**K8S 설치 가이드**

**Ver. 1.0**

**목 차**

[1. Pre-installation works (For each server) 3](#_Toc21876724)

[1.1. IP & Hostname Settings 3](#_Toc21876725)

[1.2. Hosts registration 3](#_Toc21876726)

[1.3. epel installation & yum update 3](#_Toc21876727)

[1.4. Disable firewall 3](#_Toc21876728)

[2. Docker installation (For each server) 4](#_Toc21876729)

[3. KubeAdm & Kubelet & Util installation 5](#_Toc21876730)

[3.1. Kube repo Registration (For each server) 5](#_Toc21876731)

[3.2. Configuration about docker-specific (For each server) 5](#_Toc21876732)

[3.3. KubeAdm & Kubelet & Util installation (For master server) 5](#_Toc21876733)

[3.4. KubeAdm & Kubelet & Util installation (For node server) 5](#_Toc21876734)

[3.5. KubeAdm & Kubelet & Util configuration (For each server) 6](#_Toc21876735)

[4. kubernetes initial configuration 7](#_Toc21876736)

[4.1. Master configuration (For master server) 7](#_Toc21876737)

[4.2. Minion configuration (For node server) 9](#_Toc21876738)

[4.3. Master checks (For master server) 9](#_Toc21876739)

[5. Weave Net(Network Plug-in) installation 10](#_Toc21876740)

[6. Addon installation 11](#_Toc21876741)

[6.1. Metrics Server installation 11](#_Toc21876742)

[6.2. Heapster Installation 12](#_Toc21876743)

[6.3. Dashboard installation 13](#_Toc21876744)

[7. Container(HTTPD) Test 15](#_Toc21876745)

[8. Docker(Altibase) Image Management 17](#_Toc21876746)

[8.1. Produce by Docker(Altibase) Image 17](#_Toc21876747)

[8.2. Docker(Altibase) Image deployment 18](#_Toc21876748)

1. Pre-installation works (For each server)
   1. IP & Hostname Settings

|  |
| --- |
| [root@k8s-master ~]# vi /etc/sysconfig/network-scripts/ifcfg-ens33  [root@k8s-master ~]# hostnamectl set-hostname master  [root@k8s-master ~]# service network restart |

* 1. Hosts registration

|  |
| --- |
| [root@k8s-master ~]# vi /etc/hosts  ...  192.168.1.204 k8s-master  192.168.1.201 k8s-node01  192.168.1.202 k8s-node02  192.168.1.203 k8s-node03 |

* 1. epel installation & yum update

|  |
| --- |
| [root@k8s-master ~]# yum makecache fast  [root@k8s-master ~]# yum update -y  [root@k8s-master ~]# yum install -y epel-release |

* 1. Disable firewall

|  |
| --- |
| [root@k8s-master ~]# systemctl stop firewalld && systemctl disable firewalld  [root@k8s-master ~]# setenforce 0 |

1. Docker installation (For each server)

|  |
| --- |
| [root@k8s-master ~]# yum install -y docker  [root@k8s-master ~]# systemctl enable docker && systemctl start docker  [root@k8s-master ~]# docker version |

1. KubeAdm & Kubelet & Util installation
   1. Kube repo Registration (For each server)

|  |
| --- |
| [root@k8s-master ~]# cat <<EOF > /etc/yum.repos.d/kubernetes.repo  [kubernetes]  name=Kubernetes  baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86\_64  enabled=1  gpgcheck=1  repo\_gpgcheck=1  gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg  https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg  EOF |

* 1. Configuration about docker-specific (For each server)

|  |
| --- |
| [root@k8s-master ~]# cat <<EOF > /etc/sysctl.d/k8s.conf  net.bridge.bridge-nf-call-iptables=1  net.bridge.bridge-nf-call-ip6tables=1  net.netfilter.nf\_conntrack\_max = 786432  EOF    [root@k8s-master ~]# sysctl -p /etc/sysctl.d/k8s.conf |

* 1. KubeAdm & Kubelet & Util installation (For master server)

|  |
| --- |
| [root@k8s-master ~]# yum install -y kubelet kubeadm kubectl kubernetes-cni nmap bind-utils net-tools chrony wget fping jq git bash-completion rdate |

* 1. KubeAdm & Kubelet & Util installation (For node server)

|  |
| --- |
| [root@node1 ~]# yum install -y kubelet kubeadm kubernetes-cni net-tools chrony rdate |

* 1. KubeAdm & Kubelet & Util configuration (For each server)

|  |
| --- |
| [root@k8s-master ~]# swapoff -a  [root@k8s-master ~]# free  [root@k8s-master ~]# systemctl enable kubelet && systemctl start kubelet  [root@k8s-master ~]# rdate -s time.bora.net  [root@k8s-master ~]# vi /etc/chrony.conf  ...  server time.bora.net iburst  local stratum 10  ...  [root@k8s-master ~]# systemctl enable chronyd && systemctl restart chronyd  [root@k8s-master ~]# chronyc tracking  [root@k8s-master ~]# chronyc sources -v |

1. kubernetes initial configuration
   1. Master configuration (For master server)

|  |
| --- |
| [root@k8s-master ~]# kubeadm init  [init] Using Kubernetes version: v1.15.3  [preflight] Running pre-flight checks  [preflight] Pulling images required for setting up a Kubernetes cluster  [preflight] This might take a minute or two, depending on the speed of your internet connection  [preflight] You can also perform this action in beforehand using 'kubeadm config images pull'  [kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"  [kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"  [kubelet-start] Activating the kubelet service  [certs] Using certificateDir folder "/etc/kubernetes/pki"  [certs] Generating "front-proxy-ca" certificate and key  [certs] Generating "front-proxy-client" certificate and key  [certs] Generating "etcd/ca" certificate and key  [certs] Generating "apiserver-etcd-client" certificate and key  [certs] Generating "etcd/server" certificate and key  [certs] etcd/server serving cert is signed for DNS names [k8s-master localhost] and IPs [192.168.1.204 127.0.0.1 ::1]  [certs] Generating "etcd/peer" certificate and key  [certs] etcd/peer serving cert is signed for DNS names [k8s-master localhost] and IPs [192.168.1.204 127.0.0.1 ::1]  [certs] Generating "etcd/healthcheck-client" certificate and key  [certs] Generating "ca" certificate and key  [certs] Generating "apiserver-kubelet-client" certificate and key  [certs] Generating "apiserver" certificate and key  [certs] apiserver serving cert is signed for DNS names [k8s-master kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc.cluster.local] and IPs [10.96.0.1 192.168.1.204]  [certs] Generating "sa" key and public key  [kubeconfig] Using kubeconfig folder "/etc/kubernetes"  [kubeconfig] Writing "admin.conf" kubeconfig file  [kubeconfig] Writing "kubelet.conf" kubeconfig file  [kubeconfig] Writing "controller-manager.conf" kubeconfig file  [kubeconfig] Writing "scheduler.conf" kubeconfig file  [control-plane] Using manifest folder "/etc/kubernetes/manifests"  [control-plane] Creating static Pod manifest for "kube-apiserver"  [control-plane] Creating static Pod manifest for "kube-controller-manager"  [control-plane] Creating static Pod manifest for "kube-scheduler"  [etcd] Creating static Pod manifest for local etcd in "/etc/kubernetes/manifests"  [wait-control-plane] Waiting for the kubelet to boot up the control plane as static Pods from directory "/etc/kubernetes/manifests". This can take up to 4m0s  [apiclient] All control plane components are healthy after 37.504167 seconds  [upload-config] Storing the configuration used in ConfigMap "kubeadm-config" in the "kube-system" Namespace  [kubelet] Creating a ConfigMap "kubelet-config-1.15" in namespace kube-system with the configuration for the kubelets in the cluster  [upload-certs] Skipping phase. Please see --upload-certs  [mark-control-plane] Marking the node k8s-master as control-plane by adding the label "node-role.kubernetes.io/master=''"  [mark-control-plane] Marking the node k8s-master as control-plane by adding the taints [node-role.kubernetes.io/master:NoSchedule]  [bootstrap-token] Using token: sliqf3.rh0r6ywr037hynia  [bootstrap-token] Configuring bootstrap tokens, cluster-info ConfigMap, RBAC Roles  [bootstrap-token] configured RBAC rules to allow Node Bootstrap tokens to post CSRs in order for nodes to get long term certificate credentials  [bootstrap-token] configured RBAC rules to allow the csrapprover controller automatically approve CSRs from a Node Bootstrap Token  [bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster  [bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace  [addons] Applied essential addon: CoreDNS  [addons] Applied essential addon: kube-proxy  Your Kubernetes control-plane 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/  Then you can join any number of worker nodes by running the following on each as root:  kubeadm join 192.168.1.204:6443 --token sliqf3.rh0r6ywr037hynia \  --discovery-token-ca-cert-hash sha256:4cb93a8d26c20aed723f46c1f6561e50f6674062219b8a755e136aea66da2074  [root@k8s-master ~]# mkdir -p $HOME/.kube  [root@k8s-master ~]# cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  [root@k8s-master ~]# chown $(id -u):$(id -g) $HOME/.kube/config  [root@k8s-master ~]# export KUBECONFIG=$HOME/.kube/config  [root@k8s-master ~]# echo "export KUBECONFIG=$HOME/.kube/config" | tee -a ~/.bashrc  [root@k8s-master ~]# source /etc/profile.d/bash\_completion.sh  [root@k8s-master ~]# source <(kubectl completion bash) |

* 1. Minion configuration (For node server)

|  |
| --- |
| [root@node1 ~]# kubeadm join 192.168.1.204:6443 --token sliqf3.rh0r6ywr037hynia --discovery-token-ca-cert-hash sha256:4cb93a8d26c20aed723f46c1f6561e50f6674062219b8a755e136aea66da2074 |

* 1. Master checks (For master server)

|  |
| --- |
| [root@k8s-master ~]# kubectl get nodes  NAME STATUS ROLES AGE VERSION  k8s-master NotReady master 3m52s v1.15.3  k8s-node1 NotReady <none> 50s v1.15.3  k8s-node2 NotReady <none> 49s v1.15.3  k8s-node3 NotReady <none> 48s v1.15.3 |

1. Weave Net(Network Plug-in) installation

|  |
| --- |
| [root@k8s-master ~]# kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=$(kubectl version | base64 | tr -d '\n')"  serviceaccount/weave-net created  clusterrole.rbac.authorization.k8s.io/weave-net created  clusterrolebinding.rbac.authorization.k8s.io/weave-net created  role.rbac.authorization.k8s.io/weave-net created  rolebinding.rbac.authorization.k8s.io/weave-net created  daemonset.extensions/weave-net created  [root@k8s-master ~]# kubectl get nodes  NAME STATUS ROLES AGE VERSION  k8s-master Ready master 6m15s v1.15.3  k8s-node1 Ready <none> 3m13s v1.15.3  k8s-node2 Ready <none> 3m12s v1.15.3  k8s-node3 Ready <none> 3m11s v1.15.3  [root@k8s-master ~]# kubectl get pods --all-namespaces  NAMESPACE NAME READY STATUS RESTARTS AGE  kube-system coredns-5c98db65d4-blnmj 1/1 Running 0 6m13s  kube-system coredns-5c98db65d4-xdqhd 1/1 Running 0 6m13s  kube-system etcd-k8s-master 1/1 Running 0 5m21s  kube-system kube-apiserver-k8s-master 1/1 Running 0 5m33s  kube-system kube-controller-manager-k8s-master 1/1 Running 0 5m2s  kube-system kube-proxy-26r75 1/1 Running 0 3m22s  kube-system kube-proxy-fftt9 1/1 Running 0 6m13s  kube-system kube-proxy-grtpr 1/1 Running 0 3m20s  kube-system kube-proxy-rplrh 1/1 Running 0 3m21s  kube-system kube-scheduler-k8s-master 1/1 Running 0 5m28s  kube-system weave-net-7lhd6 2/2 Running 0 59s  kube-system weave-net-8274x 2/2 Running 0 59s  kube-system weave-net-mmk8s 2/2 Running 0 59s  kube-system weave-net-qw8f9 2/2 Running 0 59s |

1. Addon installation
   1. Metrics Server installation

|  |
| --- |
| [root@k8s-master ~]# git clone https://github.com/kubernetes-incubator/metrics-server.git  [root@k8s-master ~]# kubectl create -f metrics-server/deploy/1.8+/  [root@k8s-master ~]# kubectl edit deploy -n kube-system metrics-server  ...  template:  metadata:  creationTimestamp: null  labels:  k8s-app: metrics-server  name: metrics-server  spec:  containers:  - image: k8s.gcr.io/metrics-server-amd64:v0.3.4  imagePullPolicy: Always  name: metrics-server  args:  - --kubelet-insecure-tls  - --kubelet-preferred-address-types=InternalIP,ExternalIP,Hostname  resources: {}  terminationMessagePath: /dev/termination-log  terminationMessagePolicy: File  ...  [root@k8s-master ~]# kubectl get apiservices |egrep metrics  v1beta1.metrics.k8s.io kube-system/metrics-server True 2m15s  [root@k8s-master ~]# kubectl get deployment metrics-server -n kube-system  NAME READY UP-TO-DATE AVAILABLE AGE  metrics-server 1/1 1 1 4m34s  [root@k8s-master ~]# kubectl top nodes  NAME CPU(cores) CPU% MEMORY(bytes) MEMORY%  k8s-master 164m 4% 1344Mi 17%  k8s-node1 57m 1% 842Mi 10%  k8s-node2 56m 1% 843Mi 10%  k8s-node3 62m 1% 870Mi 11%  [root@k8s-master ~]# kubectl top pods --all-namespaces  NAMESPACE NAME CPU(cores) MEMORY(bytes)  kube-system coredns-5c98db65d4-blnmj 5m 12Mi  kube-system coredns-5c98db65d4-xdqhd 4m 10Mi  kube-system etcd-k8s-master 31m 30Mi  kube-system kube-apiserver-k8s-master 42m 281Mi  kube-system kube-controller-manager-k8s-master 25m 49Mi  kube-system kube-proxy-26r75 5m 14Mi  kube-system kube-proxy-fftt9 1m 13Mi  kube-system kube-proxy-grtpr 1m 12Mi  kube-system kube-proxy-rplrh 1m 14Mi  kube-system kube-scheduler-k8s-master 4m 15Mi  kube-system metrics-server-6b6cd9b466-5htpr 1m 11Mi  kube-system weave-net-7lhd6 2m 125Mi  kube-system weave-net-8274x 5m 125Mi  kube-system weave-net-mmk8s 2m 110Mi  kube-system weave-net-qw8f9 7m 123Mi |

* 1. Heapster Installation

|  |
| --- |
| [root@k8s-master ~]# git clone https://github.com/kubernetes/heapster.git  [root@k8s-master ~]# kubectl create -f /root/heapster/deploy/kube-config/influxdb  [root@k8s-master ~]# kubectl create -f /root/heapster/deploy/kube-config/rbac/heapster-rbac.yaml |

* 1. Dashboard installation

|  |
| --- |
| [root@master dashboard]# curl https://raw.githubusercontent.com/kubernetes/dashboard/v1.10.1/src/deploy/recommended/kubernetes-dashboard.yaml -O  [root@k8s-master ~]# kubectl create -f kubernetes-dashboard.yaml  serviceaccount/kubernetes-dashboard created  clusterrolebinding.rbac.authorization.k8s.io/kubernetes-dashboard created  deployment.apps/kubernetes-dashboard created  service/kubernetes-dashboard created  [root@k8s-master ~]# kubectl get pods --all-namespaces  NAMESPACE NAME READY STATUS RESTARTS AGE  ...  kube-system kubernetes-dashboard-6f8d67df77-sszff 1/1 Running 0 15s  ...  [root@k8s-master ~]# kubectl get svc --all-namespaces  NAMESPACE NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  default kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 26m  kube-system heapster ClusterIP 10.99.187.61 <none> 80/TCP 9m18s  kube-system kube-dns ClusterIP 10.96.0.10 <none> 53/UDP,53/TCP,9153/TCP 26m  kube-system kubernetes-dashboard NodePort 10.96.33.5 <none> 80:30000/TCP 118s  kube-system metrics-server ClusterIP 10.96.91.84 <none> 443/TCP 17m  kube-system monitoring-grafana NodePort 10.102.117.138 <none> 80:30080/TCP 9m18s  kube-system monitoring-influxdb ClusterIP 10.96.70.181 <none> 8086/TCP 9m18s |

1. Container(HTTPD) Test

|  |
| --- |
| [root@k8s-master tHttpd]# vi httpd.yaml  apiVersion: v1  kind: Namespace  metadata:  name: web  ---  apiVersion: apps/v1beta1  kind: Deployment  metadata:  name: httpd  namespace: web  spec:  replicas: 1  template:  metadata:  name: httpd  labels:  app: httpd  spec:  containers:  - name: httpd  image: docker.io/httpd  ports:  - containerPort: 80  name: web-port  protocol: TCP  ---  apiVersion: v1  kind: Service  metadata:  labels:  name: httpd  name: httpd  namespace: web  spec:  ports:  - port: 80  protocol: TCP  targetPort: web-port  selector:  app: httpd  type: NodePort  [root@k8s-master tHttpd]# kubectl create -f httpd.yaml  namespace/web created  deployment.apps/httpd created  service/httpd created  [root@k8s-master tHttpd]# kubectl get pods -n web  NAME READY STATUS RESTARTS AGE  httpd-66cf4b9bd5-ccjst 1/1 Running 0 9m49s  [root@k8s-master tHttpd]# kubectl get svc -n web  NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  httpd NodePort 10.96.37.164 <none> 80:32472/TCP 9m56s  [root@k8s-master tHttpd]# telnet 192.168.1.202 32472  Trying 192.168.1.202...  Connected to 192.168.1.202.  Escape character is '^]'.  GET / HTTP/1.0  HTTP/1.1 200 OK  Date: Thu, 19 Sep 2019 14:36:21 GMT  Server: Apache/2.4.41 (Unix)  Last-Modified: Mon, 11 Jun 2007 18:53:14 GMT  ETag: "2d-432a5e4a73a80"  Accept-Ranges: bytes  Content-Length: 45  Connection: close  Content-Type: text/html  <html><body><h1>It works!</h1></body></html>  Connection closed by foreign host. |

1. Docker(Altibase) Image Management
   1. Produce by Docker(Altibase) Image

|  |
| --- |
| [root@nginx altibase]# more Dockerfile  FROM centos  MAINTAINER YongSung Hwang <blueseam@gmail.com>  COPY root/opt/ /opt/  COPY root/root/.bash\_profile /root/  EXPOSE 20300  ENTRYPOINT ["/opt/altibase-server-7.1.0/bin/altibase-entrypoint.sh"]  [root@nginx altibase]# vi root/opt/altibase-server-7.1.0/bin/altibase-entrypoint.sh  #!/bin/bash  source /root/.bash\_profile  #exec server start  ADMIN="${ALTIBASE\_HOME}/bin/isql -u sys -p MANAGER -sysdba -noprompt"  ${ADMIN} << EOF  startup  EOF  tail -f /dev/null  …  [root@nginx altibase]# docker build -t blueseam/altibase:v0.02 .  Sending build context to Docker daemon 783.9 MB  Step 1/6 : FROM centos  ---> 67fa590cfc1c  Step 2/6 : MAINTAINER YongSung Hwang <blueseam@gmail.com>  ---> Using cache  ---> 2e41128256c7  Step 3/6 : COPY root/opt/ /opt/  ---> Using cache  ---> c9e32533e125  Step 4/6 : COPY root/root/.bash\_profile /root/  ---> Using cache  ---> 1cfc2ab60665  Step 5/6 : EXPOSE 20300  ---> Using cache  ---> d27e36bed793  Step 6/6 : ENTRYPOINT /opt/altibase-server-7.1.0/bin/altibase-entrypoint.sh  ---> Using cache  ---> 2ed71ec4c040  Successfully built 2ed71ec4c040 |

* 1. Docker(Altibase) Image deployment

|  |
| --- |
| [root@master ~]# vi altibase.yaml  …  apiVersion: v1  kind: Namespace  metadata:  name: hys  ---  apiVersion: apps/v1beta1  kind: Deployment  metadata:  name: altibase  namespace: hys  spec:  replicas: 1  template:  metadata:  name: altibase  labels:  app: altibase  spec:  containers:  - name: altibase  image: docker.io/blueseam/altibase:v0.01  ports:  - containerPort: 20300  name: db-port  protocol: TCP  ---  apiVersion: v1  kind: Service  metadata:  labels:  name: altibase  name: altibase  namespace: hys  spec:  ports:  - port: 20300  protocol: TCP  targetPort: db-port  selector:  app: altibase  type: NodePort  …  [root@master ~]# kubectl create -f altibase.yaml  namespace/hys created  deployment.apps/altibase created  service/altibase created |