Kubernetes 클러스터 설치하기

3대 서버에 마스터 1개, 노드1, 노도2의 쿠버네티스 클러스터를 구성하기 위한 설치 방법을 설명합니다.
각 서버의 호스트 이름 및 IP 는 아래와 같이 가정합니다. 사용자의 클러스터 환경에 맞게 IP를 수정하시기 바랍니다.
csle1 : 192.168.0.5 (master)
csle2 : 192.168.0.6 (node1)
csle3 : 192.168.0.7 (node2)

모든 서버에 공통으로 설치하기

[모든 서버 공통] java 8 설치하기

KSB 툴박스 설치의 java-8-oracle 설치 부분을 참고합니다.

[모든 서버 공통] 기타 우분투 프로그램 설치하기

```
sudo apt-get update && \
sudo apt-get install -y python-software-properties && \
sudo add-apt-repository -y ppa:webupd8team/java && \
sudo apt-get update && \
echo "oracle-java8-installer \
shared/accepted-oracle-license-v1-1 select true" \
| sudo debconf-set-selections && \
sudo apt-get install -y oracle-java8-installer
```

[모든 서버 공통] SSHD port 변경 및 root 로그인 가능하게 SSHD config 수정하기

아래의 명령을 수행하여 포트 정보를 2243로 수정하고, root 로그인을 허용합니다.

```
sudo sed -ri 's/^Port 22/Port 2243/g' /etc/ssh/sshd_config
sudo sed -ri \
's/^PermitRootLogin prohibit-password/PermitRootLogin yes/g' \
/etc/ssh/sshd_config
```

[모든 서버 공통] SSH config 수정하기

SSH 접속시 known_hosts에 호스트 정보 저장 질문을 하지 않도록 설정합니다.
아래의 명령을 수행하여 ssh_config의 기존 내용을 모두 삭제하고 설정을 추가합니다.

```
sudo truncate -s0 /etc/ssh/ssh_config
echo "Host localhost" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "StrictHostKeyChecking no" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "Host 0.0.0.0" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "StrictHostKeyChecking no" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "Host 127.0.0.1" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "StrictHostKeyChecking no" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "Host csle*" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "StrictHostKeyChecking no" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "UserKnownHostsFile=/dev/null" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "Host master" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "StrictHostKeyChecking no" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
echo "UserKnownHostsFile=/dev/null" \
| sudo tee -a /etc/ssh/ssh_config >/dev/null
sudo service ssh restart
```

[모든 서버 공통] SSH 서비스 재시작하기

```
sudo service ssh restart
```

[모든 서버 공통] docker 설치하기

아래의 명령으로 docker-ce을 설치합니다. 현재 최신 버전은 18.06.1~ce~3-0~ubuntu 입니다.

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | \
sudo apt-key add - && \
sudo add-apt-repository \
"deb [arch=amd64]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" && \
sudo apt-get update && \
apt-cache policy docker-ce && \
sudo apt-get install -y docker-ce && \
sudo systemctl status docker
```

[참고사항] 기존 docker-engine 삭제

```
sudo apt-get purge -y docker docker-engine docker.io
```

[모든 서버 공통] docker sudo 명령없이 사용하기 위한 설정하기

Docker 명령시 root 권한이 필요합니다. root가 아닌 사용자가 sudo 없이 사용하려면 해당 사용자를 docker 그룹에 추가합니다.

```
sudo usermod -aG docker csle
# 사용자가 로그인 중이라면 다시 로그인 후 권한이 적용됩니다.
# 단, 현재 터미널에서만 적용됩니다.
sudo su csle
PC 재부팅
```

PC 재부팅을 통해서 sudo 명령없이 docker 명령을 사용할 수 있습니다.

[모든 서버 공통] nvidia-docker 2.0 설치

```
curl -s -L https://nvidia.github.io/nvidia-docker/gpgkey | \
sudo apt-key add - && \
curl -s -L \
https://nvidia.github.io/nvidia-docker/\
ubuntu16.04/amd64/nvidia-docker.list | \
sudo tee /etc/apt/sources.list.d/nvidia-docker.list && \
sudo apt-get update && \
sudo apt-get install -y nvidia-docker2 && \
sudo pkill -SIGHUP dockerd
```

아래의 질의에 대해서 Y를 누르면 설치가 완료됩니다.

```
Configuration file '/etc/docker/daemon.json'
 ==> File on system created by you or by a script.
 ==> File also in package provided by package maintainer.
   어떻게 하시겠습니까? 다음 중에 하나를 선택할 수 있습니다:
    Y 또는 I : 패키지 관리자의 버전을 설치합니다
    N 또는 O : 현재 설치된 버전을 유지합니다
       D     : 버전 간의 차이점을 표시합니다
       Z     : 프로세스를 백그라운드로 하고 상황을 알아봅니다
기본값으로 현재 버전을 그대로 유지합니다.
*** daemon.json (Y/I/N/O/D/Z) [기본값=N] ? Y
```

nvidia-docker 버전을 확인합니다.

```
csle@csle1:~$ nvidia-docker version
NVIDIA Docker: 2.0.3
Client:
 Version:           18.06.1-ce
 API version:       1.38
 Go version:        go1.10.3
 Git commit:        e68fc7a
 Built:             Tue Aug 21 17:24:56 2018
 OS/Arch:           linux/amd64
 Experimental:      false
```

```
Server:
 Engine:
  Version:          18.06.1-ce
  API version:      1.38 (minimum version 1.12)
  Go version:       go1.10.3
  Git commit:       e68fc7a
  Built:            Tue Aug 21 17:23:21 2018
  OS/Arch:          linux/amd64
  Experimental:     false
```

[참고사항] 기존 1.0 버전을 삭제하기 위해서는 아래의 명령을 수행합니다.

```
docker volume ls -q -f driver=nvidia-docker | \
xargs -r -I{} -n1 docker ps -q -a -f volume={} | xargs -r docker rm -f
sudo apt-get purge nvidia-docker
```

[모든 서버 공통] Kubernetes 설치하기

```
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | \
sudo apt-key add - && \
echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" | \
sudo tee /etc/apt/sources.list.d/kubernetes.list && \
sudo apt-get update -q && \
sudo apt-get install -qy kubelet=1.9.6-00 kubectl=1.9.6-00 kubeadm=1.9.6-00
kubectl version
```

[참고사항] 기존 Kubernetes 버전 삭제 방법

```
sudo su -
kubeadm reset
rm /usr/local/bin/kubectl
rm -r /etc/kubernetes
rm -r /var/lib/etcd
rm -rf /var/lib/cni/
ifconfig cni0 down && ip link delete cni0
ifconfig flannel.1 down && ip link delete flannel.1
apt-get autoremove --purge -y kubelet kubeadm kubectl kubernetes-cni
```

[모든 서버 공통] /etc/fstab 수정 후 재부팅하기

쿠버네티스가 Swap 기능을 관리해야 하므로, 아래의 swap이 적용된 file 시스템 부분을 주석 처

리한다.

```
sudo vi /etc/fstab

UUID=c3ec283e-682f-4e8c-b822-43eef87a725f /  ext4    errors=remount-ro 0     1
# /boot/efi was on /dev/nvme0n1p1 during installation
UUID=7D67-73C7  /boot/efi       vfat    umask=0077      0       1
# swap was on /dev/nvme0n1p3 during installation

# (주의!!!) swap 설정 해제
#UUID=257a989b-d1fa-4a7c-8ae7-9ccad9b9c0c6 none    swap  sw      0      0
```

[모든 서버 공통] Kubernetes master /etc/hosts 변경하기

```
sudo vi /etc/hosts

127.0.0.1       localhost
# 아래 주석처리. Hdfs 연동시 문제가 생김.
#127.0.1.1      csle1

# The following lines are desirable for IPv6 capable hosts
::1     ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters

# 쿠버네티스 master, node1, node2 정보 등록
192.168.0.5 csle1 master
192.168.0.6 csle2
192.168.0.7 csle3
```

[모든 서버 공통] sysctl kernel parameter값 수정하기

커널 차원에서 보안 및 최적화 관련 설정을 변경합니다 .

먼저 root계정으로 변경합니다.

```
csle@csle1:~# sudo su -
root@csle1:~#
```

아래와 같이 설정 후 반영합니다.

```
root@csle1:~# vi /etc/sysctl.conf
# 아래의 내용을 마지막에 추가합니다.
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-ip6tables=1
net.ipv4.ip_forward=1
net.netfilter.nf_conntrack_max = 786432

root@csle1:~# sysctl -p
```

[모든 서버 공통] kubectl 명령 옵션의 자동 완성 기능 설정하기

```
source /etc/profile.d/bash_completion.sh
source <(kubectl completion bash)
echo "source /etc/profile.d/bash_completion.sh" | tee -a ~/.bashrc
echo "source <(kubectl completion bash)" | tee -a ~/.bashrc
```

[모든 서버 공통] Security Enhanced Linux 모드 disable 하기

갑작스러운 리부팅 이후에 apiserver, etcd 등이 정상 기동되지 않는 경우가 자주 발생을 막기 위해 permissive로 설정합니다.

```
apt install -y selinux-utils
setenforce 0
```

[모든 서버 공통] 10-kubeadm.conf 수정하기

ExecStart= 위의 라인에 Environment 설정값을 추가합니다.

```
root@csle1:~$ vi /etc/systemd/system/kubelet.service.d/10-kubeadm.conf

[Service]
Environment="KUBELET_KUBECONFIG_ARGS=--bootstrap-kubeconfig=\
/etc/kubernetes/bootstrap-kubelet.conf --kubeconfig=\
/etc/kubernetes/kubelet.conf"
Environment="KUBELET_SYSTEM_PODS_ARGS=--pod-manifest-path=\
/etc/kubernetes/manifests --allow-privileged=true"
Environment="KUBELET_NETWORK_ARGS=--network-plugin=\
cni --cni-conf-dir=/etc/cni/net.d --cni-bin-dir=/opt/cni/bin"
Environment="KUBELET_DNS_ARGS=--cluster-dns=10.96.0.10 \
--cluster-domain=cluster.local"
Environment="KUBELET_AUTHZ_ARGS=--authorization-mode=Webhook \
--client-ca-file=/etc/kubernetes/pki/ca.crt"
Environment="KUBELET_CADVISOR_ARGS=--cadvisor-port=0"
Environment="KUBELET_CERTIFICATE_ARGS=--rotate-certificates=true \
--cert-dir=/var/lib/kubelet/pki"

# 아래의 내용을 추가합니다.
Environment="KUBELET_EXTRA_ARGS=--fail-swap-on=false \
--runtime-cgroups=/systemd/system.slice \
--kubelet-cgroups=/systemd/system.slice"

ExecStart=
ExecStart=/usr/bin/kubelet \
$KUBELET_KUBECONFIG_ARGS $KUBELET_SYSTEM_PODS_ARGS \
$KUBELET_NETWORK_ARGS $KUBELET_DNS_ARGS $KUBELET_AUTHZ_ARGS \
$KUBELET_CADVISOR_ARGS $KUBELET_CERTIFICATE_ARGS $KUBELET_EXTRA_ARGS
```

아래의 명령을 통해 kubelet을 재시작 합니다.

```
root@csle1:~$ swapoff -a
root@csle1:~$ sudo systemctl daemon-reload
root@csle1:~$ sudo systemctl restart kubelet
root@csle1:~$ sudo systemctl status kubelet

● kubelet.service - kubelet: The Kubernetes Node Agent
   Loaded: loaded (/lib/systemd/system/kubelet.service; enabled;
   vendor preset: enabled)
  Drop-In: /etc/systemd/system/kubelet.service.d
           └─10-kubeadm.conf
   Active: activating (auto-restart) (Result: exit-code) since
   수 2018-07-25 09:55:44 KST; 3s ago
     Docs: http://kubernetes.io/docs/
  Process: 10517 ExecStart=/usr/bin/kubelet $KUBELET_KUBECONFIG_ARGS
  $KUBELET_SYSTEM_PODS_ARGS $KUBELET_NETWORK_ARGS $KUBELET_DNS_ARGS
  $KUBELET_AUTHZ_ARGS $KUBELET_CADVISOR
 Main PID: 10517 (code=exited, status=1/FAILURE)
 7월 25 09:55:44 csle4 systemd[1]: kubelet.service: Unit entered failed state.
 7월 25 09:55:44 csle4 systemd[1]: kubelet.service: Failed with result
 'exit-code'.
```

systemctl status kubelet 실행시 status=1/FAILURE로 나타나지만 무시해도 됩니다.
kubeadm이 정상 실행 되면 정상화됩니다.

모든 노드에 대해 동일하게 설정하는 부분을 완료하였습니다.

master와 node들 각각에 대한 설정 방법을 설명합니다.

Kubernetes master (csle1) 설정하기

[Master: csle1] kubeadm 초기화하기

자신의 ip에 맞도록 수정하고 kubeadm으로 초기화합니다. (예: master ip : 192.168.0.5)

```
kubeadm init --apiserver-advertise-address=192.168.0.5 \
--pod-network-cidr=10.244.0.0/16 --service-cidr 10.96.0.0/12 --token-ttl 0
```

아래와 같은 로그가 나오면서 초기화 되면 정상입니다.

```
[init] Using Kubernetes version: v1.9.9
[init] Using Authorization modes: [Node RBAC]
[preflight] Running pre-flight checks.
        [WARNING FileExisting-crictl]: crictl not found in system path
[certificates] Generated ca certificate and key.
[certificates] Generated apiserver certificate and key.
[certificates] apiserver serving cert is signed for DNS names [csle1
kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.
svc.cluster.local] and IPs [10.96.0.1 129.254.xxx.xxx]
[certificates] Generated apiserver-kubelet-client certificate and key.
[certificates] Generated sa key and public key.
[certificates] Generated front-proxy-ca certificate and key.
[certificates] Generated front-proxy-client certificate and key.
[certificates] Valid certificates and keys now exist in "/etc/kubernetes/pki"
[kubeconfig] Wrote KubeConfig file to disk: "admin.conf"
[kubeconfig] Wrote KubeConfig file to disk: "kubelet.conf"
[kubeconfig] Wrote KubeConfig file to disk: "controller-manager.conf"
[kubeconfig] Wrote KubeConfig file to disk: "scheduler.conf"
[controlplane] Wrote Static Pod manifest for component kube-apiserver to
"/etc/kubernetes/manifests/kube-apiserver.yaml"
[controlplane] Wrote Static Pod manifest for component kube-controller-manager
to "/etc/kubernetes/manifests/kube-controller-manager.yaml"
[controlplane] Wrote Static Pod manifest for component kube-scheduler to
"/etc/kubernetes/manifests/kube-scheduler.yaml"
[etcd] Wrote Static Pod manifest for a local etcd instance to
"/etc/kubernetes/manifests/etcd.yaml"
[init] Waiting for the kubelet to boot up the control plane as Static Pods
from directory "/etc/kubernetes/manifests".
[init] This might take a minute or longer if the control plane images have
to be pulled.
[apiclient] All control plane components are healthy after 46.007752 seconds
[uploadconfig] Storing the configuration used in ConfigMap "kubeadm-config"
in the "kube-system" Namespace
[markmaster] Will mark node csle1 as master by adding a label and a taint
[markmaster] Master csle1 tainted and labelled with key/value:
node-role.kubernetes.io/master=""
[bootstraptoken] Using token: a96953.b5d1418a5099b5de
[bootstraptoken] Configured RBAC rules to allow Node Bootstrap tokens to
post CSRs in order for nodes to get long term certificate credentials
[bootstraptoken] Configured RBAC rules to allow the csrapprover controller
automatically approve CSRs from a Node Bootstrap Token
[bootstraptoken] Configured RBAC rules to allow certificate rotation for
all node client certificates in the cluster
[bootstraptoken] Creating the "cluster-info" ConfigMap in the "kube-public"
namespace
[addons] Applied essential addon: kube-dns
```

```
[addons] Applied essential addon: kube-proxy

Your Kubernetes master has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

  mkdir -p $HOME/.kube
  sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
  sudo chown $(id -u):$(id -g) $HOME/.kube/config

You should now deploy a pod network to the cluster.
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
  https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of machines by running the following on each node
as root:

  kubeadm join --token a96953.b5d1418a5099b5de 129.254.xxx.xxx:6443
  --discovery-token-ca-cert-hash sha256:648c1dd27c51edc0e90ff356f47599e1
  a8bee800f77ada9f23fdacd8d69e62a0
```

다른 노드에서 master에 접속하기 위해서 아래의 부분을 저장해 둡니다.

```
kubeadm join --token a96953.b5d1418a5099b5de 192.168.0.5:6443 \
  --discovery-token-ca-cert-hash \
  sha256:648c1dd27c51edc0e90ff356f47599e1a8bee800f77ada9f23fdacd8d69e62a0
```

[Master: csle1] K8s master 노드 kubectl 실행환경 설정하기

먼저 사용자 계정을 csle로 변경합니다.

```
sudo su csle
csle@csle1:~$
```

```
mkdir -p $HOME/.kube
yes | sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
export KUBECONFIG=$HOME/.kube/config
echo "export KUBECONFIG=$HOME/.kube/config" | tee -a ~/.bashrc
kubectl get componentstatus
```

Kubernetes node1, node2 설정하기 (csle2, csle3)

먼저 root 계정으로 변경합니다.

```
sudo su -
root@csle2:~$
```

[csle2, csle3] kubernetes master에 join 하기

kublet을 재부팅합니다.

```
swapoff -a
systemctl daemon-reload
systemctl restart kubelet
systemctl status kubelet
```

kubernetes master에서 init시 생성된 script와 추가 옵션을 이용해 master에 join 합니다.

```
kubeadm join --token c8908d.5dc02f87aca87415 192.168.0.5:6443 \
--discovery-token-ca-cert-hash \
sha256:756ac632e63878260dc6b3a3497b8ba69d29087ba0428beaea061e4f5241b29d \
--ignore-preflight-errors=all --discovery-token-unsafe-skip-ca-verification
```

kubernetes master에서 확인하기 (csle1)

[Master: csle1] node 및 pod 동작 확인하기

node1, node2가 join이 성공했는지 아래의 명령을 이용해서 확인합니다.

```
root@csle1:~$ sudo su csle
csle@csle1:~$ kubectl get nodes
NAME      STATUS     ROLES     AGE       VERSION
csle1     NotReady   master    5m        v1.9.6
csle2     NotReady   <none>    1m        v1.9.6
csle3     NotReady   <none>    1m        v1.9.6
```

[Master: csle1] kubernetes 클러스터 환경 구성을 위한 yaml

환경 설정하기

kubernetes master에서 클러스터 환경 구성을 위한 yaml 파일들을 이용해서 서비스를 생성합니다.

```
cd /home/csle/ksb-csle/kubernetes/env_set
kubectl create -f kube-flannel.yml
kubectl create -f kubernetes-dashboard.yaml
kubectl create -f kubernetes-dashboard-admin-rbac.yaml
kubectl create -f ServiceAccount.yml
kubectl create -f ClusterRoleBinding.yml
kubectl create -f k8s-heapster/
```

```
csle@csle1:~$ kubectl get nodes
NAME      STATUS    ROLES     AGE       VERSION
csle1     Ready     master    5m        v1.9.6
csle2     Ready     <none>    1m        v1.9.6
csle3     Ready     <none>    1m        v1.9.6
```

```
csle@csle1:~$ kubectl get pods --all-namespaces
kube-system   etcd-csle1                     1/1       Running   2       1d
kube-system   kube-apiserver-csle1           1/1       Running   2       1d
kube-system   kube-controller-manager-csle1 1/1       Running   2       1d
kube-system   kube-dns-6f4fd4bdf-vtqbt       3/3       Running   6       1d
kube-system   kube-proxy-vwp26               1/1       Running   2       1d
kube-system   kube-proxy-zs2sw               1/1       Running   2       1d
kube-system   kube-scheduler-csle1           1/1       Running   2       1d
```

클러스터 환경 구성을 위한 pod가 정상적으로 구동되었는지 확인합니다.

```
csle@csle1:~/ksb-csle/kubernetes/env_set$ kubectl get pods --all-namespaces
NAMESPACE     NAME                                READY   STATUS   RESTARTS AGE
kube-system   etcd-csle1                          1/1     Running 2        1d
kube-system   heapster-dfd674df9-bz45s            1/1     Running 2        1d
kube-system   kube-apiserver-csle1                1/1     Running 2        1d
kube-system   kube-controller-manager-csle1       1/1     Running 2        1d
kube-system   kube-dns-6f4fd4bdf-vtqbt            3/3     Running 6        1d
kube-system   kube-flannel-ds-c6mpq               1/1     Running 2        1d
kube-system   kube-flannel-ds-rc74q               1/1     Running 2        1d
kube-system   kube-proxy-vwp26                    1/1     Running 2        1d
kube-system   kube-proxy-zs2sw                    1/1     Running 2        1d
kube-system   kube-scheduler-csle1                1/1     Running 2        1d
kube-system   kubernetes-dashboard-5bd6f767c7-4qzj7  1/1  Running 2    1d
kube-system   monitoring-grafana-76848b566c-4n4xj    1/1  Running 2    1d
kube-system   monitoring-influxdb-6c4b84d695-ts7dk   1/1  Running 2    1d
```

[Master: csle1] Ingress 환경 설정하기

kubernetes master에서 ingress를 설정합니다.

```
cd /home/csle/ksb-csle/kubernetes/env_set
kubectl create -f default-backend-for-ingress.yaml
kubectl create -f configmap-nginx-ingress-controller.yaml
kubectl create -f deploy-nginx-ingress-controller.yaml
kubectl create -f ingress-rule.yaml
kubectl create -f svc-expose-by-nodeport.yaml
```

[Master: csle1] Dashboard URL에 접속하기

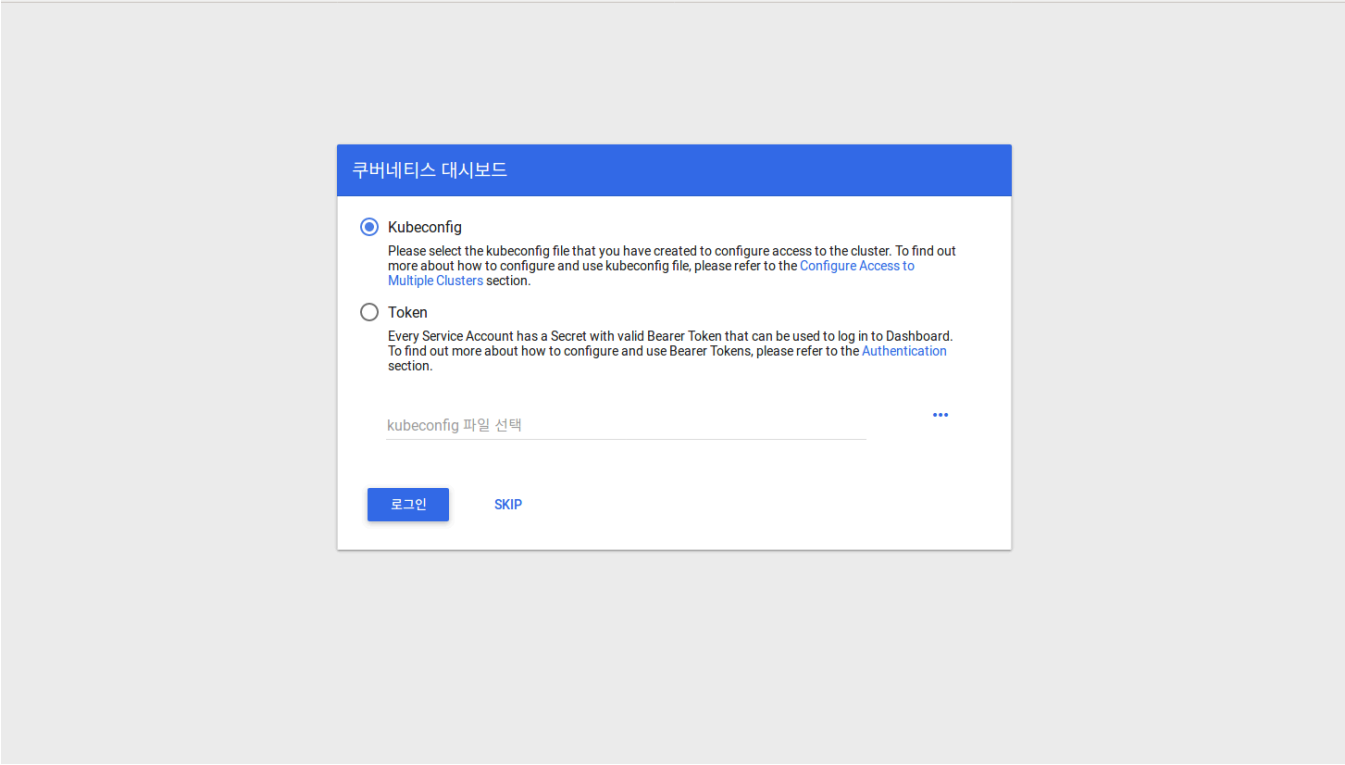
아래의 명령으로 dashboard 프록시를 동작시킵니다.

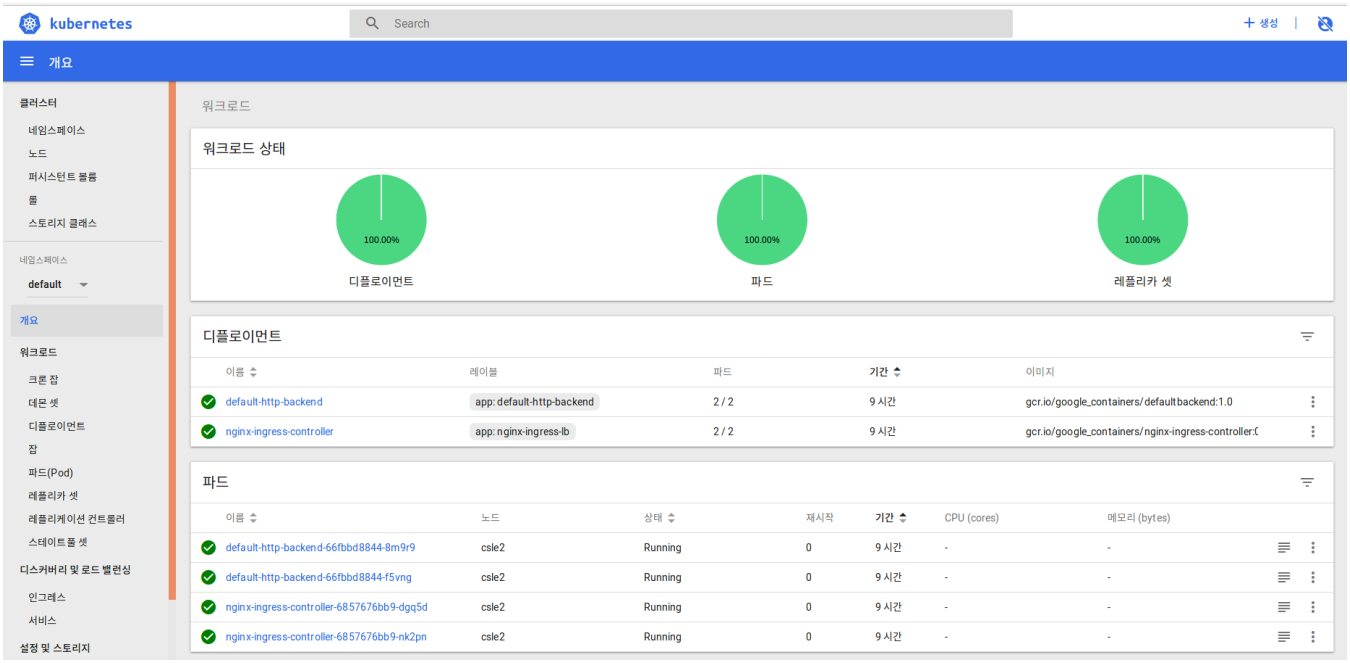
```
kubectl create clusterrolebinding add-on-cluster-admin \
--clusterrole=cluster-admin \
--serviceaccount=kube-system:kubernetes-dashboard

kubectl proxy --port=9999 --address='192.168.0.5' --accept-hosts="^*$"
```

크롬을 열어 아래의 주소로 쿠버네티스 dashboard에 접속한다.

http://192.168.0.5:9999/ui

아래에서 skip을 선택하여 dashboard에 접속합니다.





KSB 프레임워크와 연동을 위한 환경설정하기

[모든 서버 공통] KSB 툴박스에 포함된 SSH 인증키 공유하기

KSB 프레임워크의 엔진이 Kubernetes 마스터에 접속할때 인증 키를 묻지않도록 배포된 KSB 툴박스에 포함된 .ssh 폴더를 쿠버네티스 서버의 홈에 복사합니다.

```
cp -r ~/ksb-csle/tools/.ssh/ /home/csle/
```

[모든 서버 공통] hadoop 프로그램을 설치하기

~/ksb-csle/tools/hadoop-2.7.3 폴더를 master, node1, node2의 csle 홈 폴더에 복사합니다.

```
cp -r ~/ksb-csle/tools/hadoop-2.7.3 /home/csle/
```

그리고 아래와 같이 설정합니다.

```
ln -s hadoop-2.7.3 hadoop
```

[모든 서버 공통] ~/.bashrc 수정하기

```
sudo vi ~/.bashrc

export JAVA_HOME=/usr/lib/jvm/java-8-oracle
export PATH=$PATH:$JAVA_HOME/bin
export KSB_HOME=/home/csle/ksb-csle
export PYTHONPATH=/home/csle/ksb-csle/pyML/:$PYTHONPATH
export PYTHONPATH=./:/home/csle/ksb-csle/ksblib:$PYTHONPATH
export KUBECONFIG=/home/csle/.kube/config

export HADOOP_HOME=/home/csle/hadoop-2.7.3
export HADOOP_PREFIX=$HADOOP_HOME
export PATH=$PATH:$HADOOP_PREFIX/bin
export PATH=$PATH:$HADOOP_PREFIX/sbin
export HADOOP_CONF_DIR=$HADOOP_HOME/etc/hadoop
export HADOOP_MAPRED_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_HOME=${HADOOP_PREFIX}
export HADOOP_HDFS_HOME=${HADOOP_PREFIX}
export YARN_HOME=${HADOOP_PREFIX}
export HADOOP_COMMON_LIB_NATIVE_DIR=${YARN_HOME}/lib/native
export HADOOP_OPTS="-Djava.library.path=$YARN_HOME/lib/native"

source ~/.bashrc
```

이상으로 쿠버네티스와 KSB 툴박스와 연동을 하기 위한 모든 준비가 완료되었습니다.

FAQ

1. 서버 재부팅 후, connection 에러가 발생하는 경우

```
csle@csle1:~$ kubectl get nodes
The connection to the server 192.168.0.5:6443 was refused
- did you specify the right host or port?
```

- 해결 방법

```
csle@csle1:~$ sudo -i
root@csle1:~# swapoff -a
root@csle1:~# exit
csle@csle1:~$ strace -eopenat kubectl version
```

2. 서버 재부팅 후, dashboard가 ContainerCreating 상태로 에러가 발생한 경우

```
csle@csle1:~/ksb-csle/kubernetes/env_set$ kubectl get pods --all-namespaces
NAMESPACE   NAME                                READY  STATUS  RESTARTS AGE
kube-system etcd-csle1                          1/1    Running 2        1d
kube-system heapster-dfd674df9-bz45s            1/1    Running 2        1d
kube-system kube-apiserver-csle1                1/1    Running 2        1d
kube-system kube-controller-manager-csle1       1/1    Running 2        1d
kube-system kube-dns-6f4fd4bdf-vtqbt            3/3    Running 6        1d
kube-system kube-flannel-ds-c6mpq               1/1    Running 2        1d
kube-system kube-flannel-ds-rc74q               1/1    Running 2        1d
kube-system kube-proxy-vwp26                    1/1    Running 2        1d
kube-system kube-proxy-zs2sw                    1/1    Running 2        1d
kube-system kube-scheduler-csle1                1/1    Running 2        1d
kube-system kubernetes-dashboard-5bd6f767c7-4qzj7  1/1 ContainerCreating 1d
kube-system monitoring-grafana-76848b566c-4n4xj    1/1  Running 2    1d
kube-system monitoring-influxdb-6c4b84d695-ts7dk   1/1  Running 2    1d
```

dashboard 관련 설정파일을 이용해서 pod를 delete 후, 재생성합니다.

```
cd /home/csle/ksb-csle/kubernetes/env_set
kubectl delete -f kube-flannel.yml
kubectl delete -f kubernetes-dashboard.yaml
kubectl delete -f kubernetes-dashboard-admin-rbac.yaml
kubectl delete -f ServiceAccount.yml
kubectl delete -f ClusterRoleBinding.yml
kubectl delete -f k8s-heapster/

kubectl create -f kube-flannel.yml
kubectl create -f kubernetes-dashboard.yaml
kubectl create -f kubernetes-dashboard-admin-rbac.yaml
kubectl create -f ServiceAccount.yml
kubectl create -f ClusterRoleBinding.yml
kubectl create -f k8s-heapster/
```

3. KSB 툴박스 docker 컨테이너에서 host pc에 ssh 접속시

password를 묻는지 확인하는 방법

KSB 툴박스 docker 컨테이너에서 아래의 명령을 수행하여 각 kubernete 마스터와 노드에 접속시 password를 묻는지 확인합니다.

```
csle@csle1:~$ ssh -l csle csle1 -p 2243 mkdir -p \
/home/csle/ksb-csle/kubernetes/modelDB/kangnam/0001
Warning: Permanently added '[csle1]:2243,[192.168.0.5]:2243'
(ECDSA) to the list of known hosts.
csle@csle1:~$
csle@csle1:~$ ssh -l csle csle2 -p 2243 mkdir -p \
/home/csle/ksb-csle/kubernetes/modelDB/kangnam/0001
Warning: Permanently added '[csle2]:2243,[192.168.0.6]:2243'
(ECDSA) to the list of known hosts.
csle@csle1:~$
```

만약 password를 묻는다면, .ssh 폴더의 인증키가 공유되지 않은 상태입니다.
password를 묻는 마스터 혹은 노드 HOST에서 대해서 아래의 절차대로 인증키를 복사하시기 바랍니다.

[Host PC] KSB 툴박스 docker 이미지내 SSH 키를 host pc에 복사하기

KSB 툴박스 docker 컨테이너의 인증키를 공유하기 위해 ksb-csle/tools 폴더에 있는 .ssh 폴더를 csle 홈에 카피합니다.

```
cp -r ~/ksb-csle/tools/.ssh/ /home/csle/
chmod 600 ~/.ssh/authorized_keys
chmod 600 ~/.ssh/id_rsa
chmod 700 ~/.ssh/
rm ~/.ssh/known_hosts
sudo service ssh restart
```

4. 쿠버네티스 설치 후, KSB 프레임워크 동작시 namenode가 동작하지 않을 경우

거의 드물게 발생하는 에러로 jps 명령을 통해 KSB 툴박스에서 프로세스를 확인했을때 namenode가 검색되지 않는 경우가 있습니다.
쿠버네티스의 etcd 가 사용하는 port와 hadoop이 사용하는 port 충돌로 인해 namenode가 초기화되지 않을 경우가 있습니다.
확인하는 방법은 다음과 같습니다. 이러한 문제를 해결하는 방법은 kubernetes를 삭제하고 다시

설치하는 것이 가장 쉬운 방법입니다.
위에서 설명한 삭제 및 설치를 다시 하고, 바로 10-kubeadm.conf 수정하기 부분을 수행하시고, kubeadm init을 하시면 쉽게 재설치가 됩니다.
또한 node들에서도 kuberet 후, kubejoin으로 다시 클러스터를 설정하시면 됩니다.

[Host PC] etcd 사용 port 확인하기

netstat -tnlpa | grep 2379

[Host PC] 프로세스 port사용 확인

ps aux | grep 2379

[Host PC] 열려있는 파일 목록 확인하기

sudo lsof -i TCP:2379

hadoop에서 사용하는 default port 목록

Daemon	Default Port	Configuration Parameter
Namenode	50070	dfs.http.address
Datanodes	50075	dfs.datanode.http.address
Secondarynamenode	50090	dfs.secondary.http.address
Backup/Checkpoint node	50105	dfs.backup.http.address
Tasktrackers	50060	mapred.task.tracker.http.address
Jobracker	50030	mapred.job.tracker.http.address

5. 쿠버네티스 kubectl 명령 alias 방법

```
echo "alias k=kubectl" >> ~/.bashrc
source <(kubectl completion bash | sed s/kubectl/k/g)
```