

# **Dell EMC Ready Bundle for Red Hat OpenStack Platform**

**Deploying OpenShift Container Platform 3.4  
Version 10.0.1**



**Dell EMC Converged Platforms and Solutions**

# Contents

---

List of Figures.....	iv
List of Tables.....	v
Trademarks.....	6
Notes, Cautions, and Warnings.....	7
Chapter 1: Executive Summary.....	8
About the Dell EMC Ready Bundle for Red Hat OpenStack Platform.....	9
About OpenShift on OpenStack.....	9
About This Document.....	9
Intended Audience.....	10
Chapter 2: Background.....	11
Architecture.....	12
Work Resources.....	14
Chapter 3: Plan and Prepare to Install OpenShift.....	15
Prerequisites.....	16
Prepare the OpenShift Environment.....	16
Ensure Subscriptions Credentials and Pool ID.....	16
Obtain the Guest Image.....	16
Project Quotas.....	17
Configure Heat Services.....	18
Increase Keystone Token Expiration Value.....	20
Upload the Image into Glance.....	21
Prepare the Heat YAML Files.....	21
Chapter 4: Configure Dynamic DNS.....	26
Overview.....	27
Prepare the DNS Environment.....	27
Download the DNS Playbook.....	27
Prepare the DNS YAML File.....	27
Deploy DNS.....	29
Deploy the Service.....	29
Generate a DNS Update Key.....	29
Verify DNS Functionality.....	29
Chapter 5: Installation and Configuration.....	31
Obtain the OpenShift Heat Templates RPM.....	32
Execute Heat Templates.....	32

Troubleshoot and Debug Failures.....	33
Heat Events.....	33
Debugging in Proper Order.....	33
About Ansible Log Files.....	34
Chapter 6: Validate the Installation.....	35
List the Nodes.....	36
Ensure a Working Router.....	36
Deploy the Router Pod to a Master.....	37
Chapter 7: Configure a Docker Registry with Persistent Storage.....	39
Create a Cinder Volume.....	40
Configure Red Hat Ceph Storage and Cinder Persistent Storage.....	41
Create the Docker-Registry.....	42
Create a Sample User in OpenShift.....	47
Chapter 8: Configure OpenShift Web GUI Access.....	48
Make DNS World Accessible.....	49
Add DNS to the Windows Bastion Host.....	49
Navigate to the OpenShift Web Console.....	49
Chapter 9: Deploy a Sample Application.....	51
Application Deployment Procedure.....	52
Chapter 10: Next Steps.....	56
Appendix A: Getting Help.....	57
Contacting Dell EMC.....	58
References.....	58
To Learn More.....	58

## List of Figures

---

Figure 1: Solution Architecture.....	13
Figure 2: Browse OpenShift Container Platform Catalog.....	52
Figure 3: Example PHP Application.....	53
Figure 4: New Application.....	53
Figure 5: Overview.....	54
Figure 6: Build Complete.....	55

## List of Tables

---

Table 1: Work Resources.....	14
Table 2: RHOSP Resource Minimum Quotas.....	17
Table 3: openshift_parameters.yaml Parameters.....	23
Table 4: dns-vars.yaml Parameters.....	28
Table 5: Cinder Volume Values.....	40

## Trademarks

---

Copyright © 2014-2017 Dell Inc. or its subsidiaries. All rights reserved.

Microsoft® and Windows® are registered trademarks of Microsoft Corporation in the United States and/or other countries.

Red Hat®, Red Hat Enterprise Linux®, and Ceph are trademarks or registered trademarks of Red Hat, Inc., registered in the U.S. and other countries. Linux® is the registered trademark of Linus Torvalds in the U.S. and other countries. Oracle® and Java® are registered trademarks of Oracle Corporation and/or its affiliates.

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

## Notes, Cautions, and Warnings

---

-  A **Note** indicates important information that helps you make better use of your system.
-  A **Caution** indicates potential damage to hardware or loss of data if instructions are not followed.
-  A **Warning** indicates a potential for property damage, personal injury, or death.

This document is for informational purposes only and may contain typographical errors and technical inaccuracies. The content is provided as is, without express or implied warranties of any kind.

---

# Chapter 1

---

## Executive Summary

---

### Topics:

- [About the Dell EMC Ready Bundle for Red Hat OpenStack Platform](#)
- [About OpenShift on OpenStack](#)
- [About This Document](#)
- [Intended Audience](#)

In order to meet the demands put on an organization by customers who require the agility, efficiency, and innovation of cloud-based services and applications, developers need a way to build, deploy, and manage applications in a containerized format, and deploy them in a self-service fashion.

IT Operations needs to be able to provide this with a secure, enterprise-grade environment that can have policy based control for automation of cluster services, scheduling and orchestration of the applications. By incorporating OpenShift Container Platform, based on Docker containers and Kubernetes orchestration, along with OpenStack virtual machine clusters, these demands can be met quickly and effectively.

## About the Dell EMC Ready Bundle for Red Hat OpenStack Platform

The Dell EMC Ready Bundle for Red Hat OpenStack Platform is an integrated hardware and software solution for OpenStack cloud that has been jointly designed and validated by Dell EMC and Red Hat. The Ready Bundle enables organizations to easily and rapidly deploy a highly reliable, optimized, and scalable Infrastructure as a Service (IaaS) cloud solution.

The Ready Bundle architecture is built on Dell EMC PowerEdge servers for the Controller, Compute, and Storage nodes, with Red Hat OpenStack Platform and Red Hat Ceph Storage software, plus a number of validated extensions to the core architecture to enable capabilities such as Platform as a Service (PaaS), Containers as a Service (CaaS), and cloud native development and operations. This release of the Ready Bundle is based on Red Hat OpenStack Platform 10 (RHOSP 10), which is the OpenStack release called Newton.

## About OpenShift on OpenStack

Red Hat OpenShift Container Platform 3.4 is a Platform as a Service (PaaS) product. Its developer-centric approach enables developers to create and deploy applications with more predictability, greater ease, and less operator intervention. It manages deployments and provides application scalability services.

In the data center, OpenShift Container Platform 3.4 is deployed on Red Hat Enterprise Linux Server 7.3, and is comprised of application containers powered by Docker, and orchestration and management provided by Kubernetes.

Integration of OpenShift with OpenStack allows the organization to leverage existing operational techniques and organizational policies, adding a layer of deployment and redeployment flexibility not common in non-virtual deployments. This solution provides an example demonstrating how OpenShift Container Platform 3.4 basic usage can occur on a robust OpenStack infrastructure.

The configuration described in this document consists of three OpenShift master virtual machines and four OpenShift node virtual machines in single datacenter environment. The addition of more nodes is possible, but not documented here. In addition to the configuration, operational management tasks are shown to demonstrate functionality.

This version of the documentation introduces High Availability features of OpenShift Container Platform 3.4, to a robust, production-ready deployment of OpenShift on OpenStack.

More information on OpenShift Container Platform 3.4 can be found at <http://www.openshift.com>. It is based upon OpenShift Origin, the open source software project hosted at <http://www.openshift.org>.

## About This Document

This document contains code and configuration samples in monospace fonts. While it is tempting for the user to copy and paste those values from this document into their system, it is inadvisable and not supported. While we make every effort to ensure that the documentation is correct and complete, documents rendered via some client applications make unpredictable changes to the actual spacing of the data elements, and lose fidelity to what a proper code or configuration setting should actually be to work properly. We see very impactful changes, for example, between the Firefox PDF display and the Adobe Acrobat Reader PDF display.

Copy and paste from this document only with full understanding of the necessary formatting changes that you'll have to make. We have made efforts to provide online verbatim copies of the essential data, as well as pointing the user to appropriate external documentation to achieve the proper formatting.

This guide is especially important with regard to configuring DNS and networking. Ensure that you refer to this document during OpenShift and CloudForms installation.

## Intended Audience

This technical guide shows the administrator how to build and deploy OpenShift in their Dell EMC Ready Bundle for Red Hat OpenStack Platform. The end user is not directly addressed in this document.

Find out more about developing and managing the OpenShift Container Platform by accessing the Red Hat documentation here: <https://docs.openshift.com/container-platform/3.4/welcome/index.html>.

---

# Chapter 2

---

## Background

---

**Topics:**

- [Architecture](#)
- [Work Resources](#)

OpenShift will be running on multiple VMs. Some of them will require Internet access.

## Architecture

*Figure 1: Solution Architecture* on page 13 displays a conceptual visualization of the Dell EMC Ready Bundle for Red Hat OpenStack Platform architecture with OpenShift and CloudForms:

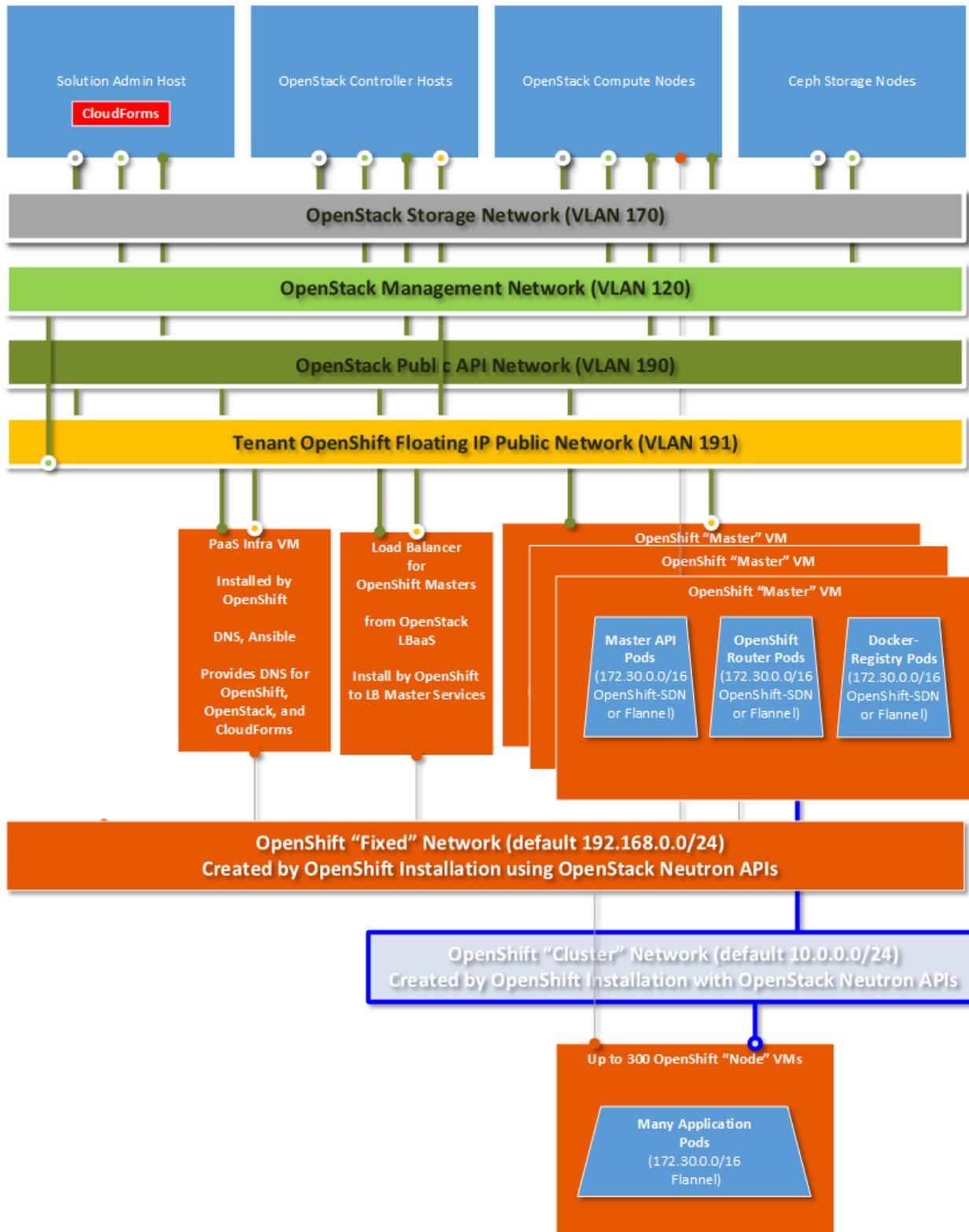


Figure 1: Solution Architecture

## Work Resources

This solution uses several Open Source projects to install OpenShift, listed in [Table 1: Work Resources](#) on page 14:

**Table 1: Work Resources**

Project	Description	URL
openshift-heat-templates-0.9.9-5.el7ost.noarch.rpm	Heat templates and scripts to set up the OpenStack environment and kick off the OpenShift Ansible installation. Packaged by Red Hat.	<a href="https://access.redhat.com/errata/RHEA-2017:1788">https://access.redhat.com/errata/RHEA-2017:1788</a>
OpenShift Ansible Playbooks	Ansible playbooks to deploy all manner of OpenShift on all manner of infrastructures. Packaged by Red Hat and available in repositories.	<a href="https://github.com/openshift/openshift-ansible">https://github.com/openshift/openshift-ansible</a>
OpenShift Container Platform 3.4 Documentation	Information required to set up and manage an OpenShift Container Platform environment, as a cluster administrator or an application developer.	<a href="https://docs.openshift.com/container-platform/3.4/welcome/index.html">https://docs.openshift.com/container-platform/3.4/welcome/index.html</a>

---

# Chapter

# 3

---

## Plan and Prepare to Install OpenShift

---

**Topics:**

- [Prerequisites](#)
- [Prepare the OpenShift Environment](#)

You must carefully plan the OpenShift installation, and prepare the environment.

## Prerequisites

This procedure depends upon a number of services and settings that must be in place before proceeding. Prerequisites include:

- Red Hat OpenStack Platform version 10
- RHOSP port security controls must be enabled in the `neutron` service
- Increase `keystone` token expiration
- RHOSP user account and credentials
- A Red Hat Enterprise Linux Server 7.3 cloud image pre-loaded into `glance`
- Red Hat Enterprise Linux update subscription credentials (user/password or Satellite Server)
- An SSH keypair pre-loaded into `nova`
- A publicly-available `neutron` network for inbound access
- A pool of floating IP addresses to provide inbound access points for instances
  - These instances require Internet access in order to download packages
- A host running `haproxy` to provide load balancing
- A host running `bind` with:
  - A properly-delegated sub-domain for publishing
  - A key for dynamic updates
- An existing LDAP or Active Directory service for user identification and authentication (see [Deploying Red Hat OpenShift Container Platform 3.4 on Red Hat OpenStack Platform 10](#))
  - Or, you can use `htpasswd` instead (see [Create a Sample User in OpenShift](#) on page 47)

## Prepare the OpenShift Environment

Follow these procedures to prepare the OpenShift installation environment:

- [Ensure Subscriptions Credentials and Pool ID](#) on page 16
- [Project Quotas](#) on page 17
- [Configure Heat Services](#) on page 18
- [Increase Keystone Token Expiration Value](#) on page 20
- [Upload the Image into Glance](#) on page 21
- [Prepare the Heat YAML Files](#) on page 21

### Ensure Subscriptions Credentials and Pool ID

You must have Red Hat subscriptions to:

- Red Hat OpenStack Platform 10
- OpenShift Container Platform 3.4

Have ready the username, password, and pool ID for the Red Hat OpenStack Platform and OpenShift.

### Obtain the Guest Image

Ensure that you have a RHEL 7.3 KVM Guest Image, in QCOW2 format.

The `rhel-guest-image-7.3-35.x86_64.qcow2` file can be found at [https://access.redhat.com/downloads/content/69/ver=/rhel---7/7.3/x86\\_64/product-software](https://access.redhat.com/downloads/content/69/ver=/rhel---7/7.3/x86_64/product-software).

## Project Quotas

Each project in RHOSP has a set of resource quotas which are set by default.

Several of these values should be increased to allow the OpenShift Container Platform stack to fit, as described in [Table 2: RHOSP Resource Minimum Quotas](#) on page 17:

**Table 2: RHOSP Resource Minimum Quotas**

Resource	Minimum	Recommended
Instances	9	20
VCPUs	20	60
RAM (GB)	50	450
Floating IP Addresses	9	15
Security Groups	5	5
Volumes	10	30
Volume Storage (GB)	800	2,000

These numbers are for a basic general-purpose installation with low to moderate use, and allow for scaling up to 10 more application nodes. The correct values for a specific installation depend upon the expected use. They are calculated using a detailed analysis of the actual resources needed and available.

 **Note:** The number of nodes, selection of instance flavors, and disk volume sizes here are for demonstration purposes. To deploy a production service consult the OpenShift Container Platform sizing guidelines for each resource.

To update the project quotas:

1. Log into the Director Node.
2. Source your *overcloudrc* file:

```
$ source overcloudrc
```

3. Set all your quotas to very high values for the admin project. This enables you to launch as many VMs as you require, create security groups, etc.:

```
$ openstack quota set admin --<parameters> <value>
```

For example:

```
$ openstack quota set admin --volumes 200
```

4. Ensure that the project quotas were updated, by executing the following command:

```
$ openstack quota show admin
```

The output will appear similar to this example:

```
+-----+-----+
| Field | Value |
+-----+-----+
| backup_gigabytes | 1000 |
| backups | 10 |
| cores | 2000 |
| fixed-ips | 200 |
| floating-ips | 200 |
```

```

gigabytes | 10000 |
gigabytes_rbd_backend | -1 |
graph | -1 |
healthmonitor | -1 |
injected-file-size | 102400 |
injected-files | 500 |
injected-path-size | 2550 |
instances | 10000 |
key-pairs | 1000 |
l7policy | -1 |
listener | -1 |
loadbalancer | 10 |
networks | 100 |
per_volume_gigabytes | -1 |
pool | 10 |
ports | 500 |
project | 490ff875ebbd4b4cbbf38f53ed58a7c2 |
properties | 1280 |
ram | 512000 |
rbac-policies | 10 |
routers | 100 |
secgroup-rules | 1000 |
secgroups | 1000 |
server-group-members | 100 |
server-groups | 100 |
snapshots | 100 |
snapshots_rbd_backend | -1 |
subnetpools | -1 |
subnets | 100 |
trunk | -1 |
volumes | 500 |
volumes_rbd_backend | -1 |
+-----+

```

## Configure Heat Services

Follow these procedures to set up and configure more Heat services:

- [Confirm or Create the heat-cfn Service](#) on page 18
- [Set Metadata Server URLs](#) on page 19

### Confirm or Create the heat-cfn Service

The `heat-cfn` service must be present. If it is not, it must be created. For more information see <https://access.redhat.com/documentation/en/red-hat-openstack-platform/8/installation-reference/92-configure-the-orchestration-service>.

1. From the Director, Source the `overcloudrc` file to authenticate to the Overcloud, by executing the following command:

```
# . ./overcloudrc
```

2. Check the output of the following command to see if the service has been created:

```
# openstack service list | grep heat-cfn
```

3. If there is no output, then the service has not been created. Create the service by executing the following command:

```
# openstack service create --name heat-cfn cloudformation
```

4. Check the output of the following command to see if the `heat-cfn` endpoint has already been created:

```
# openstack endpoint list | grep heat-cfn
```

5. If there is no output (other than warnings associated with command deprecation), then create the endpoint using the value substitutions as in the following example.

 **Note:** The `<Heat VIP>` value can either be the VIP dedicated to Heat, or the IP address of the current Controller host on the OpenStack Public API network. In this example, the `<Heat VIP>` is `192.168.190.70`, as shown in the `ip address` command.

```
# openstack endpoint create heat-cfn \
--publicurl 'http://<OpenStack Public API IP>:8000/v1' \
--adminurl 'http://<OpenStack Public API IP>:8000/v1' \
--internalurl 'http://<Heat VIP>:8000/v1' \
--region 'regionOne'
```

Property	Value
adminurl	http://192.168.190.70:8000/v1
id	996190feff1c48fbb8029f1723ff627c
internalurl	http://192.168.140.71:8000/v1
publicurl	http://192.168.190.70:8000/v1
region	regionOne
service_id	d6653ae8e8294890b81b03de9056226a

### Set Metadata Server URLs

The default installation has the Heat APIs incorrectly sending the `heat_metadata_server_url` and `heat_waitcondition_server_url` to clients as the `192.168.140.x` addresses of the Controllers. They should be telling the clients to access these services through the VIP (i.e., the OpenStack Public API IP address).

To set the proper metadata server URLs:

1. On **all** Controller Nodes, ensure that the `heat_metadata_server_url` is the URL with the OpenStack Public API IP Address port **8000**, by editing the `/etc/heat/heat.conf` file:

```
# heat things to listen

# URL of the Heat metadata server. (string value)
#heat_metadata_server_url =
heat_metadata_server_url=http://<OpenStack Public API IP Address>:8000

# URL of the Heat waitcondition server. (string value)
#heat_waitcondition_server_url = <None>
heat_waitcondition_server_url = http://<OpenStack Public API IP
Address>:8000/v1/waitcondition

# URL of the Heat CloudWatch server. (string value)
#heat_watch_server_url =
heat_watch_server_url=http://<OpenStack Public API IP Address>:8003
```

2. On **any one** Controller Node **only**, restart all the services by executing the following command:

```
# systemctl restart openstack-heat-*
```

3. Ensure proper service restart:

```
# systemctl | grep -A1 heat
session-168.scope loaded active running Session 168 of user heat-admin
```

```

session-c1.scope loaded active abandoned Session c1 of user rabbitmq
--
openstack-heat-api-cfn.service loaded active running Openstack Heat CFN-
compatible API Service
openstack-heat-api-cloudwatch.service loaded active running OpenStack Heat
CloudWatch API Service
openstack-heat-api.service loaded active running OpenStack Heat API
Service
openstack-heat-engine.service loaded active running Openstack Heat Engine
Service
openstack-nova-api.service loaded active running OpenStack Nova API
Server
--
user-1001.slice loaded active active User Slice of heat-admin
user-980.slice loaded active active User Slice of rabbitmq

```

4. Ensure that the endpoints that can be accessed from the VMs are created by examining the endpoint with the following command:

```

# openstack
(openstack) endpoint show heat-cfn
+-----+-----+
| Field          | Value                                     |
+-----+-----+
| adminurl       | http://192.168.140.70:8000/v1           |
| enabled        | True                                     |
| id             | 84784ef00c3540a08bdd941b238f258f      |
| internalurl    | http://192.168.140.70:8000/v1         |
| publicurl      | http://192.168.190.125:8000/v1        |
| region         | regionOne                               |
| service_id     | e80d0db1d99f42c784333072bf4bb61f     |
| service_name   | heat-cfn                                |
| service_type   | cloudformation                         |
+-----+-----+
(openstack)

```

5. Ensure that the watch server is listening by executing the following command:

```

$ ss -lntp | grep 8003

LISTEN      0          128          192.168.140.73:8003      *:*
LISTEN      0          128          192.168.190.125:8003   *:*
LISTEN      0          128          192.168.140.70:8003   *:*

```

## Increase Keystone Token Expiration Value

Automated portions of the installation can exceed the default timeout of the Keystone token that the Heat client obtains when it first begins the automated installation.

To increase the token expiration value:

1. On **all** Controller Nodes, increase the value by executing the following command:

```

# . ~/overcloudrc
# sed -i -e 's/#expiration = .*/expiration = 7200/' \
/etc/keystone/keystone.conf

```

2. On **any one** Controller Node **only**, restart the Keystone service by executing the following command:

```

# systemctl restart httpd

```

3. Ensure that the new 24-hour setting (86400 seconds) appears by executing the following command:

```
# . ~/overcloudrc
# openstack token issue -c expires
```

## Upload the Image into Glance

Now you can upload the RHEL 7.3 QCOW2 image, obtained in [Obtain the Guest Image](#) on page 16, into Glance.

To upload the image:

1. On the Director Node, become user *stack*.
2. Source the *overcloudrc* file to authenticate to the Overcloud, by executing the following command:

```
# . ./overcloudrc
```

3. Check the Glance version by executing the following command:

```
# glance --version
2.5.0
```

4. Create the Glance image:



**Note:** Some versions of the Glance client express "--is-Public True" differently.

```
# openstack image create --public --disk-format qcow2 \
  --container-format bare \
  --file rhel-guest-image-7.3-35.x86_64.qcow2 rhel73
```

Property	Value
checksum	486900b54f4757cb2d6b59d9bce9fe90
container_format	bare
created_at	2017-07-24T18:38:19.000000
deleted	False
deleted_at	None
disk_format	qcow2
id	c0d6cf3d-196e-4fc8-a62c-39bae4a36152
is_public	True
min_disk	0
min_ram	0
name	rhel73
owner	c1b740d8571f4cb5aa66f8ddcfdec015
protected	False
size	474909696
status	active
updated_at	2017-07-24T18:38:26.000000
virtual_size	None

## Prepare the Heat YAML Files

The installation is directed by Heat YAML files. You must prepare them with the proper parameters before they can be utilized.

Heat uses one or more YAML input files to customize the stacks that it creates. This configuration uses an external DNS service, as detailed in [Configure Dynamic DNS](#) on page 26, and a dedicated load balancer created as part of the Heat stack. DNS records are created to direct inbound traffic for the masters and the OpenShift router through this load balancer. This configuration uses the `flannel` SDN.

 **Note:** The Docker storage is set to a low value, as this reference environment is for demonstration purposes only. When deploying in production environments, ensure that you tune these values to accommodate for appropriate container density.

To prepare the YAML files:

1. Log into the Director Node.
2. Create a file, named *openshift\_parameters.yaml*.
3. Paste the following into that file:

 **Note:** Ensure that this file is properly indented when you create your version. YAML is very whitespace-sensitive.

```
parameters:
  ocp_version: "3.4"
  osp_version: 10
  #ansible_version: "2.2.0.0"

  # OpenShift service characteristics
  deployment_type: openshift-enterprise
  domain_name: "r11b.oss.labs"
  app_subdomain: "apps.r11b.oss.labs"
  lb_hostname: "devs"
  loadbalancer_type: dedicated
  openshift_sdn: flannel
  deploy_router: true
  deploy_registry: true

  # Number of each server type
  master_count: 2
  infra_count: 1
  node_count: 1

  # OpenStack network characteristics
  external_network: public
  internal_subnet: 192.168.3.0/24
  container_subnet: 10.10.3.0/24

  # DNS resolver and updates
  dns_nameserver: 100.82.35.203
  dns_update_key: kOZMgt3EDHJnHl9fILUJZQ==
  # Instance access
  ssh_key_name: ocp3kp
  ssh_user: cloud-user

  # Image selection
  bastion_image: rhel73
  master_image: rhel73
  infra_image: rhel73
  node_image: rhel73
  loadbalancer_image: rhel73

  # Docker Storage controls
  master_docker_volume_size_gb: 10
  infra_docker_volume_size_gb: 10
  node_docker_volume_size_gb: 10

  # OpenStack user credentials
  os_auth_url: http://100.82.35.190:5000/v2.0
  os_username: admin
  os_password: "CATkdFk8NBtpjGDhGgJzefJG"
  os_region_name: regionOne
  os_tenant_name: admin
```

```

# Red Hat Subscription information
rhn_username: "username"
rhn_password: "correctpassword"
rhn_pool: '8a85f9815b23f4ab015b240bd4123456'
extra_rhn_pools: '8a85f98c5a2714eb015a2784ab123456'

resource_registry:
# Adjust path for each entry
OOShift::LoadBalancer: /usr/share/openshift-heat-templates/ \
  loadbalancer_dedicated.yaml
OOShift::ContainerPort: /usr/share/openshift-heat-templates/ \
  sdn_flannel.yaml
OOShift::IPFailover: /usr/share/openshift-heat-templates/ \
  ipfailover_keepalived.yaml
OOShift::DockerVolume: /usr/share/openshift-heat-templates/ \
  volume_docker.yaml
OOShift::DockerVolumeAttachment: /usr/share/openshift-heat-templates/ \
  volume_attachment_docker.yaml
OOShift::RegistryVolume: /usr/share/openshift-heat-templates/ \
  registry_ephemeral.yaml

```

4. Review [Table 3: `openshift\_parameters.yaml` Parameters](#) on page 23 to ensure the correct parameters for your environment.



**Note:** The values below are stamp-specific examples. Your environment may require different values.

**Table 3: `openshift_parameters.yaml` Parameters**

Parameter: Example Value	Description
deployment_type: openshift-enterprise	The OpenShift Container Platform product. The other possible option is <i>origin</i> , but OpenShift Origin is not supported by Red Hat.
domain_name: "r11b.oss.labs"	The domain that will be served by the DNS server on the setup DNS master VM.
app_subdomain: "apps.r11b.oss.labs"	The subdomain used for applications deployed on OpenShift.
lb_hostname: "devs"	A dedicated node is created during stack creation, and the <code>HAPROXY</code> load balancer is configured on it. In this case the FQDN of the load balancer is <i>devs.ocp3.r11b.oss.labs</i> .
loadbalancer_type: dedicated	This configuration uses a dedicated load balancer because when deploying multiple master nodes, access to both the nodes and OpenShift router pods (which run on Infrastructure [infra] nodes) have to be load-balanced.
openshift_sdn: flannel	This configuration uses flannel as the dedicated private Software Defined Network (SDN) used for OpenShift container traffic.
deploy_router: true	Whether or not to deploy a Docker router. Set to <i>true</i> to deploy the router.
deploy_registry: true	Whether or not to deploy a Docker registry. Set to <i>true</i> to deploy the registry.

Parameter: Example Value	Description
master_count: 2	The number of Master replicas to launch and cluster. OpenShift has its own multi-master replication built in.
infra_count: 1	The number of Infra nodes to launch.
node_count: 1	The number of Nodes (OpenShift Compute VMs) to launch. Set higher for multi-user deployments. OpenShift installation spreads the nodes out over <i>nova-compute</i> nodes.
external_network: public	The name of the network in OpenStack that has access to the Internet. <i>public</i> is the external network name created by the Dell EMC Ready Bundle for Red Hat OpenStack Platform post-installation testing.
internal_subnet: 192.168.3.0/24	The subnet that is attached to <b>all</b> hosts.
container_subnet: 10.10.3.0/24	The subnet that is attached to all hosts, <b>except</b> the Infra host VM.
dns_nameserver: 100.82.35.203	DNS nameservers with public Internet query access. Use the created DDNS master server.
dns_update_key: kOZMgt3EDACCDRLUJZQ==	The special key string used to make DNS updates that can be generated by <code>ddns-confgen</code> , which is part of the <i>bind-utils</i> RPM.
ssh_key_name: ocp3	The SSH key name used to log into the OpenShift Bastion, Master, Infra, and other nodes.
ssh_user: cloud-user	The SSH username that the imaged being used allows to login via <code>ssh_key_name</code> . RHEL 7.3 image uses <i>cloud-user</i> .
bastion_image: rhel73 master_image: rhel73 infra_image: rhel73 node_image: rhel73 loadbalancer_image: rhel73	The base image for each instance within the RHOCV deployment. Use the name of the RHEL 7.3 image you uploaded to Glance.
master_docker_volume_size_gb: 10	Volume size for Master nodes. Will be provisioned out of Cinder volumes, backed by Red Hat Ceph Storage.
infra_docker_volume_size_gb: 10	Volume size for Infra nodes. Will be provisioned out of Cinder volumes, backed by Red Hat Ceph Storage.
node_docker_volume_size_gb: 10	Volume size for Node servers. Will be provisioned out of Cinder volumes, backed by Red Hat Ceph Storage.

Parameter: Example Value	Description
os_auth_url: http://100.82.35.190:5000/v2.0	The os_auth_url is found in your <i>overcloudrc</i> file as <code>export OS_AUTH_URL=http://192.82.35.190:5000/v2.0</code> .
os_username: admin	The username in the <i>overcloudrc</i> file.
os_password: "CATkdFk8NBPtpjGDhGgJzefJG"	The password in the <i>overcloudrc</i> file.
os_region_name: regionOne	The OpenStack region for your Overcloud. Use <i>regionOne</i> in this case.
os_tenant_name: admin	The tenant name in the <i>overcloudrc</i> file.
rhn_username: "username"	The Red Hat Subscription Management (RHSM) username to register your subscriptions.
rhn_password: "correctpassword"	The password of your RHSM user.
rhn_pool: '8a85f9815b23f4ab015b240bd4123456'	The Pool ID that give you entitlements to OpenShift 3.4.
extra_rhn_pools: '8a85f98c5a2714eb015a2784ab123456'	The Pool IDs that give you entitlements to the Red Hat OpenStack Platform It provides the important configuration management and OpenStack integration with OpenShift.

# Chapter

# 4

---

## Configure Dynamic DNS

---

**Topics:**

- [Overview](#)
- [Prepare the DNS Environment](#)
- [Deploy DNS](#)
- [Verify DNS Functionality](#)

This topic describes the procedures you will use to configure Dynamic DNS (DDNS).

## Overview

The installation process for Red Hat Open Container Platform (RHOCP) depends upon a reliable name service that contains an address record for each of the target instances. When installing RHOCP in a cloud environment, where the IP address of each instance is not known at the beginning, the address record for each instance must be created dynamically by the orchestration system as the instances themselves are created. Red Hat OpenStack Platform (RHOSP) does not have an integrated DNS service, but it is possible to create new records in a DNS service using dynamic updates as defined in [RFC2137](#). This method uses a symmetric key to authenticate and authorize updates to a specific zone.

Installation of RHOCP on RHOSP requires a DNS service capable of accepting DNS update records for the new instances. This section describes a method to create a suitable DNS service within RHOSP as a Heat stack. The resulting stack contains a DNS master and two DNS slaves. The DNS is capable of accepting DNS queries and update requests for the RHOSP service. It is suitable for delegation to make the RHOCP service accessible to users. The Ansible playbook that creates and configures the heat stack and configures the DNS service is part of the *openshift-ansible-contrib* repository on [Github](#). The RHOSP DNS service is in the *osp-dns* section for reference architectures.

## Prepare the DNS Environment

Follow these procedures to prepare the DNS installation environment:

1. [Download the DNS Playbook](#) on page 27
2. [Prepare the DNS YAML File](#) on page 27

### Download the DNS Playbook

To download the Ansible DNS playbook:

1. Log into the Director Node.
2. Execute the following command:

```
$ git clone https://github.com/openshift/openshift-ansible-contrib
```

### Prepare the DNS YAML File

The Ansible playbook requires a set of inputs that define the characteristics of the DNS service. These inputs are provided as a YAML formatted data file.

The input parameters are broadly divided into three groups:

- DNS Service settings
- OpenStack host parameters
- Software update subscription

To configure DNS:

1. Open the *dns-vars.yaml* file in a text editor.
2. Edit the values below so they are updated for your target environment.

```
domain_name: r11b.oss.labs
contact: admin
dns_forwarders: [100.82.32.10, 8.8.8.8]
update_key: "kOZMgt3EDHJnHl9cFUJZQ=="
slave_count: 2
```

```

stack_name: dns-service
external_network: public

image: rhel73
flavor: m1.small
ssh_user: cloud-user
ssh_key_name: ocp3

# NOTE: For Red Hat Enterprise Linux:
rhn_username: "username"
rhn_password: "correctpassword"
rhn_pool: "8a85f98c5a2714eb015a2bbbbbeed"
# Either RHN or Sat6
# sat6_hostname: ""
# sat6_organization: ""
# sat6_activationkey: ""

```

- Review [Table 4: dns-vars.yaml Parameters](#) on page 28 to ensure the correct parameters for your environment.



**Note:** The values below are stamp-specific examples. Your environment may require different values.

**Table 4: dns-vars.yaml Parameters**

Parameter: Example Value	Description
domain_name: r11b.oss.labs	This is the domain that the server will manage. It must be a fully-qualified domain name (FQDN). If it will be delegated, then it must be a proper sub-domain of an established parent domain.
dns_forwarders: [100.82.32.10, 8.8.8.8]	This is a comma-separated list of IP addresses enclosed in square brackets. Each must be the address of a DNS server that is capable of accepting forwarded DNS queries.
update_key: "kOZMgt3EDHJnHI9cFUJZQ=="	This is a DNS TSIG signing key. The key can be generated using <code>ddns-confgen</code> or <code>rndc-confgen</code> . See <a href="#">Generate a DNS Update Key</a> on page 29.
slave_count: 1	The slave count defines the number of DNS slaves. A functional DNS service can have no slaves, but for legal delegation a service must have a master and at least 1 slave. The default is 2.
stack_name: dns-service	This value defines the name of the Heat stack that is created. It defaults to <code>dns-service</code> .
external_network: public	This value must be the name of the public or external network in the Red Hat OpenStack Platform service.
image: rhel73 flavor: m1.small ssh_user: cloud-user ssh_key_name: ocp3	These parameters set the characteristics of the instances which host the DNS service. The image and flavor must already be defined in the Red Hat OpenStack Platform service. The <code>ssh_user</code> is the user which allows login with the matching key.

Parameter: Example Value	Description
rh_n_username: "username"	These parameters enable access to software updates. Use either the <code>rh_n</code> or <code>sat6</code> values, but not both.
rh_n_password: "password"	
rh_n_pool: "8a85f98c5a2714eb015a2bbbbbeed"	

## Deploy DNS

Follow these procedures to deploy DNS:

1. [Deploy the Service](#) on page 29
2. [Generate a DNS Update Key](#) on page 29

### Deploy the Service

The DNS service is defined as an Ansible playbook. This playbook creates a Heat stack that implements the network and the host instances for the DNS service. It then applies the DNS service configuration to each instance to complete the process.

To deploy DNS:

1. If not already logged in, log into the Director Node.
2. Execute the following commands:

```
$ export ANSIBLE_HOST_KEY_CHECKING=False
$ yum install ansible
$ ansible-playbook --private-key <keyfile> -e @dns-vars.yaml \
  openshift-ansible-contrib/reference-architecture/osp-dns/deploy-
  dns.yaml
```

### Generate a DNS Update Key

DNS updates require a special key string that can be generated by `ddns-confgen`, which is part of the `bind-utils` RPM. The only element required is the secret string.

To generate a DNS update key:

1. If not already logged in, log into the Director Node.
2. Execute the following commands:

```
$ rndc-confgen -a -c update.key -k update-key -r /dev/urandom
$ cat update.key key "update-key" {
  algorithm hmac-md5;
  secret "key string"; };
```

The significant part of the key output is the secret field. The value can be passed via the command line, or by setting and exporting the `DNS_UPDATE_KEY` environment variable for the deployment script that follows.

## Verify DNS Functionality

Verify both the operation of the DNS service using `dig`; and the ability to update the zone contents using `nsupdate`, which are both part of the `bind-utils` RPM.

The installation process reports the IP addresses of the DNS servers. The IP addresses can be retrieved using the `openstack server list` command as well.

1. If not already logged in, log into the Director Node.
2. Verify that the queries work, and that the nameserver *NS* and *A* records are present:

```
$ dig @100.82.x.x r11b.oss.labs axfr
```

3. Send the proper updates via the master server:

```
$ nsupdate -k update.key
server 100.82.x.x
zone r11b.oss.labs
update add test.r11b.oss.labs 300 A 1.2.3.4
send
quit
```

4. Verify that the new address is present:

```
$ dig @100.32.x.x add-test.r11b.oss.labs
```

5. Once the records are verified for accuracy, remove the test record:

```
$ nsupdate -k update.key
server 100.32.x.x
zone r11b.oss.labs
update delete test.r11b.oss.labs A
send
quit
```

---

# Chapter

# 5

---

## Installation and Configuration

---

### Topics:

- [Obtain the OpenShift Heat Templates RPM](#)
- [Execute Heat Templates](#)
- [Troubleshoot and Debug Failures](#)

Now that the solution installation environment is set up, you can install and configure the solution itself.

## Obtain the OpenShift Heat Templates RPM

Before you can execute the OpenShift Heat templates, you must first obtain the RPM.

To acquire the OpenShift Heat templates RPM:

1. Log into the Director Node.
2. Download and install `openshift-heat-templates-0.9.9-5.el7ost.noarch.rpm` for x86\_64 from <https://access.redhat.com/errata/RHEA-2017:1788> by executing the following command:

```
$ sudo yum install \
  openshift-heat-templates-0.9.9-5.el7ost.noarch.rpm
```

The template files are installed in `/usr/share/openshift-heat-templates`.

3. Update the configuration YAML files by executing the following commands:

```
$ sudo -i
# sed --in-place '451d;475,481d' \
  /usr/share/openshift-heat-templates/infra.yaml
# sed --in-place '443d;467,473d' \
  /usr/share/openshift-heat-templates/master.yaml
# sed --in-place '625d;650,657d' \
  /usr/share/openshift-heat-templates/node.yaml
```

4. Ensure the files are updated by executing the following command:

```
# cd /usr/share/openshift-heat-templates/
# grep node_etc_host */*
```

## Execute Heat Templates

This procedure strings together several YAML files to feed into Heat, in order to change the deployment configurations.

To execute the Heat templates:

1. Log into the Director Node.
2. Source the `overcloudrc` file by executing the following command:

```
$ . ./overcloudrc
```

3. Execute the templates with the following command, replacing `<stackname>` with a unique name without punctuation or numbers, to define your stack:



**Note:** As indicated in [Prepare the Heat YAML Files](#) on page 21, our example uses the stack name `stack`.

```
$ openstack stack create --wait --timeout 120 \
  -e openshift_parameters.yaml \
  -e /usr/share/openshift-heat-templates/env_loadbalancer_dedicated.yaml \
  -t /usr/share/openshift-heat-templates/openshift.yaml ocp3
```

You will see output for about 30 minutes over the course of the entire installation. The `--poll` argument ensures that you see updates from the Heat subsystem.

## Troubleshoot and Debug Failures

If the stack-create operation fails you will see some basic debugging output:

- If the problem was in OpenStack, the error message will likely be very clear and helpful.
- If the error message is in the OpenShift Ansible deployment portion, it is likely to be obscure.
- If you encounter the following error:

```
failed to connect to os-collect-config metadata urls
```

1. Verify your network configurations.
2. See [https://bugzilla.redhat.com/show\\_bug.cgi?id=1452677](https://bugzilla.redhat.com/show_bug.cgi?id=1452677) for more information.

Further information can be found in the troubleshooting document in the *openshift-on-openstack* code repository: [https://github.com/redhat-openstack/openshift-on-openstack/blob/master/README\\_debugging.adoc](https://github.com/redhat-openstack/openshift-on-openstack/blob/master/README_debugging.adoc).

Some common troubleshooting information is presented in the following topics:

- [Heat Events](#) on page 33
- [Debugging in Proper Order](#) on page 33
- [About Ansible Log Files](#) on page 34

### Heat Events

Here are some common Heat events troubleshooting steps:

1. Get a list of all the events leading up to your error, by executing the following command on the Director.

```
# openstack stack event list --nested-depth 2 ocp3
```

2. Ensure that all services are running correctly. See the [Dell EMC Ready Bundle for Red Hat OpenStack Platform Software Deployment Guide - Version 10.0.1](#) for suggestions about how to fix these errors.
3. You see the following error message:

```
Error: No such flavor
```

You might be set into the Undercloud, not the Overcloud. Remember to source the *overcloudrc* file with the following command:

```
# . ./overcloudrc
```

### Debugging in Proper Order

Proper order is crucial to successfully debugging the installation.

Follow the procedures below in the order presented:

1. Watch `heat event-list <stackname>` for failing events until the Bastion VM comes up. This takes about 10 seconds.
2. Log into the Bastion VM, and watch `/var/log/cloud-init.log` while it configures itself and prepares Ansible. This takes about 15 minutes.
3. From the Director Node or a Controller, watch `heat event-list <stackname>` again, for it to create the masters and nodes VMs.  
This takes about 10 minutes, until the long wait at "openshift-nodes" with the notification of "loadbalancer" CREATE\_COMPLETE.
4. Log into the Bastion VM again, and watch `/var/log/ansible*` for Ansible events.
  - a) Find failing events with `grep '^failed:' /var/log/ansible*`, or

b) Search for `^failed:` in an editor or pager of your choice.

You may have to log into the individual VMs, to track their `/var/log/cloud-init.log` and `/var/log/messages` for Ansible logging locally.

## About Ansible Log Files

Ansible runs on the Infra VM, and installs servers independently to hasten deployment. There can be up to three Ansible log files. If there are none, then your installation did not succeed in `/var/log/cloud-init.log`, so you should be looking there first.

The `/var/log/cloud-init.log` files from the masters and nodes are not brought to the Infra VM. If there is an installation problem early in the boot process of the masters or nodes, you must `ssh` to them to examine them.

The `/var/log/ansible.*` files are **very** verbose. You can more easily watch Ansible progress by executing the following command:

```
$ tail -f /var/log/ansible.* | grep '^TASK'
```

---

# Chapter 6

---

## Validate the Installation

---

**Topics:**

- [List the Nodes](#)
- [Ensure a Working Router](#)
- [Deploy the Router Pod to a Master](#)

This topic describes the procedures you will use to validate the installation.

## List the Nodes

Validate the installation by making sure that the system knows about its components.

To list the nodes:

1. From the Director Node, `ssh` to a Controller node.
2. Source the `overcloudrc` file:

```
$ . ./overcloudrc
```

3. Display the nodes' floating IP addresses by executing the following command:

```
$ nova list
```

4. Then `ssh` to the floating IP address of a Master node from the list:

```
$ ssh -i ocp3.pem cloud-user@100.82.35.215
```

5. List the nodes by executing the following command:

```
$ oc get nodes
```

The output should display as many masters and nodes as you requested in the Heat template.

## Ensure a Working Router

Sometimes the Ansible installer does not properly deploy the router. Check it by ensuring that there is a pod associated with it.

To check the router status:

Execute the following command:

```
$ oc status
In project default on server https://ocp3-devs.r11b.oss.labs:8443

https://docker-registry-default.apps.r11b.oss.labs (passthrough) to pod
port 5000-tcp (svc/docker-registry)
dc/docker-registry deploys docker.io/openshift3/ose-docker-
registry:v3.4.1.44
deployment #2 deployed 21 hours ago - 1 pod
deployment #1 failed 21 hours ago: newer deployment was found running

svc/kubernetes - 172.30.0.1 ports 443, 53, 53

https://registry-console-default.apps.r11b.oss.labs (passthrough) to pod
port registry-console (svc/registry-console)
dc/registry-console deploys registry.access.redhat.com/openshift3/
registry-console:3.4
deployment #1 deployed 21 hours ago - 1 pod

svc/router - 172.30.102.211 ports 80, 443, 1936
dc/router deploys docker.io/openshift3/ose-haproxy-router:v3.4.1.44
deployment #1 deployed 21 hours ago - 1 pod
```

```
View details with 'oc describe <resource>/<name>' or list everything with 'oc get all'.
```

If there are no pods in the router deployment, you must perform the procedures in [Deploy the Router Pod to a Master](#) on page 37.

## Deploy the Router Pod to a Master

To deploy the router pod to a Master node:

 **Note:** Our example uses *Master-0*.

1. List the nodes:

```
$ oc get nodes
NAME STATUS AGE
ocp3-infra-0.r11b.oss.labs Ready 1h
ocp3-master-0.r11b.oss.labs Ready,SchedulingDisabled 1h
ocp3-master-1.r11b.oss.labs Ready,SchedulingDisabled 1h
ocp3-node-15e15vq5.r11b.oss.labs Ready 1h
```

2. Set the *Master-0* node to schedulable:

```
$ oadm manage-node --ocp3-master-0.r11b.oss.labs
schedulable=true
NAME STATUS AGE
ocp3-master-0.r11b.oss.labs Ready 1h
```

3. Verify that *Master-0* is now schedulable:

```
$ oc get nodes
NAME STATUS AGE
ocp3-infra-0.r11b.oss.labs Ready 1h
ocp3-master-0.r11b.oss.labs Ready 1h
ocp3-master-1.r11b.oss.labs Ready,SchedulingDisabled 1h
ocp3-node-15e15vq5.r11b.oss.labs Ready 1h
```

4. Delete the existing router deployment configuration and service, and redeploy the router.

a) Examine the services created by the `oc adm router` command. Most importantly, you should see "1 pod" running under the `dc/router`.

```
$ oc delete dc router; oc delete service router; oc adm router --
selector="region=infra"
```

5. Watch the pods that are available for the complete creation of the OpenShift Router on one of the Master node VMs:

```
$ oc get pods
NAME READY STATUS RESTARTS AGE
router-2-198lm 0/1 ContainerCreating 0 8s
$ oc get pods
NAME READY STATUS RESTARTS AGE
router-2-198lm 1/1 Running 0 44s
$ oc status
In project default on server https://ocp3-devs.r11b.oss.labs:8443
svc/kubernetes - 172.30.0.1 ports 443, 53, 53
svc/router - 172.30.15.96 ports 80, 443, 1936
dc/router deploys docker.io/openshift3/ose-haproxy-router:v3.4.0.44
deployment #2 deployed 4 minutes ago - 1 pod
```

```
deployment #1 deployed about an hour ago
```

The OpenShift router is now properly deployed.

---

# Chapter 7

---

## Configure a Docker Registry with Persistent Storage

---

**Topics:**

- [Create a Cinder Volume](#)
- [Configure Red Hat Ceph Storage and Cinder Persistent Storage](#)
- [Create the Docker-Registry](#)
- [Create a Sample User in OpenShift](#)

This topic describes the procedures you will follow to configure a Docker registry with persistent storage.

## Create a Cinder Volume

From the Director Node, you can create a Cinder Volume for the registry.

To create a Cinder volume:

1. Check to see if there was already a registry created. If so, note its ID.



**Note:** In this example, a Cinder volume for the docker-registry was created: `osdell-registry_volume-cr4v3l33qm5n`.

```
$ cinder list | grep registry_volume

| 94249a7c-bdc3-43f0-823e-126ec77fcb76 | in-use | ocp3-registry_volume-
qpkjty6pmdhk-volume-b3yuh5kjiat3 | 10 | - | false |
df93e42d-090d-43fd-855d-2f7bcf68238c |
```

2. If no volume was created, create one and note its ID:

```
$ cinder create --name docker-registry 100
```

The output will contain information similar to [Table 5: Cinder Volume Values](#) on page 40.

**Table 5: Cinder Volume Values**

Property	Value
attachments	[]
availability_zone	nova
bootable	false
consistencygroup_id	None
created_at	2017-07-23T02:49:46.000000
description	None
encrypted	False
id	81fd05bf-8653-45eb-baf4-6df7fffb643e
metadata	{}
migration_status	None
multiattach	False
name	docker-registry
os-vol-host-attr:host	rbd:volumes@tripleo_ceph#tripleo_ceph
os-vol-mig-status-attr:migstat	None
os-vol-mig-status-attr:name_id	None
os-vol-tenant-attr:tenant_id	c1b740d8571f4cb5aa66f8ddcfdec015
os-volume-replication:driver_data	None
os-volume-replication:extended_status	None
replication_status	disabled

Property	Value
size	100
snapshot_id	None
source_volid	None
status	creating
user_id	3d0e3655d9224e5292b5f0cd3646a5ac
volume_type	None

## Configure Red Hat Ceph Storage and Cinder Persistent Storage

To configure Red Hat Ceph Storage and Cinder persistent storage for the registry:

1. Access an OpenShift master.
2. Create a new file, called *pv.yaml*.
  - a) Paste the values below into *pv.yaml*. Be sure to substitute the `volume_ID` at the end for the `id` from [Table 5: Cinder Volume Values](#) on page 40

 **Note:** Ensure that the following formatting pastes correctly into your new file. Use two spaces for each level of indentation. This is YAML.

```
apiVersion: "v1"
kind: "PersistentVolume"
metadata:
  name: "docker-reg-volume"
spec:
  capacity:
    storage: "100Gi"
  accessModes:
    - "ReadWriteOnce"
  cinder:
    fsType: "ext3"
    volumeID: "<volume_ID>"
```

3. Create the persistent volume objects, and verify them by executing the following commands:

```
# oc create -f pv.yaml
persistentvolume "docker-reg-volume" created

# oc get pv
NAME                CAPACITY  ACCESSMODES  STATUS      CLAIM REASON  AGE
docker-reg-volume  100Gi    RWO          Available   36s
```

4. Have the *docker-registry* pod claim that persistent volume by creating a Persistent Volume Claim file, called *pvc.yaml*, on the same Master host:

 **Note:** Ensure that the following formatting pastes correctly into your new file. Use two spaces for each level of indentation. This is YAML.

```
apiVersion: "v1"
kind: "PersistentVolumeClaim"
metadata:
  name: "docker-registry-pvc1"
spec:
  accessModes:
```

```
- "ReadWriteOnce"
resources:
  requests:
    storage: "100Gi"
```

5. Create those Persistent Volume Claim objects, and verify them by executing the following commands on the Master host:

```
# oc create -f pvc.yaml
persistentvolumeclaim "docker-registry-pvc1" created

# oc get pvc
NAME                                STATUS VOLUME                CAPACITY ACCESSMODES AGE
docker-registry-pvc1 Bound   docker-reg-volume 100Gi    RWO      2d
```

## Create the Docker-Registry

To create the `docker-registry` right the first time, you must create a custom `docker-registry` YAML file with the proper IP address.

 **Note:** Other pods cannot access this registry unless it resides on a network managed by Flannel; only one network per host is managed by Flannel. That network must be selected for the `docker-registry` to be accessible.

To create the `docker-registry`:

1. On a Master node, create a registry YAML file, called, `reg.yaml`:

```
# oc create -f pvc.yaml
persistentvolumeclaim "docker-registry-pvc1" created

# oc get pvc
NAME                                STATUS VOLUME                CAPACITY ACCESSMODES AGE
docker-registry-pvc1 Bound   docker-reg-volume 100Gi    RWO      2d
```

2. Find an appropriate `clusterIP` address by finding out which `172.30.x.0/24` address range is hosted on **this** server.
  - a) Note which range is on the `docker0` bridge interface.
  - b) In the example below, `172.30.9.0/24` is on the `docker0` bridge interface. So, choose an unused IP address on the `172.30.9.0/24` network.
  - c) `192.168.9.215` appears to be unused, so check that by pinging that IP address and waiting for a failed response, indicating that there is no host listening on the IP address.
  - d) That will be the `clusterIP` address, since that is Kubernetes' terminology.

```
# ip r

default via 192.168.10.1 dev eth0 proto static metric 100
10.10.0.0/24 dev eth1 proto kernel scope link src 10.10.0.5 metric 100
172.30.9.0/24 dev docker0 proto kernel scope link src 172.30.9.1
172.30.21.0/24 via 10.10.0.4 dev eth1
172.30.90.0/24 via 10.10.0.8 dev eth1
172.30.92.0/24 via 10.10.0.7 dev eth1
172.30.94.0/24 via 10.10.0.6 dev eth1
192.168.10.0/24 dev eth0 proto kernel scope link src 192.168.10.7 metric 100
```

3. Edit `~/reg.yaml` to add a `clusterIP` and `portalIP` (same as `clusterIP`) address to the "**kind: Service**" section as in the following example, then save the file:

-  **Note:** Ensure that the following formatting pastes correctly into your new file. Use two spaces for each level of indentation. This is YAML.

```
apiVersion: v1
kind: Service
metadata:
  creationTimestamp: null
  labels:
    docker-registry: default
  name: docker-registry
spec:
  clusterIP: 172.30.9.215
  port: 5000
  ports:
  - name: 5000-tcp
    port: 5000
    targetPort: 5000
```

4. Build the `docker-registry` objects, and kick off the build and deployment of the `docker-registry` pods and containers, by executing the following command:

```
# oc create -f ~/reg.yaml
```

5. Watch the deployment build for success by executing the following command:

-  **Note:** It might take some time for the results to appear. You can keep checking by executing the `oc status` command several times.

```
#oc status

In project default on server https://ocp3-devs.r11b.oss.labs:8443
svc/docker-registry - 172.30.105.235:5000

dc/docker-registry deploys registry.access.redhat.com/openshift3/ose-
docker-registry:v3.4.0.44
deployment #1 deployed 3 minutes ago - 1 pod
svc/kubernetes - 172.30.0.1 ports 443, 53, 53
svc/router - 172.30.15.96 ports 80, 443, 1936
dc/router deploys docker.io/openshift3/ose-haproxy-router:v3.4.0.44

deployment #2 deployed 15 minutes ago - 1 pod
deployment #1 deployed about an hour ago

View details with 'oc describe <resource>/<name>' or list everything with
'oc get all'.
```

6. Now that the `docker-registry` is running, update it to use the Persistent Volume Claim by executing the following commands on the Master host:

```
# oc volume deploymentconfigs/docker-registry --add \
--name=docker-registry-volume -t pvc \
--claim-name=docker-registry-pvc1 \
--overwrite deploymentconfigs/docker-registry

# oc describe dc/docker-registry
-----Output truncated-----
Volumes:
  registry-storage:
    Type:          EmptyDir (a temporary directory that shares a pod's
lifetime)
```

```

Medium:
docker-registry-volume:
  Type: PersistentVolumeClaim (a reference to a
PersistentVolumeClaim in the same namespace)
  ClaimName: docker-registry-pvc1
  ReadOnly: false
-----Output truncated-----

```

7. You can observe the `docker-registry-2-deploy` pod coming up to deploy your `docker-registry` again with the persistent storage attached, by executing the following commands:

```

# oc status
In project default on server https://ocp3-devs.r11b.oss.labs:8443
svc/docker-registry - 172.30.105.235:5000
dc/docker-registry deploys registry.access.redhat.com/openshift3/ose-
docker-registry:v3.4.0.44

deployment #2 pending 41 seconds ago
deployment #1 deployed 34 minutes ago - 1 pod
svc/kubernetes - 172.30.0.1 ports 443, 53, 53
svc/router - 172.30.15.96 ports 80, 443, 1936
dc/router deploys docker.io/openshift3/ose-haproxy-router:v3.2.0.44
deployment #2 deployed 46 minutes ago - 1 pod
deployment #1 deployed 2 hours ago
View details with 'oc describe <resource>/<name>' or list everything with
'oc get all'.
# oc get pods
NAME READY STATUS RESTARTS AGE
docker-registry-1-vghsw 1/1 Running 0 33m
docker-registry-2-deploy 0/1 ContainerCreating 0 46s
router-2-198lm 1/1 Running 0 43m

```

8. Examine the `docker-registry`, and ensure that its second deployment comes up cleanly, by executing the following commands:

```

# oc get pods
NAME READY STATUS RESTARTS AGE
docker-registry-2-xjmuo 1/1 Running 0 2m

# oc describe pod docker-registry-2-xjmuo

-----Output truncated-----

Events: (truncated to show only From and Message fields)
  From Message
  ----
  {default-scheduler} Successfully
assigned docker-registry-2-xjmuo to ocp3-node-vm4047k3.example.com
  {kubelet ocp3-node-vm4047k3.example.com} Container image
"registry.access.redhat.com/openshift3/ose-docker-registry:v3.2.0.20"
already present on machine
  {kubelet ocp3-node-vm4047k3.example.com} Created container with docker
id 2a204472c9fc
  {kubelet ocp3-node-vm4047k3.example.com} Started container with docker
id 2a204472c9fc

# oc describe service docker-registry
Name: docker-registry
Namespace: default
Labels: docker-registry=default
Selector: docker-registry=default
Type: ClusterIP

```

```

IP: 172.30.19.92
Port: 5000-tcp 5000/TCP
Endpoints: 10.1.6.2:5000
Session Affinity: ClientIP
No events.

# oc get pods
NAME READY STATUS RESTARTS AGE
docker-registry-1-vghsw 1/1 Running 0 34m
docker-registry-2-deploy 1/1 Running 0 1m
docker-registry-2-po9az 0/1 ContainerCreating 0 28s
router-2-198lm 1/1 Running 0 43m

# oc get pods
NAME READY STATUS RESTARTS AGE
docker-registry-2-po9az 1/1 Running 0 55s
router-2-198lm 1/1 Running 0 44m

```

 **Note:** By using PersistentVolumes, these docker-registry pods can fail at any time, and the underlying replicated storage is still intact. OpenShift will quickly bring up another docker-registry, thanks to the deployment configuration: docker-registry.

9. To test that the docker-registry has been properly deployed and networked, curl the docker-registry **ClusterIP** that you set above from the Master host, and from any of the other OpenShift masters or nodes:

```

# curl -v http://<ClusterIP>:5000
* About to connect() to 172.30.207.92 port 5000 (#0)
* Trying 172.30.207.92...
* Connected to 172.30.207.92 (172.30.207.92) port 5000 (#0)
> GET / HTTP/1.1
> User-Agent: curl/7.29.0
> Host: 172.30.207.92:5000
> Accept: */*
>
< HTTP/1.1 200 OK

```

- a) If you can access the **ClusterIP** only from one of the masters and not from any of the other masters and nodes, then add the route in all the other master and OpenShift nodes:

```
# ip r add <ClusterIP> via <docker0IP> dev docker0
```

10. Make sure your registry is populated by image streams and templates, **and very importantly**, have the same correct ClusterIP for the docker-registry, by executing the following commands on a Master VM:

```

# oc get is
NAME DOCKER REPO TAGS UPDATED
jenkins 172.30.21.10:5000/default/jenkins
mongodb 172.30.21.10:5000/default/mongodb
mysql 172.30.21.10:5000/default/mysql
nodejs 172.30.21.10:5000/default/nodejs
perl 172.30.21.10:5000/default/perl
php 172.30.21.10:5000/default/php
postgres 172.30.21.10:5000/default/postgres
python 172.30.21.10:5000/default/python
ruby 172.30.21.10:5000/default/ruby

```

11. If the docker-registry's ClusterIP does not match the docker-registry ClusterIP, or there is no output from the command, perform the following steps:

- a) Remove all the image streams and templates by executing the following once on a Master VM:

```
# oc delete is --all
```

- b) On **each** Master, restart all the OpenShift processes:

```
# systemctl restart atomic-openshift-node
# systemctl restart atomic-openshift-master-controllers
# systemctl restart atomic-openshift-master-api
```

- c) Recreate the image streams by executing the following commands on a master-0

```
# oc create -f /usr/share/openshift/examples/image-streams/ \
image-streams-rhel7.json

# oc get is
```

NAME	DOCKER REPO	TAGS	UPDATED
jenkins	172.30.21.10:5000/default/jenkins	1,latest	8 minutes ago
mongodb	172.30.21.10:5000/default/mongodb	2.4,2.6,latest	8 minutes ago
mysql	172.30.21.10:5000/default/mysql	5.5,5.6,latest	8 minutes ago
nodejs	172.30.21.10:5000/default/nodejs	0.10,latest	7 minutes ago
perl	172.30.21.10:5000/default/perl	5.16,5.20,latest	8 minutes ago
php	172.30.21.10:5000/default/php	5.5,5.6,latest	7 minutes ago
postgresql	172.30.21.10:5000/default/postgre..	latest,9.2,9.4	8 minutes ago
python	172.30.21.10:5000/default/python	2.7,3.3,3.4 ...	8 minutes ago
ruby	172.30.21.10:5000/default/ruby	latest,2.0,2.2	8 minutes ago

- d) Reinststate the templates by executing the following commands on a master-0:

```
# ls /usr/share/openshift/examples/db-templates/*.json | xargs -n 1 \
oc create -f template "mariadb-ephemeral" createdtemplate \
"mariadb-persistent" createdtemplate "mongodb-ephemeral" \
createdtemplate "mongodb-persistent" \
createdtemplate "mysql-ephemeral" createdtemplate \
"mysql-persistent" createdtemplate "postgresql-ephemeral" \
createdtemplate "postgresql-persistent" createdtemplate \
"redis-ephemeral" createdtemplate "redis-persistent" created
```

- e) Reinststate the quickstart-templates by executing the following commands on a master-0:



**Note:** Descriptions are truncated in the output below.

```
# ls /usr/share/openshift/examples/quickstart-templates/*.json | \
xargs -n 1 oc create -f
template "cakephp-mysql-example" created
template "cakephp-mysql-persistent" created
template "dancer-mysql-example" created
template "dancer-mysql-persistent" created
template "django-psql-example" created
template "django-psql-persistent" created
template "dotnet-example" created
template "dotnet-pgsql-persistent" created
template "jenkins-ephemeral" created
```

```

template "jenkins-persistent" created
template "nodejs-mongodb-example" created
template "nodejs-mongo-persistent" created
template "rails-postgresql-example" created
template "rails-pgsql-persistent" created
# oc get templates

```

NAME	DESCRIPTION	PARAMETERS	OBJECTS
cakephp-mysql-example	An example C...	19 (4 blank)	8
cakephp-mysql-persistent	An example C...	20 (4 blank)	9
dancer-mysql-example	An example D...	16 (5 blank)	8
dancer-mysql-persistent	An example D...	17 (5 blank)	9
django-psql-example	An example D...	17 (5 blank)	8
django-psql-persistent	An example D...	18 (5 blank)	9
dotnet-example	An example ....	16 (5 blank)	5
dotnet-pgsql-persistent	An example ....	23 (6 blank)	8
jenkins-ephemeral	Jenkins serv...	7 (all set)	6
jenkins-persistent	Jenkins serv...	8 (all set)	7
mariadb-ephemeral	MariaDB data...	7 (3 generated)	3
mariadb-persistent	MariaDB data...	8 (3 generated)	4
mongodb-ephemeral	MongoDB data...	8 (3 generated)	3
mongodb-persistent	MongoDB data...	9 (3 generated)	4
mysql-ephemeral	MySQL databa...	8 (3 generated)	3
mysql-persistent	MySQL databa...	9 (3 generated)	4
nodejs-mongo-persistent	An example N...	17 (4 blank)	9
nodejs-mongodb-example	An example N...	16 (4 blank)	8
postgresql-ephemeral	PostgreSQL d...	7 (2 generated)	3
postgresql-persistent	PostgreSQL d...	8 (2 generated)	4
rails-pgsql-persistent	An example R...	21 (4 blank)	9
rails-postgresql-example	An example R...	20 (4 blank)	8
redis-ephemeral	Redis in-mem...	5 (1 generated)	3
redis-persistent	Redis in-mem...	6 (1 generated)	4

You have now successfully created a `docker-registry` in the OpenStack cluster.

## Create a Sample User in OpenShift

The final procedure before logging into the OpenShift GUI is to create a sample user in OpenShift.

To create a sample user in OpenShift:

1. Execute the following command on **all** of the OpenShift masters, creating a password of your choosing.

 **Caution:** You must use the same one on each host.

```

# htpasswd /etc/origin/openshift-passwd osadmin
New password:
Re-type new password:
Adding password for user osadmin

```

You can now proceed to [Configure OpenShift Web GUI Access](#) on page 48.

---

# Chapter

# 8

---

## Configure OpenShift Web GUI Access

---

### Topics:

- [Make DNS World Accessible](#)
- [Add DNS to the Windows Bastion Host](#)
- [Navigate to the OpenShift Web Console](#)

This topic describes the procedures you will perform to configure OpenShift web GUI access.

## Make DNS World Accessible

To ensure that DNS is accessible by everyone:

1. On the Director Node execute the following command:

```
$ nova list
```

2. Note the floating IP address of the OpenShift *master-0* VM.
3. `ssh` into your Dynamic DNS (DDNS) Master VM.
4. Fix `named/bind` to accept queries from anywhere, by editing the `/etc/named.conf` file, changing the `allow-query` value to be **any**;

```
# vi /etc/named.conf
allow-query { any; };
```

## Add DNS to the Windows Bastion Host

To add DNS to your Windows bastion host:

1. In Windows, navigate to **Network and Sharing Center**, then select **Change Adapter Settings**.
2. Right-click on the **adapter that you use to connect to the OpenShift console** (usually the public network), then choose **Properties**.
3. Double-click on **Internet Protocol Version 4**, then change the **Preferred DNS Server** to the public IP address of the Dynamic DNS Master VM.
4. Click on **OK**, then on **OK** again to close the Network Properties dialogue box and ensure that the setting was saved.
5. Run `cmd.exe` to display a command console.
6. Ensure that the DNS server was accepted by the configuration, and applied to the Network settings, by executing the following command:

```
C:\> ipconfig /all
```

7. Ensure network connectivity by executing the following command:

```
C:\> ping <ddns host IP address>
```

8. Execute the following command to return the IP address of the load balancer:

```
C:\> nslookup <lb FQDN> <ddns host IP address>
```

## Navigate to the OpenShift Web Console

To log into the console on the bastion host:

1. In a Web browser, navigate to `ocp3-devs.r11b.oss.labs/console`.
2. Accept the Security Exception for the self-signed certificate.
3. Login with these credentials:
  - a. Username: `osadmin`

- b. Password: *password* (the password that you set, above)
4. For CLI access to the console, use the following commands:

```
$ oc login ocp3-heat-devs.ocp3.example.com --username openshift \
  --insecure-skip-tls-verify

Authentication required for https://ocp3-heat-devs.ocp3.example.com:8443
(openshift)
Username: openshift
Password:
Login successful.
Using project "test-project".
```

5. Do not log out - you will need this access for the next procedure, [Deploy a Sample Application](#) on page 51.



**Note:** If the URL is still not accessible, try adding the URL, with its public IP address, in the *hosts* file on your Windows computer; and then retry.

---

# Chapter 9

---

## Deploy a Sample Application

---

**Topics:**

- [Application Deployment Procedure](#)

Now you can deploy a sample application in the OpenShift Container Platform web GUI.

## Application Deployment Procedure

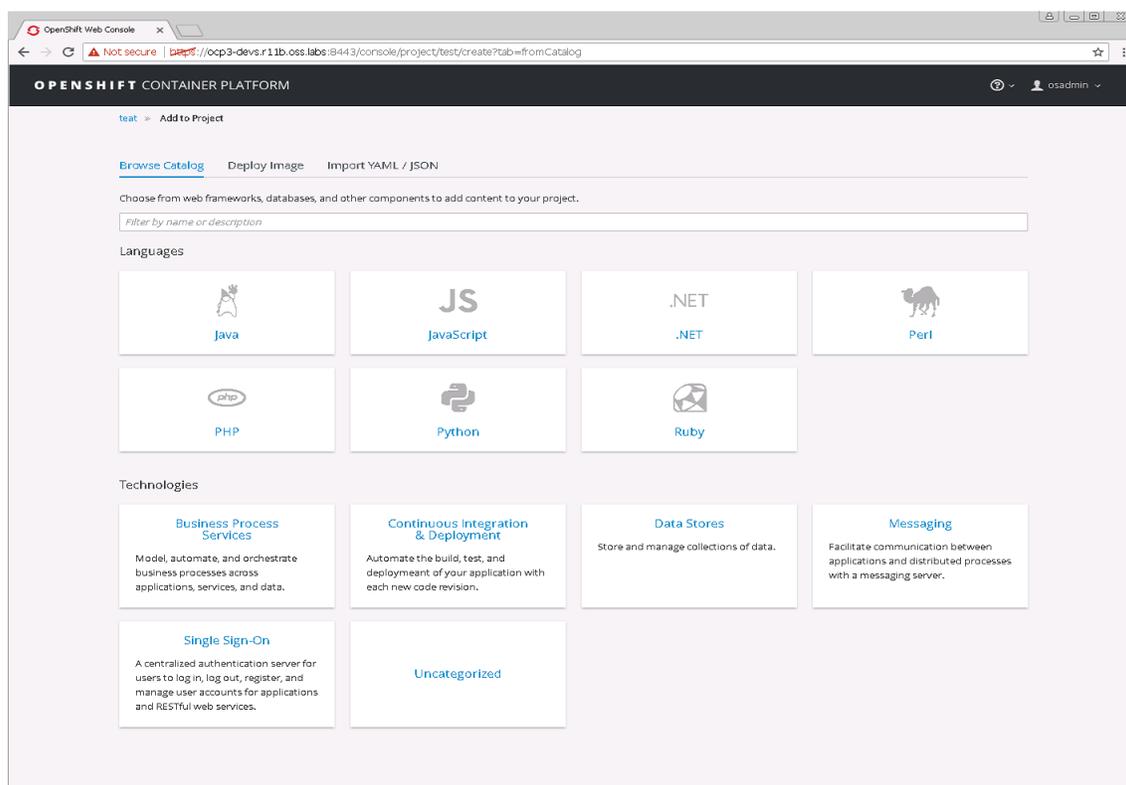
To deploy a sample application:

1. Log into the OpenShift Enterprise console.
2. Create a new project with a name that you like.



**Note:** Projects are namespaces in which you deploy one or more applications that will share resources.

3. Click on the **Browse Catalog** link to explore the Quickstart catalog, from which you can launch a sample application. See [Figure 2: Browse OpenShift Container Platform Catalog](#) on page 52.



**Figure 2: Browse OpenShift Container Platform Catalog**

4. Create a new application from the Quickstart Templates. See [Figure 3: Example PHP Application](#) on page 53.

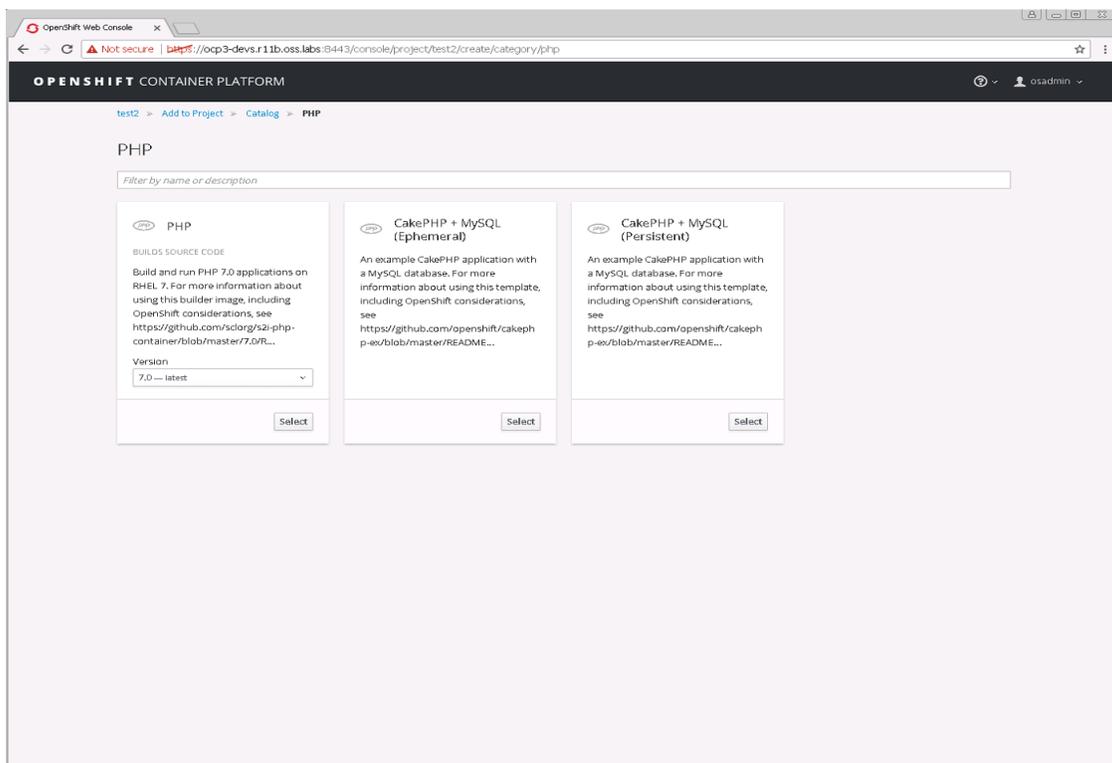


Figure 3: Example PHP Application

5. Accept all the defaults and click on **Create**. See [Figure 4: New Application](#) on page 53.

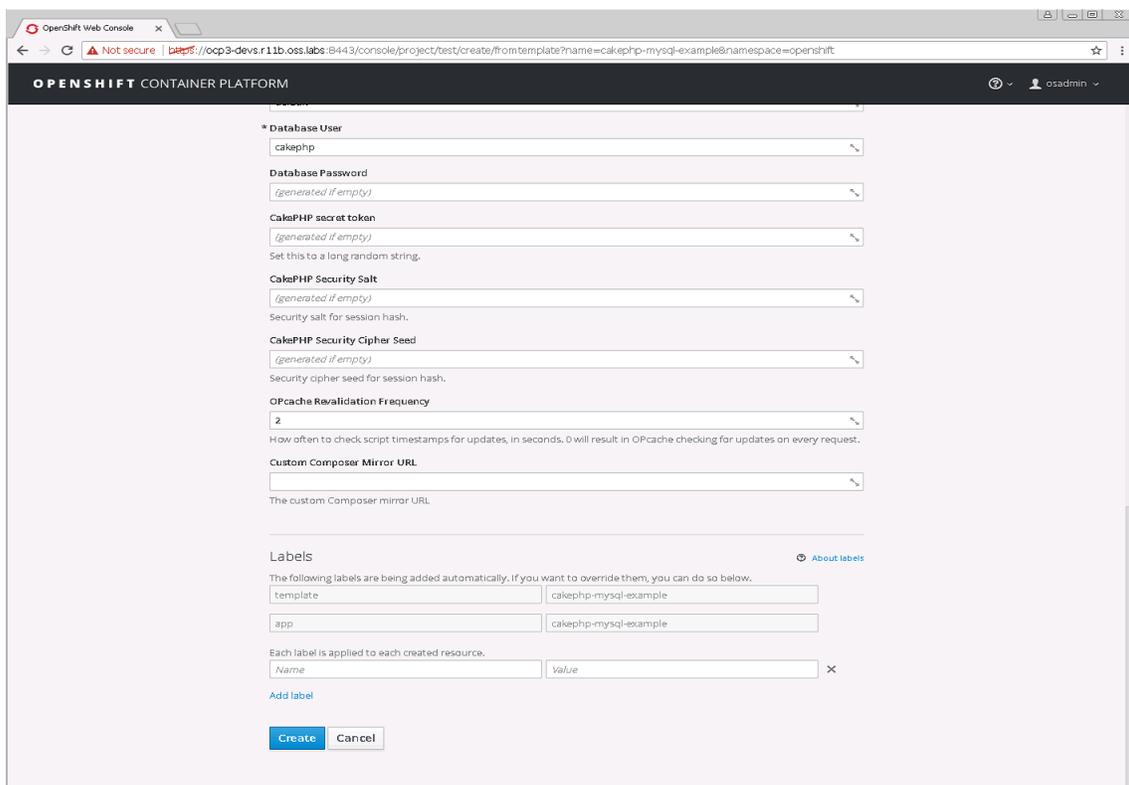
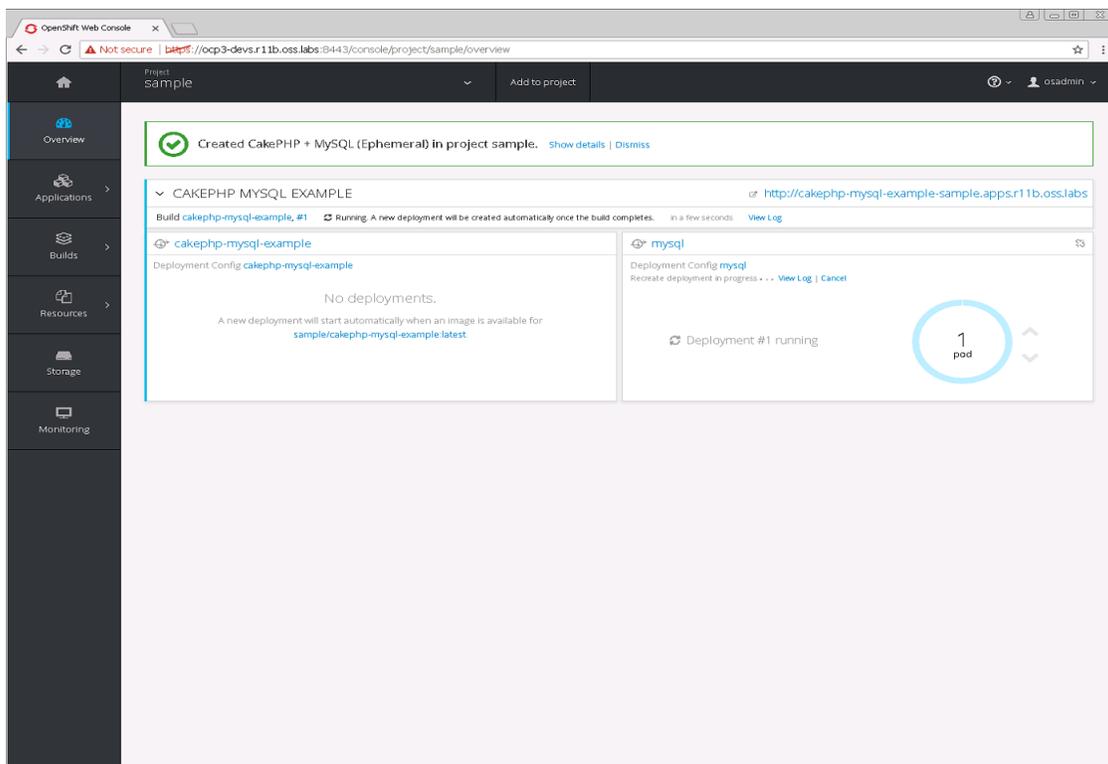


Figure 4: New Application

- Click through the informational screen, and watch the application build on the overview screen. See [Figure 5: Overview](#) on page 54.

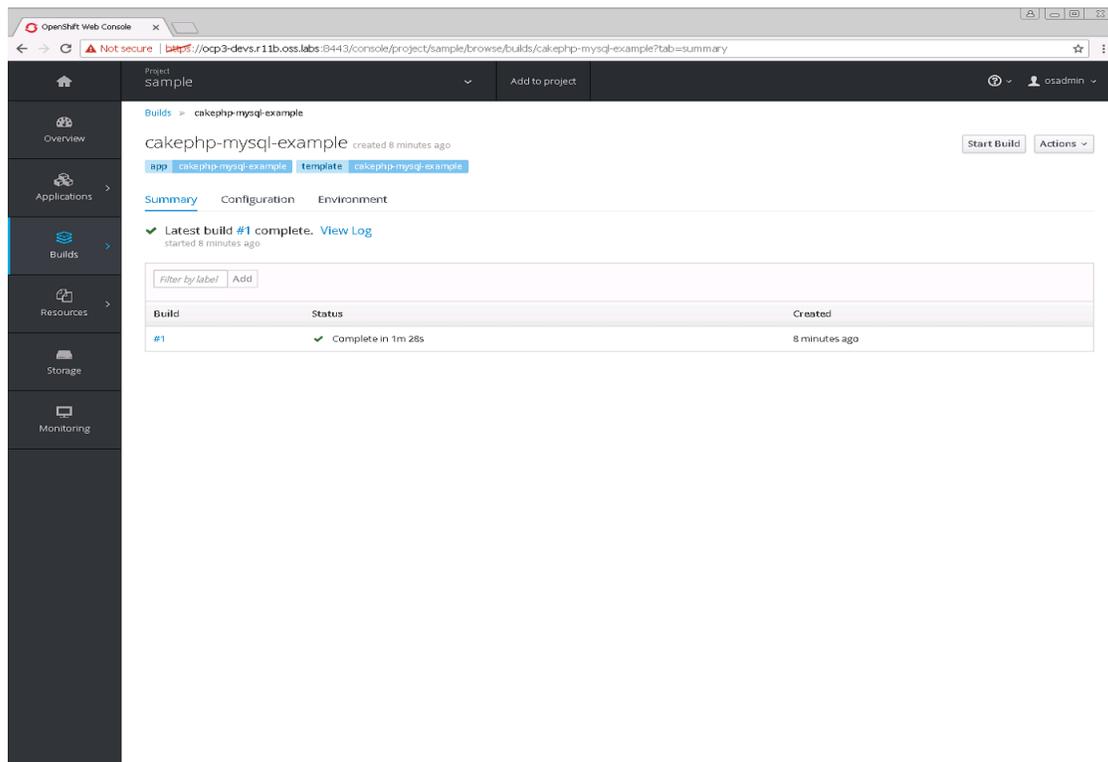


**Figure 5: Overview**

- In a matter of moments, the build is complete; the deployment completes shortly thereafter. See [Figure 6: Build Complete](#) on page 55.



**Note:** You can click the **View Log** link to see the progress.



**Figure 6: Build Complete**

8. Log into one of the master nodes and verify the application is running.



**Note:** You can access the application by clicking on its name, provided that the application name is properly resolved.

```
$ oc login https://ocp3-devs.r11b.oss.labs:8443
```

- a) Access the project with its name:

```
$ oc project test2
```

- b) Check the project's status:

```
$ oc status

In project test2 on server https://ocp3-devs.r11b.oss.labs:8443

http://py-test2.apps.r11b.oss.labs to pod port 8080-tcp (svc/py)
dc/py deploys istag/py:latest <-
bc/py source builds https://github.com/openshift/django-ex.git#master on
openshift/python:3.5
deployment #1 deployed 1 hour ago - 1 pod

3 warnings identified, use 'oc status -v' to see details.
```

- c) Access the launched application:

```
$ curl -v http://py-test2.apps.r11b.oss.labs
```

The OpenShift Container Platform is now deployed in the Dell EMC Ready Bundle for Red Hat OpenStack Platform.

---

# Chapter 10

---

## Next Steps

---

Now that the OpenShift Container Platform is now deployed in the Dell EMC Ready Bundle for Red Hat OpenStack Platform, follow the procedures in the guide listed below to deploy Red Hat CloudForms on the Dell EMC Ready Bundle for Red Hat OpenStack Platform:

- [Technical Guide - Deploying CloudForms 4.2 in the Dell EMC Ready Bundle for Red Hat OpenStack Platform - Version 10.0.1](#)

---

# Appendix

# A

---

## Getting Help

---

**Topics:**

- [Contacting Dell EMC](#)
- [References](#)

This appendix details contact and reference information for the Dell EMC Ready Bundle for Red Hat OpenStack Platform.

## Contacting Dell EMC

For customers in the United States, call 800-WWW-DELL (800-999-3355).

 **Note:** If you do not have an active Internet connection, you can find contact information on your purchase invoice, packing slip, bill, or Dell EMC product catalog.

Dell EMC provides several online and telephone-based support and service options. Availability varies by country and product, and some services may not be available in your area. To contact Dell EMC for sales, technical support, or customer service issues:

1. Visit [dell.com/support](http://dell.com/support).
2. Click your country/region at the bottom of the page. For a full listing of country/region, click **All**.
3. Click **All Support** from the **Support** menu.
4. Select the appropriate service or support link based on your need.
5. Choose the method of contacting Dell EMC that is convenient for you.

## References

Additional information can be obtained at <http://www.dell.com/en-us/work/learn/openstack-cloud> or by e-mailing [openstack@dell.com](mailto:openstack@dell.com).

If you need additional services or implementation help, please contact your Dell EMC sales representative.

## To Learn More

For more information on the Dell EMC Ready Bundle for Red Hat OpenStack Platform visit <http://www.dell.com/learn/us/en/04/solutions/red-hat-openstack>.

Copyright © 2014-2017 Dell Inc. or its subsidiaries. All rights reserved. Trademarks and trade names may be used in this document to refer to either the entities claiming the marks and names or their products. Specifications are correct at date of publication but are subject to availability or change without notice at any time. Dell EMC and its affiliates cannot be responsible for errors or omissions in typography or photography. Dell EMC's Terms and Conditions of Sales and Service apply and are available on request. Dell EMC service offerings do not affect consumer's statutory rights.

Dell EMC, the DELL EMC logo, the DELL EMC badge, and PowerEdge are trademarks of Dell Inc.