Stack.

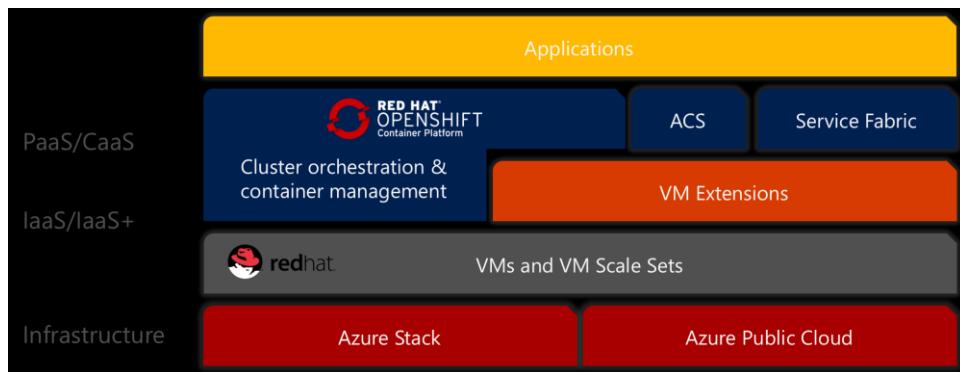
OpenShift offers multiple access modes including: developer CLI, admin CLI, web console and IDE plugins. Click2cloud is a plugin that allows Visual studio to deploy code to OpenShift, directly.



## EXERCISE-1: DEPLOY OPENSHIFT ON AZURE

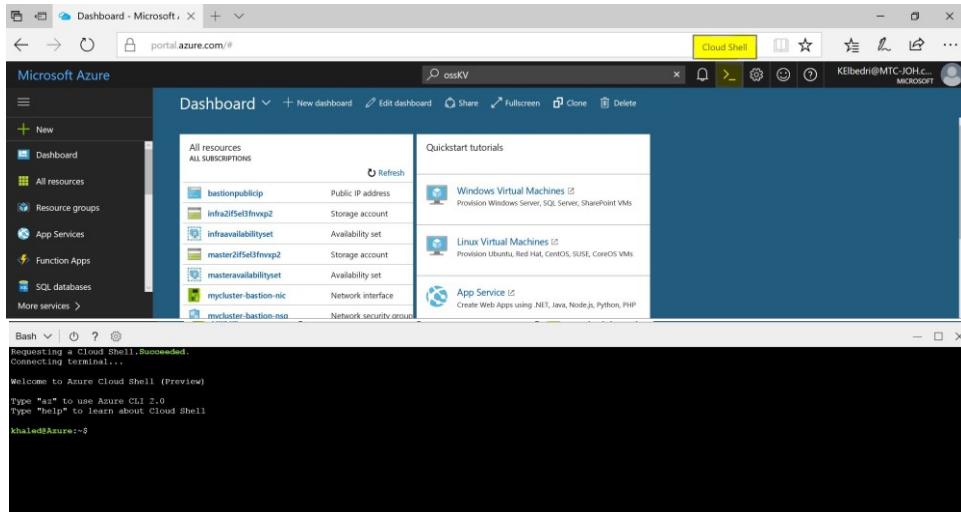
OpenShift, leverages multiple Azure services such as VM *extensions*, Azure disks, and *Key vaults*... to provide an Enterprise grade offering for customers who would like to containerize and manage their applications, without investing long time and hard effort configuring and integrating various tools.

OpenShift offers another alternative to multiple CaaS (container as a service) solutions available on Azure, such as *Azure container service*, *Azure service fabric* and *Pivotal* from *CloudFoundry*...



OpenShift container platform is available as an Azure Resource Manager solution at <https://github.com/Microsoft/openshift-container-platform>.

1. Login to Azure portal and start a new Bash Cloud shell session.



- From the open terminal, create a new ssh key pair with the name “osslab\_rsa” and save it under .ssh directory

```
$ ssh-keygen
```

```
Bash √ | ⌂ ? ⓘ
khaled@Azure:~$ ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/home/khaled/.ssh/id_rsa): .ssh/osslab_rsa
Enter passphrase (empty for no passphrase):
Enter passphrase again:
Your identification has been saved in .ssh/osslab_rsa.
Your public key has been saved in .ssh/osslab_rsa.pub.
The key fingerprint is:
SHA256:8XNjBzvVwvM1J2idrkn8zpgQ2kgDuk0XPWbVE khaled@cc-72f9-2a0b94c3-4087925635-4xxx
The key's randomart image is:
+---[RSA 2048]---+
| .o...o*o*o*o*o*o |
| +o+o+o+o+o+o+o+o |
| o+o+o+o+o+o+o+o+o |
| .o+o+o+o+o+o+o+o+o |
| o+o+o+o+o+o+o+o+o+o |
| o+o+o+o+o+o+o+o+o+o |
| o+o+o+o+o+o+o+o+o+o |
| o+o+o+o+o+o+o+o+o+o |
+---[SHA256]---+
khaled@Azure:~$
```

- Use the Azure CLI v2 to create a new resource group to host the lab resources

```
$ az group create -n ossdemo -l 'West Europe'
```

```
khaled@Azure:~$ az group create -n ossdemo -l 'West Europe'
{
  "id": "/subscriptions/f3a5dfdb-e863-40d9-b23c-752b886c0260/resourceGroups/ossmemo",
  "location": "westeurope",
  "managedBy": null,
  "name": "ossmemo",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null
}
```

- Create a Key Vault and add your ssh private key, created in the previous step.

```
$ az keyvault create -n ossKV -g ossdemo -l 'West Europe' --enabled-for-template-deployment true
```

```
khaled@Azure:~$ az keyvault create -n ossKV -g ossdemo -l 'West Europe' --enabled-for-template-deployment true
{
  "id": "/subscriptions/f3a5dfdb-e863-40d9-b23c-752b886c0260/resourceGroups/ossdemo/providers/Microsoft.KeyVault/vaults/ossKV",
  "location": "westeurope",
  "name": "ossKV",
  "properties": {
    "accessPolicies": [
      {
        "tenantId": "00000000-0000-0000-0000-000000000000"
      }
    ]
  }
}
```

```
$ az keyvault secret set --vault-name ossKV -n ossSecret
--file ~/.ssh/openssl_rsa
```

```
khaled@Azure:~$ az keyvault secret set --vault-name ossKV -n ossSecret --file ~/.ssh/openssl_rsa
{
  "attributes": {
    "created": "2017-09-25T09:12:12+00:00",
    "enabled": true,
    "expires": null,
    "notBefore": null,
    "recoveryLevel": "Purgeable",
    "updated": "2017-09-25T09:12:12+00:00"
  },
  "contentType": null,
  "id": "https://osskv.vault.azure.net/secrets/ossSecret/58a92e073f4f4245beb529cbab54afce",
  "kid": null,
  "managed": null,
  "tags": {
    "file-encoding": "utf-8"
  },
  "value": "-----BEGIN RSA PRIVATE KEY-----\nMIIEowIBAAKCAQEAwWViXCUWaAaUgsWqBAo8wPTtLw69zdK+wjHAr\n-----END RSA PRIVATE KEY-----"
}
```

The deployment of OpenShift relies on Ansible scripts that should be configured for Azure as the cloud provider. During and post-installation, OpenShift requires access to some Azure resources, like provisioning an Azure managed disk for persistence storage backend. When you have an application that needs to access or modify resources, you must set up an Azure Active Directory (AD) application and assign the required permissions to it. The service principal object defines the policy and permissions for an application's use in a specific tenant, providing the basis for a security principal to represent the application at run-time.

5. Create an Azure Active Directory Service Principal and choose a different password. Copy the resource group scope from step number 3.

```
$ az ad sp create-for-rbac -n openshiftcloudprovider --
password changeMePassword --role contributor --scopes
/subscriptions/f3a5dfdb-e863-40d9-b23c-
752b886c0260/resourceGroups/ossdemo
```

```

Khaled@Azure:~$ az ad sp create-for-rbac -n osscloudprovider --password changeMePassword --role contributor --scopes /subscriptions/f3a5dfdb-e863-40d9-b23c-752b886c0260/resourceGroups/ossdemo
{
  "appId": "83fc34ee-5be2-4c89-92fd-496c620c6f6e",
  "displayName": "osscloudprovider",
  "name": "http://osscloudprovider",
  "password": "changeMePassword",
  "tenant": "72f988bf-86f1-41af-91ab-2d7cd011db47"
}

```

- Now, go to the Azure portal and assign the required permissions to the service principal “osscloudprovider” on the resource group “ossdemo”.

The image contains two screenshots of the Microsoft Azure portal interface, illustrating the steps to assign permissions to a service principal.

**Screenshot 1: Resource groups - ossdemo - Access control (IAM)**

- The left sidebar shows the 'Resource groups' section with 'ossdemo' selected.
- The main pane shows the 'Access control (IAM)' blade for 'ossdemo'. The 'Add' button is highlighted.
- The 'Role' dropdown is set to 'Contributor'.
- The 'Select' dropdown shows 'osscloudprovider' selected.
- The 'Role' dropdown is set to 'Contributor'.
- The 'Save' button is at the bottom right.

**Screenshot 2: Add permissions - Microsoft Azure - ossdemo - Access control (IAM) - Add permissions**

- The interface is identical to the first screenshot, showing the 'osscloudprovider' service principal selected for the 'ossdemo' resource group.
- The 'Role' dropdown is set to 'Contributor'.
- The 'Select' dropdown shows 'osscloudprovider' selected.
- The 'Role' dropdown is set to 'Contributor'.
- The 'Save' button is at the bottom right.

7. Note the application id of your service principal.

```
$ az ad sp show --id http://openshiftcloudprovider
```

8. Use the following Azure resource manager solutions [template](#) to deploy your OpenShift environment:

<https://portal.azure.com/#create/Microsoft.Template/uri/https%3A%2F%2Frwa.githubusercontent.com%2FMicrosoft%2Fopenshift-container-platform%2Fmaster%2Fazureddeploy.json>

- At this stage we will need Red Hat Network or Red Hat satellite credentials to register RHEL vms with RedHat and get access to the required software channels during the deployment. More specifically, we will need:
  - o Red Hat Network/Satellite name
  - o The associated password or activation key
- Pool Id provides access to the required software channels for Openshift and Cloudforms. PoolId can be listed by invoking the command 'subscription-manager list -available' from a registered RHEL system. Contact your Red Hat admin or Red Hat support if you are missing information
- For high availability consideration, we are deploying 3 vms for each type (master, infra and node). If you want to deploy the lab with minimal cost, you can reduce the number of vms to one per each. Also, you can scaledown the vm families, but preferably stick to vms with SSD disks for faster deployment.

- The master and infra load balancer DNSs (in red circles) should be globally unique. Choose your own names.

**TEMPLATE**

Customized template  
34 resources

Edit template Edit parameters Learn more

**BASICS**

\* Subscription Microsoft Azure Internal Consumption 2

\* Resource group Create new Use existing ossdemo

\* Location West Europe

**SETTINGS**

\_artifacts Location https://raw.githubusercontent.com/Microsoft/openshift-container-platform/master/

Custom Vhd Or Gallery gallery

Custom Storage Account https://osdiskstorageaccount.blob.core.windows.net/

Custom Storage Account https://osdiskstorageaccount.blob.core.windows.net/

Custom Os Disk Name images/customosdisk.vhd

Master Vm Size Standard\_DS3\_V2

Infra Vm Size Standard\_DS3\_V2

Node Vm Size Standard\_DS12\_V2

Openshift Cluster Prefix oss

\* Openshift Master Public Ip Dns Label ossmaster

\* Infra Lb Public Ip Dns Label ossinfra

Master Instance Count 3

Infra Instance Count 3

Node Instance Count 3

Data Disk Size 128

Admin Username ossadmin

* Openshift Password ⓘ	••••••••	✓
Enable Metrics ⓘ	true	✓
Enable Logging ⓘ	true	✓
Enable Cockpit ⓘ	true	✓
Rhsm Username Password Or Activation Key ⓘ	usernamepassword	✓
* Rhsm Username Or Org Id ⓘ	••••••••••••	✓
* Rhsm Password Or Activation Key ⓘ	••••••••	✓
* Rhsm Pool Id ⓘ	8a85f9819be87e78015be89906382ad7	✓
* Ssh Public Key ⓘ	•••••••••••••••••••••••••••••••	✓
* Key Vault Resource Group ⓘ	ossdemo	✓
* Key Vault Name ⓘ	osskV	✓
* Key Vault Secret ⓘ	••••••••	✓
* Aad Client Id ⓘ	your service principal appId	✓
* Openshift Password ⓘ	••••••	✓
Enable Metrics ⓘ	true	✓
Enable Logging ⓘ	true	✓
Enable Cockpit ⓘ	true	✓
Rhsm Username Password Or Activation Key ⓘ	usernamepassword	✓
* Rhsm Username Or Org Id ⓘ	••••••••••••	✓
* Rhsm Password Or Activation Key ⓘ	••••••••	✓
* Rhsm Pool Id ⓘ	8a85f9819be87e78015be89906382ad7	✓
* Ssh Public Key ⓘ	•••••••••••••••••••••••••••••••	✓
* Key Vault Resource Group ⓘ	ossdemo	✓
* Key Vault Name ⓘ	osskV	✓
* Key Vault Secret ⓘ	••••••••	✓
* Aad Client Id ⓘ	your service principal appId	✓

## TEMPLATE

 Customized template  
34 resources

 Edit template  Edit parameters  Learn more

## BASICS

\* Subscription

\* Resource group  Create new  Use existing

\* Location

## SETTINGS

\_artifacts Location

Custom Vhd Or Gallery

Custom Storage Account

\* Aad Client Secret  

Default Sub Domain Type

Default Sub Domain

## TERMS AND CONDITIONS

[Azure Marketplace Terms](#) | [Azure Marketplace](#)

By clicking "Purchase," I (a) agree to the applicable legal terms associated with the offering; (b) authorize Microsoft to charge or bill my current payment method for the fees associated the offering(s), including applicable taxes, with the same billing frequency as my Azure subscription, until I discontinue use of the offering(s); and (c) agree that, if the deployment involves 3rd party offerings, Microsoft may share my contact information and other details of such deployment with the publisher of that offering.

Pin to dashboard

**Purchase**

- From the Azure portal, go to your resource group “ossdemo”, track the progress of the deployment and make sure the deployment finishes, successfully.

DEPLOYMENT NAME	STATUS	TIMESTAMP	DURATION
ossedeployment	Deploying	9/25/2017 6:04:45 PM	6 seconds
Microsoft.Template	Succeeded	9/24/2017 7:09:09 PM	35 minutes 11 seconds
OpenShiftDeployment	Succeeded	9/24/2017 7:09:52 PM	22 minutes 18 seconds
nodeVmDeployment	Succeeded	9/24/2017 6:37:53 PM	2 minutes 30 seconds
masterVmDeployment0	Succeeded	9/24/2017 6:37:32 PM	1 minute 49 seconds
bastionVmDeployment	Succeeded	9/24/2017 6:37:03 PM	1 minute 41 seconds
infraVmDeployment	Succeeded	9/24/2017 6:36:55 PM	1 minute 26 seconds
cleanupops	Succeeded	6/26/2017 6:51:30 AM	11 minutes 20 seconds
b15ffab2-d0da-4775-9b7...	Succeeded	6/26/2017 5:21:08 PM	2 minutes 49 seconds
azuresql1495485922.1163...	Succeeded	6/26/2017 4:16:40 PM	11 minutes 15 seconds
a4ab9b04-5b04-47e7-901...	Succeeded	6/26/2017 14:39 PM	8 minutes 59 seconds
Microsoft.ContainerRegistry	Succeeded	6/26/2017 4:09:50 PM	28 seconds
azuresql149542000.95569...	Succeeded	6/25/2017 10:29:14 PM	9 minutes 11 seconds

The following diagram explains the physical architecture of the deployed cluster.

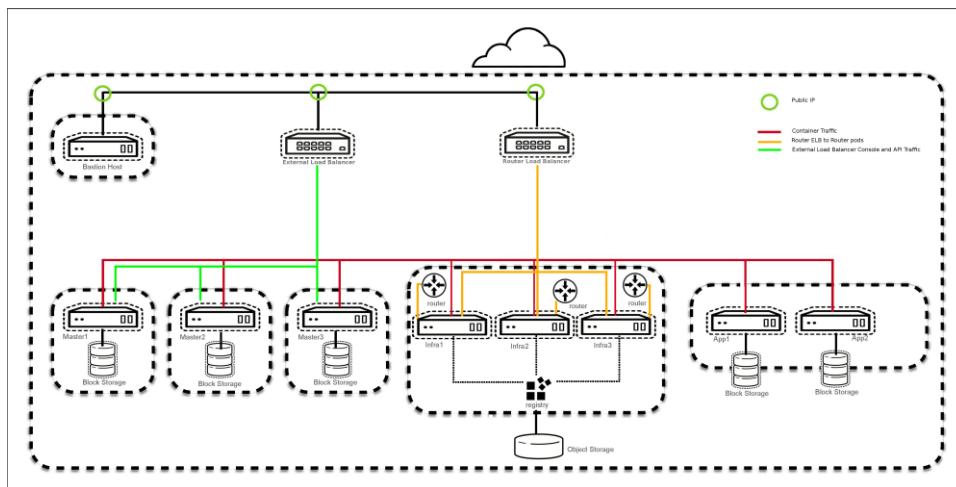


Figure 6: Deployment diagram

The bastion server implements mainly two distinct functions. One is that of a secure way to connect to all the nodes, and second that of the "installer" of the system. The bastion host is the only ingress point for SSH in the cluster from external entities. When connecting to the OpenShift Container Platform infrastructure, the bastion forwards the request to the appropriate server.

The next diagram, explains the role and tasks of the Openshift master/agents and the logical architecture of the solution.

## How OpenShift Enterprise Works

OpenShift Enterprise 3 is built around a core of application containers powered by Docker, with orchestration and management provided by Kubernetes, on a foundation of Red Hat Enterprise Linux.

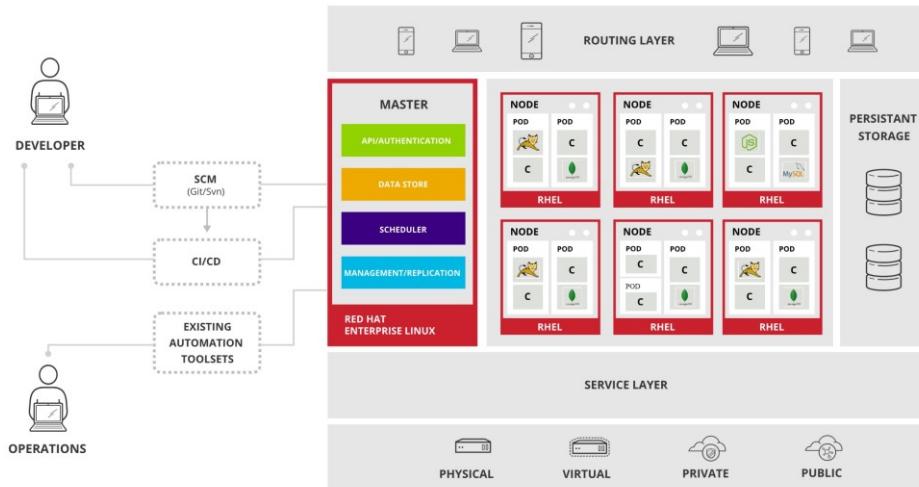


Figure 7: OpenShift logical architecture

## EXERCISE-2: CREATE AND MANAGE PROJECTS

There are many ways to launch images within an OpenShift project. In this exercise we will focus on the graphical portal.

To create an *application*, you must first create a new *project*, then select an *InstantApp* template. From there, OpenShift begins the build process, and creates a new deployment.

1. Login to your *github* account, or create one if you didn't.
2. Browse to *openshift/ruby-ex* repository and fork it into your *github* account

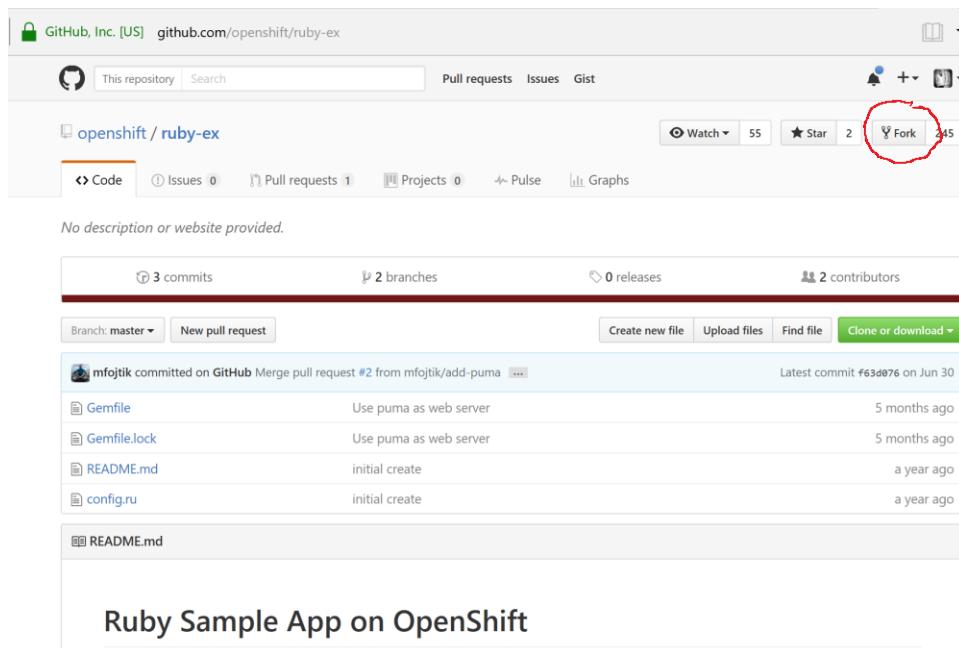
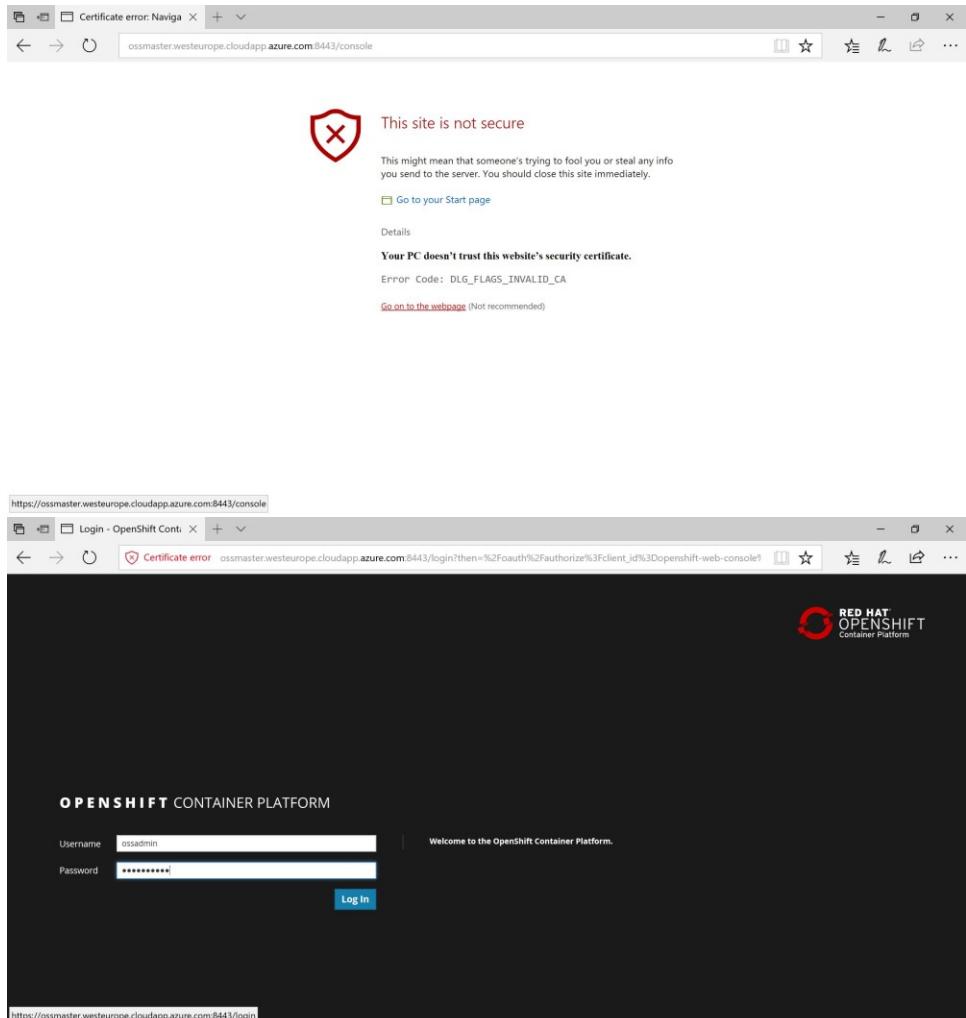


Figure 8: Fork ruby-ex sample application

3. From your browser, visit the OpenShift web console at <https://FQDN-master-node:8443>. The web site, uses a self-signed certificate, so if prompted, continue and ignore the browser warning.
4. Log in using your username and password.



5. To create a new project, click **New Project**.
6. Type a unique name, display name, and description for the new project.
7. Click **Create**. The web console's welcome screen should start loading.

The screenshot shows the OpenShift Web Console interface. The title bar reads "OpenShift Web Console". Below it, a banner indicates a "Certificate error" for the URL "ossmaster.westeurope.cloudapp.azure.com:8443/console". The main area is titled "OPENSHIFT CONTAINER PLATFORM" and displays a list of projects under the heading "Projects". The projects listed are:

- default (created an hour ago)
- kube-system (created an hour ago)
- logging (created an hour ago)
- management-infra (Management Infrastructure, created an hour ago)
- openshift (created an hour ago)
- openshift-infra (created an hour ago)

Each project entry has three small icons on the right: a gear, a pencil, and a trash can.

The screenshot shows the "New Project" dialog box from the OpenShift Web Console. The title bar reads "OpenShift Web Console". Below it, a banner indicates a "Certificate error" for the URL "ossmaster.westeurope.cloudapp.azure.com:8443/console/create-project". The main area is titled "OPENSHIFT CONTAINER PLATFORM" and displays the "New Project" form. The fields are as follows:

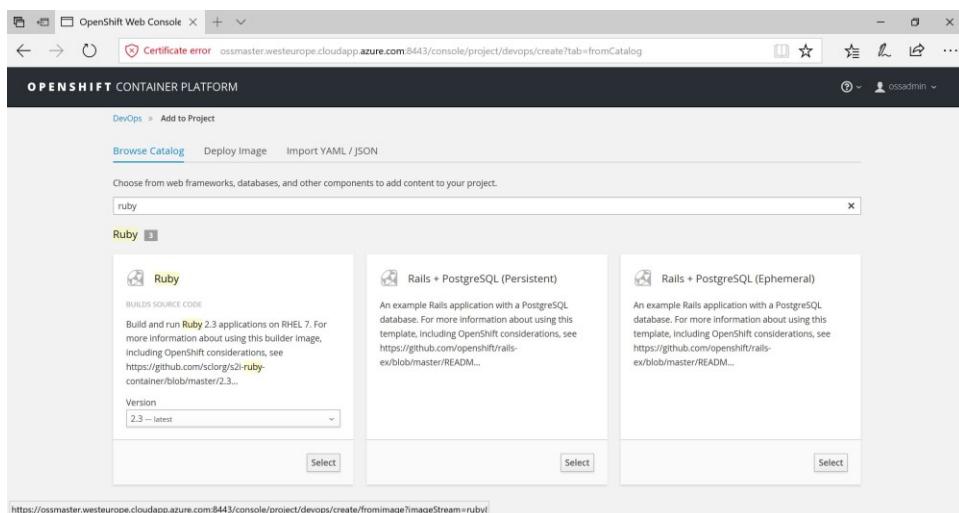
- \* Name: devops (A unique name for the project.)
- Display Name: DevOps
- Description: My Web Application

At the bottom of the dialog are two buttons: "Create" (highlighted in blue) and "Cancel".

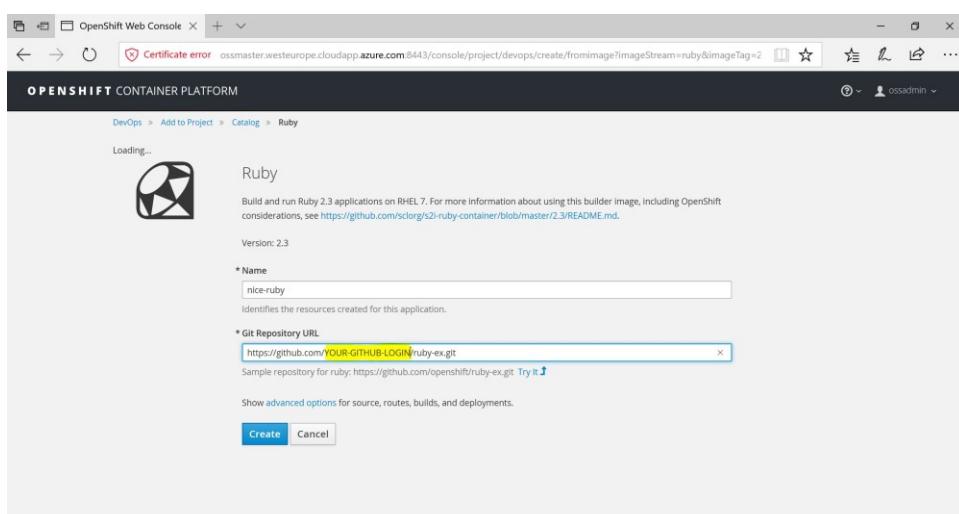
## EXERCISE-3: CREATE AND MANAGE APPLICATIONS

The *Select Image or Template* page gives you the option to add an application to your project from a publicly accessible *Git* repository, or from a *template*:

1. If creating a new project did not automatically redirect you to the *Select Image or Template page*, you might need to click **Add to Project**.
2. Click **Browse**, then select
3. Click the **ruby:latest** builder image.



4. Type a **name** for your application, and specify the git repository URL you previously forked: [https://github.com/<your\\_github\\_username>/ruby-ex.git](https://github.com/<your_github_username>/ruby-ex.git).



5. Optionally, click **Show advanced routing, build, and deployment options**. Explore the build configuration and other options and note that this example application automatically creates a route, *webhook* trigger, and builds change triggers. A *webhook* is an HTTP call-back triggered by a specific event.
6. Click **Create**. Creating your application might take some time. note the *payload url*, we will use it later to set a *webhook* in your *github* repository.
7. You can follow along on the **Overview** page of the web console to see the new resources being created, and watch the progress of the build and deployment. Click on “view log”, you will notice that Openshift is pulling the code of the application from Github and building the layers of the container image that will host the application. Once the build is complete, Openshift will start a new pod.

The screenshot shows the OpenShift Web Console interface. On the left is a sidebar with navigation links: Overview, Applications, Builds, Resources, Storage, and Monitoring. The main area is titled 'Project DevOps'. A prominent message at the top says 'Created application nice-ruby in project DevOps.' Below this, under the 'Builds' section, there is a card for 'NICE RUBY' which is currently 'running'. It includes a link to its temporary URL: <http://nice-ruby-devops.52.166.253.96.xip.io>. A note below the card states: 'nice-ruby has containers without health checks, which ensure your application is running correctly. Add Health Checks'.

This screenshot shows the 'Logs' tab for the 'NICE RUBY' build. The logs output is as follows:

```

3 Author: khalid999 (khalid999@yahoo.com)
4 Date: Mon May 15 18:44:40 2017 +0000
5 ----> Installing application source ...
6 ----> Building custom Ruby application from source Mon Sep 25 19:33:41 UTC 2017 ...
7 ----> Running "bundle install --deployment --without development:test" ...
8 Warning: the running version of Bundler is older than the version that created the lockfile. We suggest you upgrade to the latest version of Bundler by running `gem install bundler`.
9 Fetching gem metadata from https://rubygems.org/...
10 Fetching version metadata from https://rubygems.org/...
11 Warning: the running version of Bundler is older than the version that created the lockfile. We suggest you upgrade to the latest version of Bundler by running `gem install bundler`.
12 Installing puma 3.4.0 with native extensions
13 Installing rack 1.6.4
14 Using Bundler 1.15.3
15 Bundler installed 2 Gemfile dependencies, 3 gems now installed.
16 Gems in the groups development and test were not installed.
17 Bundled gems are installed into ./vendor.
18 ----> Cleaning up unused ruby gems ...
19 Warning: the running version of Bundler is older than the version that created the lockfile. We suggest you upgrade to the latest version of Bundler by running `gem install bundler`.
20 Pushing image 172.30.36.121:5000/devops/nice-ruby:latest ...
21 Pushed 8/6 layers, 2% complete
22 Pushed 1/6 layers, 14% complete
23 Pushed 2/6 layers, 39% complete
24 Pushed 3/6 layers, 55% complete
25 Pushed 4/6 layers, 78% complete

```

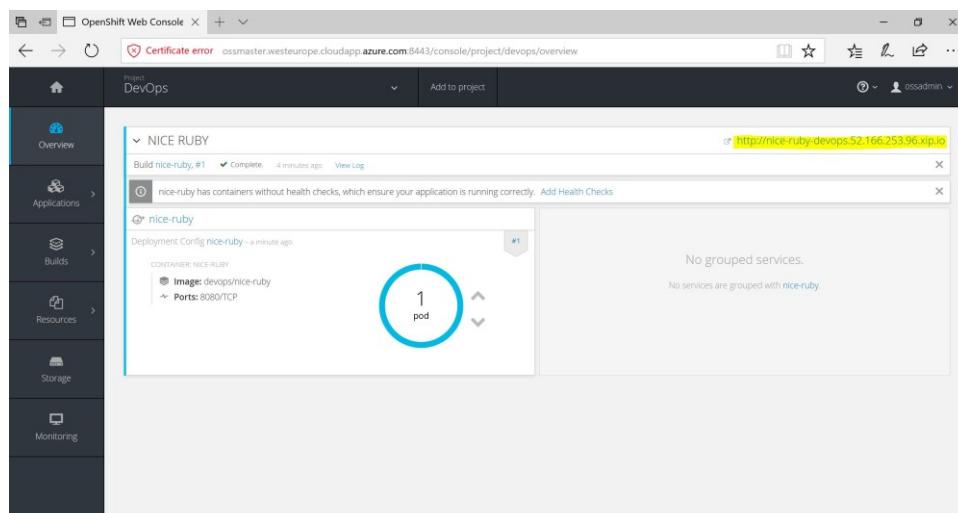
OpenShift leverages the Kubernetes concept of a pod, which is one or more containers deployed together on one host. A pod is the smallest compute unit that can be defined, deployed, and managed.

Pods are the rough equivalent of a machine instance (physical or virtual) to a container. Each pod is allocated its own internal IP address, therefore owning its entire port space. Containers within pods can share their local storage and networking.

While the Ruby *pod* is being created, its status is shown as pending. The Ruby *pod* then starts up and displays its newly-assigned IP address. When the Ruby *pod* is running, the build is complete.

Pods have a lifecycle; they are defined, then they are assigned to run on a node, then they run until their container(s) exit or they are removed for some other reason. Pods, depending on policy and exit code, may be removed after exiting, or may be retained in order to enable access to the logs of their containers.

8. From the overview page, click the web address for the application in the upper right corner. Verify that the web application is up and available.



- Return to the *OpenShift* admin console. Browse to the project's overview page, and test scaling out and in your application by increasing or decreasing the number of *pods*, using the up and down arrow signs on the web console.
- Scale out the app into 3 pods and watch the progress.

- Browse to **Applications -> Pods**, and make sure 3 pods serving the same application are now up and running.

The screenshot shows the OpenShift Web Console interface. The top navigation bar displays the URL "ossmaster.westeurope.cloudapp.azure.com:8443/console/project/devops/browse/pods". The left sidebar menu includes "Overview", "Applications", "Builds", "Resources", "Storage", and "Monitoring". The main content area is titled "Pods" and lists four pods:

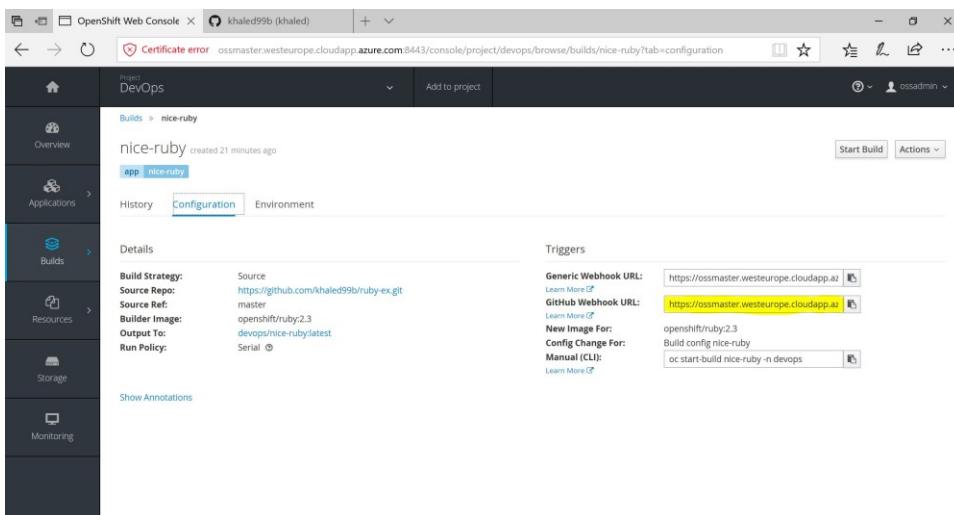
Name	Status	Containers Ready	Container Restarts	Age
nice-ruby-1-4bc8	Running	1/1	0	2 minutes
nice-ruby-1-hkxkn	Running	1/1	0	2 minutes
nice-ruby-1-48fhv	Running	1/1	0	6 minutes
nice-ruby-1-build	Completed	0/1	0	10 minutes

## EXERCISE-4: CONFIGURING AUTOMATED BUILDS

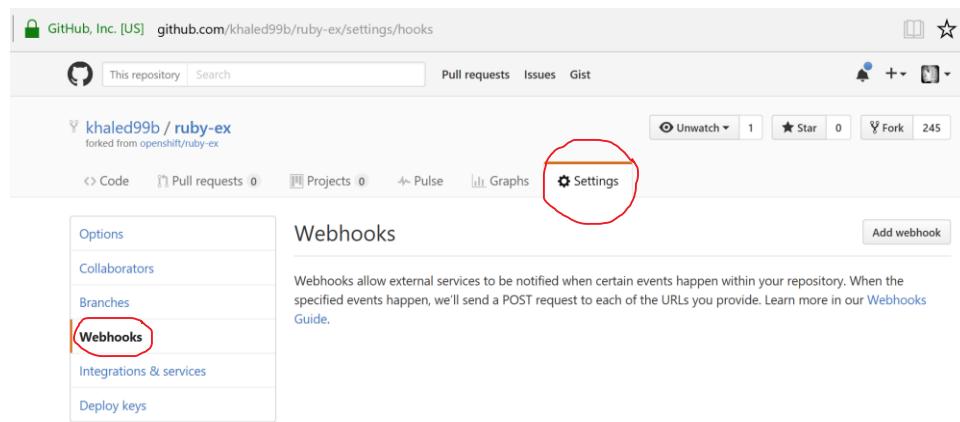
Since we forked the source code of the application from the [OpenShift GitHub repository](#), we can use a *webhook* to automatically trigger a rebuild of the application whenever code changes are pushed to the forked repository.

To set up a *webhook* for your application:

1. From the Web Console, navigate to the project containing your application.
2. Click the **Browse** tab, then click **Builds**.
3. Click your build name, then click the **Configuration** tab.
4. Click next to **GitHub webhook URL** to copy your *webhook* payload URL.



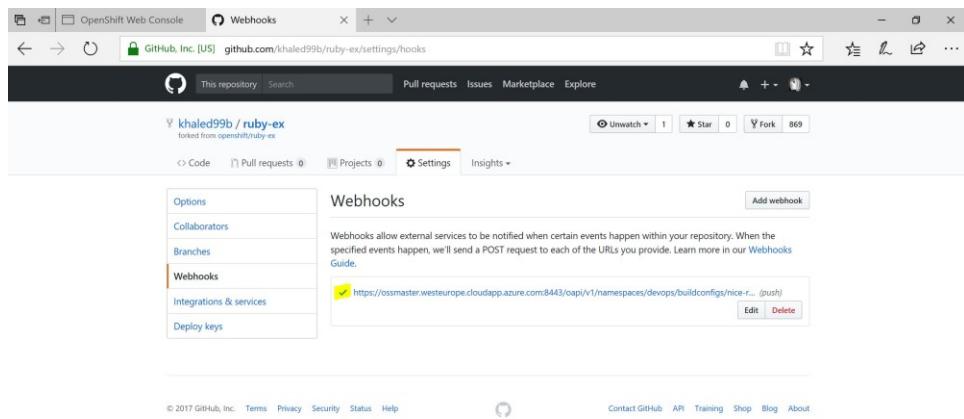
5. Navigate to your forked repository on GitHub, then click **Settings**.
6. Click **Webhooks & Services** and Click **Add webhook**.



7. Paste your *webhook* URL into the **Payload URL** field.
8. As Content Type choose application/json
9. **Disable SSL verification** and click **Add webhook** to save.

GitHub will now attempt to send a ping payload to your *OpenShift* server to ensure that communication is successful. If you see a green check mark appear next to your *webhook* URL, then it is correctly configured.

Hover your mouse over the check mark to see the status of the last delivery.



Next time you push a code change to your forked repository, your application will automatically rebuild.

## EXERCISE-5: CONTINUOUS DEPLOYMENT

In this section, we demonstrate one of the most powerful features of *OpenShift*. We will see how we can trigger a continuous deployment pipeline, just by committing code change to Github.

Once there is a code change, the Github *webhook* will trigger the build of a new container image that combines a blueprint image from the registry with the updated code and generate a new image. This feature is called *S2I*, or source to image. Once the build finishes, *OpenShift* will automatically deploy the new application based on the new image. This capability enables multiple deployment strategies such as A/B testing, Rolling upgrades...

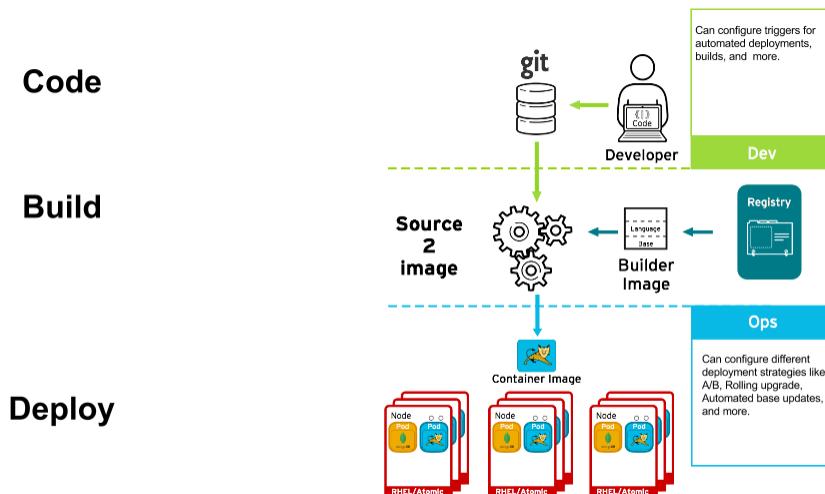


Figure 21: Continuous deployment pipeline

1. Use Azure cloud shell or install *Git* into your local machine

PS: If you don't want to use *git*, you still can perform this exercise by moving to step-5 and editing the file *config.ru*, directly from the web interface of *github* and then go to step-8.

Create a “dev” folder and change into.

```
$ mkdir dev && cd dev
```

2. Clone the forked repository to your local system

```
$ git clone  
https://github.com/<YourGithubUsername>/ruby-ex.git
```

3. Make sure your local *git* repository is referencing to your *ruby-ex git*, on *github*:

```
$ cd ruby-ex  
$ git remote -v
```

4. On your local machine, use your preferred text editor to change the sample application's source for the file **config.ru**

Make a code change that will be visible from within your application. For example: on line 229, change the title to “Welcome to your Ruby application on OpenShift POWERED BY AZURE!”, then save your changes.

5. Verify the working tree status

```
$ git status
```

6. Add config.ru content to the index, Commit the change in *git*, and push the change to your fork. You will need to authenticate with your *github* credentials

```
$ git add config.ru  
$ git commit -m "simple message"  
$ git status  
$ git push
```

7. If your *webhook* is correctly configured, your application will immediately rebuild itself, based on your changes. Monitor the build from the graphical console. Once the rebuild is successful, view your updated application using the route that was created earlier. Now going forward, all you need to do is push code updates and OpenShift handles the rest.

The image consists of two vertically stacked screenshots of the OpenShift Web Console. Both screenshots show the 'Project DevOps' view for a project named 'NICE RUBY'. In the top screenshot, under the 'Builds' section, there is a build step labeled 'Build nice-ruby, #5' with a status of 'Running'. A yellow box highlights this status. Below it is another build step labeled 'Build nice-ruby, #4' with a status of 'Complete'. To the right, a summary indicates '3 pods' with a circular icon. In the bottom screenshot, the deployment status has changed to 'Rolling deployment in progress'. The circular icon now shows '1 pod' transitioning to '3 pods'.

## 8. From the web browser refresh the page and note the new change

The image shows a screenshot of the application's landing page at the URL 'nice-ruby-devops.52.166.253.96.xip.io'. The main heading reads 'Welcome to your Ruby application on OpenShift POWERED BY AZURE!!'. Below this, there are several sections: 'Deploying code changes', 'Managing your application', 'Web Console', 'Command Line', 'Development Resources', and a terminal window showing git clone commands. A yellow box highlights the word 'POWERED BY AZURE!!'.

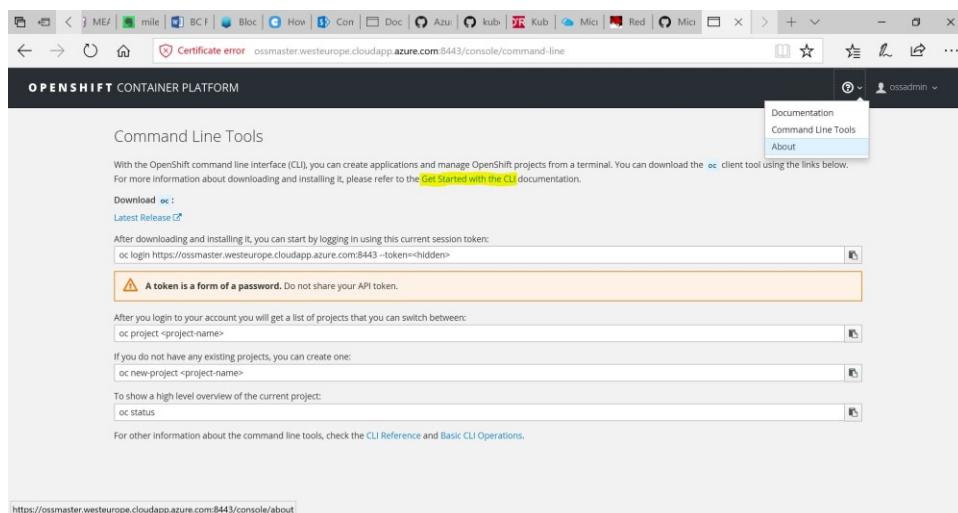
9. You may find it useful to manually rebuild an image if your *webhook* is not working, or if a build fails and you do not want to change the code before restarting the build. To manually rebuild the image based on your latest committed change to your forked repository:
  - a. Click the **Browse** tab, then click **Builds**.
  - b. Find your build, then click **Start Build**.

## EXERCISE-6: MINI PROJECT: JBOSS EAP APPLICATION

In this mini project, we will deploy a three tiers application consisting of Leaflet mapping front end with a JaxRS and MongoDB back end. The application, allows to visualize the locations of major National Parks and Historic Sites (<https://github.com/OpenShiftDemos/restify-mongodb-parks>).

During this exercise, we will leverage the CLI tool of OpenShift.

1. Download and install the OpenShift CLI related to your operating system.  
The easiest way to download the CLI is by accessing the About page on the web console if your cluster administrator has enabled the download links.  
Another alternative is to ssh into your bastion host that has the CLI tool installed already.



2. Use your openshift url endpoint to login to your environment from the CLI

```
ossadmin@oss-bastion:~$ oc login https://ossmaster.westeurope.cloudapp.azure.com:8443
Authentication required for https://ossmaster.westeurope.cloudapp.azure.com:8443 (openshift)
Username: ossadmin
Password:
Login successful.

You have access to the following projects and can switch between them with 'oc project <projectname>':
* default
  devops
  kube-system
  logging
  management-infra
  openshift
  openshift-infra

Using project "default".
[ossadmin@oss-bastion ~]$
```

3. Create a new project “nationalparks”

```

[ossadmin@oss-bastion ~]$ oc new-project nationalparks
Now using project "nationalparks" on server "https://ossmaster.westeurope.cloudapp.azure.com:8443".
You can add applications to this project with the 'new-app' command. For example, try:

  oc new-app centos/ruby-22-centos7~https://github.com/openshift/ruby-ex.git

to build a new example application in Ruby.
[ossadmin@oss-bastion ~]$

```

- From the web console, add a new Java application using the following git lab repository <https://gitlab.com/gshipley/nationalparks.git>

The screenshot shows the OpenShift Web Console interface. The URL is <https://ossmaster.westeurope.cloudapp.azure.com:8443/console/project/nationalparks/create/category/java/eap>. The page title is "OPENSHIFT CONTAINER PLATFORM". The navigation path is "nationalparks > Add to Project > Catalog > Java > Red Hat JBoss EAP". The main content area is titled "Red Hat JBoss EAP" and lists several application templates:

- Red Hat JBoss EAP 6.4 (Builds source code, JBoss EAP 6.4 S2I Images, Version 1.5, Select button)
- Red Hat JBoss EAP 7.0 (Builds source code, JBoss EAP 7.0 S2I Images, Version 1.5, Select button)
- Red Hat JBoss EAP 6.4 + A-MQ (Persistent with https) (Application template for EAP 6 A-MQ applications with persistent storage built using S2I, Select button)
- Red Hat JBoss EAP 6.4 + A-MQ (Ephemeral with https) (Application template for EAP 6 A-MQ applications built using S2I, Select button)
- Red Hat JBoss EAP 6.4 (no https) (Application template for EAP 6 applications built using S2I, Select button)
- Red Hat JBoss EAP 6.4 (with https) (Application template for EAP 6 applications built using S2I, Select button)

The screenshot shows the OpenShift Web Console interface. The URL is <https://ossmaster.westeurope.cloudapp.azure.com:8443/console/project/nationalparks/create/fromImage?imageStream=jboss-eap70>. The page title is "OPENSHIFT CONTAINER PLATFORM". The navigation path is "nationalparks > Add to Project > Catalog > Red Hat JBoss EAP 7.0". The main content area is titled "Red Hat JBoss EAP 7.0" and shows the following form fields:

- Name:** nationalparklocator (Identifies the resources created for this application)
- Git Repository URL:** <https://gitlab.com/gshipley/nationalparks.git>

At the bottom of the form, there are "Create" and "Cancel" buttons.

- List builds operations:

```

[ossadmin@oss-bastion ~]$ oc get builds
NAME          TYPE      FROM      STATUS    STARTED           DURATION
nationalparklocator-1   Source    Git@46aad91  Running   About a minute ago
[ossadmin@oss-bastion ~]$

```

6. List existing projects, pods and view logs in real time:

The image displays three separate terminal windows, each showing command-line output. The top window shows the results of the command `oc get projects`, listing various OpenShift projects with their status. The middle window shows the results of `oc get pods -w`, monitoring pod status over time. The bottom window shows the results of `oc logs -f` for a specific pod, displaying log messages in real-time.

```
[ossadmin@oss-bastion ~]$ oc get projects
NAME          DISPLAY NAME   STATUS
default        DevOps        Active
devops         DevOps        Active
kube-system   DevOps        Active
logging        DevOps        Active
management-infra  DevOps        Active
nationalparks  DevOps        Active
openshift      DevOps        Active
openshift-infra DevOps        Active
[ossadmin@oss-bastion ~]$
```

```
[ossadmin@oss-bastion ~]$ oc get pods -w
NAME      READY   STATUS    RESTARTS   AGE
^C[ossadmin@oss-bastion ~]$ oc get pods -w
NAME      READY   STATUS    RESTARTS   AGE
E
nationalparklocator-1-build  0/1     ContainerCreating  0          8s
NAME      READY   STATUS    RESTARTS   AGE
nationalparklocator-1-build  1/1     Running   0          10s
```

```
[ossadmin@oss-bastion ~]
^C[ossadmin@oss-bastion ~]$ oc logs -f nationalparklocator-1-build
Cloning "https://gitlab.com/gshipley/nationalparks.git" ...
Commit: 46aad9156b00bdca13b7d84763219d700d845126 (Update index.htm
1)
Author: dev <dev@email.com>
Date: Mon Sep 5 20:05:03 2016 +0000
Found pom.xml... attempting to build with 'mvn -e -Popenshift -DskipTests
-Dcom.redhat.xpaas.repo.redhatga package --batch-mode -Djava.net.preferIPv
4Stack=true '
Using MAVEN_OPTS '-XX:+UseParallelGC -XX:MinHeapFreeRatio=20 -XX:MaxHeapFr
eeRatio=40 -XX:GCTimeRatio=4 -XX:AdaptiveSizePolicyWeight=90 -XX:MaxMetasp
aceSize=100m -XX:+ExitOnOutOfMemoryError'
```

Output truncated ....

```

ossadmin@oss-bastion:~ ap/standalone/deployments for later deployment...
Pushing image 172.30.36.121:5000/nationalparks/nationalparklocator:latest
...
Pushed 0/7 layers, 1% complete
Pushed 1/7 layers, 27% complete
Pushed 2/7 layers, 48% complete
Pushed 3/7 layers, 58% complete
Pushed 4/7 layers, 84% complete
Pushed 5/7 layers, 88% complete
Pushed 6/7 layers, 88% complete
Pushed 6/7 layers, 98% complete
Pushed 7/7 layers, 100% complete
Push successful
[ossadmin@oss-bastion ~]$
```

## 7. Browse the newly created application and verify it is available

The screenshot shows two browser windows. The top window is the OpenShift Web Console for the project 'nationalparks'. It displays the 'NATIONALPARKLOCATOR' application, which has a single pod running. The bottom window shows a map titled 'National Parks on OpenShift' covering most of Europe, with various national parks outlined in red across different countries.

8. We can see the map but not the attraction points. The reason is that we only deployed the front-end application. What we will need now is to add a backend data base. From the web console, add a new persistent mongodb data store. And set the needed environment variables and specification as bellow:

## MongoDB (**Persistent**)

MongoDB database service, with persistent storage. For more information about using this template, including OpenShift considerations, see <https://github.com/sclorg/mongodb-container/blob/master/3.2/README.md>.

NOTE: Scaling to more than one replica is not supported. You must have persistent volumes available in your cluster to use this template.

### Images

**mongodb:3.2** from parameter Version of MongoDB Image

### Parameters

#### \* Memory Limit

Maximum amount of memory the container can use.

#### Namespace

The OpenShift Namespace where the ImageStream resides.

#### \* Database Service Name

\* MongoDB Connection Username  
 

Username for MongoDB user that will be used for accessing the database.

\* MongoDB Connection Password  
 

Password for the MongoDB connection user.

\* MongoDB Database Name  
 

Name of the MongoDB database accessed.

\* MongoDB Admin Password  
 

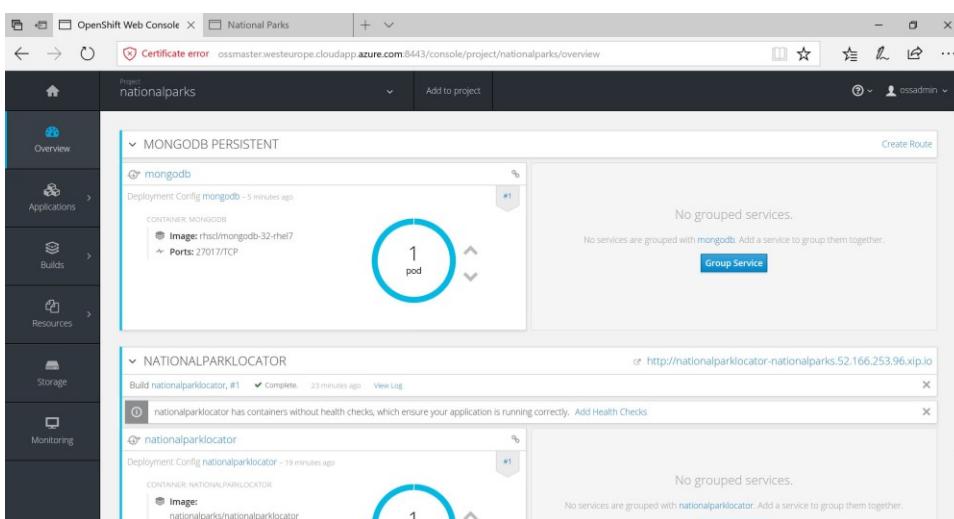
Password for the database admin user.

\* Volume Capacity  
 

Volume space available for data, e.g. 512Mi, 2Gi.

\* Version of MongoDB Image  
 

## 9. The graphical portal should show two applications in our project



10. From the left menu in the web console, click on storage and verify that OpenShift created a persistent storage volume using Azure storage.
11. Change the deployment configuration of the front-end application to include the environment variables required to access the database

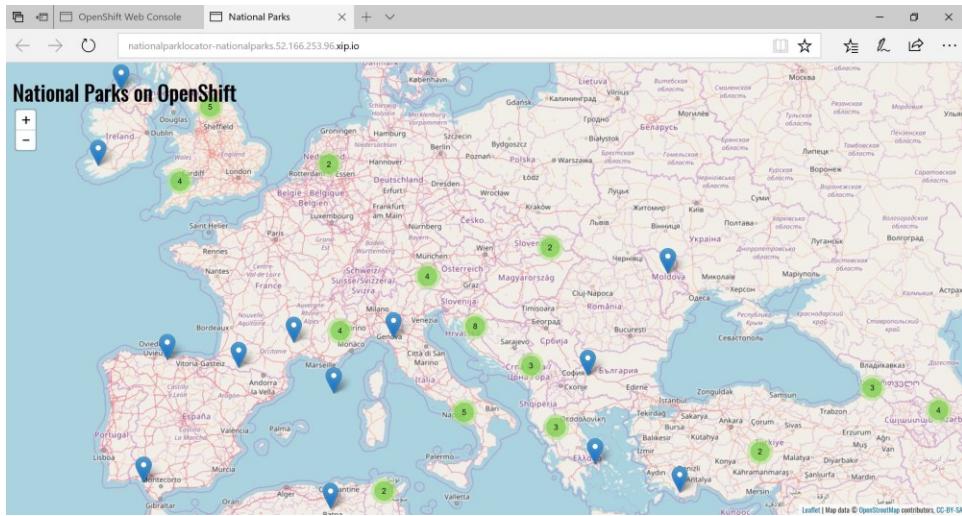
```
ossadmin@oss-bastion:~$ oc env dc nationalparklocator -e MONGODB_USER=mongodb -e MONGODB_PASSWORD=mongodb -e MONGODB_DATABASE=mongodb
deploymentconfig "nationalparklocator" updated
[ossadmin@oss-bastion ~]$
```

12. verify the last modification took place by running “oc get dc nationalparklocator -o json”

```
ossadmin@oss-bastion:~  
[{"spec": {  
    "containers": [  
        {  
            "env": [  
                {  
                    "name": "MONGODB_USER",  
                    "value": "mongodb"  
                },  
                {  
                    "name": "MONGODB_PASSWORD",  
                    "value": "mongodb"  
                },  
                {  
                    "name": "MONGODB_DATABASE",  
                    "value": "mongodb"  
                }  
            ],  
            "image": "172.30.36.121:5000/nationalparks/nationalparklocator@sha256:99338ae0c6362a4fd7b87957a2403c7c93b37e50b78a8156ad402eac2fd1bbd8",  
            "imagePullPolicy": "Always",  
            "name": "nationalparklocator",  
            "ports": [  
                {"containerPort": 27017, "hostPort": 27017, "protocol": "TCP"}  
            ]  
        }  
    ]  
},  
"status": {}  
}  
]  
ossadmin@oss-bastion:~
```

13. Back to the graphical console, note the automatic migration to a new pod based on the new configuration

13. Navigate to the application end-point and verify that the parks are now showing on the map



14. Our new application became very popular, and we need to scale out our front end to two pods.

```
[ossadmin@oss-bastion:~]$ oc get dc
NAME          REVISION  DESIRED  CURRENT  TRIGGERED BY
mongodb        1          1        1        config,image(mongodb:3.2)
nationalparklocator  2          1        1        config,image(nationalparklocator:latest)
[ossadmin@oss-bastion ~]$ 
[ossadmin@oss-bastion ~]$ oc scale --replicas=2 dc/nationalparklocator
deploymentconfig "nationalparklocator" scaled
[ossadmin@oss-bastion ~]$ 
[ossadmin@oss-bastion ~]$ oc get dc
NAME          REVISION  DESIRED  CURRENT  TRIGGERED BY
mongodb        1          1        1        config,image(mongodb:3.2)
nationalparklocator  2          2        2        config,image(nationalparklocator:latest)
[ossadmin@oss-bastion ~]$
```

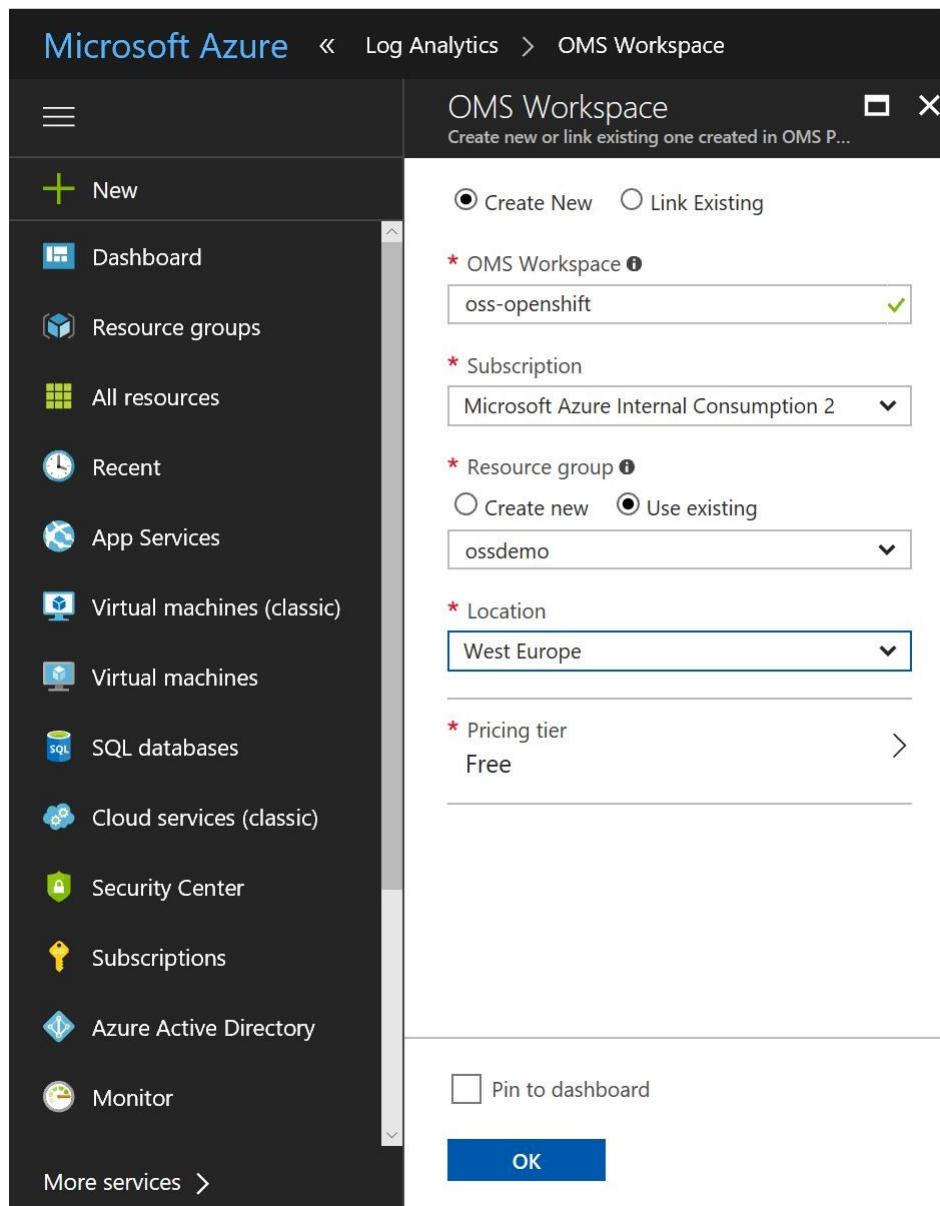
15. Now, let's test the self-healing capabilities of OpenShift by deleting one of the running pods. Because, the desired state of the replication controller is 2 pods for the application “nationalparklocator”, OpenShift will automatically and instantly trigger the deployment of a new pod.

```
[ossadmin@oss-bastion:~]$ oc get pods
NAME          READY  STATUS      RESTARTS  AGE
mongodb-1-twrzb  0/1   ContainerCreating  0       1h
nationalparklocator-1-build  0/1   Completed   0       2h
nationalparklocator-2-2cqz2  1/1   Running    0       4m
nationalparklocator-2-nz14j  1/1   Running    0       1h
[ossadmin@oss-bastion ~]$ 
[ossadmin@oss-bastion ~]$ oc delete pod nationalparklocator-2-2cqz2
pod "nationalparklocator-2-2cqz2" deleted
[ossadmin@oss-bastion ~]$ 
[ossadmin@oss-bastion ~]$ oc get pods
NAME          READY  STATUS      RESTARTS  AGE
mongodb-1-twrzb  0/1   ContainerCreating  0       1h
nationalparklocator-1-build  0/1   Completed   0       2h
nationalparklocator-2-nz14j  1/1   Running    0       1h
nationalparklocator-2-sqda4  0/1   ContainerCreating  0       9s
[ossadmin@oss-bastion ~]$ 
[ossadmin@oss-bastion ~]$ oc get pods
NAME          READY  STATUS      RESTARTS  AGE
mongodb-1-twrzb  0/1   ContainerCreating  0       1h
nationalparklocator-1-build  0/1   Completed   0       2h
nationalparklocator-2-nz14j  1/1   Running    0       1h
nationalparklocator-2-sqda4  1/1   Running    0       1m
[ossadmin@oss-bastion ~]$
```

## EXERCISE-7: MONITORING OEPNSHIFT WITH AZURE OMS

Azure Operations Management Suite (OMS) provides native support to OpenShift. In this exercise, we will walk through the steps of configuring OpenShift to export monitoring metrics directly to OMS

1. From the Azure portal create a new OMS workspace. Open the OMS portal and note the workspace id and one of the primary keys.



The top screenshot shows the Microsoft Azure portal interface. On the left, the navigation menu includes options like Dashboard, Resource groups, All resources, Recent, App Services, Virtual machines (classic), Virtual machines, SQL databases, Cloud services (classic), Security Center, Subscriptions, Azure Active Directory, and Monitor. The main area is titled 'oss-openshift Log Analytics' and shows a search bar and a 'Report a bug' button. A banner at the top right says 'Customers have already upgraded 1000s workspaces to the new and improved platform and are enjoying the benefits. Don't wait for automatic upgrades. Start planning, team more and upgrade now.' Below the banner, there are sections for 'Essentials' (Resource group (change) ossidemo, Status Active, Location West Europe, Subscription name (change) Microsoft Azure Internal Consumption 2, Subscription ID [REDACTED]), 'Management' (Overview, Log Search, OMS Portal, View Designer), and 'Pricing tier' (Free OSS-OPENSHIFT). The bottom screenshot shows the Microsoft Operations Management Suite (OMS) portal settings page for the 'oss-openshift' workspace. It has a sidebar with 'Solutions', 'Connected Sources' (selected), 'Data', 'Computer Groups', 'Accounts', 'Alerts', and 'Preview Features'. The main content area shows 'Windows Servers' (0 WINDOWS COMPUTERS CONNECTED), 'Linux Servers', 'Azure Storage', 'System Center', and 'Windows Telemetry'. It also includes sections for 'WORKSPACE ID' ([REDACTED]), 'PRIMARY KEY' ([REDACTED] Regenerate), 'SECONDARY KEY' ([REDACTED] Regenerate), and 'OMS Gateway' (Download OMS Gateway).

- From Cloud Shell ssh into the bastion host, then ssh into one of the master node and create an OpenShift project and account for OMS.

```
[ossadmin@oss-bastion ~]$ ssh oss-master-0
[ossadmin@oss-master-0 ~]$ oadm new-project omslogging --node-selector='zone=default'
[ossadmin@oss-master-0 ~]$ oc project omslogging
[ossadmin@oss-master-0 ~]$ oc create serviceaccount omsagent
[ossadmin@oss-master-0 ~]$ oadm policy add-cluster-role-to-user cluster-reader system:serviceaccount:omslogging:omsagent
[ossadmin@oss-master-0 ~]$ oadm policy add-scc-to-user privileged system:serviceaccount:omslogging:omsagent
```

- Use wget to download the ocp-\* files from <https://github.com/Microsoft/OMS-docker/tree/master/OpenShift>
- Make the file "secretgen.sh" executable. Run it, and provide our workspace id and key.

```
[ossadmin@oss-master-0 ~]$ chmod +x secretgen.sh
[ossadmin@oss-master-0 ~]$ ./secretgen.sh
```

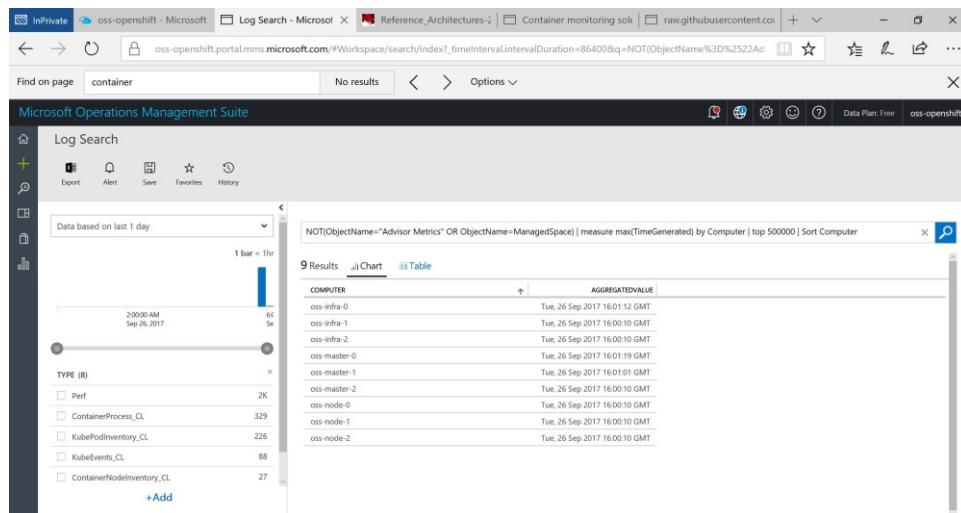
4. Create an OMS daemon set. A DaemonSet ensures that all the openshift cluster nodes run a copy of the oms pod.

```
[ossadmin@oss-master-0 ~]$ oc create -f ocp-secret.yaml
```

5. Validate that the daemon set is working properly

```
[ossadmin@oss-master-0 ~]$ oc get daemonset
NAME      DESIRED  CURRENT  READY   NODE-SELECTOR   AGE
oms       9         9         6        zone=default   1m
[ossadmin@oss-master-0 ~]$ oc describe daemonset oms
Name:           oms
Image(s):      microsoft/oms
Selector:      name=omsagent
Node-Selector: zone=default
Labels:         agentVersion=1.4.0-45
                dockerProviderVersion=10.0.0-25
                name=omsagent
Desired Number of Nodes Scheduled: 9
Current Number of Nodes Scheduled: 9
Number of Nodes Mischeduled: 0
Pods Status:   6 Running / 3 Waiting / 0 Succeeded / 0 Failed
Events:
FirstSeen  LastSeen  Count  From            SubObjectPath  Type    Reason          Message
-----  -----  -----  -----  -----  -----  -----  -----
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-sb5c1
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-2gxk0
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-krqsl
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-zmhj8
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-t18kn
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-c8fvj
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-g1pk1
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-rft2d
1m        1m        1     {daemon-set}        Normal  SuccessfulCreate  Created pod: oms-mr6nz
[ossadmin@oss-master-0 ~]$
```

6. Back to OMS portal, you will find that there are new data sources exporting metrics.



7. Create your custom dashboard and start exploring the data exported by OpenShift under different visualization formats

**All Computers with their most recent data**  
9

Computer	Last Seen
oss-infra-0	9/26/2017 6:01:12,110 PM
oss-infra-1	9/26/2017 6:00:10,801 PM
oss-infra-2	9/26/2017 6:00:10,803 PM

**Distribution of data Types**  
8

Type	Count
Perf	1,88
ContainerProcess_CL	329
KubePodInventory_CL	226

**All Events**  
0

Data based on last 1 day  
1 bar = 1hr  
2:00:00 AM Sep 26, 2017

**TYPE (8)**

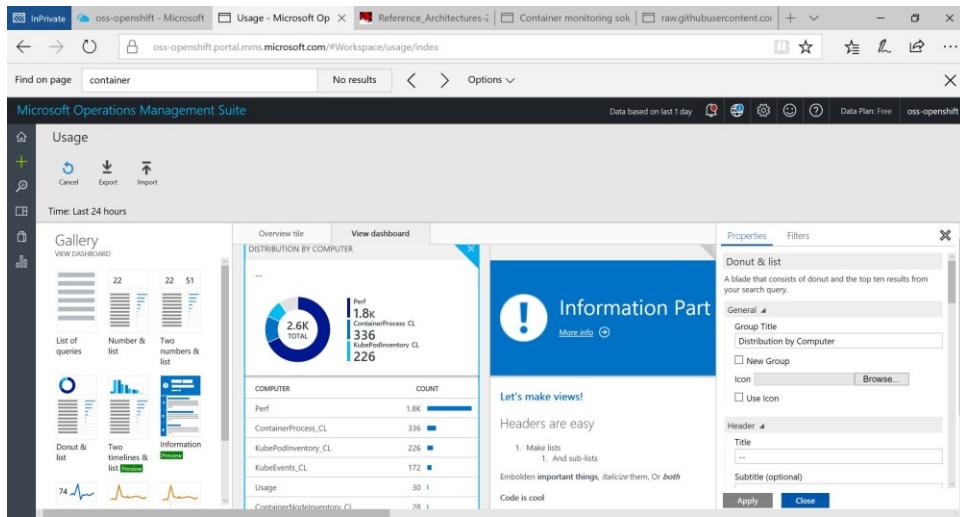
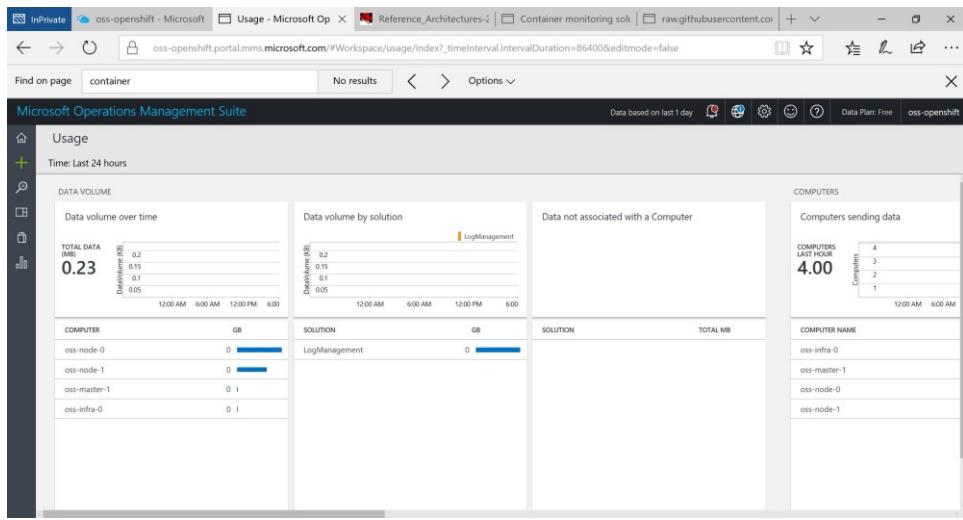
- Perf
- ContainerProcess\_CL
- KubePodInventory\_CL
- KubeEvents\_CL
- ContainerNodeInventory\_CL

+Add

\* | Measure count() by Type

8 Results

TYPE	AGGREGATEDVALUE
Perf	1,750
ContainerProcess_CL	336
KubePodInventory_CL	226
KubeEvents_CL	88
ContainerNodeInventory_CL	28
Syslog	11
Usage	7
Heartbeat	2



## EXERCISE-8: RED HAT CLOUD FORMS ON AZURE

Red Hat Cloud Forms is an open source cloud management solution providing a unified and consistent set of management capabilities across:

- Virtualization platforms like Red Hat Virtualization, VMware vCenter, and Microsoft Hyper-V.
- Private cloud platforms based on OpenStack.
- Public cloud platforms like Microsoft Azure and Amazon Web Services.
- Red Hat OpenShift

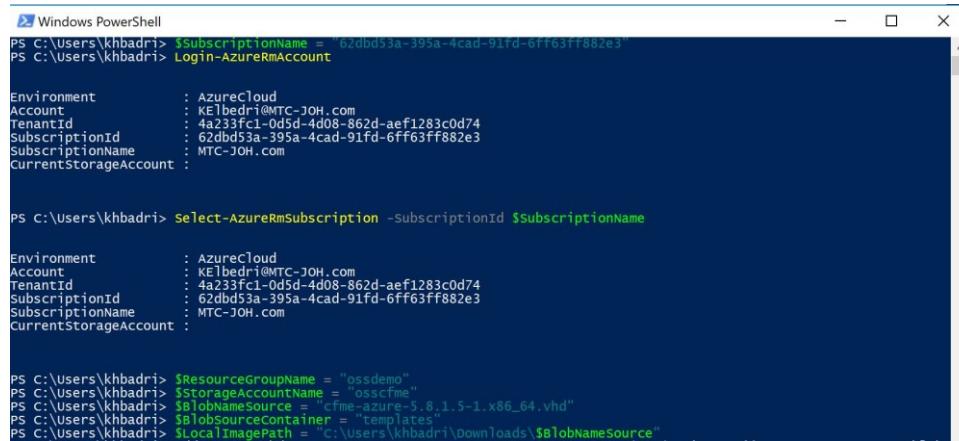
CloudForms can see and manage both the guest and host systems, allowing management of workloads and infrastructure within the same system.

CloudForms is a first-class citizen on Azure and provides a solid hybrid cloud management solution. During this exercise we will walk through the installation and configuration of CloudForms on Azure and the integration with the OpenShift environment created in the previous steps. In this section we will leverage another powerful open source SDK to perform the steps: Powershell.

1. Login to Red Hat network portal and download the latest cloudfoms vhd image for Azure. You will need an active Red Hat subscription.  
(<https://www.redhat.com/en/technologies/management/cloudfoms>)
2. We will need to round up the size of the vhd image to a multiple number of MB, otherwise the provisioning will fail in the next steps. We can use powershell to perform this operation

```
> Resize-VHD -Path $LocalImagePath -SizeBytes 32770MB
```

3. Configure some variables to use during the deployment. Change \$BlobNameSource and \$LocalImagePath to reflect your environment.



```
PS C:\Users\kbadri> $SubscriptionName = "62dbd3a-395a-4cad-91fd-6ff63ff882e3"
PS C:\Users\kbadri> Login-AzureRmAccount

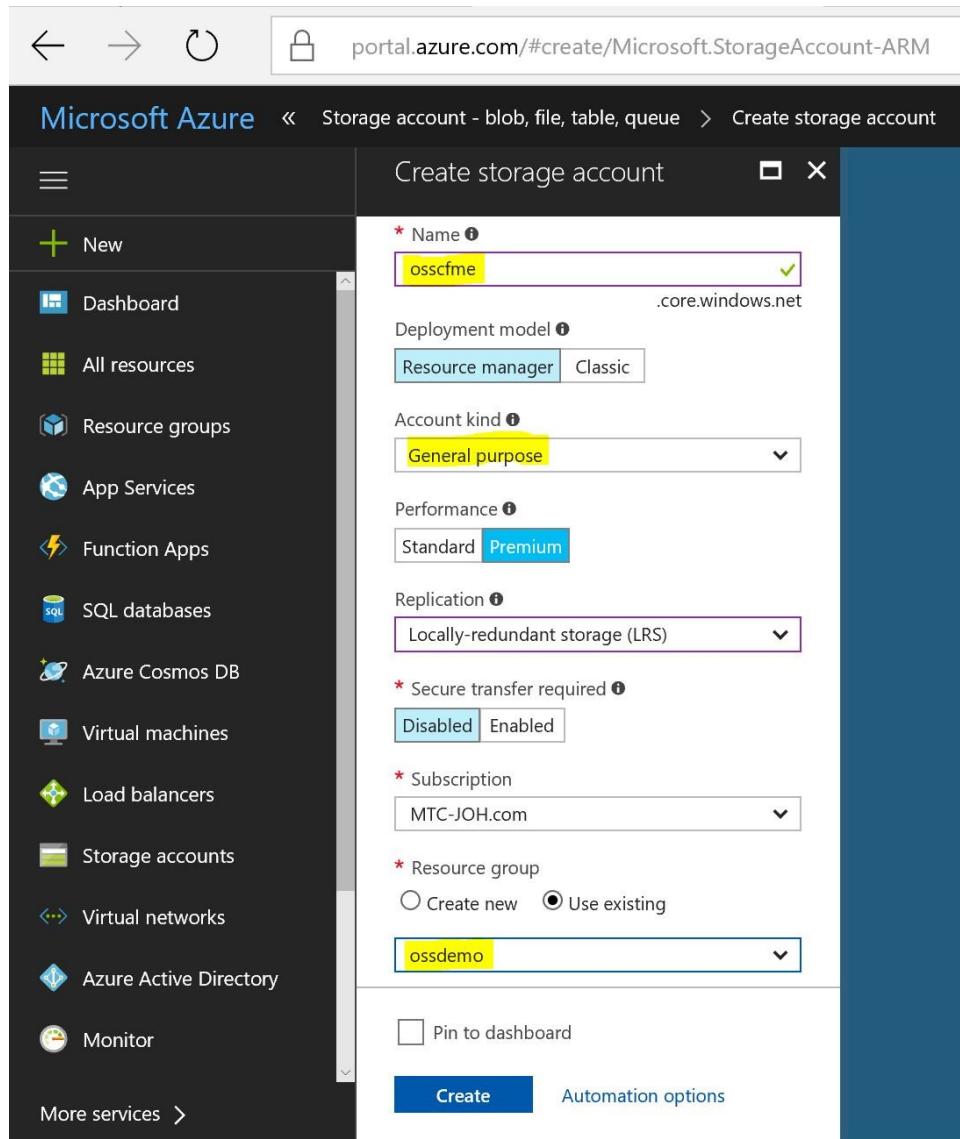
Environment      : Azurecloud
Account         : KElbedri@MTC-JOH.com
TenantId        : 4a233fc1-0d5d-4d08-862d-aef1283c0d74
SubscriptionId  : 62dbd3a-395a-4cad-91fd-6ff63ff882e3
SubscriptionName : MTC-JOH.com
CurrentStorageAccount : 

PS C:\Users\kbadri> Select-AzureRmSubscription -SubscriptionId $SubscriptionName

Environment      : Azurecloud
Account         : KElbedri@MTC-JOH.com
TenantId        : 4a233fc1-0d5d-4d08-862d-aef1283c0d74
SubscriptionId  : 62dbd3a-395a-4cad-91fd-6ff63ff882e3
SubscriptionName : MTC-JOH.com
CurrentStorageAccount : 

PS C:\Users\kbadri> $resourceGroupName = "osscdemo"
PS C:\Users\kbadri> $storageAccountName = "osscfme"
PS C:\Users\kbadri> $blobNameSource = "cfme-azure-5.8.1.5-1.x86_64.vhd"
PS C:\Users\kbadri> $blobSourceContainer = "templates"
PS C:\Users\kbadri> $localImagePath = "C:\Users\kbadri\Downloads\$blobNameSource"
```

4. Create a new storage account “osscfme” in the “osscdemo” resource group to deploy the vhd image to.



5. Upload the vhd image into the newly created storage account. This operation will take 15 minutes. Time for a break!

```
> Add-AzureRmVhd -ResourceGroupName $ResourceGroupName -  
Destination  
https://$StorageAccountName.blob.core.windows.net/$BlobSour  
ceContainer/$BlobNameSource -LocalFilePath $LocalImagePath  
-NumberOfUploaderThreads 8
```

```

PS C:\Users\kbadri> Add-AzureRmVhd -ResourceGroupName $ResourceGroupName -Destination https://$storageAccountName.blob.
FilePath $localImagePath -NumberofUploaderThreads 8
Calculating MD5 Hash
11.8% complete; Remaining Time: 00:01:21; Throughput: 2836.9Mbps
[oooooooooooooooooooooo
00:01:21 remaining.

.....
S: C:\Users\kbadri> Add-AzureRmVhd -ResourceGroupName $ResourceGroupName -Destination https://$storageAccountName.blob.core.windows.net
FilePath $localImagePath -NumberofUploaderThreads 8
MD5 hash is being calculated for the file C:\users\kbadri\Downloads\cfme-azure-5.8.1.5-1.x86_64.vhd.
MD5 hash calculation is completed.
Lapsed time for the operation is 00:01:31
Creating new page blob of size 34361836032...
Lapsed time for upload: 00:49:22
localFilePath          DestinationUri
-----          -----
:\Users\kbadri\Downloads\cfme-azure-5.8.1.5-1.x86_64.vhd https://osscfme.blob.core.windows.net/templates/cfme-azure-5.8.1.5-1.x86_64.vhd

S: C:\Users\kbadri>

```

## 6. Customize the Azure environment and create the CloudForms vm

```

$BlobNameDest = "cfme-azure-5.8.1.5-1.x86_64.vhd"
$BlobDestinationContainer = "vhds"
$VMName = "cfme-5.8"
$DeploySize= "Standard_DS4_V2"
$vmUserName = "ossadmin"

$InterfaceName = "cfmenic"
$VNetName = "openshiftvnet"
$PublicIPName = "cfme-public-ip"

$SSHKey = "ssh-rsa
AAAAB3NzaC1yc2EAAAQABAAQDQOd6tNwPPYBQ+wveI+dmBmdpmaBB
87qOG/dHe/ZEFJIXLDRILytVh2kevgeXn/SzbqL3DJ4qQWVGmsaZwELhlQK
dcc/Aybwn9tQ94H2WgPQ79RnRd8BRgdu5sVWTpyGZc5OFdAyFvkIftiasH
Ag3jb7u6oJ7f3HCH4tax/sbdqhSkTnivH4Uxq0Vx1DQQt1z4WfbCYxmG9cL
c2zNzqc6/d7Y/g33iW94FZ5CCPFUoY+HdyOSu5cy/rWtreCskWQZwNR8xkv
DOKI1c2bnBTCosN79FMzZYyiocvMIJbtE9KYH9G49G6p2tXDByhOQVj/4aI
gRc2S5SW+l6e7VSn71 khaled@kbadri"

$StorageAccount = Get-AzureRmStorageAccount -ResourceGroup
$ResourceGroupName -Name $StorageAccountName

$SourceImageUri =
"https://$StorageAccountName.blob.core.windows.net/templates/$BlobNameSource"
$Location = $StorageAccount.Location
$OSDiskName = $VMName

```

```

# Network

$PIP = New-AzureRmPublicIpAddress -Name $PublicIPName -
ResourceGroupName $ResourceGroupName -Location $Location -
AllocationMethod Dynamic -Force

$VNet = Get-AzureRmVirtualNetwork -Name openshiftVnet -
ResourceGroupName ocpkhaled


$Interface = New-AzureRmNetworkInterface -Name
$InterfaceName -ResourceGroupName $ResourceGroupName -
Location $Location -SubnetId $VNet.Subnets[1].Id -
PublicIpAddressId $PIP.Id -Force

# Specify the VM Name and Size

$VirtualMachine = New-AzureRmVMConfig -VMName $VMName -
VMSize $DeploySize


# Add User

$cred = Get-Credential -UserName $VMUserName -Message
"Setting user credential - use blank password"

$VirtualMachine = Set-AzureRmVMOperatingSystem -VM
$VirtualMachine -Linux -ComputerName $VMName -Credential
$cred


# Add NIC

$VirtualMachine = Add-AzureRmVMNetworkInterface -VM
$VirtualMachine -Id $Interface.Id


# Add Disk

$OSDiskUri =
$StorageAccount.PrimaryEndpoints.Blob.ToString() +
$BlobDestinationContainer + "/" + $BlobNameDest


$VirtualMachine = Set-AzureRmVMOSDisk -VM $VirtualMachine -
Name $OSDiskName -VhdUri $OSDiskUri -CreateOption fromImage
-SourceImageUri $SourceImageUri -Linux


# Set SSH key

Add-AzureRmVMSShPublicKey -VM $VirtualMachine -Path
"/home/$VMUserName/.ssh/authorized_keys" -KeyData $SSHKey


# Create the VM

```

**New-AzureRmVM -ResourceGroupName \$ResourceGroupName -Location \$Location -VM \$VirtualMachine**

```

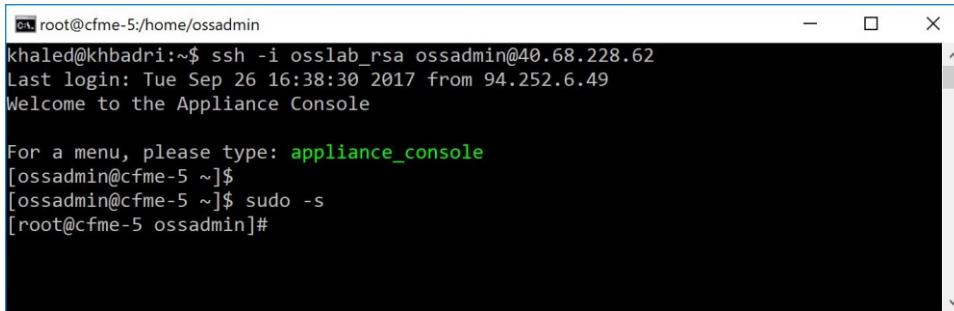
PS C:\Users\kbadri> $blobNameDest = "cime-azure5.8.1.0+2.x80_04.vhd"
PS C:\Users\kbadri> $blobDestinationContainer = "vhds"
PS C:\Users\kbadri> $vName = "cfme-5.8"
PS C:\Users\kbadri> $deploySize= "Standard_DS4_V2"
PS C:\Users\kbadri> $vmUserName = "osssadmin"
PS C:\Users\kbadri> $interfaceName = "cfmenic"
PS C:\Users\kbadri> $vNetName = "openshiftvnet"
PS C:\Users\kbadri> $publicIPName = "cfme-public-ip"
PS C:\Users\kbadri> $storageKey = "rNzCgY2EAAAADQABAAQDQOD6tNwPPVQ+wveI+dmBmpmBB87qOG/dhe/ZEFJIXLDRILytvh2kei
.../vmtbres4kyu9c496p...rcord...v1...ftiastAg31b7u6o1773H04tax/sbdghsk7n1vh4ux0vxdQO1z4wfbcyxmc9clc22nzqc6/d7Y/g331w94F25CPfUoy+/
PS C:\Users\kbadri> $storageAccount = Get-AzureRmStorageAccount -ResourceGroup $ResourceGroupName -Name $StorageAccountName
PS C:\Users\kbadri> $sourceImageUri = "https://$storageAccountName.blob.core.windows.net/templates/$blobNameSource"
PS C:\Users\kbadri> $location = $storageAccount.Location
PS C:\Users\kbadri> $osDiskName = $VMName
PS C:\Users\kbadri>
<
PS C:\Users\kbadri> $ip = New-AzureRmPublicIpAddress -Name $publicIPName -ResourceGroupName $ResourceGroupName -Location $location -AllocationMethod Dynamic -Force
WARNING: The output object type of this cmdlet will be modified in a future release.
> PS C:\Users\kbadri>
PS C:\Users\kbadri> $vnet = Get-AzureRmVirtualNetwork -Name openshiftvnet -ResourceGroupName ossdemo
PS C:\Users\kbadri> $interFace = New-AzureRmNetworkInterface -Name $interfaceName -ResourceGroupName $ResourceGroupName -Location $location -SubnetId $vnet.Subnets[1].Id -PublicIp
addressId $ip.Id -Force
WARNING: The output object type of this cmdlet will be modified in a future release.
> PS C:\Users\kbadri>
PS C:\Users\kbadri> $vm = Set-AzureRmVmConfig -VMName $vName -DeploymentSize $deploySize
PS C:\Users\kbadri> $cred = Get-Credential -UserName $vmUserName -Message "Setting user credential - use blank password"
PS C:\Users\kbadri> $virtualMachine = Set-AzureRmOperatingSystem -VM $virtualMachine -Linux -ComputerName $vName -Credential $cred
PS C:\Users\kbadri> $osDisk = $osDiskName + $storageAccount.PrimaryEndpoints.Blob.ToString() + $blobDestinationContainer + "/" + $blobNameDest
PS C:\Users\kbadri> $virtualMachine = Set-AzureRmOsDisk -VM $virtualMachine -Name $osDiskName -VhdUri $osDiskUri -CreateOption FromImage -SourceImageUri $sourceImageUri -Linux
PS C:\Users\kbadri> Add-AzureRmMSShPublicKey -VM $virtualMachine -Path "/home/$vmUserName/.ssh/authorized_keys" -KeyData $sshKey

Name          : cfme-5.8
HardwareProfile  : VmSize
NetworkProfile   : NetworkInterface
OSProfile        : ComputerName, AdminUsername, AdminPassword, LinuxConfiguration
StorageProfile    : OsDisk
NetworkingInterfaceIDs : /subscriptions/62dbd53a-395a-4cad-91fd-6ff63ff882e3/resourceGroups/ossdemo/providers/Microsoft.Network/networkInterfaces/cfmenic

PS C:\Users\kbadri> New-AzureRmVM -ResourceGroupName $ResourceGroupName -Location $location -VM $virtualMachine
WARNING: Since the VM is created using premium storage, existing standard storage account, $vsa1, is used for boot diagnostics.
RequestID IsSuccessStatusCode StatusCode ReasonPhrase
----- -----------
True      OK
OK
PS C:\Users\kbadri>
```

7. Navigate into the “cfme” vm from the Azure Portal, and note the IP address.

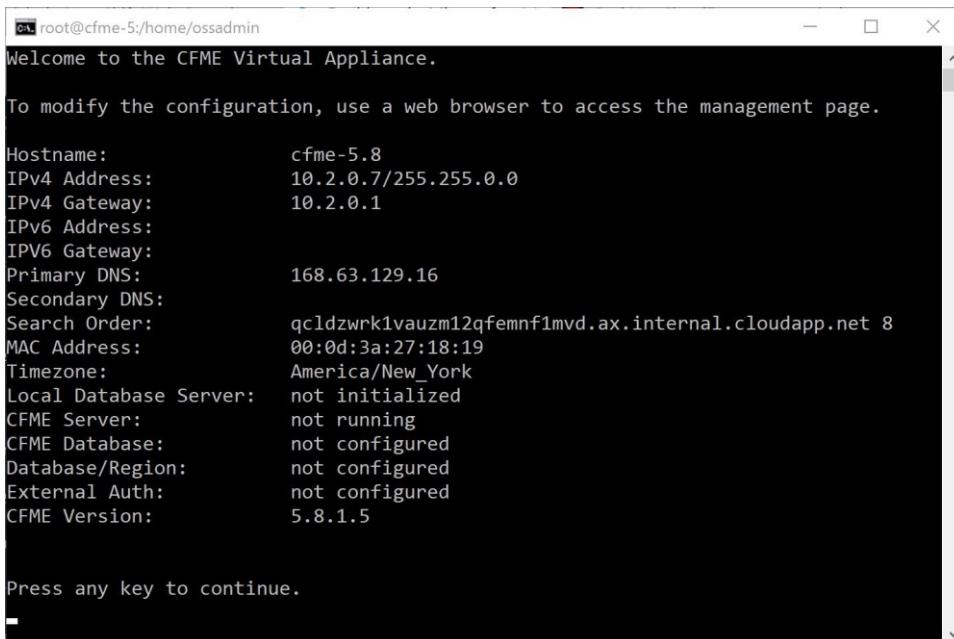
8. From the Cloud Shell, ssh into the appliance using the SSH key.



```
root@cfme-5:/home/ossadmin
khaled@khbadri:~$ ssh -i osslab_rsa ossadmin@40.68.228.62
Last login: Tue Sep 26 16:38:30 2017 from 94.252.6.49
Welcome to the Appliance Console

For a menu, please type: appliance_console
[ossadmin@cfme-5 ~]$ 
[ossadmin@cfme-5 ~]$ sudo -s
[root@cfme-5 ossadmin]#
```

9. Switch to root “sudo -s” and enter “appliance\_console” command. The Red Hat CloudForms appliance summary screen displays.



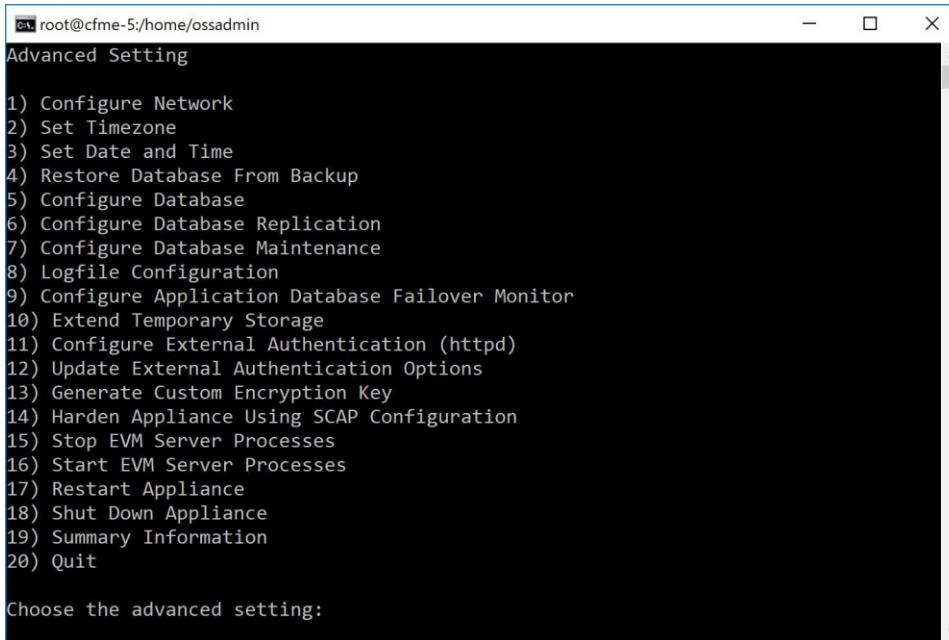
```
root@cfme-5:/home/ossadmin
Welcome to the CFME Virtual Appliance.

To modify the configuration, use a web browser to access the management page.

Hostname: cfme-5.8
IPv4 Address: 10.2.0.7/255.255.0.0
IPv4 Gateway: 10.2.0.1
IPv6 Address:
IPV6 Gateway:
Primary DNS: 168.63.129.16
Secondary DNS:
Search Order: qcldzwrk1vauzm12qfemnf1mvd.ax.internal.cloudapp.net 8
MAC Address: 00:0d:3a:27:18:19
Timezone: America/New_York
Local Database Server: not initialized
CFME Server: not running
CFME Database: not configured
Database/Region: not configured
External Auth: not configured
CFME Version: 5.8.1.5

Press any key to continue.
-
```

10. Press Enter to manually configure settings.



```
root@cfme-5:/home/ossadmin
Advanced Setting

1) Configure Network
2) Set Timezone
3) Set Date and Time
4) Restore Database From Backup
5) Configure Database
6) Configure Database Replication
7) Configure Database Maintenance
8) Logfile Configuration
9) Configure Application Database Failover Monitor
10) Extend Temporary Storage
11) Configure External Authentication (httpd)
12) Update External Authentication Options
13) Generate Custom Encryption Key
14) Harden Appliance Using SCAP Configuration
15) Stop EVM Server Processes
16) Start EVM Server Processes
17) Restart Appliance
18) Shut Down Appliance
19) Summary Information
20) Quit

Choose the advanced setting:
```

11. Press the number for the item you want to change, and press Enter. The options for your selection are displayed.
12. Follow the prompts to make the changes.
13. Press Enter to accept a setting where applicable. Configure the time zone (2) and quit (20)
14. Back to the Azure portal. We need to add a new disk for the CloudForms data base.

Microsoft Azure Virtual machines > cfme-5.8 - Disks

**OS disk**

NAME	SIZE	STORAGE ACCOUNT TYPE	ENCRYPTION
cfme-5.8	32 GiB	Premium_LRS	Not enabled

**Data disks**  
None

Microsoft Azure Virtual machines > cfme-5.8 - Disks > Attach unmanaged disk

Name: cfme-5.8-20170926-224610

Source type: New (empty disk)

Account type: Premium (SSD)

Size (GiB): 20

Storage container:

Storage blob name: cfme-5.8-20170926-224610.vhd

OK

Microsoft Azure Virtual machines > cfme-5.8 - Disks > Attach unmanaged disk

Name: cfme-5.8-20170926-224610

Source type: New (empty disk)

Account type: Premium (SSD)

Size (GiB): 20

Storage container: https://ossdemo.blob.core.windows.net/templates

Storage blob name: cfme-dbdisk.vhd

OK

15. Use the command fdisk to verify the newly added disk

```
[root@cfme-5 ossadmin]# fdisk -l | tail -n 5
Disk /dev/sdc: 21.5 GB, 21474836480 bytes, 41943040 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 4096 bytes
I/O size (minimum/optimal): 4096 bytes / 4096 bytes

[root@cfme-5 ossadmin]#
```

16. Run “appliance\_console” again, hit enter and choose (5) to configure the database
17. Choose (1) to create a key

```
root@cfme-5:/home/ossadmin
Configure Database

No encryption key found.
For migrations, copy encryption key from a hardened appliance.
For worker and multi-region setups, copy key from another appliance.
If this is your first appliance, just generate one now.

Encryption Key

1) Create key
2) Fetch key from remote machine

Choose the encryption key: |1| 1
```

18. Choose (1) to create internal database for the database location.

```
root@cfme-5:/home/ossadmin

1) Create key
2) Fetch key from remote machine

Choose the encryption key: |1| 1

Encryption key now configured.

Database Operation

1) Create Internal Database
2) Create Region in External Database
3) Join Region in External Database
4) Reset Configured Database

Choose the database operation: 1
```

19. Choose (1) for the disk we attached previously

```
root@cfme-5:/home/ossadmin
database disk

1) /dev/sdc: 20480 MB
2) Don't partition the disk

Choose the database disk: |1| 1
```

20. Select Y to configure the appliance as a database-only appliance and create and confirm a password for the database. As a result, the appliance is configured as a basic PostgreSQL server, without a user interface

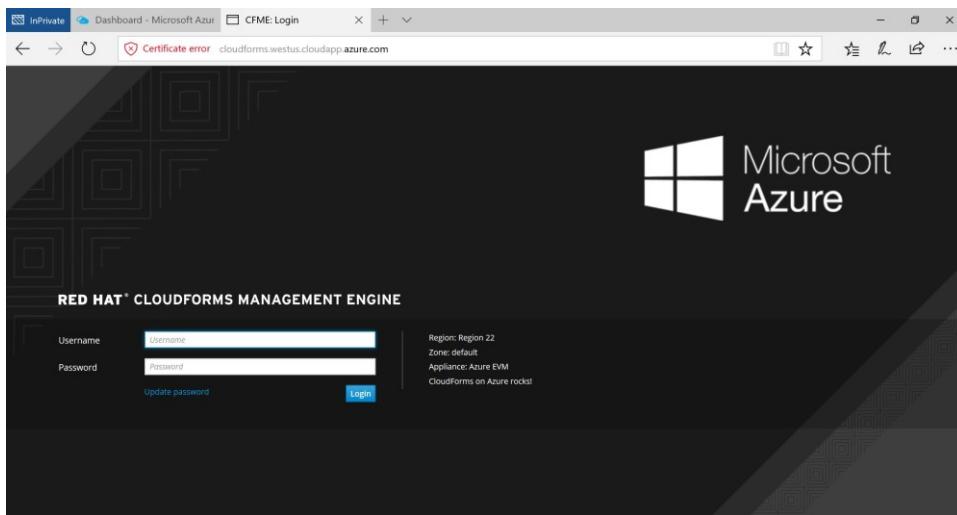
```

root@cfme-5:/home/ossadmin
NOTE:
* The CFME application will not be running.
* This is required when using highly available database deployments.
* CAUTION: This is not reversible.

? (Y/N): |N| Y
Enter the database password on localhost: *****

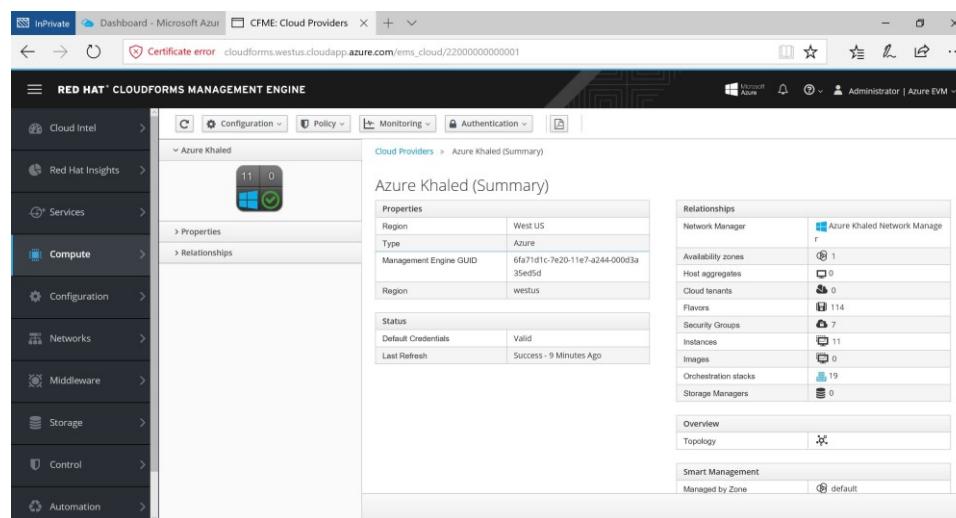
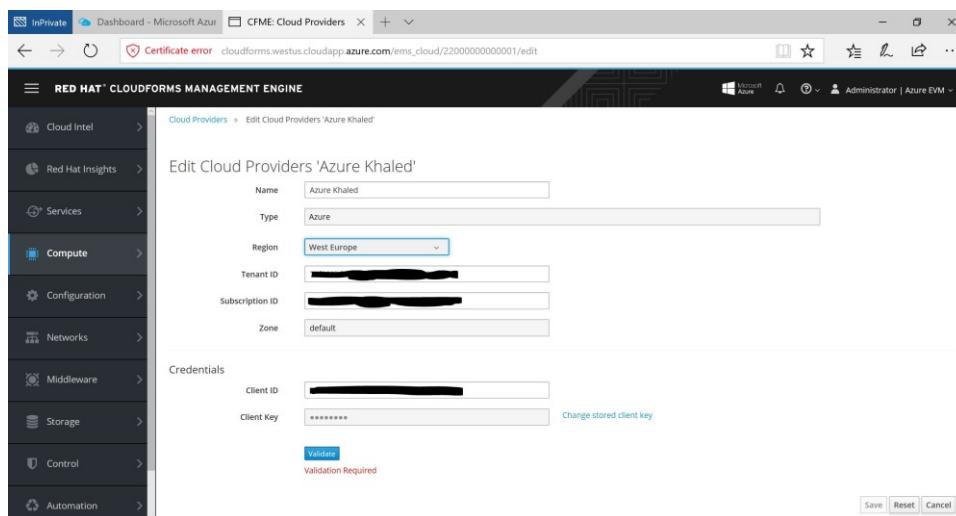
```

21. Choose (16) to start EVM processes. If it fails, choose () to restart the appliance
22. Once Red Hat CloudForms is installed, you can log in and perform administration tasks.
23. Log in to Red Hat CloudForms for the first time after installing by:
24. Navigate to the URL for the login screen. (<https://xx.xx.xx.xx> on the virtual machine instance)



25. Enter the default credentials (Username: admin | Password: smartvm) for the initial login.
26. Click Login.
27. Navigate to the URL for the login screen. (<https://xx.xx.xx.xx> on the virtual machine instance)
28. Click Update Password beneath the Username and Password text fields.
29. Enter your current Username and Password in the text fields.
30. Input a new password in the New Password field.
31. Repeat your new password in the Verify Password field.
32. Click Login.
33. To Add an Azure Cloud Provider:
  - Navigate to Compute → Clouds → Providers.
  - Click (Configuration), then click (Add a New Cloud Provider).
  - Enter a Name for the provider.

- From the Type list, select Azure.
- Select a region from the Region list. One provider will be created for the selected region.
- Enter Tenant ID.
- Enter Subscription ID.
- Enter Zone.
- In the Credentials section, enter the Client ID and Client Key; click Validate.
- Click Add.



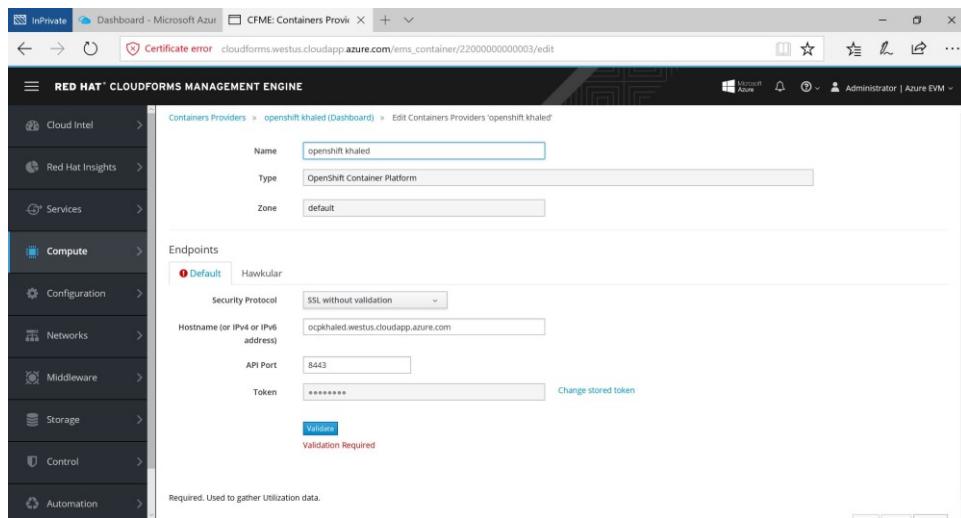
#### 34. To add OpenShift provider

- Navigate to Compute → Containers → Providers.
- Click (Configuration), then click (Add Existing Containers Provider).

- Enter a Name for the provider.
- From the Type list, select OpenShift Container Platform.
- Enter the appropriate Zone for the provider. If you do not specify a zone, it is set to default.
- Under Endpoints in the Default tab, configure the following for the OpenShift provider:
  - o Select a Security Protocol method to specify how to authenticate the provider: Choose SSL without validation
  - o Enter the Hostname or IPv4 of your OpenShift environment and keep default port
  - o Run the following to obtain the token needed to add an OpenShift Container Platform

```
# oc sa get-token -n management-infra management-admin
eyJhbGciOiJSUzI1NiI...
```

- o Enter the OpenShift management token in the Token field.
- o Enter the same token in the Confirm Token field.
- o Click Validate to confirm that Red Hat CloudForms can connect to the OpenShift Container Platform provider.



35. Explore CloudForms by navigating the left menu to get an idea on the insights and intelligence provided by the solution.

**Cloud Providers**

No filters defined.

Cloud Providers

AWS Khaled Azure Khaled

Cloud Providers

Azure Khaled (Summary)

**Properties**

Region	West US
Type	Azure
Management Engine GUID	6fa71d1c-7e20-11e7-a244-000d3a35ed5d
Region	westus

**Status**

Default Credentials	Valid
Last Refresh	Success - 9 Minutes Ago

**Relationships**

- Network Manager: Azure Khaled Network Manager (1)
- Availability zones: 1
- Host aggregates: 0
- Cloud tenants: 0
- Flavors: 114
- Security Groups: 7
- Instances: 11
- Images: 0
- Orchestration stacks: 19
- Storage Managers: 0

**Overview**

Topology

**Smart Management**

Managed by Zone: default

**Timelines**

Options

Management Events Power Activity Show Detailed Events

Apply

1 Months ending 09/26/2017

Power Activity (31) Group of 2 events Tue 09/12/2017 02:35 PM

Aug 27 Sep 03 Sep 10 Sep 17 Sep 24

Sep 05 Sep 07 Sep 09 Sep 11 Sep 13 Sep 15 Sep 17 Sep 19 Sep 21 Sep 23 Sep 25

This is a group of 2 events starting on 9/4/2017 2:48:00 PM

**CFME: Container Dashboard**

**RED HAT® CLOUDFORMS MANAGEMENT ENGINE**

Cloud Intel | Red Hat Insights | Services | Compute | Configuration | Networks | Middleware | Storage | Control | Automation

1 Providers | 9 Nodes | 40 Containers | 3 Registries | 10 Projects

39 Pods | 15 Services | 192 Images | 7 Routes

Aggregated Node Utilization

CPU: 30 Available of 32 Cores | Memory: 150 Available of 165 GB

2 Cores Used | 15 GB Used

Network Utilization Trend: 3934 kbps (Last 30 Days)

New Image Usage Trend: Images (Last 30 days)

**CFME: Container Topology**

Display Names | Refresh | Search

Replicators | Pods | Containers | Services | Routes | Nodes | VMs | Hosts

Click on the legend to show/hide entities, and double click/right click the entities in the graph to navigate to their summary pages.

**CFME: Dashboard**

Default Dashboard

Vendor and Guest OS Chart

Asset Name	Cluster Name	CPU - Usage Rate (%) (Avg)
No records found		

Updated September 06, 2017 00:00 | Next September 27, 2017 00:00

Azure | RHEL 7.3 | Linux

Top CPU Consumers (weekly)

Asset Name	Cluster Name	CPU - Usage Rate (%) (Avg)
No records found		

Updated September 06, 2017 00:00 | Next September 27, 2017 00:00

Virtual Infrastructure Platforms

No data available.

Guest OS Information

Updated September 06, 2017 00:00 | Next September 27, 2017 00:00

## END THE LAB

To end the lab, simply delete the resource group *openshiftRG* from the Azure portal or from the Azure CLI. And delete the created *webhook* from your *git* repository.

```
$ az group delete ossdemo
```

## REFERENCES

### USEFUL LINKS

[https://access.redhat.com/documentation/en-us/reference\\_architectures/2017/pdf/deploying\\_red\\_hat\\_openshift\\_container\\_platform\\_3.5\\_on\\_microsoft\\_azure/Reference\\_Architectures-2017-Deploying\\_Red\\_Hat\\_OpenShift\\_Container\\_Platform\\_3.5\\_on\\_Microsoft\\_Azure-en-US.pdf](https://access.redhat.com/documentation/en-us/reference_architectures/2017/pdf/deploying_red_hat_openshift_container_platform_3.5_on_microsoft_azure/Reference_Architectures-2017-Deploying_Red_Hat_OpenShift_Container_Platform_3.5_on_Microsoft_Azure-en-US.pdf)

<https://blogs.technet.microsoft.com/msoms/2017/08/04/container-monitoring-solution-in-red-hat-openshift/>

<https://testdrive.azure.com/#/test-drive/redhat.openshift-test-drive>

<https://github.com/Microsoft/openshift-container-platform>

<https://github.com/Microsoft/openshift-origin>

[https://access.redhat.com/documentation/en-us/red\\_hat\\_cloudfoms/4.5/html/installing\\_red\\_hat\\_cloudfoms\\_on\\_microsoft\\_azure/](https://access.redhat.com/documentation/en-us/red_hat_cloudfoms/4.5/html/installing_red_hat_cloudfoms_on_microsoft_azure/)

<http://manageiq.org/>

### REDHAT AND MICROSOFT PARTNERSHIP

<http://openness.microsoft.com/2016/04/15/microsoft-red-hat-partnership-accelerating-partner-opportunities/>

<https://www.redhat.com/en/microsoft>