

OPENSIFT v4.14 Exam Paper

Important Instructions:

Duration: 3Hrs
Total marks: 300
Pass Marks: 210

utility.lab.example.com (172.25.250.14)

master01.lab.example.com 172.25.250.12 (SNO Cluster)

Server API URL: <https://api.ocp4.example.com:6443>

1. Wild-card domain for the cluster: apps.ocp4.example.com
2. Documentation about OpenShift can be accessed at the following url:
https://access.redhat.com/documentation/enus/openshift_container_platform/4.14/
3. Kubeadmin password will be available in the location as
/home/student/kubeadmin-password in the workbench VM itself.
4. User opsadm user will be given.
5. opsadm password for login to workbench VM will be provided in instructions.

Question Outline

1. Configure the Identity Provider for the OpenShift.
2. Configure Cluster permissions
3. Configure Project permissions
4. Create Groups and configure permissions
5. Configure Quotas for Project
6. Configure Limits for Project
7. Configure and deploy a secure route
8. Scale the Application manually .
9. Configure Auto-scaling for an application.
10. Configure a Secret .
11. Use the Secret value for Application Deployment.
12. Configure a Service Account.
13. Deploy an application.
14. Deploy an application.
15. Network policy.
16. Persistent storage.
17. Install Operator.
18. Cronjob.
19. Bootstrap Project template.

- 20. Monitoring and health-check
- 21. Helm Chart.
- 22. Set liveness probe.

Login Instructions:

```
[desktop@desktop]$ ssh opsadm@workbentch.ocp4.example.com
```

```
[opsadm@workbentch]$ cat /home/opsadm/kubeadmin-password
```

```
[opsadm@workbentch]$ oc login -u kubeadmin -p admin --server
```

```
https://api.ocp4.example.com:6443
```

```
[opsadm@workbentch]$ oc whoami
```

System:admin - -> once logged cluster with kubeadmin user then start working on questions

Questions & Solutions

1) Configure the Identity Provider for the Openshift.

- Create an htpasswd Identity Provider with the name: htpass-ex280
- Create the secret for Identity provider users: htpass-idp-ex280
- Create the user account jobs with password jobs123
- Create the user account wozniak with password wozniak123
- Create the user account collins with password collins123
- Create the user account adlerin with password adlerin123
- Create the user account armstrong with password armstrong123

Ans:-

```
$ rpm -qi httpd-tools
```

```
$ htpasswd -b -c -b htpassfile jobs jobs123
```

```
$ htpasswd -b htpassfile Wozniak wozniak123
```

```
$ htpasswd -b htpassfile collins collins123
```

```
$ htpasswd -b htpassfile adlerin adlerin123
```

```
$ htpasswd -b htpassfile Armstrong armstrong123
```

```
$ oc create secret generic htpass-idp-ex280 --from-file  
htpasswd=/home/student/htpassfile -n openshift-config
```

```
$ oc get oauth cluster -oyaml > oauth.yaml
```

```
$ vim /home/student/oauth.yaml
```

spec:

identityProviders:

- name: htpass-ex280

mappingMethod: claim

```
type: HTTPasswd
htpasswd:
  fileName:
    name: htpass-idp-ex280
```

:wq!

```
$ oc apply -f /home/student/oauth.yaml
```

```
$ watch oc get pods -n openshift-authentication (verify: new pods will trigger)
```

2) Configure Cluster permissions

- User jobs is able to modify the cluster
- wozniak is able to create projects
- armstrong cannot create projects.
- wozniak cannot modify the cluster
- Remove the kubeadmin user from the cluster

Ans:-

```
$ oc adm policy add-cluster-role-to-user cluster-admin jobs
```

```
$ oc adm policy remove-cluster-role-from-group self provisioner
system:authenticated:oauth
```

```
$ oc adm policy add-cluster-role-to-user self-provisioner wozniak
```

```
$ oc delete secret kubeadmin -n kube-system
```

Note: Remove kubeadmin by end of exam (or) login cluster 1st with jobs user and remove kubeadmin here.

3) Configure Project permissions

- Create following projects
 - apollo
 - titan
 - bluebook
 - apache
 - gemini
- User armstrong is admin for the apollo and titan project
- User collins is able to view the apollo project.

Ans:-

```
$ oc whoami
```

```
$ oc new-project apollo
```

```
$ oc new-project titan
```

```
$ oc new-project gemini
$ oc new-project bluebook
$ oc new-project apache
$ oc policy add-role-to-user admin armstrong -n apollo
$ oc policy add-role-to-user admin armstrong -n titan
$ oc policy add-role-to-user view collins -n apollo
```

NOTE: Once you done with 3rd Questions login to cluster with all users who ever we created in 1st Question.

NOTE: if you want to delete kubeadmin in 2nd Question 1st login cluster with jobs and remove (or) remove kubeadmin by end of exam once login with jobs user

4) Create Groups and configure permissions

- Create a group called commander and user wozniak is the member of this group
- Create a group called pilot and user adlerin is the member of this group.
- The commander group members are able to edit the apache and gemini project
- The pilot group members are able to view apache project but not edit it.

Ans:-

```
$ oc adm groups new commander
$ oc adm policy new pilot
$ oc adm groups add-users commander wozniak
$ oc adm groups add-users pilot adlerin
$ oc adm policy add-role-to-group edit commander -n apache
$ oc adm policy add-role-to-group edit commander -n gemini
$ oc adm policy add-role-to-group view pilot -n apache
```

5) Configure Quotas for the Project

- The amount of memory consumed across all containers may not exceed 1Gi
- The amount of CPU across all containers may not exceed 2 full cores.
- The maximum number of replication controllers does not exceed 3
- The maximum number of pods does not exceed 3
- The maximum number of services does not exceed 6

Ans:-

```
$ oc project apache
```

```
$ oc create quota --help|less (# copy 1st example and modify)
```

```
$ oc create quota ex280-quota
```

```
–hard=memory=1Gi,cpu=2,replicationcontrollers=3,pods=3,services=6 -n  
apache
```

```
$ oc get quota / $ oc describe quota ex280-quota (verify)
```

6) Configure Limits for the Project

- The amount of memory consumed by a single pod is between 100Mi and 300Mi
- The amount of cpu consumed by a single pod is between 10m and 500m
- The amount of cpu consumed by a single container is between 10m and 500m with a default request value of 100m
- The amount of memory consumed by a single container is between 100Mi and 300Mi with a default request value of 100Mi

Ans:-

```
$ oc project bluebook
```

```
$ vim ex280-limits.yaml
```

```
apiVersion: v1
kind: LimitRange
metadata:
  name: ex280-limits
  namespace: bluebook
spec:
  limits:
    - type: Pod
      max:
        memory: "500Mi"
        cpu: "300m"
      min:
        memory: "100Mi"
        cpu: "10m"
    - type: Container
      max:
        memory: "500Mi"
        cpu: "300m"
      min:
        memory: "100Mi"
        cpu: "10m"
      defaultRequest:
        memory: "100Mi"
        cpu: "100m"
```

INSERT --

\$ oc create -f ex280-limits.yaml --dry-run=server --validate=true

\$ oc create -f ex280-limits.yaml

\$ oc describe limitrange ex280-limits

7). Configure and deploy a secure route Deploy an application called oxcart securely in the project called area51

- The application has self-signed certificate available at
"/C=US/ST=NC/L=Raleigh/O=RedHat/OU=RHT/CN=oxcart.apps.ocp4.example.com" Use newcert command to generate required tls objects, pass the subject as argument value
- The application should be reachable at the following url
<https://oxcart.apps.ocp4.example.com>
- Application produces a valid Output

Practice: Lab setup

In Practice create a deployment using

quay.io/redhattraining/hello-world-nginx:v1.0 image

```
$ oc new-project area51
```

```
$ oc project area51
```

```
$ oc new-app oxcart --image=quay.io/redhattraining/hello world-nginx:v1.0
```

```
$ oc expose svc oxcart
```

```
$ oc get all
```

Ans:-

```
$ oc project are51
```

```
$ oc get all
```

```
$ oc get pods
```

Note:-

- In exam newcert script file available in /usr/local/bin/newcert
- First copy newcert file to opsadm user home directory and add execute permission for script and run the script

```
$ sh newcert
```

Note:-

(while running the script it's promoting for subject, add subject as you have in question, then it's create private key, csr, cert file in current user home directory, use those files to create secure route)

First prefer to generate certificate with script, incase if you facing any issue while running script use openssl command tools to generate self sign certificate with same subject line

Using below steps:-

Step-1: generate private

```
$ openssl genrsa -out private.key
```

step-2: generate csr with private key

```
$ openssl req -new -key private.key -out redhat.csr --  
subj="/C=US/ST=NC/L=Raleigh/O=RedHat/OU=RHT/CN=oxcart.apps.oc  
p4.example.com"
```

step-3: generate certificate using private key and csr

```
$ openssl x509 -req -in redhat.csr -signkey private.key -days 366 -out redhat.crt  
$ ls -l
```

Private.key , redhat.csr, redhat.crt (output)

```
$ oc create route edge --service=oxcart --hostname= oxcart.apps.oc  
p4.example.com --key=/home/opsadm/private.key cert=/home/opsadm/redhat.crt  
$ oc get route  
https://oxcart.apps.oc p4.example.com (# verify in browser)
```

8) Scale the Application manually

- Scale an application called hydra in the project called lerna The hydra application should be scaled to five times

Practice:

```
$ oc new-project lerna  
$ oc project lerna  
$ oc create deployment hydra  
--image=quay.io/redhattraining/hello-world-nginx:v1.0
```

Ans:-

```
$ oc project lerna  
$ oc get all  
$ oc get pods  
$ oc scale deployment/hydra --replicas=5  
$ oc get pods (Verify)
```

9) Configure Autoscaling for an Application

- Configure an autoscaling for the scala application in the project gru with following specification.
 - Minimum number of replicas: 6
 - Maximum number of replicas: 40
 - Threshold CPU-Percentage: 60
 - Application resource of CPU Request: 25m
 - Application limits of CPU Limits: 100m

Practice:

```
$ oc new-project gru
$ oc project gru
$ oc create deployment scala
--image=quay.io/redhattraining/hello-world-nginx:v1.0
```

Ans:-

```
$ oc project gru
$ oc get all
$ oc get pods
$ oc get deployment/scala -oyaml > scala.yaml
$ vim scala.yaml
```

spec:

containers:

resources: (# remove braces and add resource requests and limits weather it's cpu (or) memory)

requests:

cpu: "25m"

limits:

cpu: "100m"

:wq

```
$ oc apply -f scala.yaml $ oc autoscale deployment/scala --max=40 --min=6
--cpu percentage=60
```

```
$ oc get hpa (# once hpa created 6 new pods will trigger)
```

```
$ oc get pods
```

10) Configure a Secret

- Configure a secret in the math project and the name of secret should be magic.
- The secret should have following key value pairs Decoder_Ring:
ASDA142hfh-gfrhhueo-erfdk345v

Practice:

```
$ oc new-project math
$ oc project math
$ oc create deployment qed -
image=quay.io/redhattraining/hello-world-nginx:v1.0
```

Ans:-

```
$ oc project math
$ oc create secret generic magic --from-literal
ring=ASDA142hfh-gfrhhueo-erfdk345v
$ oc get secrets
```

11) Use the Secret value for Application Deployment

- Configure the environmental variable for the application called qed in the math project so that it uses the secret “magic
- After configuring the environmental value for the application, it should stop producing the following output “App is not configured properly

Ans:-

```
$ oc project math
$ oc get all
$ oc get deployment
$ oc set env deployment/qed --from secret/magic --prefix DECODE_
$ oc get pods (# pod will restarts)
$ oc describe pod qed ( # verify env )
```

12). Configure a Service Account

- Create a service account called ex280-sa in the project called apples
- This service account should be able to run application with any user id

Practice:

```
$ oc new-project apples
$ oc project apples
$ oc new-app oranges --image=quay.io/redhattraining/hello world-nginx:v1.0
$ oc expose svc oranges
$ oc get all
```

Ans:-

```
$ oc project apples
$ oc create sa ex-280-sa
$ oc adm policy add-scc-to-user anyuid -z ex280-sa
```

13) Deploy an application

- Deploy an application called oranges in the project called apples
- This application should use the service account ex-280-sa
- The Application should produce a valid output

Ans:-

```
$ oc project apples
$ oc get all
$ oc get pods
$ oc set sa deployment/oranges ex280-sa
$ oc get route (# verify in browser , app should produce output)
$ oc describe svc oranges
```

14) Deploy an application Deploy an application called voyager in the project path-finder

- Don't add any new configuration
- Application should produce a valid output

Practice:-

```
$ oc new-project path-finder
$ oc create deployment voyager --image
quay.io/redhattraining/hello-world-nginx:v1.0
$ oc get all
```

Ans:-

Nothing to do here

15) Set active liveness probes for the application in the path-finder project.

- It should utilize the tcp socket port:8080
- Probe parameters,initial-delay-seconds=3 and timeout seconds=10

Ans:-

```
$ oc project path-finder
$ oc get all
$ oc set probe deployment.apps/voyager --liveness --open-tcp=8080
--initial-delay-seconds=3 --timeout-seconds=10
$ oc get pods (New pod will reflect)
```

16) Monitoring and health-check

- collect all cluster information log files and diagnostic information about your cluster.
- Store the result at /home/student/ex280-clusterID.tar.gz

Note:- In the question paper you will get one script file which you need to execute with your tar file to send all reports to RedHat.

Ans:-

```
$ oc adm must-gather --dest-dir /home/student/ex280-cluster-data
```

```
$ tar -cvaf /home/student/ex280-cluster-data /home/student/ex280
```

```
clusterID.tar.gz
```

```
$ sh <scriptfile> /home/student/ex280 clusterID.tar.gz
```

Note:- clusterID Means your cluster ID

17) Deploy application through Charts

- Deploy an application and its dependencies from a Helm chart
- Helm Repo: <http://helm.ocp4.example.com/charts>
- Repo Name: ex280-repo
- Install the etherpad chart to the charts-development project.
- Application Should be available at the following URL
https://etherpad-charts-development.apps.ocp4.example.com

Ans:-

```
$ oc new-project charts-development
```

```
$ helm repo list
```

```
$ helm repo add ex280-repo http://helm.ocp4.example.com/charts
```

```
$ helm search repo --versions
```

```
$ helm install etherpad ex280-repo/etherpad
```

```
$ oc get all
```

```
$ oc get route
```

18) Deploy a cronjob

- Cronjob should be part of marathon project
- Cronjob is named as the application image.
- Use the image quay.io/redhattraining/scaling
- The Job should be executed at 4.05 every 2nd day of the month.
- Use Service account called ex280-ocpsa
- Service accounts should have privileges to run job at cluster scope.
- Successful Job History Limit should be 13

Ans:-

```
$ oc new-project marathon
$ oc project marathon
$ oc create deployment.apps/scaling -- image=quay.io/redhattraining/scaling
$ oc get deployment scaling -oyaml | oc adm policy scc-subject review -f --
```

RESOURCE	ALLOWED BY
Pod/scaling-6b85dd-nhr	anyuid

Note: Once check the required scc policy for service account delete the deployment

```
$ oc create cronjob scaling -- image=quay.io/redhattraining/scaling --schedule
"05 04 02 * *"
```

```
$ oc create sa ex280-ocpsa
$ oc adm policy add-scc-to-user anyuid -z ex280-ocpsa
$ oc set sa cronjob/scaling ex280-ocpsa
$ oc edit cronjob scaling
```

Spec:

successfulJobHistoryLimit: 13

:wq

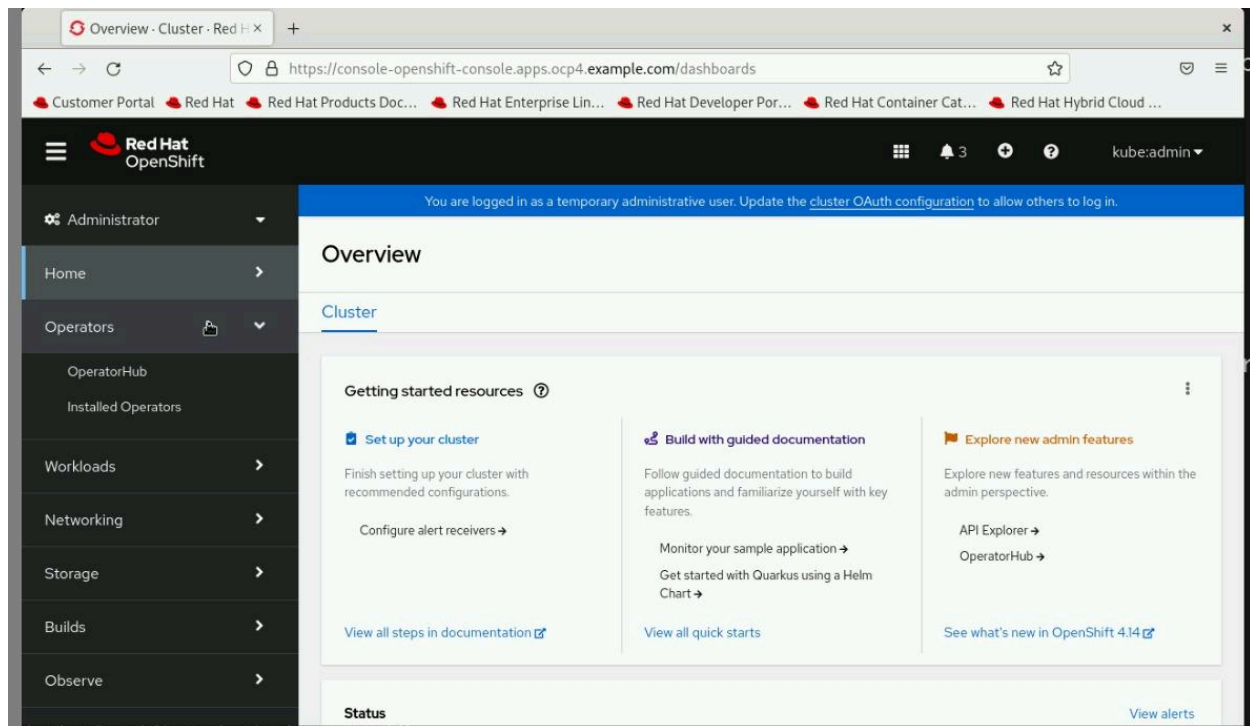
19) Install Operator

- Install File Integrity operator in the openshift-file integrity project. Operator should have stable channel
- Operator should automatically update itself through OLM

Ans:-

```
$ oc whoami --show-console
```

Open the url in firefox and login into jobs user with password



20) Bootstrap Project template

- As the administrator, update the OpenShift cluster to use a new project template
- The project template must automatically create the limit range, and add quota resources for new projects

For example, a project named test has the quota and limit range as test-quota and test-limitrange respectively

- Project should have default LimitRange as below
 - The amount of memory consumed by a single pod is between 100Mi and 300Mi
 - The amount of cpu consumed by a single pod is between 10m and 500m
 - The amount of cpu consumed by a single container is between 10m and 500m with a default request value of 100m and default 100m
 - The amount of memory consumed by a single container is between 100Mi and 300Mi with a default request value of 100Mi and default 100Mi

Or

- Project should have default ResourceQuota as below
 - Projects are limited to 10 pods

- Projects can request a maximum of 1 GiB of memory
- Projects can request a maximum of 2 CPUs
- Projects can use a maximum of 4 GiB of memory
- Projects can use a maximum of 4 CPUs

Ans:-

```
$ oc adm create-bootstrap-project-template -oyaml > project-template.yaml
```

```
$ vim quota-limits.yaml
```

```

1 - apiVersion: v1
2   kind: LimitRange
3   metadata:
4     name: ${PROJECT_NAME}-limitrange
5   spec:
6     limits:
7       - type: Pod
8         max:
9           memory: "300Mi"
10          cpu: "500m"
11        min:
12          memory: "100Mi"
13          cpu: "10m"
14       - type: Container
15         max:
16           memory: "300Mi"
17           cpu: "500m"
18         min:
19           memory: "100Mi"
20           cpu: "10m"
21         defaultRequest:
22           memory: "100Mi"
23           cpu: "100m"
24         default:
25           memory: "100Mi"
26           cpu: "100m"
27 - apiVersion: v1
28   kind: ResourceQuota
29   metadata:
30     name: ${PROJECT_NAME}-quota
31   spec:
32     hard:
33       pods: "10"
34       requests.memory: "1Gi"
35       requests.cpu: "2"
36       limits.memory: "4Gi"
-- INSERT --

```

```
$ cat quota-limits.yaml >> project-template.yaml
```

Note: Inside the project-template.yaml file make sure the **parameter** section should write at the end.

```
$ oc create -f project-template.yaml -n openshift-config
```

```
$ oc get template -n openshift-config
```

```
[student@workstation ~]$
[student@workstation ~]$ oc get template -n openshift-config
NAME              DESCRIPTION    PARAMETERS    OBJECTS
project-request           5 (5 blank)    6
[student@workstation ~]$
```

```
$ oc edit projects.config.openshift.io -oyaml
```

```
22 url: 1c038539-11a4-4ec3-8532-c0e
23 spec:
24   projectRequestTemplate:
25     name: project-request
```

```
$ watch oc get pods -n openshift-apiserver      [#pods restarts]
```

21) Persistent storage

- Deploy Application in the project
 - Create application named 'gamma' Application should be on project space
 - Application uses the image- quay.io/redhattraining/hello-world-nginx
 - Provision storage for the file system /srv
 - Application should be reachable at the following URL
https://space.apps.ocp4.example.com

Note: In exam NFS Server Details may not be given directly, you need to check NFS storage class, get the details using following commands

```
$ oc get sc
```

```
$ oc describe sc nfs-storage
```

mountOption:

Server: 192.168.50.254

Mount: /exports-ocp4

- Create a PVC named 'gamma-pvc' for space project
 - PVC should of size 1Gi
 - Claim mode should be same as PV
 - PVCshould bound on gamma-pv

Ans:-

```
$ oc new-project space
```

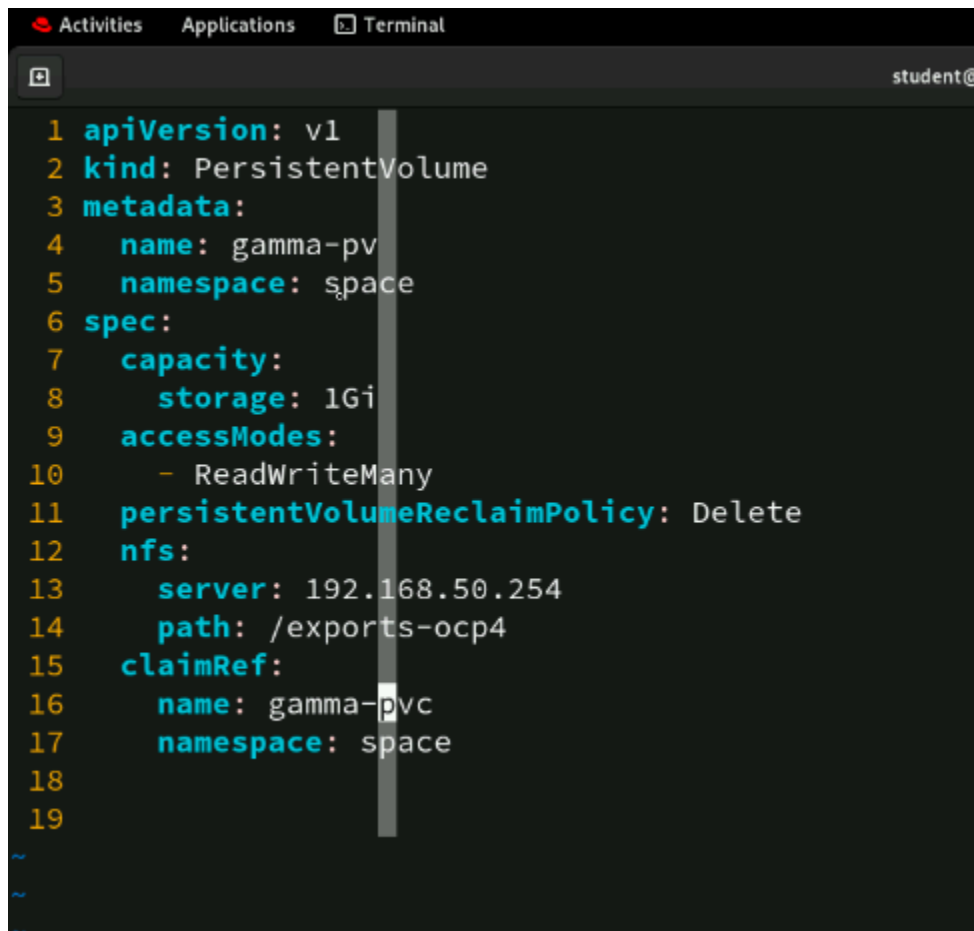
```
$ oc new-app --name gamma --image
```

quay.io/redhattraining/hello-world-nginx:v1.0

```
$ oc get all [#verify all resources are created]
```



```
$ oc get svc [#To check service resource name]
$ oc create route edge --service=gamma -
hostname=space.apps.ocp4.example.com
$ oc get route
$ oc get sc [check storage class]
$ oc get sc -oyaml [for nfs-hostname and exported path]
$ vim gamma-pv.yml
```

A screenshot of a terminal window with a dark background. The window title bar shows 'Activities', 'Applications', and 'Terminal'. The user is logged in as 'student@'. The terminal displays the contents of a YAML file named 'gamma-pv.yml'. The file defines a PersistentVolume with the following fields: apiVersion: v1, kind: PersistentVolume, metadata: {name: gamma-pv, namespace: space}, spec: {capacity: {storage: 1Gi}, accessModes: [ReadWriteMany]}, persistentVolumeReclaimPolicy: Delete, nfs: {server: 192.168.50.254, path: /exports-ocp4}, and claimRef: {name: gamma-pvc, namespace: space}. The lines are numbered 1 through 19. A vertical cursor is positioned at the end of line 17.

```
1 apiVersion: v1
2 kind: PersistentVolume
3 metadata:
4   name: gamma-pv
5   namespace: space
6 spec:
7   capacity:
8     storage: 1Gi
9   accessModes:
10    - ReadWriteMany
11 persistentVolumeReclaimPolicy: Delete
12 nfs:
13   server: 192.168.50.254
14   path: /exports-ocp4
15 claimRef:
16   name: gamma-pvc
17   namespace: space
18
19
```

```
$ vim gamma-pvc.yml
```

```

1 apiVersion: v1
2 kind: PersistentVolumeClaim
3 metadata:
4   name: gammapvc
5   namespace: space
6 spec:
7   resources:
8     requests:
9       storage: 1Gi
10  accessModes:
11    - ReadWriteMany
12

```

```

$ oc create -f gamma-pv.yaml
$ oc create -f gamma-pvc.yaml
$ oc set volumes deployment.apps/gamma --add --type pvc --
claim-name=gamma-pvc --mount-path=/svc
$ oc rsh <pod name>
$ df -Th [check mount-path and exit from pod]

```

22) Network Policy

- Create a NetworkPolicy in a database project with the name mysql-db-conn-policy to communicate the pod networking.k8s.io/v1/network=database.
- Policy uses labels to specify namespace with team=devsecops and for pods selector- Uses label deployment=web-mysql, via TCP on port 8080.

Ans:-

```
$ vim allow-network.yml
```

```
Activities Applications Terminal Mar 13 09:34
student@workstation:~ — v
1 apiVersion: networking.k8s.io/v1
2 kind: NetworkPolicy
3 metadata:
4   name: mysql-db-conn-policy
5   namespace: database
6 spec:
7   podSelector:
8     matchLabels:
9       networking.k8s.io/v1/network: database
10  policyTypes:
11    - Ingress
12  ingress:
13    - from:
14      - namespaceSelector:
15        matchLabels:
16          team: devsecops
17      - podSelector:
18        matchLabels:
19          deployment: web-mysql
20  ports:
21    - protocol: TCP
22      port: 8080
```

```
$ oc create -f allow-network.yml --dry-run=server --validate=true
```

```
$ oc create -f allow-network.yml
```

Best of Luck