

Microsoft

OPENSHIFT & CloudForms 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
CHALLENGE-1: Deploy Openshift on azure	4
CHALLENGE -2: Create and manage projects	14
CHALLENGE -3: Create and manage Applications	17
CHALLENGE -4: Configuring automated builds	22
CHALLENGE -5: Continuous deployment	24
CHALLENGE -6: Mini project: JBOSS EAP application	28
CHALLENGE -7: Monitoring oepnshift with azure oms.....	37
CHALLENGE -8: Red Hat Cloud Forms on Azure	42
End the lab	56
References	56
Useful links	56
Redhat and Microsoft partnership.....	56

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 Challenge-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 challenges cover:

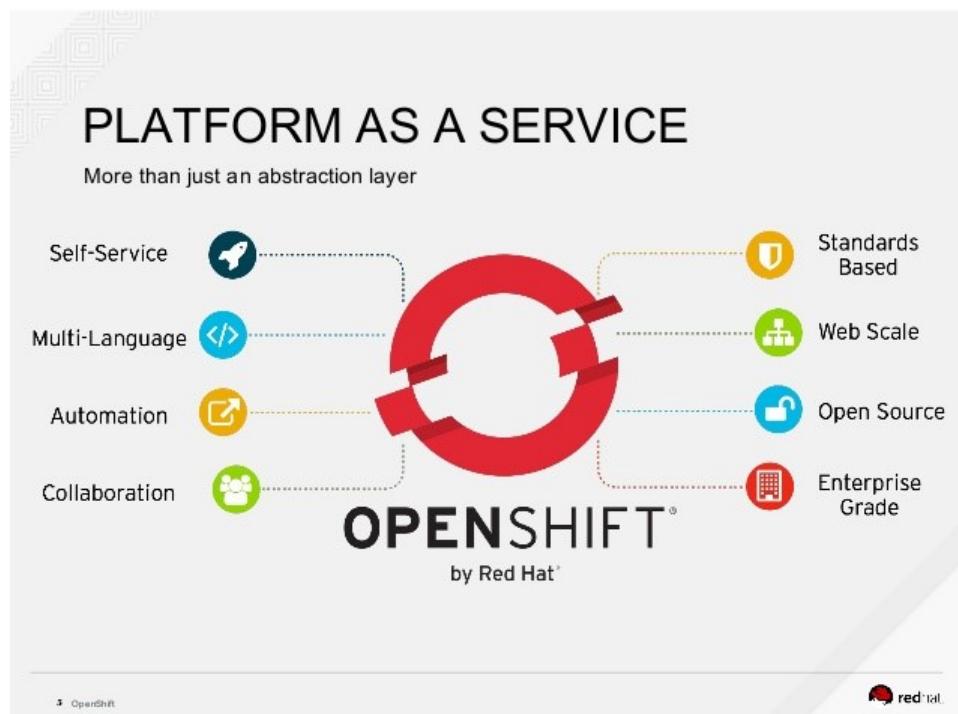
- o Introduction to OpenShift3
- o Deployment of OpenShift on Azure
- o Creating and managing OpenShift projects on azure
- o Creating and managing OpenShift applications on Azure
- o Automating builds with Linux Containers on Azure.
- o Mini project: Jboss EAP application
- o Integration of OpenShift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

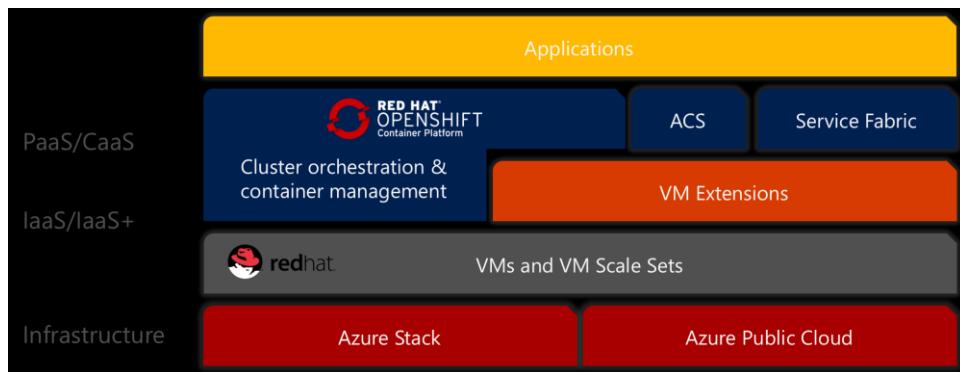
OpenShift offers multiple access modes including: developer CLI, admin CLI, web console and IDE plugins. Click2cloud is a plugin that allows Visual studio to deploy code to OpenShift, directly.



CHALLENGE-1: DEPLOY OPENSHIFT ON AZURE

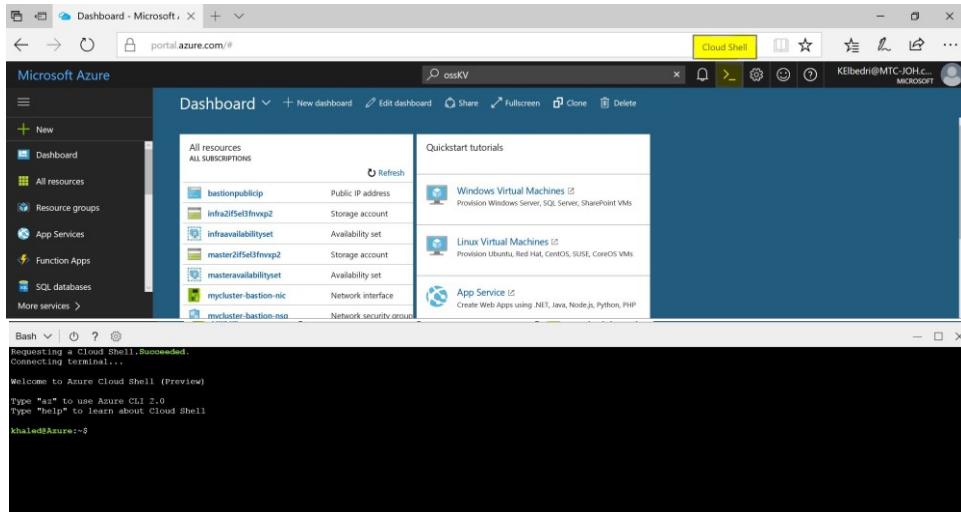
OpenShift, leverages multiple Azure services such as VM *extensions*, Azure disks, and *Key vaults*... to provide an Enterprise grade offering for customers who would like to containerize and manage their applications, without investing long time and hard effort configuring and integrating various tools.

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:
SHA256:8XyNpBzvVwM1J2idrkn8zpgQ2kgDuk0XPWbVE 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+|
+---[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. Both screenshots show the 'ossdemo - Access control (IAM)' blade for the 'ossdemo' resource group.

Screenshot 1: Initial IAM blade

- The left sidebar shows the 'Resource groups' section with 'ossdemo' selected.
- The main area shows the 'Access control (IAM)' tab selected.
- On the right, there is a table with the following columns: NAME, TYPE, ROLE, and SCOPE.
- A message at the bottom states: 'Showing a filtered set of results. Total number of role assignments: 43'.
- The table below shows 0 items.

Screenshot 2: Adding permissions

- The same interface is shown, but the 'Add' button is highlighted.
- The 'Role' dropdown is set to 'Contributor'.
- The 'Assign access to' dropdown is set to 'Azure AD user, group, or application'.
- The search bar shows 'osscloudprovider'.
- The table shows 0 items.
- A message at the bottom says: 'Selected members: No members selected. Search for and add one or more members you want to assign the role for this resource.'
- At the bottom right, there are 'Save' and 'Discard' buttons.

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
- PoolId 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 agent). If you want to deploy the lab with minimal cost, you can reduce the number of vms to one per each. Also, you can scale down 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	<input type="radio"/> Create new <input checked="" type="radio"/> 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 

TEMPLATE

 Customized template
34 resources

[Edit template](#) [Edit parameters](#) [Learn more](#)

BASICS

* Subscription	<input type="text" value="Microsoft Azure Internal Consumption 2"/>
* Resource group	<input type="radio"/> Create new <input checked="" type="radio"/> Use existing <input type="text" value="ossdemo"/>
* Location	<input type="text" value="West Europe"/>

SETTINGS

_artifacts Location 	<input type="text" value="https://raw.githubusercontent.com/Microsoft/openshift-container-platform/master/"/>
Custom Vhd Or Gallery 	<input type="text" value="gallery"/>
Custom Storage Account 	<input type="text" value="https://osdiskstorageaccount.blob.core.windows.net/"/>
* Aad Client Secret 	<input type="text" value="*****"/>
Default Sub Domain Type 	<input type="text" value="xipio"/>
Default Sub Domain 	<input type="text" value="contoso.com"/>

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.

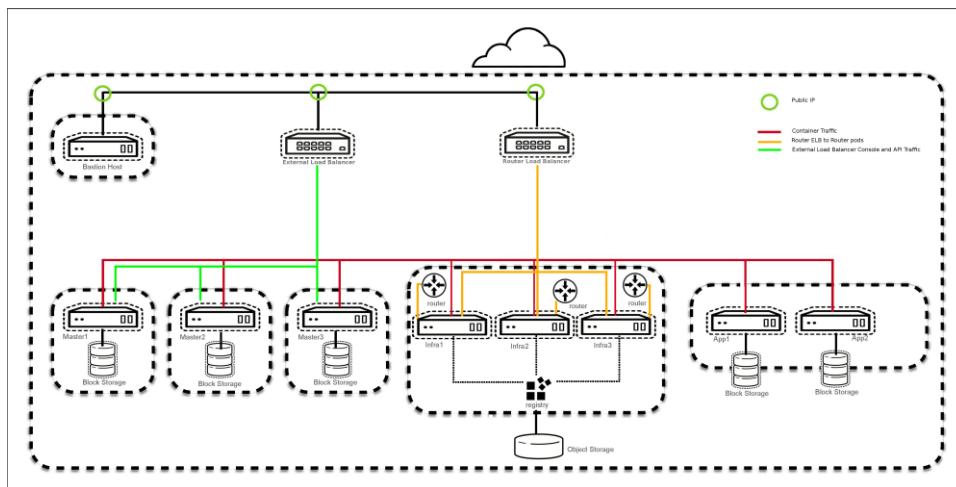
Pin to dashboard

[Purchase](#)

- From the Azure portal, go to your resource group “ossdemo”, track the progress of the deployment and make sure the deployment finishes, successfully. The process should last around 20 minutes. It is a good time to have a break.

DEPLOYMENT NAME	STATUS	TIMESTAMP	DURATION
ossdeployment	Deploying	9/25/2017 6:04:45 PM	6 seconds
Microsoft.Template	Succeeded	9/24/2017 7:10:09 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
infraVmDeployment0	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
b15ffa02-d0da-4775-9fb7-...	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.

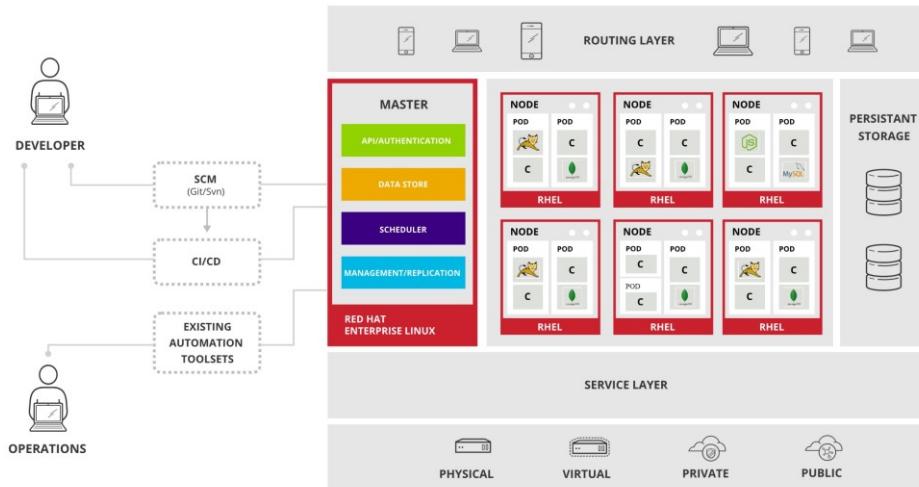


The bastion server implements mainly two distinct functions. One is that of a secure way to connect to all the nodes, and second that of the "installer" of the system. The bastion host is the only ingress point for SSH in the cluster from external entities. When connecting to the OpenShift Container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

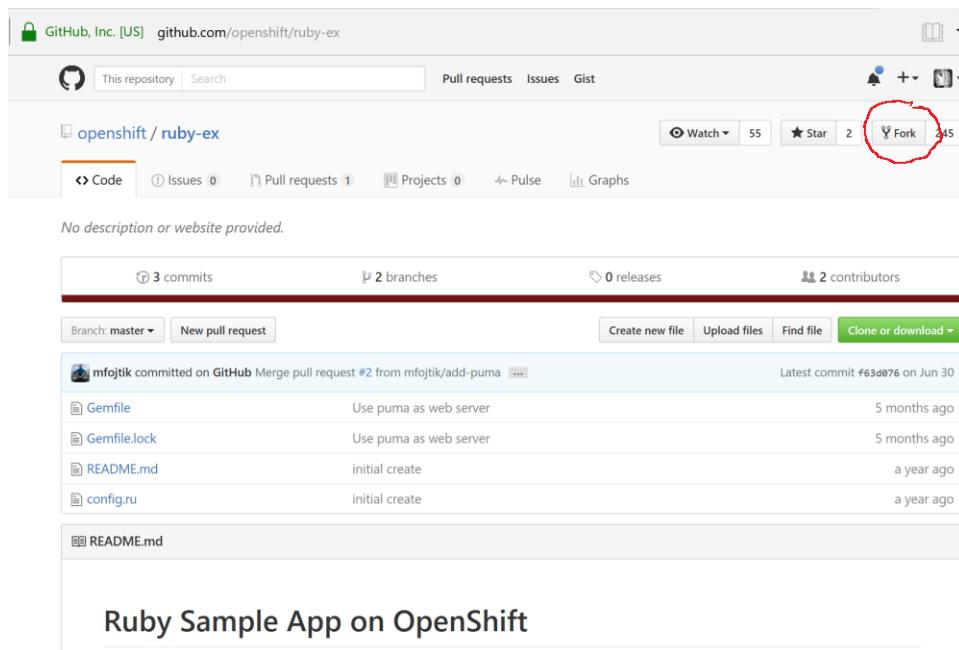


CHALLENGE -2: CREATE AND MANAGE PROJECTS

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

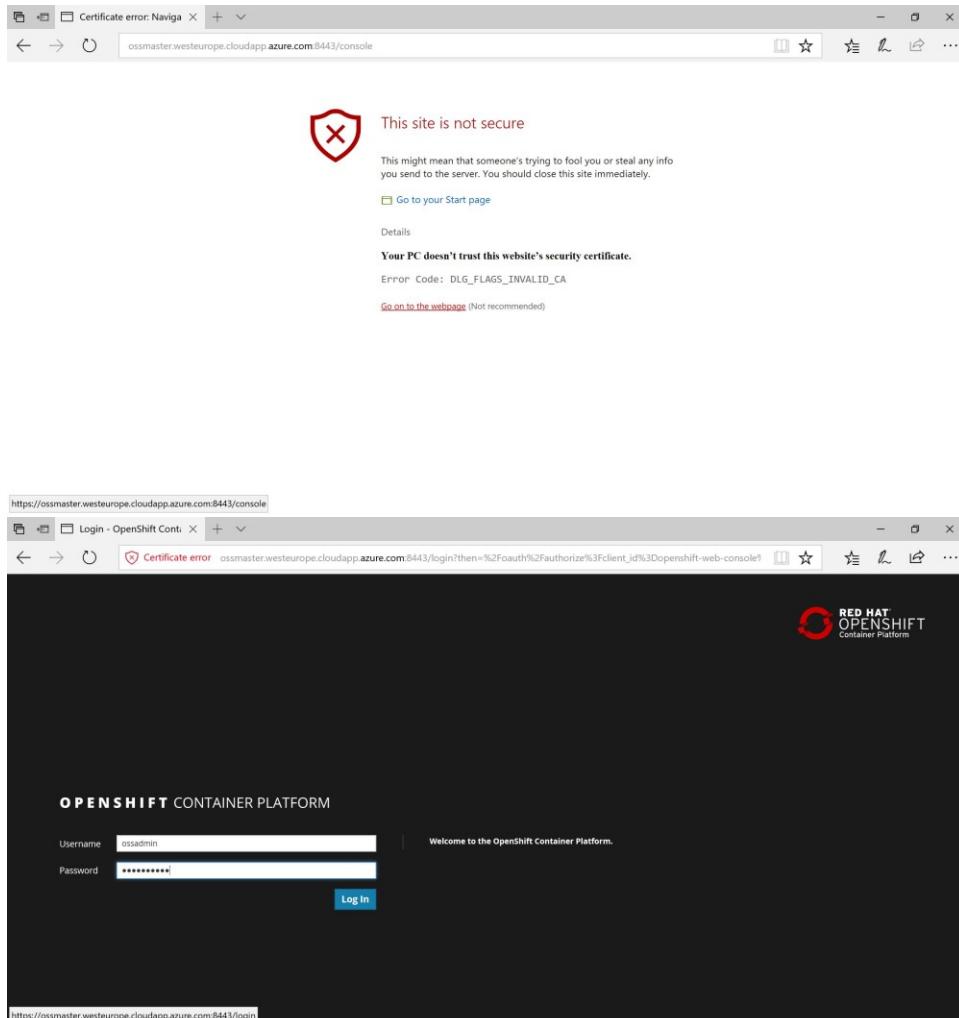
To create an *application*, you must first create a new *project*, then select an *InstantApp* template. From there, OpenShift begins the build process, and creates a new deployment.

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



The screenshot shows the GitHub repository page for 'openshift / ruby-ex'. At the top, there's a navigation bar with links for 'Pull requests', 'Issues', and 'Gist'. Below the navigation, there's a search bar and a 'Watch' button. To the right of the search bar, there are 'Star' and 'Fork' buttons; the 'Fork' button is highlighted with a red circle. Below the header, the repository name 'openshift / ruby-ex' is displayed. Underneath the repository name, there are buttons for 'Code', 'Issues 0', 'Pull requests 1', 'Projects 0', 'Pulse', and 'Graphs'. A note below the repository name says 'No description or website provided.' Below this, there are sections for '3 commits', '2 branches', '0 releases', and '2 contributors'. A commit list follows, showing entries from 'mfojtik' on Jun 30. The commits include 'Merge pull request #2 from mfojtik/add-puma', 'Gemfile', 'Gemfile.lock', 'README.md', and 'config.ru'. At the bottom of the page, there's a section titled 'Ruby Sample App on OpenShift'.

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:

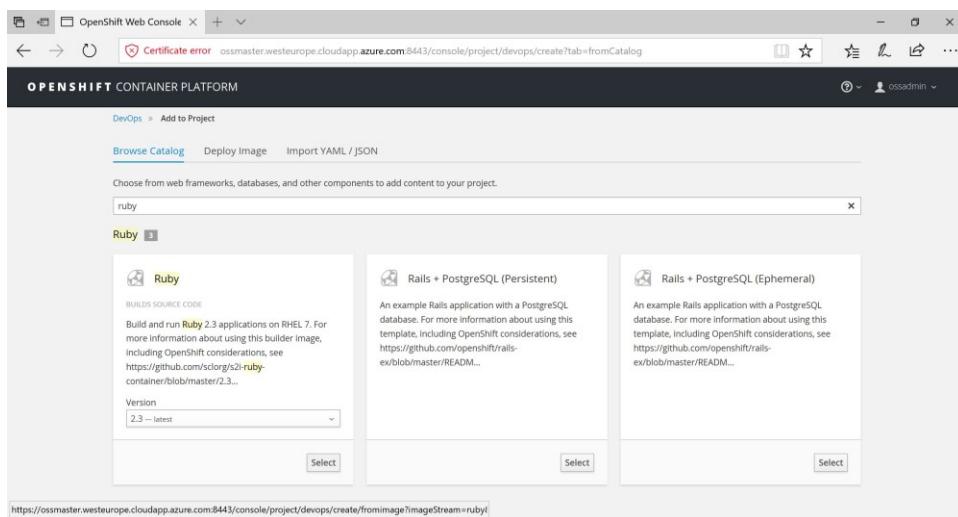
- * Name: devops (A unique name for the project.)
- Display Name: DevOps
- Description: My Web Application

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

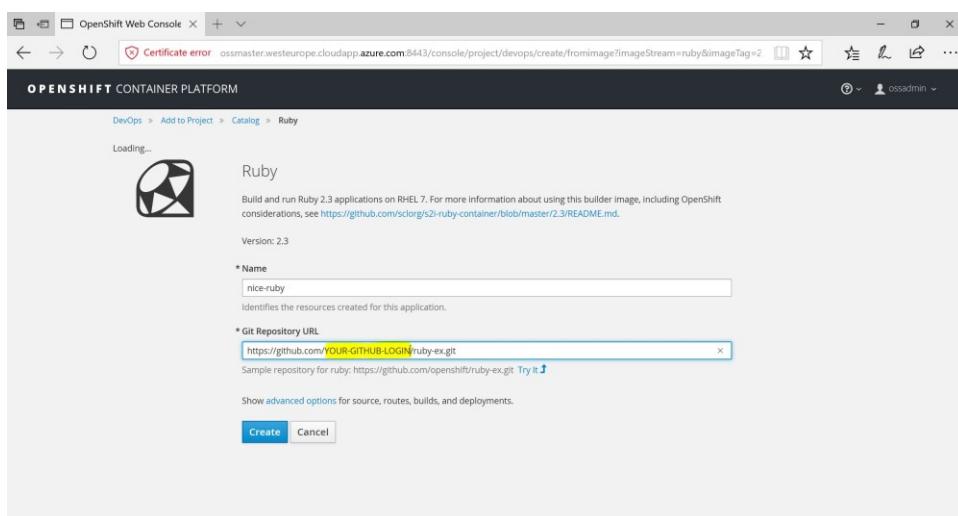
CHALLENGE -3: CREATE AND MANAGE APPLICATIONS

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.



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. The left sidebar has navigation links for Overview, Applications, Builds, Resources, Storage, and Monitoring. The main content area is titled 'Project DevOps'. It displays a message: 'Created application nice-ruby in project DevOps.' Below this, there's a section for 'NICE RUBY' with a status message: 'Build nice-ruby, #1 is running. A new deployment will be created automatically once the build completes.' A link to 'View Log' is provided. To the right, there's a note: 'nice-ruby has containers without health checks, which ensure your application is running correctly. Add Health Checks.' At the bottom, it says 'Deployment Config nice-ruby' and 'No deployments.' A note states: 'A new deployment will start automatically when an image is available for devops/nice-ruby:latest.'

The screenshot shows the logs for the 'NICE RUBY' build. The logs output the following command sequence:

```

3 Author: khale999 (khale999@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

```

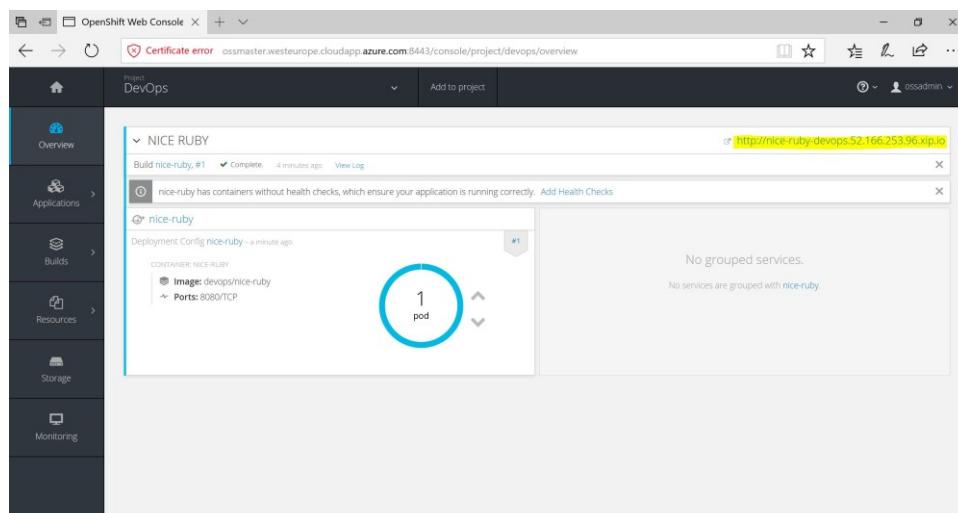
OpenShift leverages the Kubernetes concept of a pod, which is one or more containers deployed together on one host. A pod is the smallest compute unit that can be defined, deployed, and managed.

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:

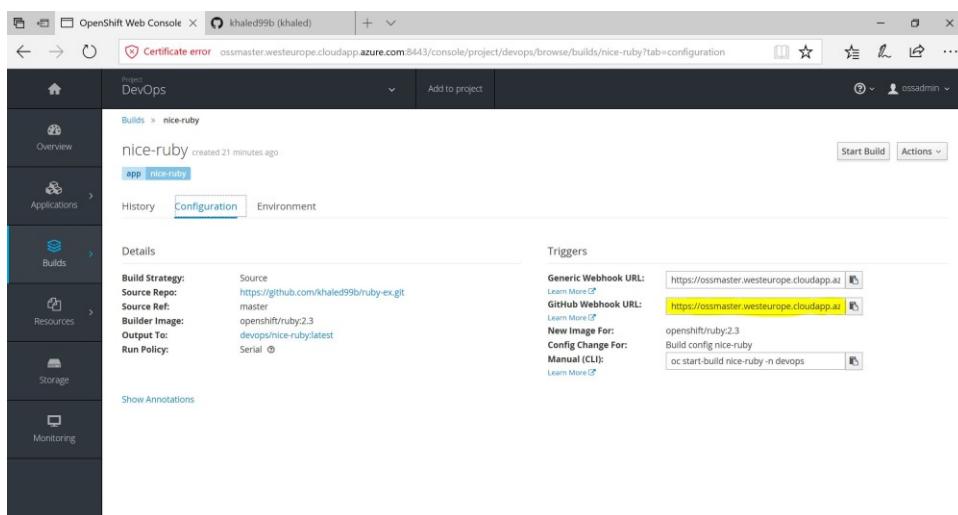
Name	Status	Containers Ready	Container Restarts	Age
nice-ruby-1-4bc8	Running	1/1	0	2 minutes
nice-ruby-1-hkxkn	Running	1/1	0	2 minutes
nice-ruby-1-48fhv	Running	1/1	0	6 minutes
nice-ruby-1-build	Completed	0/1	0	10 minutes

CHALLENGE -4: CONFIGURING AUTOMATED BUILDS

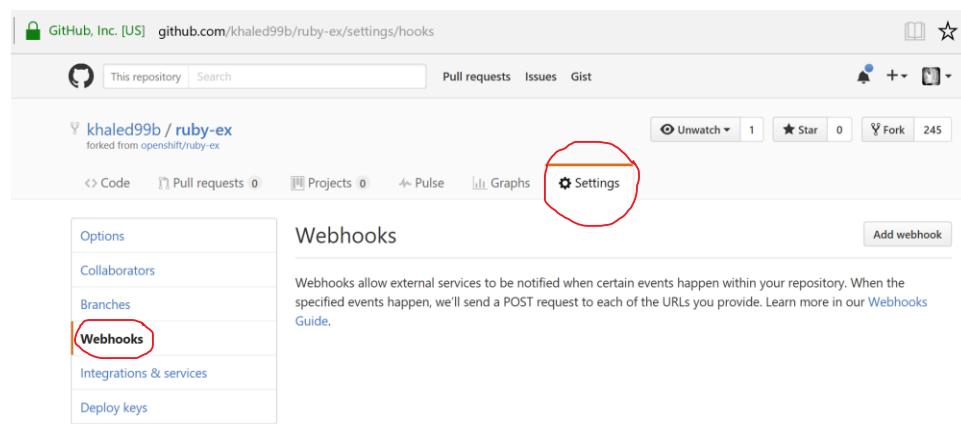
Since we forked the source code of the application from the [OpenShift GitHub repository](#), we can use a *webhook* to automatically trigger a rebuild of the application whenever code changes are pushed to the forked repository.

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.



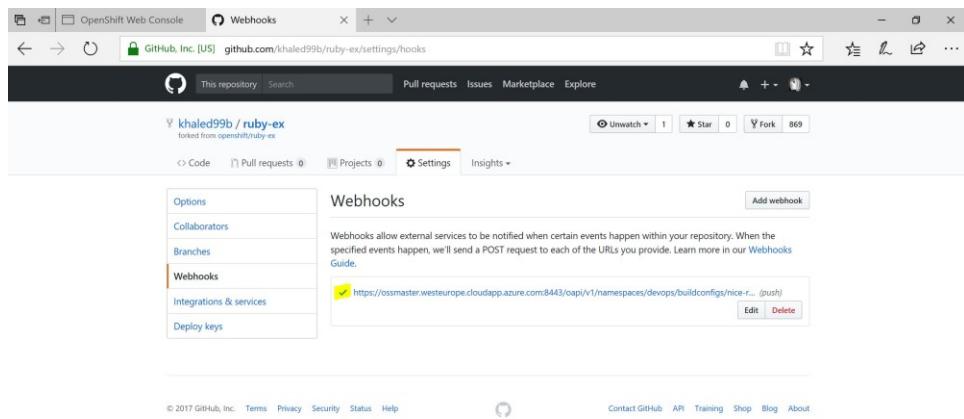
5. Navigate to your forked repository on GitHub, then click **Settings**.
6. Click **Webhooks & Services** and Click **Add webhook**.



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.

CHALLENGE -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...

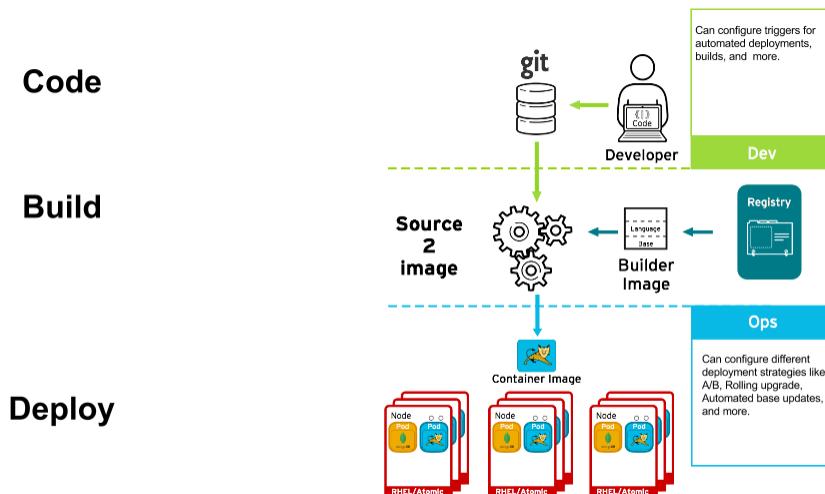


Figure 21: Continuous deployment pipeline

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 challenge by moving to step-5 and editing the file *config.ru*, directly from the web interface of *github* and then go to step-7.

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' overview page for a project named 'NICE RUBY'. In the top screenshot, a deployment named 'Build nice-ruby, #5' is shown with a status of 'Building'. A yellow box highlights this status. Below it, another deployment named 'Build nice-ruby, #4' is shown as 'Complete' with a timestamp of '3 minutes ago'. To the right, a deployment pod count of '1 pod' is shown, with an arrow pointing to '3 pods', indicating a successful deployment. In the bottom screenshot, the deployment status has changed to 'Complete' with a timestamp of '2 minutes ago'. The deployment pod count has increased from '1 pod' to '3 pods', with an arrow indicating the transition.

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

The image shows a web browser window displaying the deployed Ruby application. The title bar reads 'Welcome to your Ruby application on OpenShift POWERED BY AZURE!!'. The page content includes several sections: 'Deploying code changes' (instructions for GitHub webhooks), 'Managing your application' (link to developer guide), 'Web Console' (link to view application state), 'Command Line' (link to OpenShift CLI), 'Development Resources' (links to OpenShift documentation, GitHub, and Stack Overflow), and 'Working in your local Git repository' (instructions for cloning the repository). At the bottom, there is a terminal window showing command-line instructions for cloning the repository:

```
$ git clone <git_url> <directory_to_create>
# Within your project directory
# Commit your changes and push to OpenShift
```

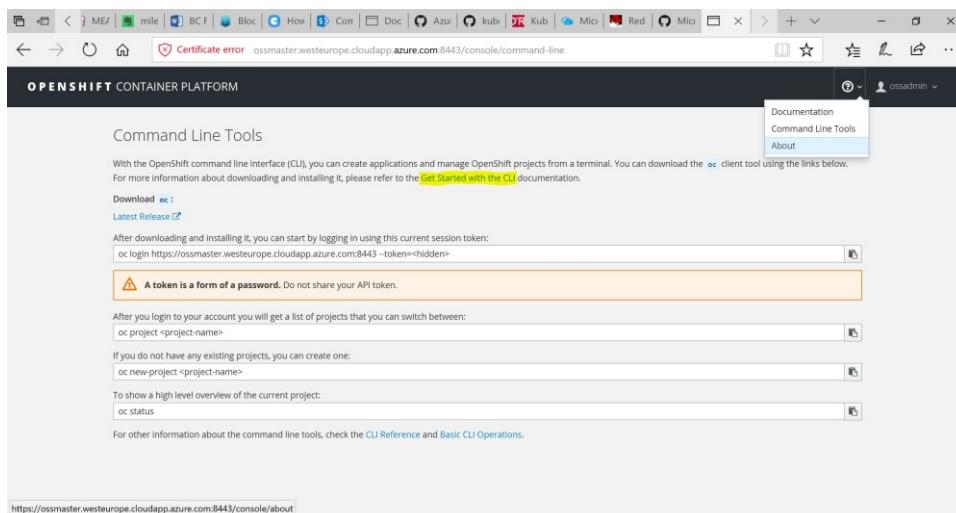
9. You may find it useful to manually rebuild an image if your *webhook* is not working, or if a build fails and you do not want to change the code before restarting the build. To manually rebuild the image based on your latest committed change to your forked repository:
 - a. Click the **Browse** tab,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

CHALLENGE -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 challenge, 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.



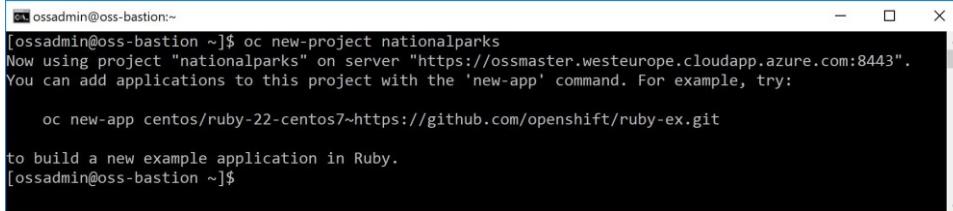
2. Use your openshift url endpoint to login to your environment from the CLI

```
ossadmin@oss-bastion:~$ [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”

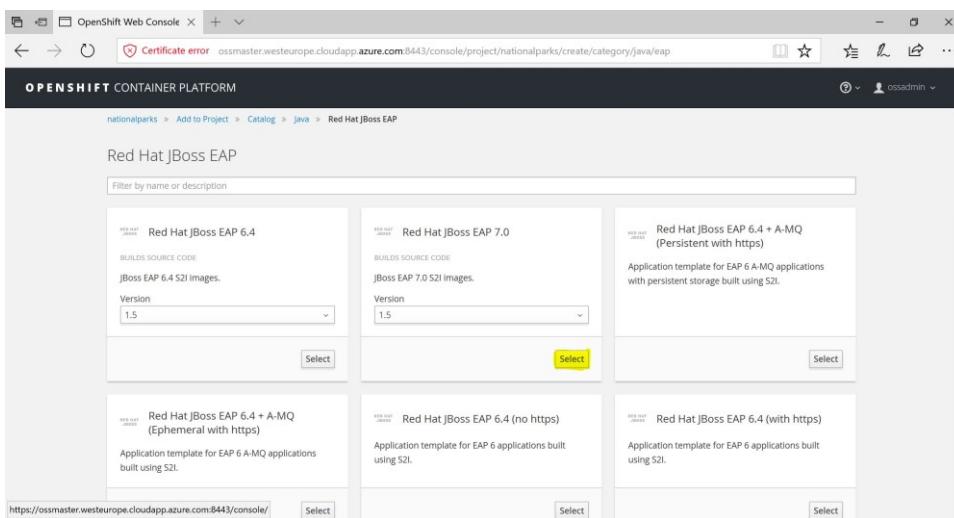


```
[ossadmin@oss-bastion:~]$ oc new-project nationalparks
Now using project "nationalparks" on server "https://ossmaster.westeurope.cloudapp.azure.com:8443".
You can add applications to this project with the 'new-app' command. For example, try:

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

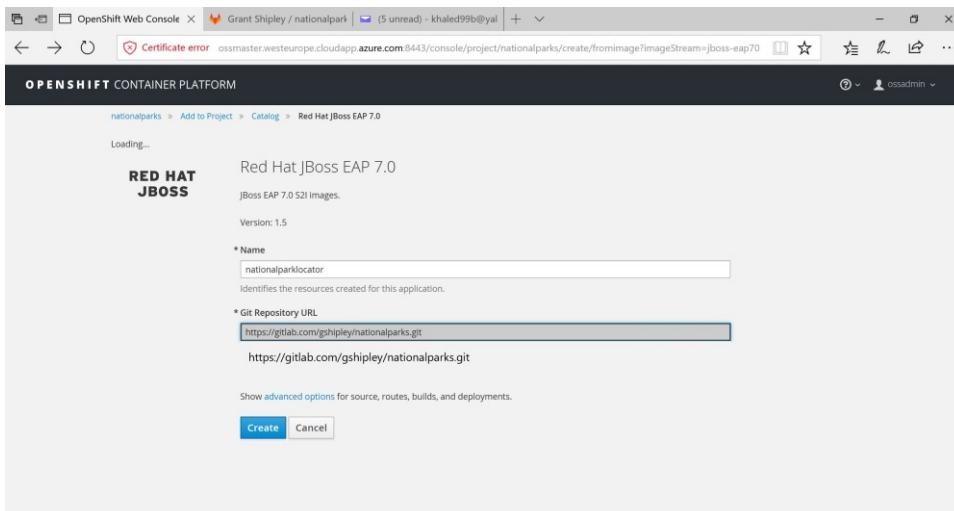
to build a new example application in Ruby.
[ossadmin@oss-bastion ~]$
```

4. 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 breadcrumb navigation shows: nationalparks > Add to Project > Catalog > Java > Red Hat JBoss EAP. The main content area displays several application templates under the heading "Red Hat JBoss EAP". The templates listed are:

- 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) (Builds source code, 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) (Builds source code, Application template for EAP 6 A-MQ applications built using S2I, Select button)
- Red Hat JBoss EAP 6.4 (no https) (Builds source code, Application template for EAP 6 applications built using S2I, Select button)
- Red Hat JBoss EAP 6.4 (with https) (Builds source code, 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 breadcrumb navigation shows: nationalparks > Add to Project > Catalog > Red Hat JBoss EAP 7.0. The main content area displays a form for creating a new application:

RED HAT JBOSS

Red Hat JBoss EAP 7.0

JBoss EAP 7.0 S2I Images.

Version: 1.5

* Name: Identifies the resources created for this application.

* Git Repository URL: https://gitlab.com/gshipley/nationalparks.git

Show advanced options for source, routes, builds, and deployments.

5. 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:

```
[ossadmin@oss-bastion ~]$ oc get projects
NAME          DISPLAY NAME  STATUS
default        default     Active
devops         DevOps      Active
kube-system   kube-system Active
logging        logging     Active
management-infra management-infra Active
nationalparks nationalparks Active
openshift      openshift   Active
openshift-infra openshift-infra 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 ~]$ 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 the OpenShift Web Console interface. On the left, a sidebar lists navigation options: Overview, Applications, Builds, Resources, Storage, and Monitoring. The main content area displays the 'natiонаlparks' project. Under the 'natiонаlparks' heading, there is a section for the 'NATIONALPARKLOCATOR' application. This section includes a deployment log entry for 'Build nationalparklocator, #1' which is 'Complete' and occurred '8 minutes ago'. A 'View Log' button is also present. Below the log, a message states: 'nationalparklocator has containers without health checks, which ensure your application is running correctly. Add Health Checks'. A deployment configuration for 'nationalparklocator' is shown, indicating one pod running the 'nationalparks/nationalparklocator' image on ports 8080/TCP, 8443/TCP, and 8778/TCP. A circled '1 pod' label is overlaid on the deployment configuration section. To the right, a note says 'No grouped services.' and 'No services are grouped with nationalparklocator.'

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

The screenshot shows the OpenShift Web Console interface. The left sidebar has a dark theme with icons for Overview, Applications, Builds, Resources, Storage, and Monitoring. The main area displays the 'nationalparks' project. Under 'Applications', there are two entries: 'MONGODB PERSISTENT' and 'NATIONALPARKLOCATOR'. Each entry has a deployment config and a single pod icon. The 'MONGODB PERSISTENT' entry also includes a 'Create Route' button.

10. From the left menu in the web console, click on storage and verify that OpenShift created a persistent storage volume using Azure storage.

Name	Status	Capacity	Access Modes	Age
mongodb using storage class generic	Bound to volume pvc-1019b324-a2b2-11e7-96f4-000d3a27e831	1 GiB	RWO (Read-Write-Once)	3 minutes

11. Change the deployment configuration of the front-end application to include the environment variables required to access the database

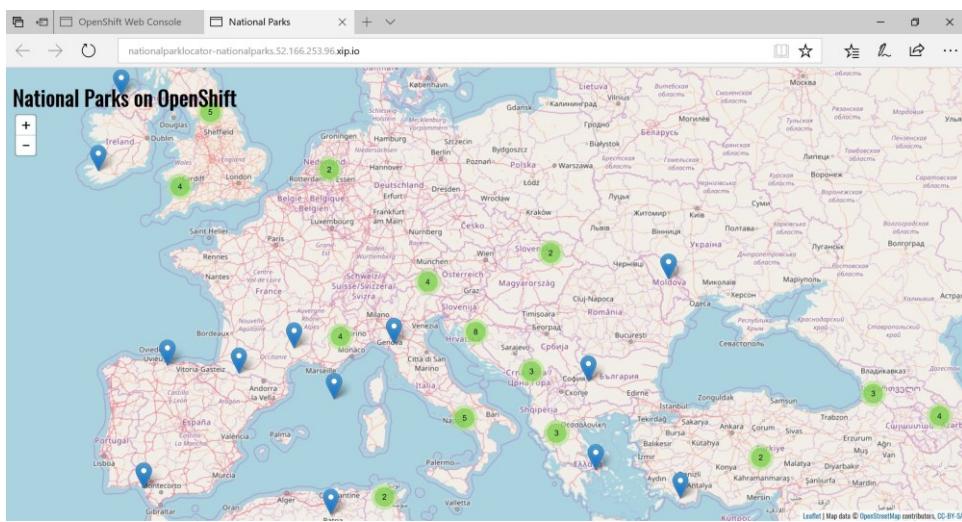
```
ossadmin@oss-bastion:~ [ossadmin@oss-bastion ~]$ oc env dc nationalparklocator -e MONGODB_USER=mongodb -e MONGODB_PASSWORD=mongodb -e MONGODB_DATABASE=mongodb
deploymentconfig "nationalparklocator" updated
[ossadmin@oss-bastion ~]$
```

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

```
ossadmin@oss-bastion:~ [ossadmin@oss-bastion ~]$ oc get dc nationalparklocator -o yaml
apiVersion: v1
kind: DeploymentConfig
metadata:
  creationTimestamp: 2017-07-10T14:43:00Z
  labels:
    app: nationalparklocator
    component: frontend
    deploymentconfig: nationalparklocator
    version: v1
  name: nationalparklocator
  namespace: nationalparks
  resourceVersion: "1019b324-a2b2-11e7-96f4-000d3a27e831"
  selfLink: /apis/config.openshift.io/v1/namespaces/nationalparks/deploymentconfigs/nationalparklocator
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
  selector:
    matchLabels:
      app: nationalparklocator
      component: frontend
      deploymentconfig: nationalparklocator
      version: v1
  strategy:
    type: Rolling
status:
  availableReplicas: 1
  currentReplicas: 1
  desiredReplicas: 1
  latestAvailablePod: pod/nationalparklocator-1
  latestCreatedPod: pod/nationalparklocator-1
  observedGeneration: 1
  readyReplicas: 1
  totalReplicas: 1
  updateProgressPercent: 100
  updatedReplicas: 1
```

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

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



15. Our new application became very popular, and we need to scale out our front end to two pods. Use “oc scale” to do it

```
[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 ~]$
```

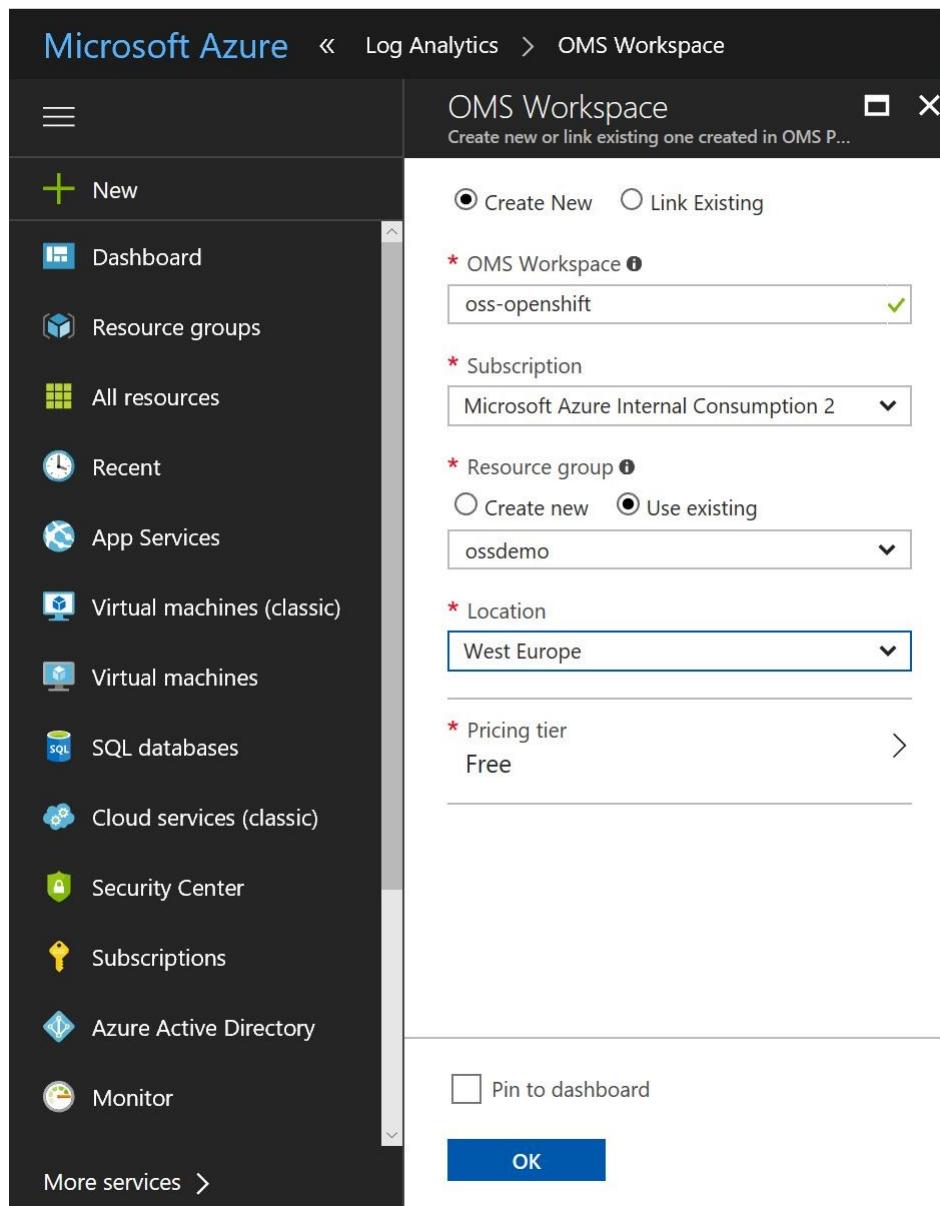
16. Now, let's test the self-healing, capabilities of OpenShift by deleting one of the running pods. Because, the desired state of the replication controller is 2 pods for the application “nationalparklocator”, OpenShift will automatically and instantly trigger the deployment of a new pod.

```
[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-sqdq4  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-sqdq4  1/1   Running     0          1m
[ossadmin@oss-bastion ~]$
```

CHALLENGE -7: MONITORING OEPNSHIFT WITH AZURE OMS

Azure Operations Management Suite (OMS) provides native support to OpenShift. In this challenge, 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



2. Open the OMS portal and note the workspace id and one of the primary keys.

3. From Cloud Shell ssh into the bastion host, then ssh into one of the master node and create an OpenShift project and user 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
```

2. Use wget to download the ocp-* files from
<https://github.com/Microsoft/OMS-docker/tree/master/OpenShift>

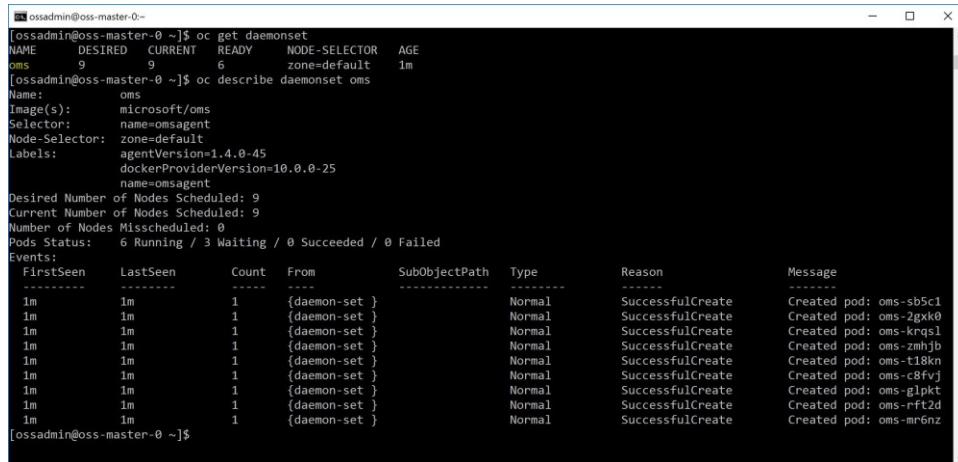
3. Make the file “secretgen.sh” executable. Run it, and provide our workspace id and key.

```
[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.

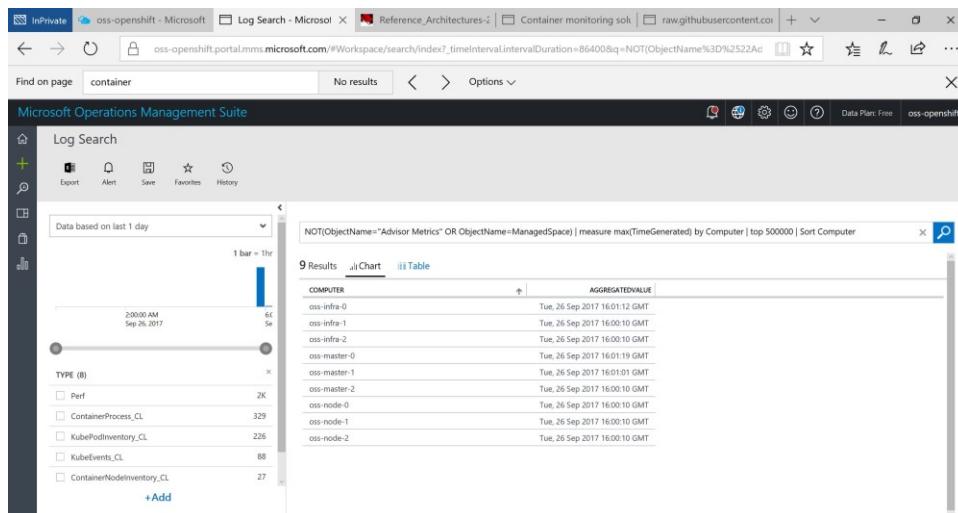
```
[ossadmin@oss-master-0 ~]$ oc create -f ocp-secret.yaml
```

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 Misscheduled: 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-2gxrk0
1m         1m         1   {daemon-set}   Normal   SuccessfulCreate   Created pod: oms-kqrsl1
1m         1m         1   {daemon-set}   Normal   SuccessfulCreate   Created pod: oms-zmhjb
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-glptk
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.



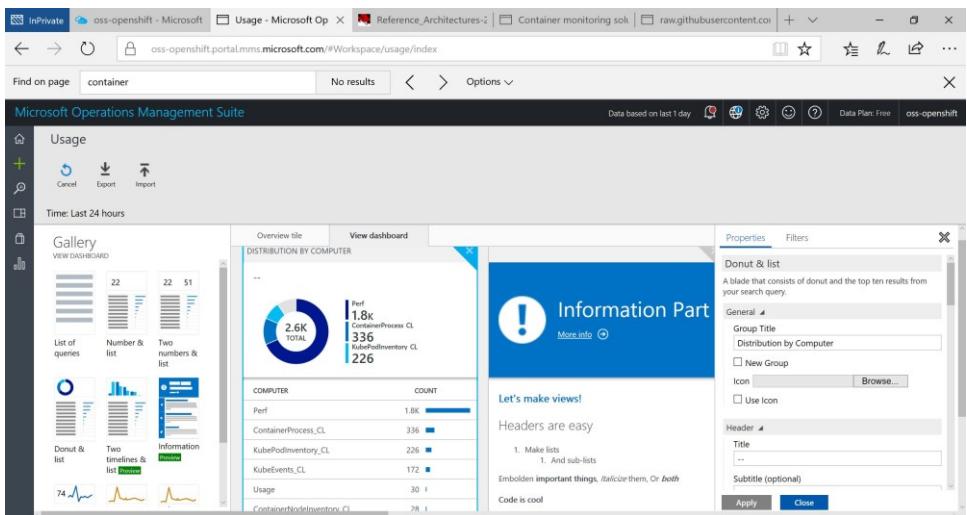
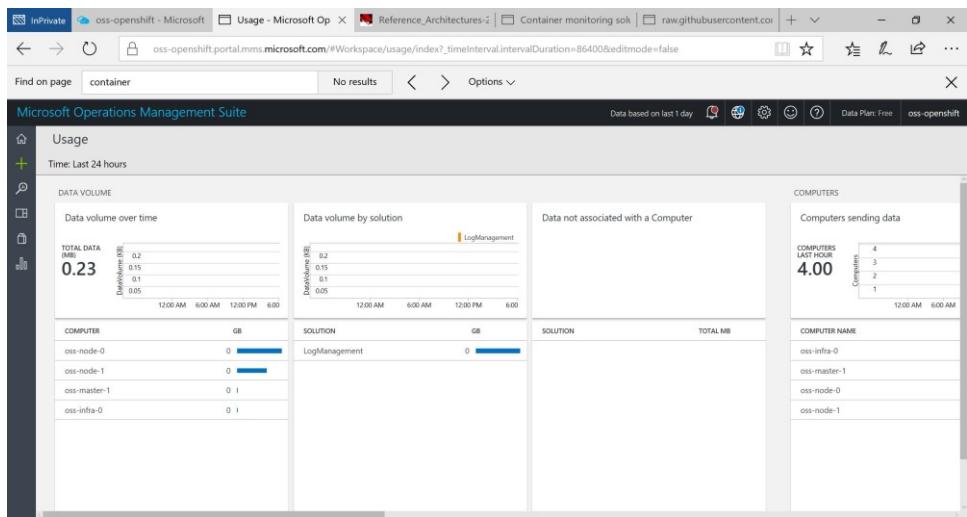
COMPUTER	AGGREGATEDVALUE
oss-infra-0	Tue, 26 Sep 2017 16:01:12 GMT
oss-infra-1	Tue, 26 Sep 2017 16:00:10 GMT
oss-infra-2	Tue, 26 Sep 2017 16:00:10 GMT
oss-master-0	Tue, 26 Sep 2017 16:01:19 GMT
oss-master-1	Tue, 26 Sep 2017 16:01:01 GMT
oss-master-2	Tue, 26 Sep 2017 16:00:10 GMT
oss-node-0	Tue, 26 Sep 2017 16:00:10 GMT
oss-node-1	Tue, 26 Sep 2017 16:00:10 GMT
oss-node-2	Tue, 26 Sep 2017 16:00:10 GMT

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

The screenshot shows the Microsoft Operations Management Suite (MOMS) My Dashboard. It displays a summary of computer data and event counts. The 'All Computers with their most recent data' section shows 9 computers, with three listed: oss-irha-0, oss-irha-1, and oss-irha-2, all last updated at 9/26/2017 6:01:12:310 PM. The 'Distribution of data Types' section shows 8 types, with Perf having 1.8K entries, ContainerProcess_CL having 109, and KubePodInventory_CL having 226. The 'All Events' section shows 0 events.

The screenshot shows the Microsoft Operations Management Suite (MOMS) Log Search interface. It displays a chart titled '* Measure count by Type' showing the count of logs for different types over the last day. The chart shows Perf with 1,750 entries, ContainerProcess_CL with 336, KubePodInventory_CL with 226, KubeEvents_CL with 88, ContainerNodeInventory_CL with 28, Syslog with 11, Usage with 7, and Heartbeat with 2.

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



CHALLENGE -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

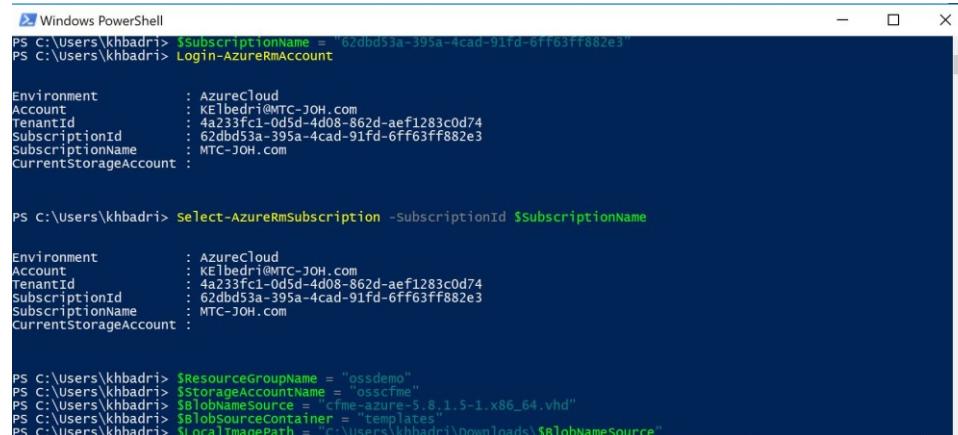
CloudForms can see and manage both the guest and host systems, allowing management of workloads and infrastructure within the same system.

CloudForms is a first-class citizen on Azure and provides a solid hybrid cloud management solution. During this challenge 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. Login to Azure and Configure some variables to use during the deployment. Change \$BlobNameSource and \$LocalImagePath to reflect your environment.



```
PS C:\Users\kbadri> $SubscriptionName = "62dbd53a-395a-4cad-91fd-6ff63ff882e3"
PS C:\Users\kbadri> Login-AzureRmAccount

Environment          : AzureCloud
Account             : KEBedri@MTC-JOH.com
TenantId            : 4a233fc1-0d5d-4d08-862d-aef1283c0d74
SubscriptionId      : 62dbd53a-395a-4cad-91fd-6ff63ff882e3
SubscriptionName    : MTC-JOH.com
CurrentStorageAccount : 

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

Environment          : AzureCloud
Account             : KEBedri@MTC-JOH.com
TenantId            : 4a233fc1-0d5d-4d08-862d-aef1283c0d74
SubscriptionId      : 62dbd53a-395a-4cad-91fd-6ff63ff882e3
SubscriptionName    : MTC-JOH.com
CurrentStorageAccount : 

PS C:\Users\kbadri> $ResourceGroupName = "ossdemo"
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 “ossdemo” resource group to deploy the vhd image to.

The screenshot shows the Microsoft Azure portal interface for creating a new storage account. The left sidebar contains a navigation menu with options like New, Dashboard, All resources, Resource groups, App Services, Function Apps, SQL databases, Azure Cosmos DB, Virtual machines, Load balancers, Storage accounts, Virtual networks, Azure Active Directory, and Monitor. The main content area is titled "Create storage account". The form fields are as follows:

- Name: osscfme
- Deployment model: Resource manager
- Account kind: General purpose
- Performance: Premium
- Replication: Locally-redundant storage (LRS)
- Secure transfer required: Disabled
- Subscription: MTC-JOH.com
- Resource group: ossdemo
- Pin to dashboard: Unchecked
- Create button: The "Create" button is highlighted with a yellow background.

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.core.windows.net/$blobName -LocalImagePath $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/$blobName -LocalImagePath $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: 00:01:31
Creating new page blob of size 34361836032...
Lapsed time for upload: 00:49:22
localFilePath          DestinationUri
-----:-----:-----
:C:\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

```

6. Customize the Azure environment and create the CloudForms vm. Use your public key

```

$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
AAAAAB3NzaC1yc2EAAAQABAAQDQOd6tNwPPYBQ+wveI+dmBmdpmaBB
87qOG/dHe/ZEFJIXLDRILytVh2kevgeXn/SzbqL3DJ4qQWVGmsaZwELhlQK
dcc/Aybwrn9tq94H2WgPQ79RnRd8BRgdu5sVWTpyGZc5OFdAyFvkIftiasH
Ag3jb7u6oJ7f3HCH4tax/sbdqhSkTnivH4Uxq0Vx1DQOt1z4WfbCYxmG9cL
c2zNzqc6/d7Y/g33iW94FZ5CCPfUoY+HdyOSu5cy/rWtreCskWQZwNR8xkv
DOKI1c2bnBTCosN79FMzZYyiOcvMIJbtE9KYH9G49G6p2tXDBByhOQVj/4aI
gRc2S5SW+16e7VSn71 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 ossdemo

$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 = "cfme-azure5-8-1-0-2.x86_64.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 = "r3nZzCgY2EAAAADQABAAQDQOD6tNwPPYBQ+HveI+dmBmpmBB87qOG/dhe/ZEFJIXLDRILytvh2kei
.../mIb3o4yv9c496p...rcordv...v1...fti...Ag31b7U6o1773H04tax/sbdghskTn...vh4ux0v...ldQO...z4wfbCYmc9...cLc22nzc6/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>

< Windows PowerShell
PS C:\Users\kbadri> $rip = 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>

< Windows PowerShell
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 $rip.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> $virtualMachine = New-AzureRmVmConfig -VMName $VMName -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 -ComputerName $VMName -Credential $cred
PS C:\Users\kbadri> $osDiskURI = $storageAccount.PrimaryEndpoints.Blob.ToString() + $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 :
  VirtualMachineSize : Standard_DS4_V2
  MemoryInGB : 8
  NetworkInterfaceCount : 1
  OSProfile :
    ComputerName : cfme-5.8
    AdminUsername : osssadmin
    AdminPassword : 
    LinuxConfiguration :
      StorageProfile :
        OSDisk :
          Uri : https://ossdemo.blob.core.windows.net/templates/cfme-5.8.vhd
      NetworkInterfaceIds :
        Value : /subscriptions/62dbd53a-395a-4cad-91fd-6ff63ff882e3/resourceGroups/ossdemo/providers/Microsoft.Network/networkInterfaces/cfmenic
  NetworkProfile : 
  VmSize : Standard_DS4_V2

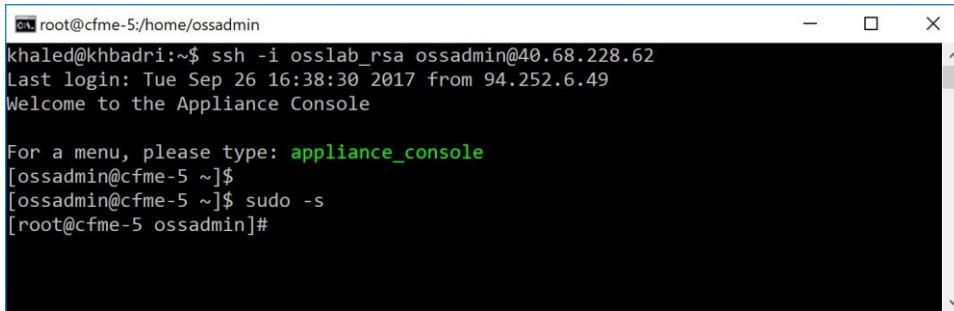
PS C:\Users\kbadri> New-AzureRmVM -ResourceGroupName $ResourceGroupName -Location $location -VM $virtualMachine
WARNING: Since the VM is created using premium storage, existing standard storage account, psvla$yoppksdvfe!, is used for boot diagnostics.
RequestID IsSuccessStatusCode StatusCode ReasonPhrase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

```

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

NAME	STATUS	SIZE	IP ADDRESS
cfme-5.8	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.62
oss-bastion	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.63
oss-infra-0	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.64
oss-infra-1	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.65
oss-infra-2	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.66
oss-master-0	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.67
oss-master-1	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.68
oss-master-2	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.69
oss-node-0	Running	Standard DS4 v2 (8 vcpus, 28 GB memory)	40.64.226.70

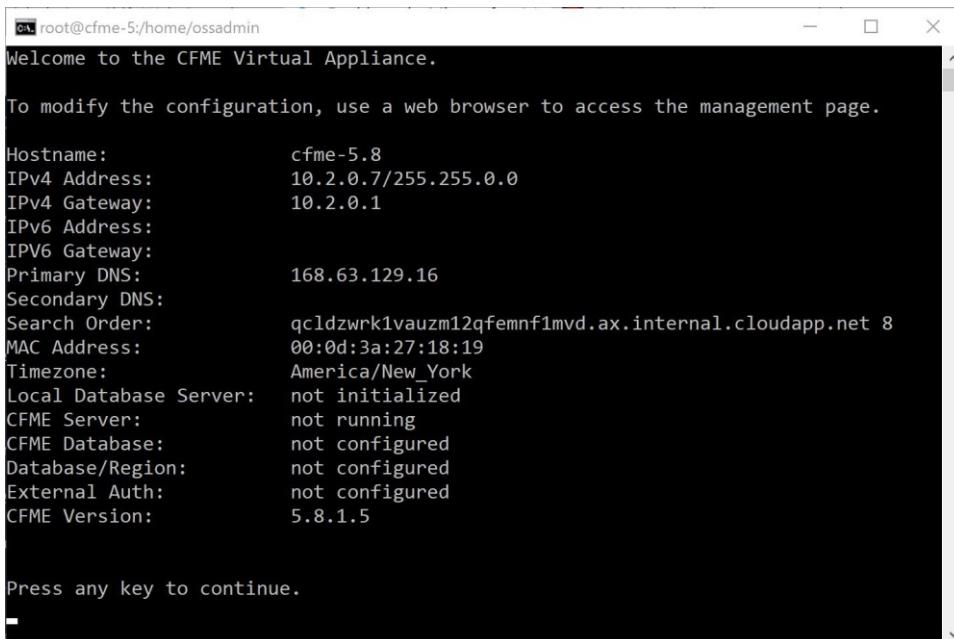
8. From the Cloud Shell, ssh into the virtual appliance using the SSH key.



```
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. For example configure the time zone (2).
12. Follow the prompts to make the changes.
13. Press Enter to accept a setting where applicable and quit (20)
14. Back to the Azure portal. We need to add a new disk for the CloudForms data base.

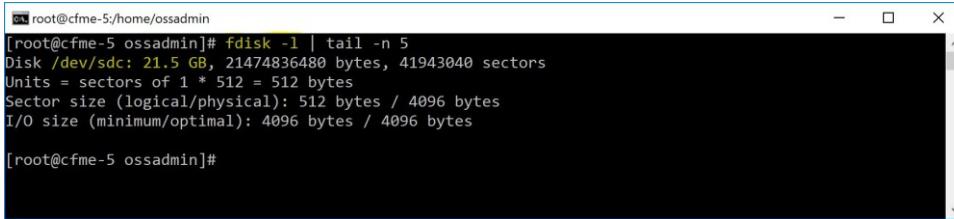
The image consists of three vertically stacked screenshots of the Microsoft Azure portal, illustrating the steps to attach an unmanaged disk to a virtual machine.

Screenshot 1: The Azure portal interface shows the 'Virtual machines' blade for the 'cfme-5.8 - Disks' resource group. On the left, the navigation menu includes 'Dashboard', 'All resources', 'Resource groups', 'App Services', 'Function Apps', 'SQL databases', 'Azure Cosmos DB', 'Virtual machines', 'Load balancers', 'Storage accounts', 'Virtual networks', 'Azure Active Directory', and 'Monitor'. The main area displays a list of virtual machines under 'Subscriptions: MTC-JOH.com'. One VM, 'cfme-5.8', is selected. The 'Disks' tab is active in the ribbon. A table shows the 'OS disk' configuration: Name 'cfme-5.8', Size '32 GiB', Storage Account 'Premium_LRS', and Encryption 'Not enabled'. Below the table, there is a section for 'Data disks' with a button '+ Add data disk'.

Screenshot 2: The 'Attach unmanaged disk' dialog box is open. It contains fields for 'Name' (set to 'cfme-5.8-20170926-224610'), 'Source type' (set to 'New (empty disk)'), 'Account type' (set to 'Premium (SSD)'), and 'Size (GiB)' (set to '20'). Under 'ESTIMATED PERFORMANCE', it shows 'IOPS limit' (500) and 'Throughput limit (MB/s)' (100). There are also fields for 'Storage container' (with a 'Browse' button) and 'Storage blob name' (set to 'cfme-5.8-20170926-224610.vhd'). At the bottom is an 'OK' button.

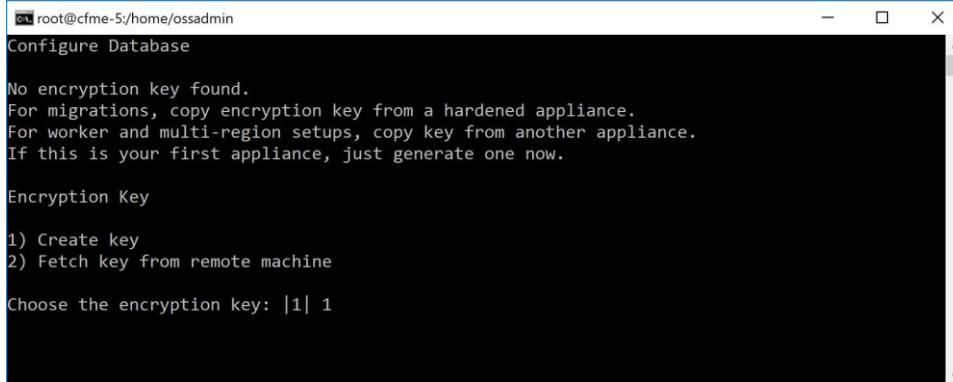
Screenshot 3: The 'Attach unmanaged disk' dialog box is shown again, but with a different value in the 'Storage container' field. Instead of 'https://osscfme.blob.core.windows.net/templates', it now shows 'https://osscfme.blob.core.windows.net/tempfiles'. All other fields remain the same as in Screenshot 2.

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



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

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



```
[root@cfme-5 ~]# Configure Database

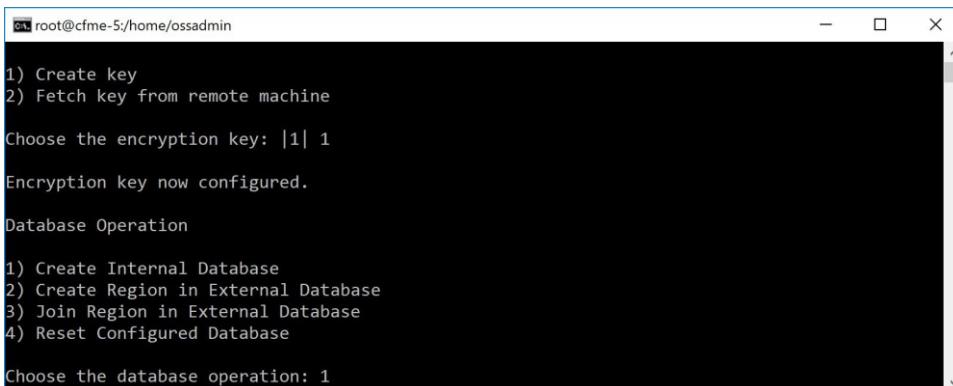
No encryption key found.
For migrations, copy encryption key from a hardened appliance.
For worker and multi-region setups, copy key from another appliance.
If this is your first appliance, just generate one now.

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 ~]# 1) Create key
2) Fetch key from remote machine

Choose the encryption key: |1| 1

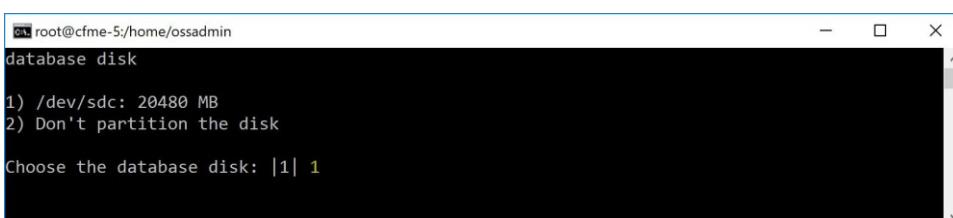
Encryption key now configured.

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

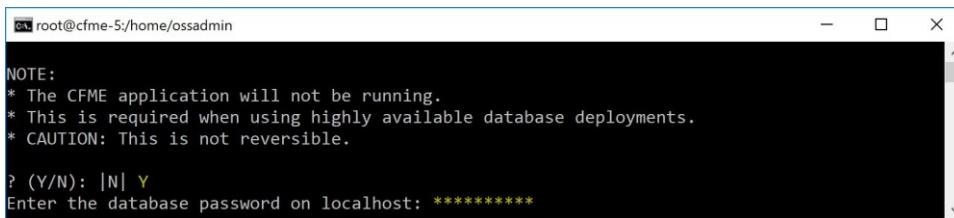


```
[root@cfme-5 ~]# database disk

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

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

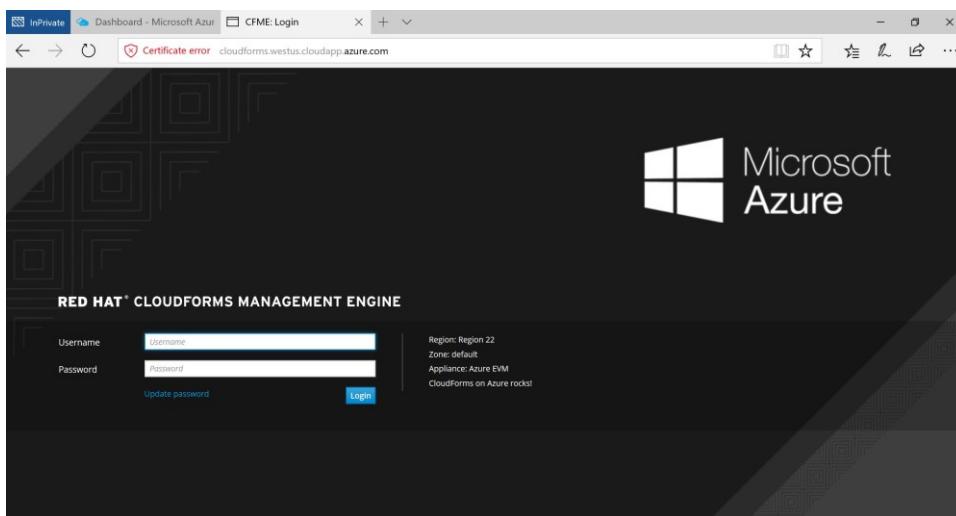
20. Select Y to configure the appliance as a database-only appliance and create and confirm a password for the database. As a result, the appliance is configured as a basic PostgreSQL server, without a user interface



```
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.

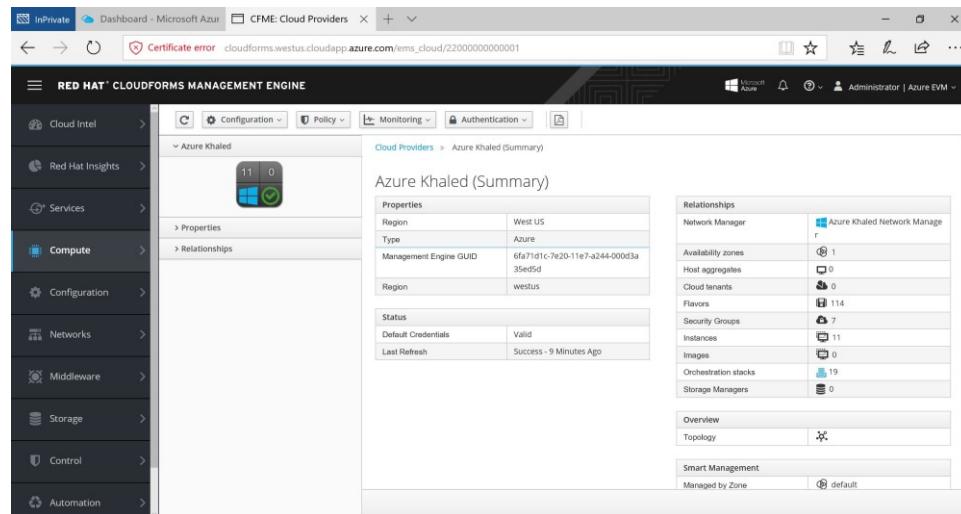
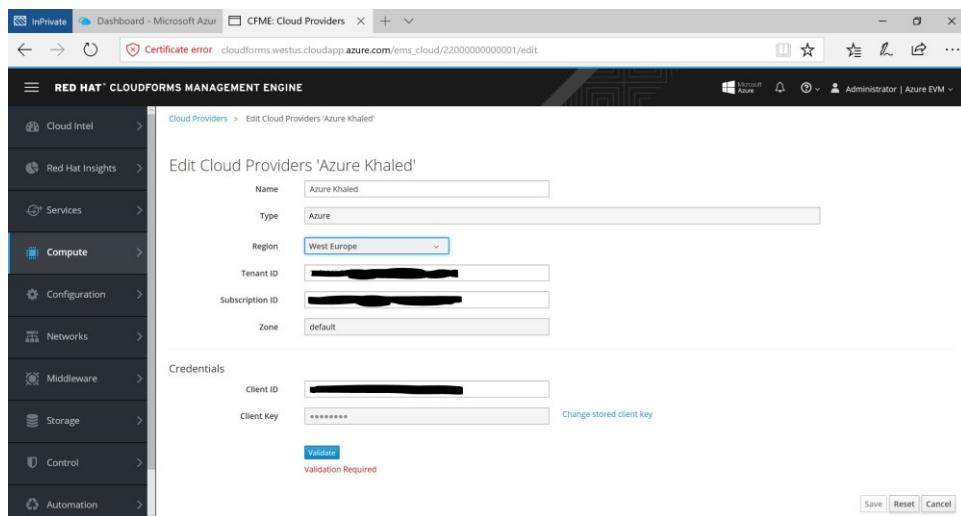
? (Y/N): |N| Y
Enter the database password on localhost: *****
```

21. Choose (16) to start EVM processes.
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:
- Navigate to the URL for the login screen. (<https://xx.xx.xx.xx> on the virtual machine instance)
 - Enter the default credentials (Username: admin | Password: smartvm) for the initial login.
 - Click Login.
 - Navigate to the URL for the login screen. (<https://xx.xx.xx.xx> on the virtual machine instance)
 - Click Update Password beneath the Username and Password text fields.
 - Enter your current Username and Password in the text fields.
 - Input a new password in the New Password field.
 - Repeat your new password in the Verify Password field.
 - Click Login.
24. 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.

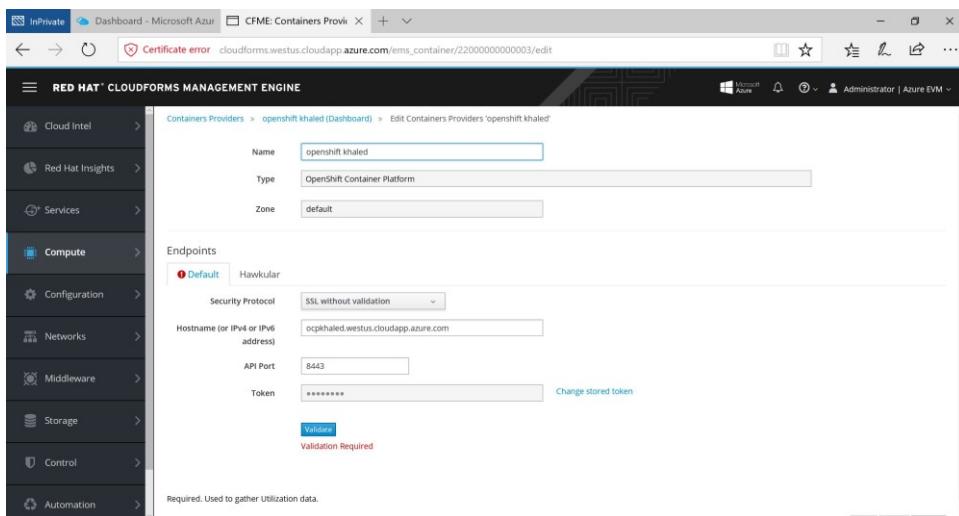


25. Add OpenShift provider

- 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
eyJhbGciOiJSUzI1Ni...
```
 - 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.



26. Explore CloudForms by navigating the left menu to get an idea on the insights and intelligence provided by the solution. CloudForms provides additional modules (compliance, management, reporting...) that won't be covered by the scope of the lab.

RED HAT® CLOUDFORMS MANAGEMENT ENGINE

Cloud Providers

No filters defined.

Cloud Providers

AWS Khaled Azure Khaled

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

Configuration Policy Monitoring Authentication

Cloud Providers

Azure Khaled

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
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 19 Sep 21 Sep 23 Sep 25

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

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

Configuration Policy Monitoring Authentication

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

Guest OS	Percentage
RHEL 7.3...	90.9%
Linux 1	9.1%

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

END THE LAB

To end the lab, simply delete the resource group *ossdemo* 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://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_cloudforms/4.5/html/installing_red_hat_cloudforms_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>