Table of Contents

Homework Assignment

Goals

- Assess hands-on proficiency with Red Hat OpenShift Container Platform Deployment advanced topics
- Complete course leading to **Red Hat Delivery Specialist - Advanced Platform-as-a-Service (PaaS) Administration** accrediation.

Criteria

- Assignments should take the average student 30-40 hours to complete
- Assignments are an individual effort
    - Each student completes his or her own assignment without collaboration
- Assignments should simulate a challenge typically encountered in a Red Hat Consulting engagement
    - Assignment requirements are intentionally a bit vague
- Grading is as follows:
    - **20%** : Basic Requirements section
    - **20%** : HA Deployment section
    - **20%** : Environment Configuration section
    - **20%** : CICD Workflow section
    - **20%** : Multitenancy section
- Passing grade: 80%

Submission - Engagement Journal and Git Repo

1. Engagement Journal
    - Submit your documentation like you would submit a client engagement journal.

- Provide sufficient documentation for each section.
- Sections without documentation will not be graded

2. Git Repo
   - The Git repo should be your own work
   - Deployment must run end-to-end without error.
   - Use as much ansible as possible. Shell is acceptable, but discouraged.
   - Extra-Credit: Use the best tools of the Containers and PaaS Community of Practice, detailed below
   - Upload your homework with
     - a comment with:
       - version of OCP used (e.g. 3.9.41, 3.10.34, 3.11.16)
       - your class date-location-instructor's name (for your grader's reference)
       - the single command to run to deploy your environment
   - Upload your homework to
     - Red Hat LMS (https://docs.google.com/document/d/1nxlvAOlSdNs3-y8AkmDjnc8vtCH9rJdI5zbN9deCK50/edit) if you are a Red Hat employee
     - Red Hat Connect (https://partner.redhat.com) (partner.redhat.com) if you are a business partner

---

1. Business Use Case

You are a consultant assigned to a telecommunications company called MitziCom. MitziCom provides hosting and cloud services to a variety of clients, from medium size companies to enterprise giants.

MitziCom has asked you to lead a 30-40 hour proof-of-concept (POC) using Red Hat OpenShift Container Platform. The purpose of the POC is to determine the feasibility of using Red Hat OpenShift Container Platform as a target for internal and client workloads.

---

2. POC Requirements

MitziCom management requires that you include all of the items listed in these subsections in your POC.

Full Automation

MitziCom will be deploying your work onto their own systems. Management wants to ease the burden on SREs, so they must be able to deploy your work on their infrastructure in an **automated fashion** in a **single command.**

Automation Requirements

- Create a public GitHub repository with all your work which can be cloned onto a homework environment bastion host and executed to complete all the steps
- Include an Ansible inventory file based on the which deploys the desired OpenShift and its hosted components.
- Customizes the Ansible inventory file for different hostnames (replace GUID, at a minimum)
  - Extra-Credit: Ansible Inventory split into multiple files - hosts file and vars file in YAML format.
- Use htpasswd authentication
- Executes the OpenShift Ansible deployer
- Post-deployment, sets up storage and other environment configurations
  - Extra-Credit: Use the Red Hat Community of Practice tools to automate `oc apply` when creating OpeShift objects: https://github.com/redhat-cop/openshift-applier (https://github.com/redhat-cop/openshift-applier)
- Deploys app and executes the CICD Workflow
  - Extra-Credit: Use the Red Hat Community of Practice style for application deployment: https://github.com/redhat-cop/container-pipelines/tree/master/basic-spring-boot (https://github.com/redhat-cop/container-pipelines/tree/master/basic-spring-boot)
- Creates all OpenShift objects necessary for multitenancy
- In the Engagement journal - Provide instructions for the MitziCom administrator to deploy all the above in a similar environment in a **single** command

> Here is a simple structure and inventory to get you started if the Red Hat CoP formats are too confusing: https://github.com/newgoliath/ocp_advanced_deployment_homework (https://github.com/newgoliath/ocp_advanced_deployment_homework)

2.1. Basic Requirements

- Ability to authenticate at the master console
- Registry has storage attached and working
- Router is configured on each infranode
- PVs of different types are available for users to consume
- Ability to deploy a simple app (`nodejs-mongo-persistent`)

2.2. HA Requirements

- There are three masters working
- There are three `etcd` instances working
- There is a load balancer to access the masters called loadbalancer.$GUID.$DOMAIN
- There is a load balancer/DNS for both infranodes called *.apps.$GUID.$DOMAIN
- There are at least two infranodes, labeled env=infra

2.3. Environment Configuration

- NetworkPolicy is configured and working with projects isolated by default
- Aggregated logging is configured and working
- Metrics collection is configured and working
- Router and Registry Pods run on Infranodes
- Metrics and Logging components run on Infranodes
- Service Catalog, Template Service Broker, and Ansible Service Broker are all working

2.4. CICD Workflow

- Jenkins pod is running in a project called `cicd-dev` with a persistent volume
- Jenkins deploys `openshift-tasks` app
- Jenkins OpenShift plugin is used in the buildconfig to create a CICD workflow
- Use `tasks-build` as your project for building. Do promotion between projects `tasks-dev`, `tasks-test`, and `tasks-prod` as your project names.
- HPA called `tasks-hpa` is configured and working on production deployment of `openshift-tasks` in project `tasks-prod`.

2.5. Multitenancy

- Multiple Clients (customers) created
  - Clients will be named Alpha Corp and Beta Corp (client=alpha, client=beta), and a "client=common" for unspecified customers.
  - Alpha Corp will have two users, Amy and Andrew
  - Beta Corp will have two users, Brian and Betty
  - Common will be for all other users workloads
- Dedicated node for each Client
- The new project template is modified so that it includes a LimitRange
- A new user template is used to create a user object with the specific label value (optional)
- Alpha and Beta Corp users are confined to projects, and all new pods are deployed to customer dedicated nodes

Build Version: 4932ae803559c5c4b7a5259dcc71b2a5cbd60481 : Last updated 2018-11-09 12:06:52 EST