RED HAT CONSULTING

redhat

OpenShift Container Platform Proof of Concept Environment setup

PREPARED FOR: MitziCom

Version : 1.0

Table of Contents

RED HAT CONSULTING

Review History

Version	Date	Contributor	Role	Description
1.0	24/08/2018	Phoon Woh Shon	Author	**initial copy**

1 Preface

1.1 Confidentiality, Copyright, and Disclaimer

1.2 About This Document

This document contains information about the work done to deploy an OpenShift Container Platform cluster for a Proof of Concept in the Client's environment.

1.3 Terminology

Term	Definition
docker	Open source application engine that is the core unit of packaging in Red Hat OpenShift
etcd	Reliable storage backend for Red Hat OpenShift to maintain cluster state. This etcd deployment is configured as a 2n+1 cluster for a recommended cluster size providing fault tolerance.
kubernetes	Kubernetes manages containerized applications across a set of containers or hosts and provides mechanisms for deployment, maintenance, and application-scaling. Docker packages, instantiates, and runs containerized applications. A Kubernetes cluster consists of one or more masters and a set of nodes.
master	The master validates and configures the data for pods, services, and replication controllers. It also assigns pods to nodes and synchronizes pod information with service configuration.
nodes	Node provides the runtime environments for containers. Each node in a Red Hat OpenShift cluster has the required services to be managed by the master.

Table 1-1: Terminology

2 Background

MitziCom is a telecommunications company that provides hosting and cloud services to a variety of clients, from medium size companies to enterprise scale companies.
This OpenShift Container Platform deployment is part of a Proof of Concept to determine the feasibility of using Red Hat OpenShift Container Platform as a target for internal and client workloads.

The POC focus on capabilities of Red Hat OpenShift Container Platform around

- automation
- support of multi-tenant workload
- CICD

3 OpenShift Container Platform Environment Overview

The OpenShift environment deployed in the Client's public cloud environment is a 3-masters native HA OpenShift Container Platform set-up, as seen in **Appendix A**.

2 infra nodes were deployed to host registry, docker and the logging / metrics containers; where their node selectors were specified as the infra nodes : ***env=infra***

3 application nodes were deployed, each labelled for different clients

- node1: client=alpha
- node2: client=beta
- node3: client=common

1 support node provides NFS storage.

Access to the cluster is via a Bastion host

The information of the OCP hosts are as follows:

3.1 OpenShift Hosts

Hostname	Description	Labels
loadbalancer1	load balancer node - running haproxy	
master1	master node 1	openshift_node_labels="{'env':'master', 'cluster': '$GUID'}
master2	master node 2	openshift_node_labels="{'env':'master', 'cluster': '$GUID'}

master3	master node 3	openshift_node_labels="{'env':'master', 'cluster': '$GUID'}
infranode1	infranode 1	openshift_node_labels="{'env':'infra', 'cluster': '$GUID'}
infranode2	infranode 2	openshift_node_labels="{'env':'infra', 'cluster': ‘$GUID’}
node1	node 1 - for hosting apps from Client alpha	openshift_node_labels="{'client':'alpha', 'cluster': ‘$GUID’}"
node2	node 2 - for hosting apps from Client beta	openshift_node_labels="{'client':'beta', 'cluster': ‘$GUID’}"
node3	node 3 - for hosting apps for common clients	openshift_node_labels="{'client':'common', 'cluster': '$GUID'}
support1	utilities node - hosting nfs service	
bastion	Bastion host	

3.2 Network Isolation

Network isolation is provided using **openshift-ovs-networkpolicy** plugin

3.3 Network Services

These were the network services provided within the customer network.

Service	Hosts
DNS	192.199.0.2

3.4 Authentication

Authentication is provided using the **HTPasswdPasswordIdentityProvider**.

3.5 High Availability

High Availability is provided by the **native** method.

3.6 Access Hostnames

3.6.1 OpenShift Console

Hostname	Purpose

https://loadbalancer1.$GUID.example.opentlc.com	web console
https://loadbalancer1-$GUID-internal:443	Internal endpoint, CLI

3.6.2 Wildcard DNS Domain

*.apps.$GUID.example.opentlc.com

3.7 CA Cert Configuration

For this deployment, the default certificates generated by the Installer were used. Custom certificates can be configured as documented here.

3.8 Load Balancers

3.8.1 Load Balanced developer/admin frontend

The developer/admin endpoint is fronted by a haproxy load-balancer deployed using native HA master capabilities built into OpenShift

Endpoint	Hostname	Notes
https://loadbalancer1.$GUID.example.opentlc.com	loadbalancer1.$GUID.example.opentlc.com	Developer/admin endpoint
https://loadbalancer1-$GUID-internal:443	loadbalancer1.$GUID.internal	Internal API

4 Deployment

4.1 Pre-Deployment

All the hosts were prepared according to [Prerequisites]() and [Host Preparation]() section of the **OpenShift Container Platform Installation and Configuration** documentation.

4.2 Running the deployment

Deployment of the OpenShift Container Platform cluster is done using the ansible playbook method.

The playbooks to install the cluster and execute post installation activities are hosted here:

https://github.com/wohshon/ocp_advanced_deployment_homework

4.2.1 Quick Start

With the assumption that the infrastructure layer has been set up and configured according to the information in Section 3 and 4.1, the installation can be invoked by following the steps below:

1. Login to Bastion Host
2. Change to root user

```
# ssh -i <path/to/key> <login_id>@bastion.$GUID.example.opentlc.com
# sudo su -
```

3. Create a workspace to clone the playbooks

   e.g. the home directory of the user

```
# cd ~
```

4. Clone the playbooks, change into the cloned directory

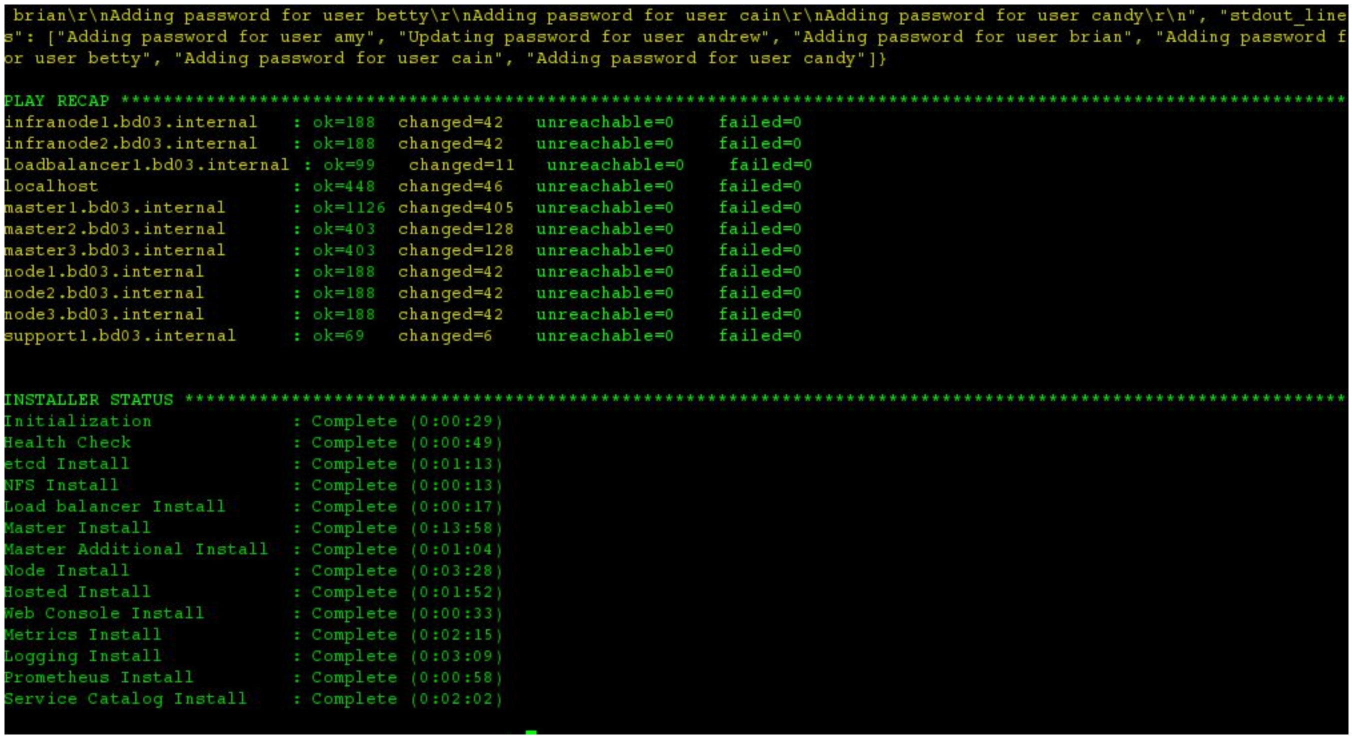
```
# git clone https://github.com/wohshon/ocp_advanced_deployment_homework
# cd ocp_advanced_deployment_homework/
```

5. The OpenShift Container Platform installation can be started by the following commands :

```
#./run.sh
```

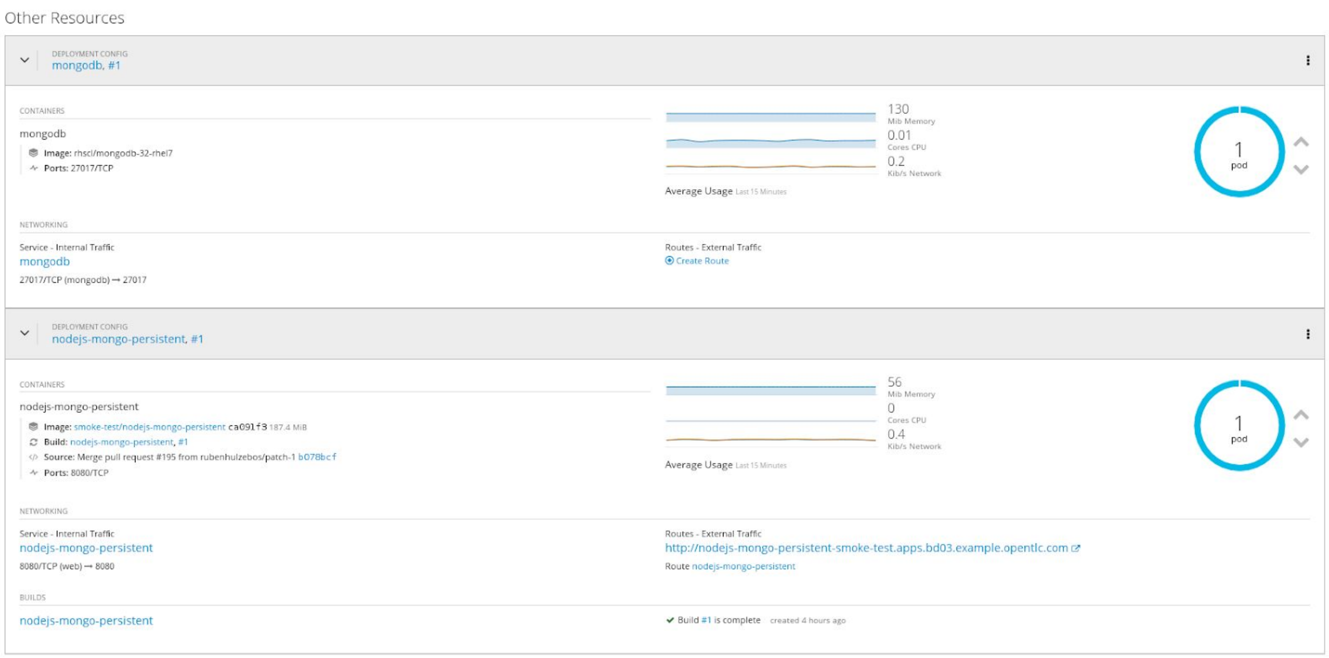
4.3 Deployment Verification

Installation playbook executed without error



```
 brian\r\nAdding password for user betty\r\nAdding password for user cain\r\nAdding password for user candy\r\n", "stdout_line
s": ["Adding password for user amy", "Updating password for user andrew", "Adding password for user brian", "Adding password f
or user betty", "Adding password for user cain", "Adding password for user candy"]}

PLAY RECAP ****************************************************************************************************************
infranode1.bd03.internal   : ok=188  changed=42   unreachable=0    failed=0
infranode2.bd03.internal   : ok=188  changed=42   unreachable=0    failed=0
loadbalancer1.bd03.internal : ok=99   changed=11   unreachable=0    failed=0
localhost                  : ok=448  changed=46   unreachable=0    failed=0
master1.bd03.internal      : ok=1126 changed=405  unreachable=0    failed=0
master2.bd03.internal      : ok=403  changed=128  unreachable=0    failed=0
master3.bd03.internal      : ok=403  changed=128  unreachable=0    failed=0
node1.bd03.internal        : ok=188  changed=42   unreachable=0    failed=0
node2.bd03.internal        : ok=188  changed=42   unreachable=0    failed=0
node3.bd03.internal        : ok=188  changed=42   unreachable=0    failed=0
support1.bd03.internal     : ok=69   changed=6    unreachable=0    failed=0


INSTALLER STATUS **********************************************************************************************************
Initialization             : Complete (0:00:29)
Health Check               : Complete (0:00:49)
etcd Install               : Complete (0:01:13)
NFS Install                : Complete (0:00:13)
Load balancer Install      : Complete (0:00:17)
Master Install             : Complete (0:13:58)
Master Additional Install  : Complete (0:01:04)
Node Install               : Complete (0:03:28)
Hosted Install             : Complete (0:01:52)
Web Console Install        : Complete (0:00:33)
Metrics Install            : Complete (0:02:15)
Logging Install            : Complete (0:03:09)
Prometheus Install         : Complete (0:00:58)
Service Catalog Install    : Complete (0:02:02)
```

Smoke Test application deployed successfully



Other Resources

DEPLOYMENT CONFIG
mongodb, #1

CONTAINERS
mongodb
Image: rhscl/mongodb-32-rhel7
Ports: 27017/TCP

130 Mib Memory
0.01 Cores CPU
0.2 Kib/s Network
Average Usage Last 15 Minutes

1 pod

NETWORKING
Service - Internal Traffic
mongodb
27017/TCP (mongodb) → 27017

Routes - External Traffic
Create Route

DEPLOYMENT CONFIG
nodejs-mongo-persistent, #1

CONTAINERS
nodejs-mongo-persistent
Image: smoke-test/nodejs-mongo-persistent ca091f3 187.4 MiB
Build: nodejs-mongo-persistent, #1
Source: Merge pull request #195 from rubenhulzebos/patch-1 b078bcf
Ports: 8080/TCP

56 Mib Memory
0 Cores CPU
0.4 Kib/s Network
Average Usage Last 15 Minutes

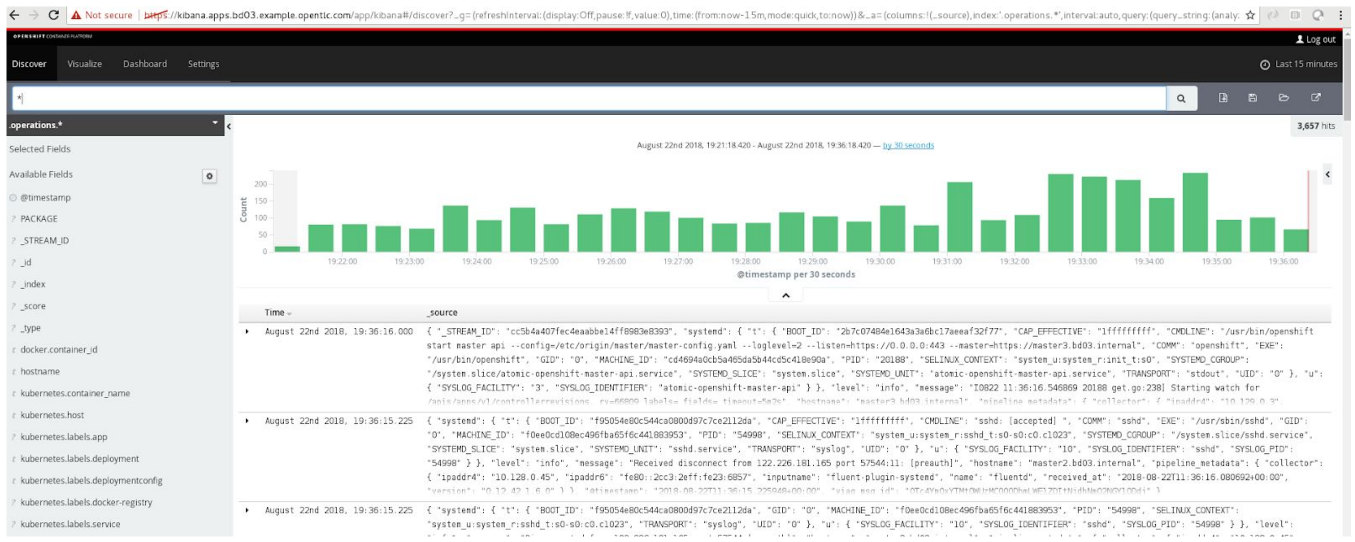
1 pod

NETWORKING
Service - Internal Traffic
nodejs-mongo-persistent
8080/TCP (web) → 8080

Routes - External Traffic
http://nodejs-mongo-persistent-smoke-test.apps.bd03.example.opentlc.com
Route nodejs-mongo-persistent

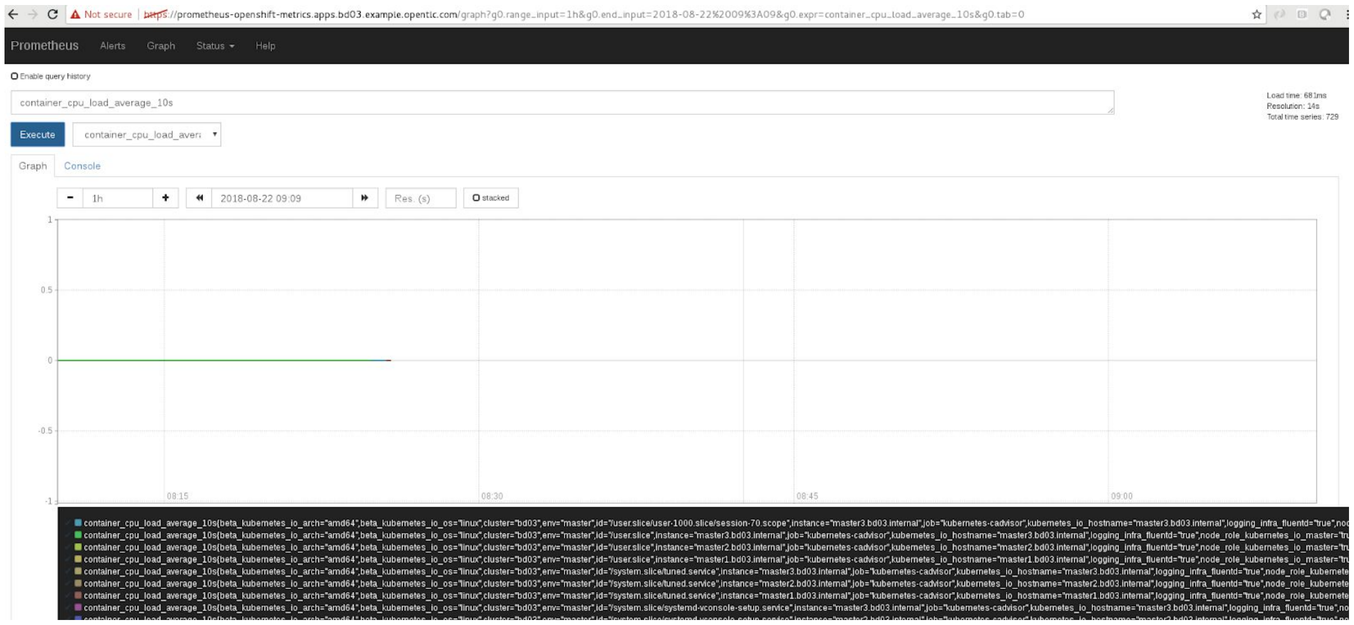
BUILDS
nodejs-mongo-persistent
✓ Build #1 is complete created 4 hours ago

Logging and Metrics component deployed



Not secure | ~~https~~://kibana.apps.bd03.example.opentlc.com/app/kibana#/discover?_g=(refreshInterval:(display:Off,pause:!f,value:0),time:(from:now-15m,mode:quick,to:now))&_a=(columns:!(_source),index:'.operations.*',interval:auto,query:(query_string:(analy:

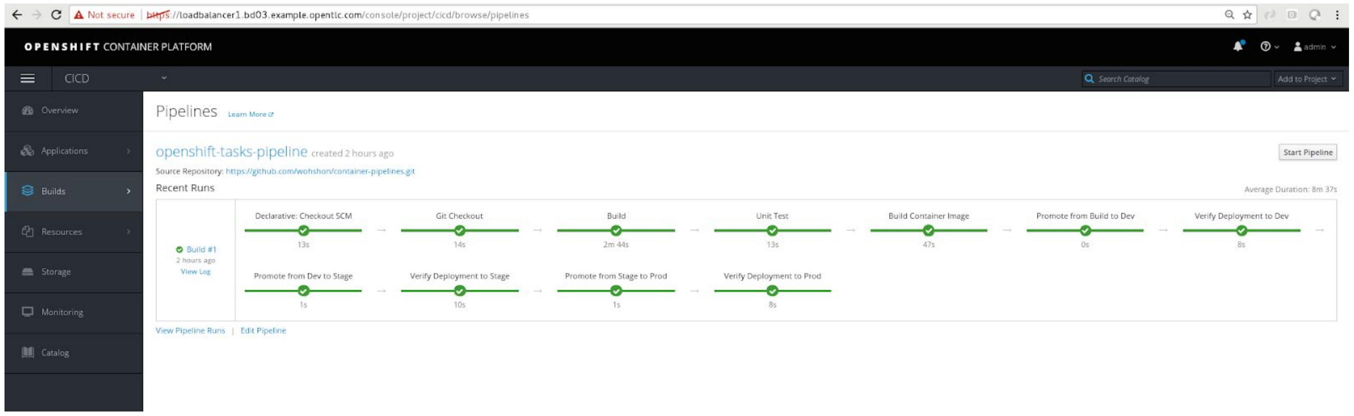
```
Time                          _source
August 22nd 2018, 19:36:16.000  { "_STREAM_ID": "cc5b4a407fec4eaabbe14ff8983e8393", "systemd": { "t": { "BOOT_ID": "2b7c07484e1643a3a6bc17aeeaf32f77", "CAP_EFFECTIVE": "1fffffffff", "CMDLINE": "/usr/bin/openshift start master api --config=/etc/origin/master/master-config.yaml --loglevel=2 --listen=https://0.0.0.0:443 --master=https://master3.bd03.internal", "COMM": "openshift", "EXE": "/usr/bin/openshift", "GID": "0", "MACHINE_ID": "cd4694a0cb5a465da5b44cd5c418e90a", "PID": "20188", "SELINUX_CONTEXT": "system_u:system_r:init_t:s0", "SYSTEMD_CGROUP": "/system.slice/atomic-openshift-master-api.service", "SYSTEMD_SLICE": "system.slice", "SYSTEMD_UNIT": "atomic-openshift-master-api.service", "TRANSPORT": "stdout", "UID": "0" }, "u": { "SYSLOG_FACILITY": "3", "SYSLOG_IDENTIFIER": "atomic-openshift-master-api" } }, "level": "info", "message": "I0822 11:36:16.546869 20188 get.go:238] Starting watch for /apis/apps/v1/controllerrevisions, rv=66809 labels= fields= timeout=5m2s", "hostname": "master3.bd03.internal", "pipeline_metadata": { "collector": { "ipaddr4": "10.129.0.3",
August 22nd 2018, 19:36:15.225  { "systemd": { "t": { "BOOT_ID": "f95054e80c544ca0800d97c7ce2112da", "CAP_EFFECTIVE": "1fffffffff", "CMDLINE": "sshd: [accepted] ", "COMM": "sshd", "EXE": "/usr/sbin/sshd", "GID": "0", "MACHINE_ID": "f0ee0cd108ec496fba65f6c441883953", "PID": "54998", "SELINUX_CONTEXT": "system_u:system_r:sshd_t:s0-s0:c0.c1023", "SYSTEMD_CGROUP": "/system.slice/sshd.service", "SYSTEMD_SLICE": "system.slice", "SYSTEMD_UNIT": "sshd.service", "TRANSPORT": "syslog", "UID": "0" }, "u": { "SYSLOG_FACILITY": "10", "SYSLOG_IDENTIFIER": "sshd", "SYSLOG_PID": "54998" } }, "level": "info", "message": "Received disconnect from 122.226.181.165 port 57544:11: [preauth]", "hostname": "master2.bd03.internal", "pipeline_metadata": { "collector": { "ipaddr4": "10.128.0.45", "ipaddr6": "fe80::2cc3:2eff:fe23:6857", "inputname": "fluent-plugin-systemd", "name": "fluentd", "received_at": "2018-08-22T11:36:16.080692+00:00", "version": "0.12.42 1.6.0" } }, "@timestamp": "2018-08-22T11:36:15.225948+00:00", "viaq_msg_id": "OTc4YmQxYTM+OWUzMC00ODhmLWE1ZDItNidhNmQ2NGY1ODdi" }
August 22nd 2018, 19:36:15.225  { "systemd": { "t": { "BOOT_ID": "f95054e80c544ca0800d97c7ce2112da", "GID": "0", "MACHINE_ID": "f0ee0cd108ec496fba65f6c441883953", "PID": "54998", "SELINUX_CONTEXT": "system_u:system_r:sshd_t:s0-s0:c0.c1023", "TRANSPORT": "syslog", "UID": "0" }, "u": { "SYSLOG_FACILITY": "10", "SYSLOG_IDENTIFIER": "sshd", "SYSLOG_PID": "54998" } }, "level":
```



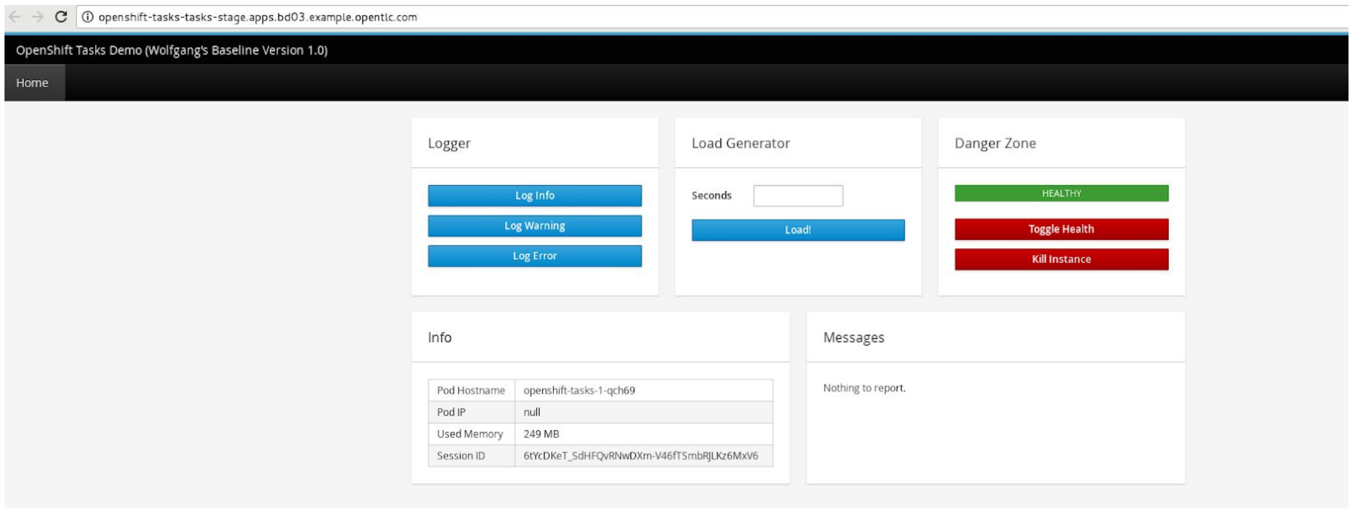
Not secure | https://prometheus-openshift-metrics.apps.bd03.example.opentlc.com/graph?g0.range_input=1h&g0.end_input=2018-08-22%2009%3A09&g0.expr=container_cpu_load_average_10s&g0.tab=0
Prometheus
Alerts
Graph
Status
Help
Enable query history
container_cpu_load_average_10s
Load time: 681ms
Resolution: 14s
Total time series: 729
Execute
container_cpu_load_aver
Graph
Console
1h
2018-08-22 09:09
Res. (s)
stacked
1
0.5
0
-0.5
-1
08:15
08:30
08:45
09:00

```
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/user.slice/user-1000.slice/session-70.scope",instance="master3.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master3.bd03.internal",logging_infra_fluentd="true",noc
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/user.slice",instance="master3.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master3.bd03.internal",logging_infra_fluentd="true",node_role_kubernetes_io_master="tru
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/user.slice",instance="master2.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master2.bd03.internal",logging_infra_fluentd="true",node_role_kubernetes_io_master="tru
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/user.slice",instance="master1.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master1.bd03.internal",logging_infra_fluentd="true",node_role_kubernetes_io_master="tru
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/system.slice/tuned.service",instance="master3.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master3.bd03.internal",logging_infra_fluentd="true",node_role_kubernete
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/system.slice/tuned.service",instance="master2.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master2.bd03.internal",logging_infra_fluentd="true",node_role_kubernete
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/system.slice/tuned.service",instance="master1.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master1.bd03.internal",logging_infra_fluentd="true",node_role_kubernete
container_cpu_load_average_10s{beta_kubernetes_io_arch="amd64",beta_kubernetes_io_os="linux",cluster="bd03",env="master",id="/system.slice/systemd-vconsole-setup.service",instance="master3.bd03.internal",job="kubernetes-cadvisor",kubernetes_io_hostname="master3.bd03.internal",logging_infra_fluentd="true",no
```

Pipeline for OpenShift Tasks created



Not secure | https://loadbalancer1.bd03.example.opentlc.com/console/project/cicd/browse/pipelines
OPENSHIFT CONTAINER PLATFORM
admin
CICD
Search Catalog
Add to Project
Overview
Applications
Builds
Resources
Storage
Monitoring
Catalog
Pipelines
Learn More
openshift-tasks-pipeline created 2 hours ago
Start Pipeline
Source Repository: https://github.com/wohshon/container-pipelines.git
Recent Runs
Average Duration: 8m 37s
Build #1
2 hours ago
View Log
Declarative: Checkout SCM
13s
Git Checkout
14s
Build
2m 44s
Unit Test
13s
Build Container Image
47s
Promote from Build to Dev
0s
Verify Deployment to Dev
8s
Promote from Dev to Stage
1s
Verify Deployment to Stage
10s
Promote from Stage to Prod
1s
Verify Deployment to Prod
8s
View Pipeline Runs | Edit Pipeline



Pod Hostname	openshift-tasks-1-qch69
Pod IP	null
Used Memory	249 MB
Session ID	6tYcDKeT_SdHFQvRNwDXm-V46fTSmbRJLKz6MxV6

Nothing to report.

5 Walkthrough of Ansible Playbook

5.1 Overview

The playbook for installing the cluster is hosted at github.com:

https://github.com/wohshon/ocp_advanced_deployment_homework

It consists of various roles that modularize the cluster setup and post installation activities.

The structure of the playbooks project is listed below:

```
ocp_advanced_deployment_homework/
├── applier
│   ├── cicd
│   ├── deploy-client-projects
│   ├── deploy-tasks-app
│   ├── inventory
│   ├── multitenant
│   └── smoke-test
├── group_vars
│   └── all.yml
├── roles
│   ├── cicd
│   ├── deploy-client-projects
│   ├── deploy-tasks-app
│   ├── inventory
│   ├── make-applier-projects-unique
│   ├── multitenant
│   ├── openshift-applier
│   ├── post-install
│   ├── prep-env
│   └── smoke-test
├── homework.yaml
├── README.adoc
├── run.sh
└── uninstall.sh
```

The roles will be explained in detailed in subsequent sections. An overview is provided in the table below.

All openshift objects are created using the ***OpenShift Applier*** role

Stage	Role	Description
Pre installation	inventory	Generate inventory hosts file
Pre installation	prep-env	Set GUID variable in all nodes Checks for docker installation Checks for nfs installation in support node
Installation	N/A	import_playbook from installation script: - prerequisites.yml, - deploy_cluster.yml Deploy a HA cluster of Openshift Container Platform
Post Installation	post-install	Setup user access to cluster Setup Persistent Volumes
Smoke Test	smoke-test	Deploys a nodejs mongodb app with persistent storage
Setup CI / CD	cicd	Deploys a jenkins instance with persistent storage
Setup CI / CD	deploy-tasks-app	Deploys a jenkins pipeline in the jenkins instance setup

		previously Pipeline automates the deployment of OpenShift Tasks over 3 environment
Setup Multi-Tenant	multitenant	Inject *admissionControl* plugin to master config to manage project node selector Create and inject new default project template into master config with limitrange. Create new project template for multi-tenant clients Multi-tenant project template are configured to role-binded to groups to allow easy onboarding of new users (just add new users to group to gain access to projects) The 2 project templates serves different purposes: - the default request template (with limitrange) is to support the conventional ***oc new-project*** new project request - the multi-tenant project template allows cluster-admin to create new project namespace catering to multi tenant usecases; with nodeselector and a group based rolebinding defined during project creation
Deploy Multi-tenant projects and applications	deploy-client-projects	Setup users, groups and projects for alpha, beta and charlie Deploy a sample application

5.2 Single Command to install / uninstall cluster

1. run.sh

This command set the GUID for the infrastructure environment as an environment variable and triggers the playbook to start installation

The GUID variable is critical as it is passed as a extra variable to the ansible playbook. This allows the hosts files to form the correct hostnames for ansible to communicate to them

```
#!/bin/bash
echo 'set log_plays to true'
export ANSIBLE_LOG_PATH=/var/log/ansible.log
echo '******************SET GUID*******************'
export GUID=`hostname | cut -d"." -f2`; echo "export GUID=$GUID" >> $HOME/.bashrc
```

```
echo 'GUID ==> '$GUID
echo '******************RUN PLAYBOOK****************'
ansible-playbook -v -f 20 homework.yaml --extra-vars GUID=$GUID
```

2. **uninstall.sh**

This command set the GUID for the infrastructure environment as an environment variable and triggers the playbook to start un-installation

```
#!/bin/bash
echo 'set log_plays to true'
export ANSIBLE_LOG_PATH=/var/log/ansible_uninstall.log
echo '******************SET GUID********************'
export GUID=`hostname | cut -d"." -f2`; echo "export GUID=$GUID" >> $HOME/.bashrc
echo 'GUID ==> '$GUID
echo '******************RUN PLAYBOOK****************'
ansible-playbook /usr/share/ansible/openshift-ansible/playbooks/adhoc/uninstall.yml
```

5.3 Main Playbook

1. homework.yaml

This is the main playbook that invokes the roles to setup and configure the cluster.

Details of the various roles will be covered in the next section.

```
---
- name: Set GUID in host file
  hosts: localhost
  roles:
    - inventory

- name: Prepare Environment Play
  hosts: all
  roles:
    - prep-env

- name: install pre-req
  import_playbook:
"/usr/share/ansible/openshift-ansible/playbooks/prerequisites.yml"
```

```
- name: install ocp
  import_playbook:
"/usr/share/ansible/openshift-ansible/playbooks/deploy_cluster.yml"

- name: Post Install
  hosts: all,localhost
  roles:
    - post-install

- name: Smoke Test
  hosts: localhost
  roles:
    - smoke-test

- name: Deploy Jenkins
  hosts: localhost
  roles:
    - cicd

- name: Deploy OpenShift Tasks Project
  hosts: localhost
  roles:
    - deploy-tasks-app

- name: Prepare cluster for Multitenant project setup
  hosts: masters,localhost
  roles:
    - multitenant

- name: On Board users groups projects and applications
  hosts: masters,localhost
  roles:
    - deploy-client-projects
```

5.4 Applier directory

This contains the object definitions and parameters for the OpenShift Container Platform Objects that needs to be created,

The applier objects will be discussed in conjunction with the related roles in the next section.

```
applier/
```

```
├──── cicd
│   ├──── params
│   ├──── projects
│   └──── templates
├──── deploy-client-projects
│   ├──── params
│   └──── templates
├──── deploy-tasks-app
│   ├──── hpa
│   ├──── limitrange
│   ├──── params
│   ├──── projects
│   └──── templates
├──── multitenant
│   ├──── project-templates
│   └──── user-group-templates
└──── smoke-test
    ├──── params
    ├──── projects
    └──── templates
```

6 Client Onboarding Process

6.1 Overview

One of the requirements in the PoC is to support multi-tenanted usecase.
Each tenant/ client is to have a dedicated node to run their container workload.

The following changes were made to the cluster to support this usecase

1. Admission Controller

   Pod Node Selector is used to manage the pod placement tied to client projects.
   The following changes in the master config were made to include the pod node selector configuration

master-config.yaml

```
admissionConfig:
  pluginConfig:
    PodNodeSelector:
      location: /etc/origin/podnodeSelectorConfig.yaml
    BuildDefaults:
….
```

podNodeSelectorConfig.yaml

```
podNodeSelectorPluginConfig:
  clusterDefaultNodeSelector: "client=common"
  clientalpha: "client=alpha"
  clientbeta: "client=beta"
```

2. New Project Template

A new project template is created to support the multitenant usecase (refer to **APPENDIX C**)
This template, extends from the default project template, to include the additional parameters

- GROUP_NAME
  - to create a rolebinding to this group with **edit** rights
- CLIENT_NODE_SELECTOR
  - specify the node selector for the client, this will be use to annotate the project to control the pod placements

This template does not support the self-provisioner role's project creation. It has to be executed by an cluster-admin
A default project request (enhanced with limitrange and resourcequota values) is created to support the default project request. The template is listed in **APPENDIX C**

master-config.yaml:

```
...
projectConfig:
  defaultNodeSelector: client=common
  projectRequestMessage: ''
  projectRequestTemplate: 'default/default-project-request'
  securityAllocator:
    mcsAllocatorRange: s0:/2
    mcsLabelsPerProject: 5
    uidAllocatorRange: 1000000000-1999999999/10000
...
```

3. New Group and User Template

A User and group template is created, the User template labels the user with the "**client=<client name>**" label.

This will be useful when there is a need to use Admission Controller to control the maximum number of projects a client can create.

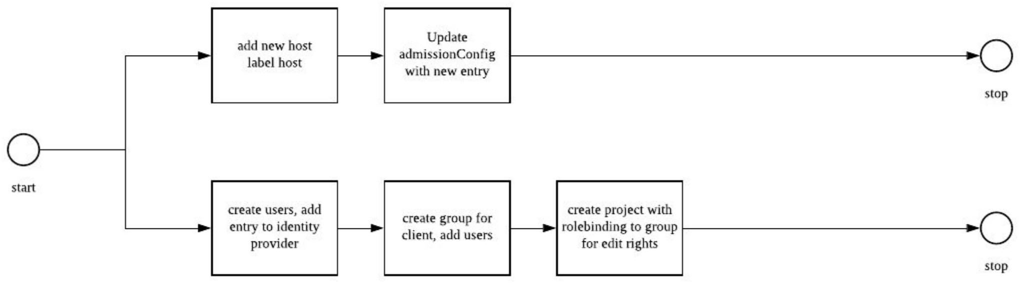
A group template is created to facilitate the creating of rolebindings to the projects

6.2 New client onboarding

High level flow for onboarding new clients

1. New worker node is allocated to every client to host their containers.
    a. The new node is added to the cluster by running the ansible playbook provided by OpenShift Container Platform.
    b. The node is labelled with the labels "***client=<client name>***"
2. Update Admission Config with the relevant label for the client
3. Create users for the client, in both openshift and the identity provider
4. Create group for users with **edit** rights in the project
5. Create project using the multi tenant project template, passing in the group name, project name, admin user.

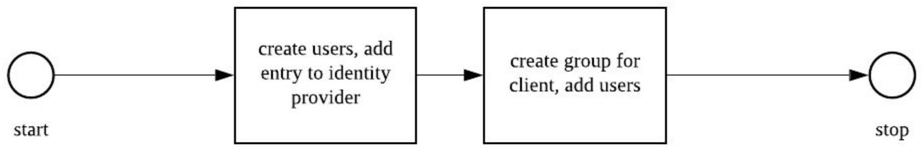
A high level flow is depicted in the diagram below



6.3 New User onboarding for existing clients

High level flow for onboarding new users

1. Create users for the client, in both openshift and the identity provider
2. Add user to group, the user will automatically have access to the project


start
create users, add entry to identity provider
create group for client, add users
stop

7 Walkthrough of Ansible Playbook Roles

7.1 Overview

The roles will be described in this section in the sequence it is being invoked in the main playbook.

7.2 Role : Inventory

The inventory role is the first role to be executed as it needs to set up the inventory hosts with the correct hostname which requires a GUID variable that is passed in by the main playbook.

Task Name	Description	Remarks
Creates the hosts file	Uses the *localhost* host to generate a inventory hosts with the correct GUID injected into the hosts file. The generated hosts file replaces the default ansible hosts file at */etc/ansible/hosts.*	Uses jinja2 template
refresh	The meta task refresh_inventory is used to reload the newly generated hosts inventory information. This sets up the correct hosts information for subsequent roles.	

The generated hosts file is listed in **APPENDIX B**

```
---
# This role setsup the host file
```

```
- name: Creates the hosts file
  template: src=hosts.j2 dest=/etc/ansible/hosts

- name: refresh
  meta: refresh_inventory
```

7.3 Role : prep-env

The prep-env role is to verify and prepare the environment and hosts prior to invoking the OpenShift Container Platform installation scripts.

Task Name	Description	Remarks
All GUID env variable	injects the GUID as environment variable into all the hosts within the cluster	
Verify Docker installation	Verify docker installation in all the masters and application nodes	
Configuration of Docker		
Verify NFS installed on nfs node	Ensure nfs packages are installed in the support node	
check exportfs	Runs exportfs to check nfs function	

```
---
- name: All GUID env variable
  shell: export GUID=`hostname | cut -d"." -f2`; echo "export GUID=$GUID" >>
$HOME/.bashrc
  tags: set-guid

- name: Verify Docker installation
  yum:
    name: docker
    state: present
  when: "'nodes' in group_names"
  tags: verify-docker
```

```
- name: Configuration of Docker
  shell: systemctl restart docker
  when: "'nodes' in group_names"
  tags: verify-docker

- name: Verify NFS installed on nfs node
  yum:
    name: "{{ packages }}"
    state: present
  vars:
    packages:
    - nfs-utils
    - rpcbind
  when: "'nfs' in group_names"
  tags: verify-nfs

- name: check exportfs
  shell : exportfs
  when: "'nfs' in group_names"
  tags: verify-nfs
```

7.4 import_playbook: "/usr/share/ansible/openshift-ansible/playbooks/prerequisites.yml"

This is not a role.
The prerequisites playbook from the OpenShift Container Platform installation script is triggered via the *import_playbook* module. This playbook runs some verification tests on the environment.

7.5 import_playbook: "/usr/share/ansible/openshift-ansible/playbooks/deploy_cluster.yml"

This is not a role.
The deploy_cluster playbook from the OpenShift Container Platform installation script is triggered via the *import_playbook* module. This playbook installs the cluster.

7.6 Role : post-install

The post-install role invokes tasks to configure the cluster after installation is successful.

Task Name	Description	Remarks
Copy Config File	copies over the kube context file for the default system:admin user from the masters to the bastion hosts	

Check system:admin role	runs a **oc whoami** command for verification	
Check nodes	Verifies nodes are ready with a **oc get nodes** command	
create pv directories	Create directories for export on support node	
configure exports	configure exportfs on support node	
create dir for pv files	Create directory to house persistent volume yaml files	
Create pv yaml - RWO	Create persistent volume yaml files for RWO	
Create pv yaml - RWX	Create persistent volume yaml files for RWX	
Load seed hosts inventory import Openshift Applier roles	Calls Openshift Applier to create Persistent Volumes seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the persistent volumes templates will be generated. The seed-hosts.yml is listed below.	
Check PV	Check Persistent volumes are created	
Create Admin User	Creates a cluster-admin user, ‘admin’. This allows login to the web console as a cluster admin user	
Fix NFS persistent	Pull downs recycler images for persistent volumes.	

```
---
- name: Copy config file
  fetch:
    src: /root/.kube/config
    dest: /root/.kube/config
    flat: yes
    tags: check-role
  when: "'masters' in group_names"

- name: Check system:admin role
  shell: oc whoami
  when: "inventory_hostname == 'localhost'"
  tags: check-role

- name: Check nodes
  shell: oc get nodes
  when: "inventory_hostname == 'localhost'"
  tags: check-nodes

- name: create pv directories
  file:
    dest: "/srv/nfs/user-vols/pv{{ item }}"
    state: directory
    group: nfsnobody
    owner: nfsnobody
    mode: 0777
  with_sequence: start=1 end=50
  when: "'nfs' in group_names"
  tags: create-pv-dir

- name: configure exports
  shell: echo /srv/nfs/user-vols/pv{{ item }} >>
/etc/exports.d/openshift-uservols.exports
  with_sequence: start=1 end=50
  when: "'nfs' in group_names"
  tags: configure-exports

- name: create dir for pv files
  file:
    dest: ./applier/pvs
    state: directory
  when: "inventory_hostname == 'localhost'"
  tags: create-pv-yaml-dir

- name: Create pv yaml - RWO
  vars:
    volsize: '5Gi'
    volume: 'pv{{ item }}'
    mode: 'ReadWriteOnce'
    reclaimPolicy: 'Recycle'
  template: src=pv.j2 dest=./applier/pvs/{{ volume }}.yml
  with_sequence: start=1 end=25
  when: "inventory_hostname == 'localhost'"

- name: Create pv yaml - RWX
```

```
  vars:
    volsize: '10Gi'
    volume: 'pv{{ item }}'
    mode: 'ReadWriteMany'
    reclaimPolicy: 'Retain'
  template: src=pv.j2 dest=./applier/pvs/{{ volume }}.yml
  with_sequence: start=26 end=50
  when: "inventory_hostname == 'localhost'"

- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost
  when: "inventory_hostname == 'localhost'"

- name: load inventory info for PV
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml
  when: "inventory_hostname == 'localhost'"

- name: Apply condition to each task in role
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"

- name: check pv
  shell: oc get pv
  when: "inventory_hostname == 'localhost'"

- name: Create Admin User
  script: create-users.sh
  when: "'masters' in group_names"

- name: Fix NFS Persistence
  shell: "{{ item }}"
  with_items:
    - "docker pull registry.access.redhat.com/openshift3/ose-recycler:latest"
    - "docker tag registry.access.redhat.com/openshift3/ose-recycler:latest
registry.access.redhat.com/openshift3/ose-recycler:v3.9.30"
```

```
  when: "'nodes' in group_names"
  tags: pull-recycler
```

seed-hosts.yml:

```
openshift_cluster_content:
- object: pv
  content:
  - name: "create pvs"
    file: "{{role_path}}/../../applier/pvs/"
    action: create
```

7.7 Role : smoke-test

The smoke-test role deploys a nodejs with mongodb application to verify the cluster is functioning.
It requires a persistent volume.
The role also runs a simple connectivity test to check for the pod availability.

Task Name	Description	Remarks
add localhost to seed hosts load inventory info for smoke test Deploy smoke test app using applier	Calls Openshift Applier to deploy the sample application seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the application templates are. The seed-hosts.yml is listed below.	
Test URL	Register the route of the pod	
wait for pod to be alive	runs a connectivity test using the url module	
Smoke Test Passed	Display a message to indicate success smoke test	

```
---
```

```
#- name: Smoke Test
#  script: deploy-nodejs-mongodb-persistent.sh

- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost
  when: "inventory_hostname == 'localhost'"

- name: load inventory info for smoke test
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml
  when: "inventory_hostname == 'localhost'"

- name: Deploy smoke test app using applier
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"

- name: Test URL
  shell: echo http://$(sudo oc get route -n smoke-test | awk 'NR>1 {print $2}')
  register: url

- name: debug
  debug:
    msg: "{{ url.stdout }}"

- name: "wait for pod to be alive"
  uri:
    url: "{{ url.stdout }}"
    status_code: 200
  register: result
  until: result.status == 200
  retries: 120
  delay: 2

- name: Smoke Test Passed
  debug:
    msg: "***SMOKE TEST PASSED***"
  when: result.status == 200
```

seed-hosts.yml:

```
openshift_cluster_content:
- object: project
  content:
  - name: "create smoke test project"
    file: "{{role_path}}/../../applier/smoke-test/projects/projects.yml"
    action: create
- object: deployments
  content:
  - name: "deploy smoke test app"
    template: "{{role_path}}/../../applier/smoke-test/templates/deployment.yml"
    params: "{{role_path}}/../../applier/smoke-test/params/smoke-test.env"
```

7.8 Role : cicd

The cicd role deploys a nodejs with mongodb application to verify the cluster is functioning.
It requires a persistent volume.
The role also runs a simple connectivity test to check for the pod availability.

Task Name	Description	Remarks
add localhost to seed-hosts load inventory info for Jenkins Deploy Jenkins using applier	Calls Openshift Applier to deploy the sample application seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the application templates are. The seed-hosts.yml is listed below.	Openshift Applier
Test URL	Register the route of the pod	
wait for pod to be alive	runs a connectivity test using the url module	status 403 is use to check for connectivity as it will not pass the authentication check
Jenkins deployed	Display a message to indicate success smoke test	

```
---
- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost
  when: "inventory_hostname == 'localhost'"

- name: load inventory info for Jenkins
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml
  when: "inventory_hostname == 'localhost'"

- name: Deploy Jenkins using applier
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"

- name: Test URL
  shell: echo https://$(sudo oc get route -n cicd | awk 'NR>1 {print $2}')
  register: url

- name: debug
  debug:
    msg: " Jenkins URL: {{ url.stdout }}"

- name: "wait for pod to be alive"
  uri:
    url: "{{ url.stdout }}"
    status_code: 403
    validate_certs: no
  register: result
  until: result.status == 403
  retries: 120
  delay: 2

- name: Jenkins Deployed
  debug:
    msg: "***JENKINS DEPLOYED***"
  when: result.status == 403
```

seed-hosts.yml:

```
openshift_cluster_content:
- object: project
  content:
  - name: "create cicd project"
    file: "{{role_path}}/../../applier/cicd/projects/projects.yml"
    action: create
- object: deployments
  content:
  - name: "deploy jenkins"
    template: "{{role_path}}/../../applier/cicd/templates/deployment.yml"
    params: "{{role_path}}/../../applier/cicd/params/jenkins.env"
```

7.9 Role : deploy-tasks-app

This role creates a jenkins pipeline in the jenkins instance created previously. It uses the pipeline to build and deploy Openshift Tasks across 3 environments, namely, development, staging and production.
It demonstrates the CI / CD capability of the platform

Task Name	Description	Remarks
add localhost to seed-hosts `load inventory info from seed-hosts` Deploy application and objects	Calls Openshift Applier to create the pipeline deploy the sample application, Openshift Tasks The pipeline is a specified in a JenkinsFile hosted at https://github.com/wohshon/container-pipelines/tree/master/basic-spring-boot seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the project / application templates etc are. Limitrange and horizontal pod autoscaler objects are also created for the projects The seed-hosts.yml is listed below.	Openshift Applier

```
---
- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost

- name: load inventory info from seed-hosts
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml

- name: Deploy application and objects
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"
```

seed-hosts.yml:

```
openshift_cluster_content:
- object: projects
  content:
  - name: "create environments"
    file: "{{role_path}}/../../applier/deploy-tasks-app/projects/projects.yml"
    action: create
- object: limitrange
  content:
  - name: "create limitrange"
    file: "{{role_path}}/../../applier/deploy-tasks-app/limitrange/limitrange.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-dev"
    action: create
    namespace: "{{tasks_dev_namespace}}"
  - name: "create limitrange stage"
    file: "{{role_path}}/../../applier/deploy-tasks-app/limitrange/limitrange.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-stage"
    action: create
    namespace: "{{tasks_stage_namespace}}"
  - name: "create limitrange prod"
    file: "{{role_path}}/../../applier/deploy-tasks-app/limitrange/limitrange.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-prod"
    action: create
    namespace: "{{tasks_prod_namespace}}"

- object: hpa
  content:
  - name: "create hpa"
    file: "{{role_path}}/../../applier/deploy-tasks-app/hpa/hpa.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-dev"
    action: create
    namespace: "{{tasks_dev_namespace}}"
  - name: "create hpa stage"
    file: "{{role_path}}/../../applier/deploy-tasks-app/hpa/hpa.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-stage"
    action: create
    namespace: "{{tasks_stage_namespace}}"
  - name: "create hpa prod "
    file: "{{role_path}}/../../applier/deploy-tasks-app/hpa/hpa.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-prod"
    action: create
    namespace: "{{tasks_prod_namespace}}"

- object: deployments
  content:
  - name: "deploy dev environment"
    template:
"{{role_path}}/../../applier/deploy-tasks-app/templates/deployment.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-dev"
  - name: "deply stage environment"
    template:
"{{role_path}}/../../applier/deploy-tasks-app/templates/deployment.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-stage"
  - name: "deply prod environment"
    template:
"{{role_path}}/../../applier/deploy-tasks-app/templates/deployment.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/deployment-prod"
- object: builds
  content:
  - name: "deploy build pipeline to dev"
    template: "{{role_path}}/../../applier/deploy-tasks-app/templates/build.yml"
    params: "{{role_path}}/../../applier/deploy-tasks-app/params/build-dev"
```

7.10 Role : multitenant

This role sets up the multi-tenant requirements for the Proof of Concept. It uses the PodNodeSelector plugin from the admissionConfig to limit the pod placements of projects into specific nodes.
Labels of 'client=<clientname>' e.g. client=alpha are used to label the projects and nodes.
In this Proof of Concept,

- node1 is labelled with *client=alpha*
- node2 , *client=beta*
- node3, *client=common*, for other clients

The plugin config specify in a **podnodeSelectorConfig.yaml** file, which is injected as a reference into the master-config.yaml.

2 new project templates are created in this role

- default-project-request:
    - this is extended from the original project request template, with the following changes
        - A limitrange is added to the template to conform to the POC requirements.
        - Networkpolicy to only accept traffic from same namespace and 'default' namespace is added
        - This template will be use when project is created via the usual **oc new-project** or via the web console
        - Note: projects created during installation will not have the network policy, they can be modified to include the policy but this is out of scope of the POC
    - this new template reference is injected into the ***projectRequestTemplate*** config in the master-config.yaml
- multitenant-project-request:
    - this is a customized project template that extends from the new ***default-project-request*** template described above, with additional
        - GROUP_NAME parameter.
        - CLIENT_NODE_SELECTOR parameter
    - It creates a project object that has a ***rolebinding*** to a specific user group (via the GROUP_NAME) with ***edit*** rights to the project
    - The CLIENT_NODE_SELECTOR sets up the annotation in the project to limit pod placement to the correct nodes.
    - THIS TEMPLATE WILL NOT BE ABLE TO SUPPORT ***oc new-project*** command

1 user and 1 group template are also created in this role

- the user template accepts parameters for
    - USER_NAME
    - CLIENT_LABEL_KEY : CLIENT_LABEL_VALUE for labelling it as a client type

- The group template accept
    - GROUP_NAME
    - GROUP_USERS (list), to specify users belonging to this group

Task Name	Description	Remarks
label default project to allow network policy	Label default namespace as 'name=default' this is to allow router to access pods as network policy now only allows traffic from same namespace and from 'default' namespace	
add localhost to seed-hosts load inventory info for multitenant project template Create templates using applier	Calls Openshift Applier to create - new project teamplates - new user and group templates seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the Objects definitions are The seed-hosts.yml is listed below.	Openshift Applier
copy podnodeselector yaml	copy podnodeselector.yaml over to all the master nodes	
backup master config update master config file with admissionConfig	Backup master config file inject master config with reference to podnodeselector.yaml	
update master config file with default project template	Inject new project request template into master config	using lineinfile module
Stop / Start master services	restart master services for changes to take effect	

```
---
- name: label default project to allow network policy
  shell: "oc label namespace default name=default"
  when: "inventory_hostname == 'localhost'"

- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost
  when: "inventory_hostname == 'localhost'"

- name: load inventory info for multitenant project template
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml
  when: "inventory_hostname == 'localhost'"

- name: Create templates using applier
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"

- name: copy podnodeselector yaml
  copy:
    src: "{{ role_path }}/files/podnodeSelectorConfig.yaml"
    dest: "/etc/origin/"
  when: "'masters' in group_names"

- name: backup master-config.yaml
  shell: 'cp /etc/origin/master/master-config.yaml
/etc/origin/master/master-config.yaml_backup_multitenant'
  when: "'masters' in group_names"

- name: update master config file with admissionConfig
  shell: 'sed -i "/pluginConfig:/ a \    PodNodeSelector:\n      location:
/etc/origin/podnodeSelectorConfig.yaml" /etc/origin/master/master-config.yaml'
  when: "'masters' in group_names"

- name: update master config file with default project template
  lineinfile:
    dest: "/etc/origin/master/master-config.yaml"
    line: "  projectRequestTemplate: 'default/default-project-request'"
    regexp: "^(.*)projectRequestTemplate: ''(.*)$"
  when: "'masters' in group_names"

- name: stop master services
  shell: 'systemctl stop {{ item }}'
  with_items:
    - atomic-openshift-master-api
    - atomic-openshift-master-controllers
  when: "'masters' in group_names"

- name: start master services
  shell: 'systemctl start {{ item }}'
  with_items:
    - atomic-openshift-master-controllers
    - atomic-openshift-master-api
  when: "'masters' in group_names"
```

seed-hosts.yml:

```
openshift_cluster_content:
- object: project
  content:
  - name: "create project teamplates"
    file: "{{role_path}}/../../applier/multitenant/project-templates/"
    action: create
- object: users-groups
  content:
  - name: "create project teamplates"
    file: "{{role_path}}/../../applier/multitenant/user-group-templates/"
    action: create
```

podnodeSelectorConfig.yaml :

```
podNodeSelectorPluginConfig:
  clusterDefaultNodeSelector: "client=common"
  clientalpha: "client=alpha"
  clientbeta: "client=beta"
```

7.11 Role : deploy-client-projects

This role simulates the onboarding of 3 clients, namely: **alpha**, **beta** and **charlie**
It assumes that there is a node labelled with ‘client=<client name>’ and relevant admissionConfig are already setup.

The onboarding of clients are described in details in the ***Clients Onboarding*** section.

This role onboards 3 clients

From the previous ‘multitenant’ role, 3 templates were setup

- a project template
- user template
- group template

To onboard new client, the following steps are required

1. Create users

```
# oc process user-request-template -p CLIENT_LABEL_KEY="client" -p
CLIENT_LABEL_VALUE="alpha" -p USER_NAME=amy | oc create -f -
# oc process user-request-template -p CLIENT_LABEL_KEY="client" -p
CLIENT_LABEL_VALUE="alpha" -p USER_NAME=andrew | oc create -f -
```

2. Create group

```
# oc process group-request-template -p GROUP_NAME=alpha-users -p
GROUP_USERS='["amy","andrew"]' | oc create -f -
```

3. Create Projects

```
oc process multitenant-project-request -p PROJECT_DESCRIPTION='' -p
PROJECT_DISPLAYNAME='' -p PROJECT_NAME='alpha' -p PROJECT_ADMIN_USER='amy' -p
GROUP_NAME=alpha-users -p CLIENT_NODE_SELECTOR='client=alpha' | oc create -f -
```

```
- apiVersion: v1
  groupNames:
  - ${GROUP_NAME}
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: normal-users
    namespace: ${PROJECT_NAME}
  roleRef:
```

```
name: edit
```

Task Name	Description	Remarks
add localhost to seed-hosts load inventory info for client projects Create projects and groups and users	Calls Openshift Applier to deploy the all the objects - users - groups - projects - sample application seed-hosts inventory is created in the *vars* directory of this role, It points to the directory where the object definitions are. The seed-hosts.yml is listed below.	Openshift Applier
Create Users in Identity Providers	Add Users created in the previous task to htpasswd file	

```
---
- name: add localhost to seed-hosts
  add_host:
    hostname: localhost
    groups:
      - seed-hosts
    ansible_connection: local
    ansible_host: localhost
  when: "inventory_hostname == 'localhost'"

- name: load inventory info for client projects
  include_vars:
    dir: vars
    files_matching: seed-hosts.yml
  when: "inventory_hostname == 'localhost'"

- name: Create projects and groups and users
```

```
  import_role:
    name: openshift-applier
  when: "inventory_hostname == 'localhost'"


- name: Create Users in identity provider
  script: add-identity-provider.sh
  when: "'masters' in group_names"
```

seed-hosts.yml:

```
# ALPHA CLIENT
- object: user
  content:
  - name: "create user amy"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/amy-env"

- object: user
  content:
  - name: "create user andrew"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/andrew-env"

- object: groups
  content:
  - name: "create groups"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/group-request-template
.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/alpha-env"

- object: project
  content:
  - name: "create project"
    template:
"{{role_path}}/../../applier/multitenant/project-templates/multitenant-project-templ
```

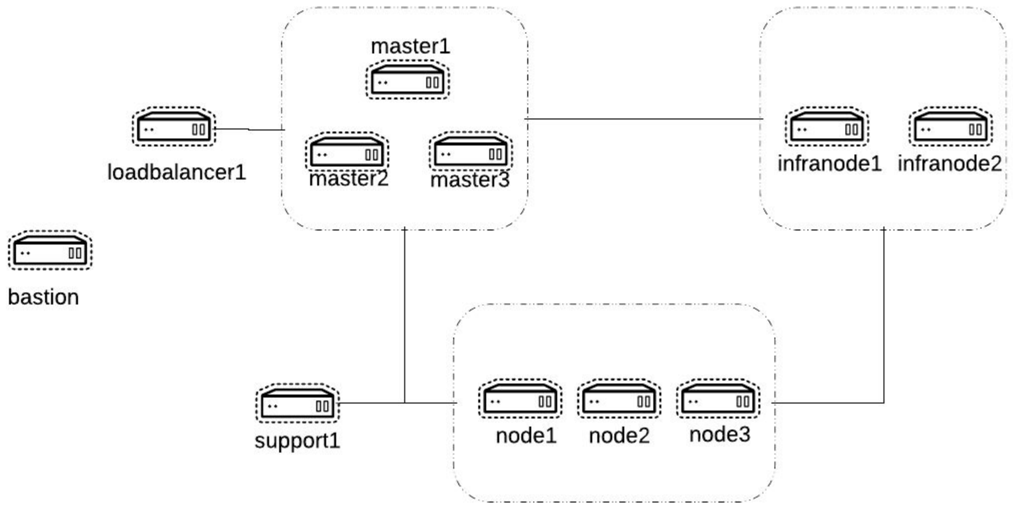
```
ate.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/alpha-env"

- object: app
  content:
  - name: "deploy app"
    template:
"{{role_path}}/../../applier/deploy-client-projects/templates/sample-app.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/alpha-env"

# BETA CLIENT

- object: user
  content:
  - name: "create user amy"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/brian-env"

- object: user
  content:
  - name: "create user andrew"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/betty-env"

- object: groups
  content:
  - name: "create groups"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/group-request-template
.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/beta-env"

- object: project
  content:
  - name: "create project"
    template:
"{{role_path}}/../../applier/multitenant/project-templates/multitenant-project-templ
ate.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/beta-env"
```

```
- object: app
  content:
  - name: "deploy app"
    template:
"{{role_path}}/../../applier/deploy-client-projects/templates/sample-app.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/beta-env"

# CHARLIE CLIENT

- object: user
  content:
  - name: "create user amy"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/cain-env"

- object: user
  content:
  - name: "create user andrew"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/user-request-template.
yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/candy-env"

- object: groups
  content:
  - name: "create groups"
    template:
"{{role_path}}/../../applier/multitenant/user-group-templates/group-request-template
.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/charlie-env"

- object: project
  content:
  - name: "create project"
    template:
"{{role_path}}/../../applier/multitenant/project-templates/multitenant-project-templ
ate.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/charlie-env"

- object: app
```

```
  content:
  - name: "deploy app"
    template:
"{{role_path}}/../../applier/deploy-client-projects/templates/sample-app.yaml"
    params: "{{role_path}}/../../applier/deploy-client-projects/params/charlie-env"
```

RED HAT CONSULTING

Appendix A: Architecture diagram



Appendix B: Ansible Host file used for installation

```
[OSEv3:vars]

###########################################################################
### Ansible Vars
###########################################################################
timeout=60
ansible_become=yes
ansible_ssh_user=ec2-user

###########################################################################
### OpenShift Basic Vars
###########################################################################
openshift_deployment_type=openshift-enterprise
deployment_type=openshift-enterprise
containerized=false
openshift_disable_check="memory_availability"

# Default node selectors
osm_default_node_selector='client=common'
openshift_hosted_infra_selector="env=infra"

###########################################################################
### OpenShift Master Vars
###########################################################################

openshift_master_api_port=443
openshift_master_console_port=443

openshift_master_cluster_method=native
openshift_master_cluster_hostname=loadbalancer1.bd03.internal
openshift_master_cluster_public_hostname=loadbalancer1.bd03.example.opentlc.com
openshift_master_default_subdomain=apps.bd03.example.opentlc.com
#openshift_master_ca_certificate={'certfile': '/root/intermediate_ca.crt', 'keyfile':
'/root/intermediate_ca.key'}
openshift_master_overwrite_named_certificates=True

# Set this line to enable NFS
openshift_enable_unsupported_configurations=True

###########################################################################
### OpenShift Network Vars
###########################################################################

os_sdn_network_plugin_name='redhat/openshift-ovs-networkpolicy'

###########################################################################
```

```
### OpenShift Authentication Vars
###########################################################################

# htpasswd Authentication
openshift_master_identity_providers=[{'name': 'htpasswd_auth', 'login': 'true',
'challenge': 'true', 'kind': 'HTPasswdPasswordIdentityProvider', 'filename':
'/etc/origin/master/htpasswd'}]
openshift_master_htpasswd_file=/root/htpasswd.openshift


###########################################################################
### OpenShift Router and Registry Vars
###########################################################################

openshift_hosted_router_replicas=2

openshift_hosted_registry_replicas=1

openshift_hosted_registry_storage_kind=nfs
openshift_hosted_registry_storage_access_modes=['ReadWriteMany']
openshift_hosted_registry_storage_nfs_directory=/srv/nfs
openshift_hosted_registry_storage_nfs_options='*(rw,root_squash)'
openshift_hosted_registry_storage_volume_name=registry
openshift_hosted_registry_storage_volume_size=20Gi
openshift_hosted_registry_pullthrough=true
openshift_hosted_registry_acceptschema2=true
openshift_hosted_registry_enforcequota=true
openshift_hosted_router_selector="env=infra"
openshift_hosted_registry_selector="env=infra"
###########################################################################
### OpenShift Service Catalog Vars
###########################################################################

openshift_enable_service_catalog=true

template_service_broker_install=true
openshift_template_service_broker_namespaces=['openshift']

ansible_service_broker_install=true
ansible_service_broker_local_registry_whitelist=['.*-apb$']

openshift_hosted_etcd_storage_kind=nfs
openshift_hosted_etcd_storage_nfs_options="*(rw,root_squash,sync,no_wdelay)"
openshift_hosted_etcd_storage_nfs_directory=/srv/nfs
openshift_hosted_etcd_storage_labels={'storage': 'etcd-asb'}
openshift_hosted_etcd_storage_volume_name=etcd-asb
openshift_hosted_etcd_storage_access_modes=['ReadWriteOnce']
openshift_hosted_etcd_storage_volume_size=10G
```

```
###########################################################################
### OpenShift Metrics and Logging Vars
###########################################################################
# Enable cluster metrics
openshift_metrics_install_metrics=True

openshift_metrics_storage_kind=nfs
openshift_metrics_storage_access_modes=['ReadWriteOnce']
openshift_metrics_storage_nfs_directory=/srv/nfs
openshift_metrics_storage_nfs_options='*(rw,root_squash)'
openshift_metrics_storage_volume_name=metrics
openshift_metrics_storage_volume_size=10Gi
openshift_metrics_storage_labels={'storage': 'metrics'}

openshift_metrics_cassandra_nodeselector={"env":"infra"}
openshift_metrics_hawkular_nodeselector={"env":"infra"}
openshift_metrics_heapster_nodeselector={"env":"infra"}

# Enable cluster logging
openshift_logging_install_logging=True

openshift_logging_storage_kind=nfs
openshift_logging_storage_access_modes=['ReadWriteOnce']
openshift_logging_storage_nfs_directory=/srv/nfs
openshift_logging_storage_nfs_options='*(rw,root_squash)'
openshift_logging_storage_volume_name=logging
openshift_logging_storage_volume_size=10Gi
openshift_logging_storage_labels={'storage': 'logging'}

openshift_logging_kibana_hostname=kibana.apps.bd03.example.opentlc.com
openshift_logging_es_cluster_size=1

openshift_logging_es_nodeselector={"env":"infra"}
openshift_logging_kibana_nodeselector={"env":"infra"}
openshift_logging_curator_nodeselector={"env":"infra"}


###########################################################################
### OpenShift Prometheus Vars
###########################################################################

## Add Prometheus Metrics:
openshift_hosted_prometheus_deploy=true
openshift_prometheus_node_selector={"env":"infra"}
openshift_prometheus_namespace=openshift-metrics

# Prometheus
openshift_prometheus_storage_kind=nfs
```

```
openshift_prometheus_storage_access_modes=['ReadWriteOnce']
openshift_prometheus_storage_nfs_directory=/srv/nfs
openshift_prometheus_storage_nfs_options='*(rw,root_squash)'
openshift_prometheus_storage_volume_name=prometheus
openshift_prometheus_storage_volume_size=10Gi
openshift_prometheus_storage_labels={'storage': 'prometheus'}
openshift_prometheus_storage_type='pvc'
# For prometheus-alertmanager
openshift_prometheus_alertmanager_storage_kind=nfs
openshift_prometheus_alertmanager_storage_access_modes=['ReadWriteOnce']
openshift_prometheus_alertmanager_storage_nfs_directory=/srv/nfs
openshift_prometheus_alertmanager_storage_nfs_options='*(rw,root_squash)'
openshift_prometheus_alertmanager_storage_volume_name=prometheus-alertmanager
openshift_prometheus_alertmanager_storage_volume_size=10Gi
openshift_prometheus_alertmanager_storage_labels={'storage': 'prometheus-alertmanager'}
openshift_prometheus_alertmanager_storage_type='pvc'
# For prometheus-alertbuffer
openshift_prometheus_alertbuffer_storage_kind=nfs
openshift_prometheus_alertbuffer_storage_access_modes=['ReadWriteOnce']
openshift_prometheus_alertbuffer_storage_nfs_directory=/srv/nfs
openshift_prometheus_alertbuffer_storage_nfs_options='*(rw,root_squash)'
openshift_prometheus_alertbuffer_storage_volume_name=prometheus-alertbuffer
openshift_prometheus_alertbuffer_storage_volume_size=10Gi
openshift_prometheus_alertbuffer_storage_labels={'storage': 'prometheus-alertbuffer'}
openshift_prometheus_alertbuffer_storage_type='pvc'

# Necessary because of a bug in the installer on 3.9
openshift_prometheus_node_exporter_image_version=v3.9

###########################################################################
### OpenShift Hosts
###########################################################################
[OSEv3:children]
lb
masters
etcd
nodes
nfs

[lb]
loadbalancer1.bd03.internal

[masters]
master1.bd03.internal
master2.bd03.internal
master3.bd03.internal

[etcd]
master1.bd03.internal
```

```
master2.bd03.internal
master3.bd03.internal

[nodes]
## These are the masters
master1.bd03.internal openshift_hostname=master1.bd03.internal
openshift_node_labels="{'env':'master', 'cluster': 'bd03'}"
master2.bd03.internal openshift_hostname=master2.bd03.internal
openshift_node_labels="{'env':'master', 'cluster': 'bd03'}"
master3.bd03.internal openshift_hostname=master3.bd03.internal
openshift_node_labels="{'env':'master', 'cluster': 'bd03'}"

## These are infranodes
infranode1.bd03.internal openshift_hostname=infranode1.bd03.internal
openshift_node_labels="{'env':'infra', 'cluster': 'bd03'}"
infranode2.bd03.internal openshift_hostname=infranode2.bd03.internal
openshift_node_labels="{'env':'infra', 'cluster': 'bd03'}"

## These are regular nodes
node1.bd03.internal openshift_hostname=node1.bd03.internal
openshift_node_labels="{'client':'alpha', 'cluster': 'bd03'}"
node2.bd03.internal openshift_hostname=node2.bd03.internal
openshift_node_labels="{'client':'beta', 'cluster': 'bd03'}"
node3.bd03.internal openshift_hostname=node3.bd03.internal
openshift_node_labels="{'client':'common', 'cluster': 'bd03'}"

[nfs]
support1.bd03.internal openshift_hostname=support1.bd03.internal
```

Appendix C: Multi-tenant project template

MULTITENANT PROJECT REQUEST TEMPLATE

```
apiVersion: v1
kind: Template
metadata:
  creationTimestamp: null
  name: multitenant-project-request
  namespace: default
objects:
```

```
- apiVersion: v1
  kind: Project
  metadata:
    annotations:
      openshift.io/description: ${PROJECT_DESCRIPTION}
      openshift.io/display-name: ${PROJECT_DISPLAYNAME}
      scheduler.alpha.kubernetes.io/node-selector: ${CLIENT_NODE_SELECTOR}
      openshift.io/node-selector: ${CLIENT_NODE_SELECTOR}
    creationTimestamp: null
    name: ${PROJECT_NAME}
  spec: {}
  status: {}
- apiVersion: networking.k8s.io/v1
  kind: NetworkPolicy
  metadata:
    name: allow-same-namespace
    namespace: ${PROJECT_NAME}
  spec:
    podSelector:
    ingress:
    - from:
      - podSelector: {}
- apiVersion: networking.k8s.io/v1
  kind: NetworkPolicy
  metadata:
    name: allow-from-default-namespace
    namespace: ${PROJECT_NAME}
  spec:
    podSelector:
    ingress:
    - from:
      - namespaceSelector:
          matchLabels:
            name: default
- apiVersion: v1
  kind: ResourceQuota
  metadata:
    name: ${PROJECT_NAME}-quota
    namespace: ${PROJECT_NAME}
  spec:
    hard:
      memory: 1024Mi
      cpu: 500m
      pods: 3
      resourcequotas: 1
- apiVersion: v1
  kind: LimitRange
  metadata:
    name: ${PROJECT_NAME}-limits
    creationTimestamp: null
    namespace: ${PROJECT_NAME}
  spec:
    limits:
      -
        type: Pod
        max:
          cpu: 500m
          memory: 750Mi
        min:
          cpu: 10m
          memory: 5Mi
      -
        type: Container
        max:
          cpu: 500m
          memory: 750Mi
        min:
          cpu: 10m
          memory: 5Mi
        default:
          cpu: 250m
          memory: 500Mi
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: admins
    namespace: ${PROJECT_NAME}
  roleRef:
    name: admin
  subjects:
  - kind: User
    name: ${PROJECT_ADMIN_USER}
  userNames:
  - ${PROJECT_ADMIN_USER}
- apiVersion: v1
  groupNames:
  - ${GROUP_NAME}
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: normal-users
    namespace: ${PROJECT_NAME}
  roleRef:
    name: edit
- apiVersion: v1
  groupNames:
  - system:serviceaccounts:${PROJECT_NAME}
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:image-pullers
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:image-puller
  subjects:
  - kind: SystemGroup
    name: system:serviceaccounts:${PROJECT_NAME}
  userNames: []
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:image-builders
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:image-builder
  subjects:
  - kind: ServiceAccount
    name: builder
  userNames:
  - system:serviceaccount:${PROJECT_NAME}:builder
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:deployers
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:deployer
  subjects:
  - kind: ServiceAccount
    name: deployer
  userNames:
  - system:serviceaccount:${PROJECT_NAME}:deployer
parameters:
- name: PROJECT_NAME
- name: PROJECT_DISPLAYNAME
- name: PROJECT_DESCRIPTION
- name: PROJECT_ADMIN_USER
- name: GROUP_NAME
- name: CLIENT_NODE_SELECTOR
```

DEFAULT PROJECT REQUEST TEMPLATE

```
apiVersion: v1
kind: Template
metadata:
  creationTimestamp: null
  name: default-project-request
  namespace: default
objects:
- apiVersion: v1
  kind: Project
  metadata:
    annotations:
      openshift.io/description: ${PROJECT_DESCRIPTION}
      openshift.io/display-name: ${PROJECT_DISPLAYNAME}
    creationTimestamp: null
    name: ${PROJECT_NAME}
  spec: {}
  status: {}
- apiVersion: networking.k8s.io/v1
```

```
  kind: NetworkPolicy
  metadata:
    name: allow-same-namespace
    namespace: ${PROJECT_NAME}
  spec:
    podSelector:
    ingress:
    - from:
      - podSelector: {}
- apiVersion: networking.k8s.io/v1
  kind: NetworkPolicy
  metadata:
    name: allow-from-default-namespace
    namespace: ${PROJECT_NAME}
  spec:
    podSelector:
    ingress:
    - from:
      - namespaceSelector:
          matchLabels:
            name: default
- apiVersion: v1
  kind: ResourceQuota
  metadata:
    name: ${PROJECT_NAME}-quota
    namespace: ${PROJECT_NAME}
  spec:
    hard:
      memory: 1024Mi
      cpu: 500m
      pods: 3
      resourcequotas: 1
- apiVersion: v1
  kind: LimitRange
  metadata:
    name: ${PROJECT_NAME}-limits
    creationTimestamp: null
    namespace: ${PROJECT_NAME}
  spec:
    limits:
      -
        type: Pod
        max:
          cpu: 500m
          memory: 750Mi
        min:
          cpu: 10m
          memory: 5Mi
      -
        type: Container
        max:
          cpu: 500m
          memory: 750Mi
        min:
          cpu: 10m
          memory: 5Mi
        default:
          cpu: 250m
          memory: 500Mi
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: admins
    namespace: ${PROJECT_NAME}
  roleRef:
    name: admin
  subjects:
  - kind: User
    name: ${PROJECT_ADMIN_USER}
  userNames:
  - ${PROJECT_ADMIN_USER}
- apiVersion: v1
  groupNames:
  - system:serviceaccounts:${PROJECT_NAME}
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:image-pullers
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:image-puller
  subjects:
  - kind: SystemGroup
    name: system:serviceaccounts:${PROJECT_NAME}
  userNames: []
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:image-builders
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:image-builder
  subjects:
  - kind: ServiceAccount
    name: builder
  userNames:
  - system:serviceaccount:${PROJECT_NAME}:builder
- apiVersion: v1
  groupNames: []
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: system:deployers
    namespace: ${PROJECT_NAME}
  roleRef:
    name: system:deployer
  subjects:
  - kind: ServiceAccount
    name: deployer
  userNames:
  - system:serviceaccount:${PROJECT_NAME}:deployer
parameters:
- name: PROJECT_NAME
- name: PROJECT_DISPLAYNAME
- name: PROJECT_DESCRIPTION
- name: PROJECT_ADMIN_USER
```

Appendix D: User and Group template

USER TEMPLATE

```
apiVersion: template.openshift.io/v1
kind: Template
metadata:
  creationTimestamp: null
  name: user-request-template
  namespace: default
objects:
- apiVersion: user.openshift.io/v1
  groups: null
  identities:
  - htpasswd_auth:${USER_NAME}
  kind: User
  metadata:
    creationTimestamp: null
    labels:
      ${CLIENT_LABEL_KEY}: ${CLIENT_LABEL_VALUE}
    name: ${USER_NAME}
parameters:
- name: USER_NAME
- name: CLIENT_LABEL_KEY
- name: CLIENT_LABEL_VALUE
```

Group Template

```
apiVersion: template.openshift.io/v1
kind: Template
metadata:
  creationTimestamp: null
  name: group-request-template
  namespace: default
objects:
- apiVersion: user.openshift.io/v1
  kind: Group
  metadata:
    creationTimestamp: null
    name: ${GROUP_NAME}
  users: ${{GROUP_USERS}}
parameters:
- name: GROUP_NAME
- name: GROUP_USERS
  value: "[]"
```