OCP 구축 후 설정

1. 계정 생성

1) htpasswd 생성 후 계정 입력

```
htpasswd -bBc ./htpasswd admin admin
```

2) secret 생성

```
oc create secret generic htpass-secret --from-file=htpasswd=./htpasswd -n
openshift-config
```

3) Oauth cluster 설정에 Identity provider 추가

```
oc edit oauth cluster
apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
annotations:
release.openshift.io/create-only: "true"
creationTimestamp: "2020-01-14714:51:33Z"
generation: 2
name: cluster
resourceVersion: "230840"
selfLink: /apis/config.openshift.io/v1/oauths/cluster
uid: 5687 1cc0-36dd-1 1ea-abc7--005056b332db
spec: <----- 아래 내용 추가 ---->
  identityProviders:
  - htpasswd:
      fileData:
        name: htpass-secret
    mappingMethod: claim
    name: my_htpasswd_provider
    type: HTPasswd
```

4) 계정에 cluster-admin 권한 부여

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

5) 로그인 확인

```
oc login -u admin -p admin
```

6) 사용자 추가

```
새로운 유저가 추가 된 htpasswd 파일 생성

htpasswd -bB /paas/opt/registry/auth/ocp.htpasswd test test

OCP 콘솔에서 openshift-config 프로젝트의 secret 에 유저 추가
Projects -> openshift-config -> Workloads -> Secrets -> htpass-secret -> Actions
-> Edit Secret -> Value -> htpassws 로 확인한 id, passwd 추가 -> Save
```

2. 그룹 생성

1) yaml 파일을 이용하여 Role 생성 / 사전에 role 을 명시한 ap-admin.yaml 파일 필요

```
oc apply -f /root/paas_work/rbac/ap-admin.yaml
```

2) 그룹 생성

```
oc adm groups new ap-admins
```

3) 그룹에 권한 부여

```
oc adm policy add-cluster-role-to-group ap-admin ap-admins
```

4) 그룹에 유저 추가

```
oc adm groups add-users ap-admins test
```

5) 그룹에서 유저 제거

```
oc adm groups remove-users ap-admins test
```

3. Image Registry 설정 변경

1) pvc 를 empty 로 구성하는 명령어

```
oc patch configs.imageregistry.operator.openshift.io cluster --type merge --
patch '{"spec":{"storage":{"emptyDir":{}}}}'

OCP 4.3 구성 직후 imageRegistry managermentState 변경
oc patch configs.imageregistry.operator.openshift.io cluster --type=merge --
patch '{"spec":{"managementState": "Managed" }}'
```

2) pvc 설정 명령어

```
oc patch configs.imageregistry.operator.openshift.io cluster --type=merge --
patch '{"spec":{"storage":{"pve":{"claim":"nfsregistry-claim"}}}}'
```

3) Replica 변경 명령어

```
oc patch configs.imageregistry.operator.openshift.io cluster --type=merge --
patch '{"spec":{"replicas":1}}'
```

4) nodeSelector 적용

```
oc patch configs.imageregistry.operator.openshift.io cluster --type=merge --
patch '{"spec":{"nodeSelector":{"node-role.kubernetes.io/infra":""}}}'
```

5) Imageregistry 에서 사용할 PV 생성

```
registry-pv.yaml

kind: PersistentVolume
apiVersion: v1
metadata:
  name: registry
spec:
  capacity:
    storage: 300Gi
  nfs:
    server: {NAS_SERVER_DOMAIN}
    path: /registry
  accessModes:
  - ReadWriteMany
  persistentVolumeReclaimPolicy: Retain
  volumeMode: Filesystem

oc create -f registry-pv.yaml
```

6) Imageregistry 에서 사용할 pvc 생성

```
regislty-pvc.yaml

kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: registry-claim
  namespace: openshift-image-registry
spec:
  accessModes :
  - ReadWriteMany
  resources:
    requests:
      storage: 300Gi
  volumeName: registry
  volumeMode: Filesystem

oc create -f registry-pvc.yam|
```

7) 설정 반영

```
oc patch configs.imageregistry.operator.openshift.io cluster --type=json -p
'[{"op":"replace", "path": "/spec/storage", "value": {"pvc": {"claim":
"nfsregistry-claim"}}}]'
```

8) Default route 설정

```
oc patch configs.imageregistry.operator.openshift.io/cluster --patch '{"spec":
{"defaultRoute":true}}' --type=merge
```

4. Ingrss nodeSelector 설정 변경

1) nodeSelector Lables 추가

```
oc patch ingresscontrollers.operator.openshift.io default -n openshift-ingress-
operator --type=merge --patch='{"spec":{"nodePlacement": {"nodeSelector":
{"matchLabels":{"node-role.kubernetes.io/router": ""}}}}}'

or

oc edit -n openshift-ingress-operator ingresscontrollers.operator.openshift.io
default

apiVersion: operator .openshift.io/v1
kind: IngressControl ler
metadata:
  creationTimestamp: '2020-01-14T14:56:232'
  finalizers:
    - ingresscontroller.operator .openshift.io/finalizer-ingresscontroller
  generation: 5
  name: default
  namespace: openshi ft-ingress-oper ator
  resourceVersion: '2514649'
  selfLink: >-
    /apis/operator.openshift.io/v1/namespaces/openshift-
ingressoperator/ingresscontrol lers/default
  uid:083a1356-36de-11ea-86f b-005056b362f7
spec:
  nodePlacement :
    nodeSelector :
      matchLabels:
        node-role.kubernetes.io/router: ''
  replicas: 3
```

5. Prometheus Monitoring 설정

openshift-monitoring 에 구성된 Prometheus, AlertManager 를 관리하기 위해 Conf igMap 생성
1) configmap 생성

```
monitoring-cm.yaml

apiVersion: v1
kind: ConfigMap
metadata:
  name: cluster-monitoring-config
  namespace: openshift-monitoring
data:
```

```
  config.yaml: |
    alertmanagerMain:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      volumeClaimTemplate:
        spec:
          resources:
            requests:
              storage: 50Gi
    prometheusK8s:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
      volumeClaimTemplate:
        spec:
          resources:
            requests:
              storage: 200Gi
    prometheusOperator:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
    grafana:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
    k8sPrometheusAdapter:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
    kubeStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
    telemeterClient:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
    openshiftStateMetrics:
      nodeSelector:
        node-role.kubernetes.io/infra: ""
```

2) pv 생성

```
apiVersion: v1
kind: PersistentVolume
metadata:
  name: pro-pv00
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 10Gi
  nfs:
    server: {NFS_SERVER}
    path: {PATH}
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
```

```
kind: PersistentVolume
metadata:
  name: pro-pv01
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 10Gi
  nfs:
    server: {NFS_SERVER}
    path: {PATH}
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolume
metadata:
  name: aler-pv00
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 10Gi
  nfs:
    server: {NFS_SERVER}
    path: {PATH}
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolume
metadata:
  name: aler-pv01
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 10Gi
  nfs:
    server: {NFS_SERVER}
    path: {PATH}
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolume
metadata:
  name: aler-pv02
spec:
  accessModes:
  - ReadWriteOnce
  capacity:
    storage: 10Gi
  nfs:
    server: {NFS_SERVER}
```

```
    path: {PATH}
  persistentVolumeReclaimPolicy: Recycle
  storageClassName: gp2
  volumeMode: Filesystem
```

3) pvc 생성

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
  name: prometheus-k8s-db-prometheus-k8s-0
  namespace: openshift-monitoring
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
  name: prometheus-k8s-db-prometheus-k8s-1
  namespace: openshift-monitoring
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 10Gi
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
  name: alertmanager-main-db-aletmanager-main-0
  namespace: openshift-monitoring
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
```

```
  annotations:
  name: alertmanager-main-db-aletmanager-main-1
  namespace: openshift-monitoring
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: gp2
  volumeMode: Filesystem

apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  annotations:
  name: alertmanager-main-db-aletmanager-main-2
  namespace: openshift-monitoring
spec:
  accessModes:
  - ReadWriteOnce
  resources:
    requests:
      storage: 2Gi
  storageClassName: gp2
  volumeMode: Filesystem
```

6. NetworkPolicy 설정 - 기존 프로젝트 추가 적용

다른 프로젝트의 Pod 간 통신을 제한하기 위한 설정 (Multitenant)

1) allow-from-openshift-ingress.yaml 생성

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-openshift-ingress
spec:
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          network.openshift.io/policy-group: ingress
  podSelector: {}
  policyTypes:
  - Ingress
```

2) allow-from-openshift-monitoring.yaml 생성

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-openshift-monitoring
spec:
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          network.openshift.io/policy-group: monitoring
  podSelector: {}
  policyTypes:
  - Ingress
```

3) deny-other-namespaces.yaml 생성

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: deny-other-namespaces
  namespace: openshift-monitoring
spec:
  podSelector: null
  ingress:
    - from:
        - podSelector: {}
```

4) 정책 적용 대상 프로젝트에 3개 정책 적용

```
oc apply -f allow-from-openshift-ingress.yaml -n <project>
oc apply -f allow-from-openshift-monitoring.yaml -n <project>
oc apply -f deny-other-namespaces.yaml -n <project>
```

7. NetworkPolicy 설정 - 신규 프로젝트 자동 적용

1) default project template 생성

```
oc adm create-bootstrap-project-template -o yaml > template.yaml
```

2) template.yaml 수정 (3개 정책 추가)

```
apiVersion: template.openshift.io/v1
kind: Template
metadata:
  creationTimestamp: null
  name: project-request
objects:
- apiVersion: project.openshift.io/v1
  kind: Project
  metadata:
    annotations:
      openshift.io/description: ${PROJECT_DESCRIPTION}
      openshift.io/display-name: ${PROJECT_DISPLAYNAME}
      openshift.io/requester: ${PROJECT_REQUESTING_USER}
```

```
      openshift.io/node-selector: node-role.kubernetes.io/worker=
    creationTimestamp: null
    name: ${PROJECT_NAME}
  spec: {}
  status: {}
- apiVersion: rbac.authorization.k8s.io/v1
  kind: RoleBinding
  metadata:
    creationTimestamp: null
    name: admin
    namespace: ${PROJECT_NAME}
  roleRef:
    apiGroup: rbac.authorization.k8s.io
    kind: ClusterRole
    name: admin
  subjects:
  - apiGroup: rbac.authorization.k8s.io
    kind: User
    name: ${PROJECT_ADMIN_USER}
  - apiVersion: networking.k8s.io/v1
    kind: NetworkPolicy
    metadata:
      name: allow-from-openshift-ingress
    spec:
      ingress:
      - from:
        - namespaceSelector:
            matchLabels:
              network.openshift.io/policy-group: ingress
      podSelector: {}
      policyTypes:
      - Ingress
  - apiVersion: networking.k8s.io/v1
    kind: NetworkPolicy
    metadata:
      name: allow-from-openshift-monitoring
    spec:
      ingress:
      - from:
        - namespaceSelector:
            matchLabels:
              network.openshift.io/policy-group: monitoring
      podSelector: {}
      policyTypes:
      - Ingress
  - apiVersion: networking.k8s.io/v1
    kind: NetworkPolicy
    metadata:
      name: deny-other-namespaces
    spec:
      podSelector: null
      ingress:
        - from:
          - podSelector: {}
parameters:
- name: PROJECT_NAME
- name: PROJECT_DISPLAYNAME
- name: PROJECT_DESCRIPTION
```

```
- name: PROJECT_ADMIN_USER
- name: PROJECT_REQUESTING_USER
```

3) template 생성, 확인, 수정

```
oc create -f template.yaml -n openshift-config
oc get template -n openshift-config
oc edit template <project_template> -n openshift-config
```

4) web-console 에서 project configuration resource 수정 (projectRequestTemplate 추가)

```
Administration –> Cluster Settings –> Global Configuration –> Project –> Edit
YAML

spec:
  projectRequest Template:
    name: project-request

or

oc edit project .config.openshi ft.io/cluster

spec:
  projectRequest Template:
    name: project-request
```

8. OCP 관리콘솔 세션 타임아웃 설정

1) oauth 설정

```
oc edit oauths.config.openshift.io cluster

apiVersion: config.openshift.io/v1
kind: OAuth
metadata:
  annotations:
    release.openshift.io/create-only: "true"
  creationTimestamp: "2020-03-04T01:34:40Z"
  generation: 3
  name: cluster
  resourceVersion: "2338073"
  selfLink: /apis/config.openshift.io/v1/oauths/cluster
  uid: adde1cdc-29dc-4c70-8275-304bc39a3461
spec:
  identityProviders:
  - htpasswd:
      fileData:
        name: htpass-secret
    mappingMethod: claim
    name: my_htpasswd_provider
    type: HTPasswd
  tokenConfig:
    accessTokenMaxAgeSeconds: 600
```

9. Project self-provisioning 해제

권한이 없는 유저의 프로젝트 생성 제한

1) cluster-admin 권한이 있는 유저로 oc login

```
oc login -u <ID>
```

2) clusterrolebinding 에 설정되어 있는 self-provisioners 상태 확인

```
oc describe clusterrolebinding.rbac self-provisioners

Name:         self-provisioners
Labels:       <none>
Annotations:  rbac.authorization.kubernetes.io/autoupdate: true
Role:
  Kind:  ClusterRole
  Name:  self-provisioner
Subjects:
  Kind   Name                         Namespace
  ----   ----                         ---------
  Group  system:authenticated:oauth
```

3) "system:authenticated:oauth" 그룹에서 self-provisioner 제거

```
oc patch clusterrolebinding.rbac self-provisioners -p '{"subjects": null}'
```

4) cluster roles 의 automatic updates 해제

```
oc patch clusterrolebinding.rbac self-provisioners -p '{"metadata": {
"annotations":{"rbac.authorization.kubernetes.io/autoupdate": "false" }}}'
```

5) 특정 계정에 프로젝트 생성 권한 추가

```
oc adm groups add-users self-provisioner <USERNAME>
```

10. Node Roles 수정

```
oc edit <NODE_NAME>

node-role.kubernetes.io/worker: ""
node-role.kubernetes.io/infra: ""
node-role.kubernetes.io/ logging: ""
node-role.kubernetes.io/router: ""
node-role.kubernetes.io/egress: ""

or

- 설정
oc label node <NODE_NAME> node-role.kubernetes.io/worker=""
```

```
- 제거
oc label node <NODE_NAME> node-role.kubernetes.io/worker-
```

11. IngressController 설정

1) IngressController 생성

```
apiVersion: v1
items:
- apiVersion: operator.openshift.io/v1
  kind: IngressController
  metadata:
    name: sharded
    namespace: openshift-ingress-operator
  spec:
    domain: <apps-sharded.basedomain.example.net>
    nodePlacement:
      nodeSelector:
        matchLabels:
          node-role.kubernetes.io/router: ""
    routeSelector:
      matchLabels:
        type: sharded
  status: {}
kind: List
metadata:
  resourceVersion: ""
  selfLink: ""
```

2) 프로젝트 별 Lables 설정

```
oc label namespace <PROJECT_NAME> type=sharded
```

3) IngressController 를 router 노드로 설정

```
일반
oc patch ingresscontrollers.operator.openshift.io default -n openshift-ingress-
operator --type=merge --patch='{"spec":{"nodePlacement": {"nodeSelector":
{"matchLabels":{"node-role.kubernetes.io/router": ""}}}}}'
```

12. egressIP 설정

```
oc patch netnamespace <PROJECT_NAME> --type=merge -p '{"egressIPs":
["10.0.0.0"]}'

egressIPs 는 다른 노드에서 사용하고 있지 않으며, Infra 환경에서 실제 사용할 수 있는 IP로
설정
```

13. Hostname 설정

1) hostnamectl 을 이용한 hostname 설정

```
ssh <NODE_NAME>
sudo -i
hostnamectl set-hostname <NODE_NAME>
```

14. NTP 설정

1) NTP 설정

```
ssh <NODE_NAME>
sudo -i
vi /etc/chrony.conf
server {TIME_SERVER_IP} iburst
```

15. nmcli 를 이용한 StaticIP 설정 / DHCP 종료 후 서버 reboot 하여 확인 필요

1) ens192

```
ssh <NODE_NAME>
sudo -i
nmcli con show ens192
nmcli con mod ens192 ipv4.addresses {SERVER_IP}/24
nmcli con mod ens192 ipv4.gateway {GATEWAY}
nmcli con mod ensi92 ipv4.dns {DNS_IP}
nmcli con mod ens192 ipv4.dns-search {DNS}
nmcli con mod ensi92 ipv4.method manual
nmcli con mod ensi92 ipv6.method ignore
nmcli con reload
```

2) ens224

```
ssh <NODE_NAME>
sudo -i
nmcli con show ens224
nmcli con mod ens224 ipv4.addresses {SERVER_IP}/24
nmcli con mod ens224 ipv4.never-default yes
nmcli con mod ens224 ipv4.method manual
nmcli con mod ens224 ipv6.method ignore
nmcli con reload
```

16. MachineConfigPool 추가 / node의 각 role 별로 수행 (infra, logging 등)

1) worker machineconfigpool export

```
oc get mcp wokrker > infra-mcp.yaml
```

2) infra-mcp.yaml 수정

```
apiVersion: machineconfiguration.openshift.io/v1
kind: MachineConfigPool
metadata:
  name: infra
spec:
  configuration:
    name: rendered-worker-af92bc208ec7f1dbfbd5bd11f23af9e6
  machineConfigSelector:
    matchLabels:
      machineconfiguration.openshift.io/role: worker
  nodeSelector:
    matchLabels:
      node-role.kubernetes.io/infra: ""
  paused: false
```

3) machineconfigpool 생성 및 확인

```
oc create -f infra-mcp.yam|

oc get mcp
```

* 참고

1) ingresscontroller scale 변경

```
oc patch ingresscontrollers.operator.openshift.io default -n openshift-ingress-
operator --type=merge --patch='{"spec":{"replicas": 3}}'
```

2) etcdctl 상태 확인 / master node에 접속 후 확인

```
### etcd 컨테이너에 접속
id=$(sudo crictl ps --name etcd-member | awk 'FNR==2{ print $1}') && sudo crictl
exec -it $id /bin/sh

### member 조회를 위한 변수 선언
export ETCDCTL_API=3 ETCDCTL_CACERT=/etc/ssl/etcd/ca.crt ETCDCTL_CERT=$(find
/etc/ssl/ -name *peer*crt) ETCDCTL_KEY=$(find /etc/ssl/ -name *peer*key)

### member 조회
etcdctl member list -w table
etcdctl endpoint status --cluster -w table
```

3) mirror-registry 로 image push / pull

3.1) podman 로그인

```
podman login –u paasadm -p paasadm demo.ocp4.com:5000
```

3.2) push

```
podman push registry.demo.ocp4.com:5000/jboss–webserver-3/webserver31-tomcat7-
openshift:latest
```

3.3) pull

```
podman pull registry.demo.ocp4.com:5000/jboss–webserver-3/webserver31-tomcat7-
openshift:latest
```

4) image-registry 로 image push / pull

4.1) podman 로그인

```
podman login -u kbadmin -p $(oc whoami -t) default-route-openshift-
imageregistry.apps.demo.ocp4.com --tls-verify=false
```

4.2) podman push

```
podman push defaul t-route-openshi f t-image-registry.apps.demo.ocp4.com/jboss-
webserver-3/webserver31-tomcat7-openshift:latest –-tls-verify=false
```

4.3) podman pull

```
podman pull defaul t-route-openshi f t-image-registry.apps.demo.ocp4.com/jboss-
webserver-3/webserver31-tomcat7-openshift:latest –-tls-verify=false
```

5) Disable the kubeadmin user

```
kubeadmin secret 확인
oc describe secret kubeadmin -n kube-system

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

6) inseucre 설정

```
oc edit image.config.openshift.io/cluster

apiVersion: config.openshift.io/v1
kind: Image
metadata:
  annotations:
    release.openshift.io/create-only: "true"
  creationTimestamp: "2020-03-26T04:58:57Z"
  generation: 1
```

```
  name: cluster
  resourceVersion: "17413"
  selfLink: /apis/config.openshift.io/v1/images/cluster
  uid: b3f274db-38ac-4b64-84ec-9e5b718382b2
spec:
  registrySources:
    insecureRegistries:
    - utilityvm.example.com:5000
    - default-route-openshift-image-registry.apps.baba.blue.osp.opentlc.com
```