部署高可用kubernetes master集群

kubernetes master节点包含的组件：

- kube-apiserver
- kube-scheduler
- kube-controller-manager

目前这三个组件需要部署在同一台机器上。

- kube-scheduler、 kube-controller-manager 和kube-apiserver 三者的功能紧密相关；
- 同时只能有一个kube-scheduler、 kube-controller-manager进程处于工作状态，如果运行多个，则需要通过选举产生一个 leader；

TLS证书文件

pem和 token.csv 证书文件我们在TLS 证书和密钥这一步中已经创建过了。再检查一下。

```
[root@vlnx251101 ~]# ls /etc/kubernetes/ssl/
admin-key.pem  admin.pem  ca-key.pem  ca.pem  kube-proxy-key.pem  kube-proxy.pem
kubernetes-key.pem  kubernetes.pem
```

下载最新版本的二进制文件

下载 server tarball 文件

https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG-1.11.md#downloads-for-v1112

server 的tarball kubernetes-server-linux-amd64.tar.gz 已经包含了 client（kubectl） 二进制文件，所以不用单独下载 kubernetes-client-linux-amd64.tar.gz 文件；

```
[root@vlnx251101 ~]# wget https://dl.k8s.io/v1.11.2/kubernetes-server-linux-amd64.tar.gz
[root@vlnx251101 ~]# tar xf kubernetes-server-linux-amd64.tar.gz
[root@vlnx251101 ~]# cd kubernetes/server/bin/
```

分发到所有节点（同时部署到node）

```
[root@vlnx251101 kubernetes]# cp -r kube-apiserver kube-controller-manager kube-
scheduler kubectl kube-proxy kubelet /usr/local/bin/
```

master:

```
kube-apiserver kube-controller-manager kube-scheduler kubectl
```

node:

```
kube-proxy kubelet
```

配置和启动kube-apiserver

创建kube-apiserver的 service配置文件

serivce 配置文件 /usr/lib/systemd/system/kube-apiserver.service 内容：

```
[root@vlnx251101 kubernetes]# vim /usr/lib/systemd/system/kube-apiserver.service

[Unit]
Description=Kubernetes API Service
Documentation=https://github.com/GoogleCloudPlatform/kubernetes
After=network.target
After=etcd.service

[Service]
EnvironmentFile=-/etc/kubernetes/config
EnvironmentFile=-/etc/kubernetes/apiserver
ExecStart=/usr/local/bin/kube-apiserver \
          $KUBE_LOGTOSTDERR \
          $KUBE_LOG_LEVEL \
          $KUBE_ETCD_SERVERS \
          $KUBE_API_ADDRESS \
          $KUBE_API_PORT \
          $KUBELET_PORT \
          $KUBE_ALLOW_PRIV \
          $KUBE_SERVICE_ADDRESSES \
          $KUBE_ADMISSION_CONTROL \
          $KUBE_API_ARGS
Restart=on-failure
Type=notify
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

/etc/kubernetes/config 文件的内容为：

```
[root@vlnx251101 kubernetes]# vim /etc/kubernetes/config

###
# kubernetes system config
#
# The following values are used to configure various aspects of all
# kubernetes services, including
#
#   kube-apiserver.service
#   kube-controller-manager.service
#   kube-scheduler.service
#   kubelet.service
#   kube-proxy.service
```

```
# logging to stderr means we get it in the systemd journal
KUBE_LOGTOSTDERR="--logtostderr=true"
# journal message level, 0 is debug
KUBE_LOG_LEVEL="--v=0"
# Should this cluster be allowed to run privileged docker containers
KUBE_ALLOW_PRIV="--allow-privileged=true"
# How the controller-manager, scheduler, and proxy find the apiserver
KUBE_MASTER="--master=http://192.168.251.101:8080"
```

该配置文件同时被 kube-apiserver、 kube-controller-manager 、kube-scheduler、 kubelet 、kube-proxy使用。**（每个节点都要有）**

apiserver 配置文件/etc/kubernetes/apiserver内容为：

```
[root@vlnx251101 kubernetes]# vim /etc/kubernetes/apiserver

###
## kubernetes system config
##
## The following values are used to configure the kube-apiserver
##
#
## The address on the local server to listen to.
KUBE_API_ADDRESS="--advertise-address=192.168.251.101 --bind-address=192.168.251.101 --
insecure-bind-address=192.168.251.101"
#
## The port on the local server to listen on.
#KUBE_API_PORT="--port=8080"
#
## Port minions listen on
#KUBELET_PORT="--kubelet-port=10250"
#
## Comma separated list of nodes in the etcd cluster
KUBE_ETCD_SERVERS="--etcd-
servers=https://192.168.251.101:2379,https://192.168.251.102:2379,https://192.168.251.103
#
## Address range to use for services
KUBE_SERVICE_ADDRESSES="--service-cluster-ip-range=10.254.0.0/16"
#
## default admission control policies
KUBE_ADMISSION_CONTROL="--admission-
control=ServiceAccount,NamespaceLifecycle,NamespaceExists,LimitRanger,ResourceQuota"
#
## Add your own!
KUBE_API_ARGS="--authorization-mode=Node,RBAC --enable-bootstrap-token-auth --runtime-
config=rbac.authorization.k8s.io/v1beta1 --kubelet-https=true --token-auth-
```

```
file=/etc/kubernetes/token.csv --service-node-port-range=30000-32767 --tls-cert-
file=/etc/kubernetes/ssl/kubernetes.pem --tls-private-key-
file=/etc/kubernetes/ssl/kubernetes-key.pem --client-ca-file=/etc/kubernetes/ssl/ca.pem
--service-account-key-file=/etc/kubernetes/ssl/ca-key.pem --etcd-
cafile=/etc/kubernetes/ssl/ca.pem --etcd-certfile=/etc/kubernetes/ssl/kubernetes.pem --
etcd-keyfile=/etc/kubernetes/ssl/kubernetes-key.pem --enable-swagger-ui=true --
apiserver-count=3 --audit-log-maxage=30 --audit-log-maxbackup=3 --audit-log-maxsize=100
--audit-log-path=/var/lib/audit.log --event-ttl=1h"
```

- 如果中途修改过--service-cluster-ip-range地址，则必须将default命名空间的kubernetes的service给删除，使用命令：kubectl delete service kubernetes，然后系统会自动用新的ip重建这个service，不然apiserver的log有报错the cluster IP x.x.x.x for service kubernetes/default is not within the service CIDR x.x.x.x/16; please recreate
- --authorization-mode=Node,RBAC，指定在安全端口使用 RBAC 授权模式，拒绝未通过授权的请求；对于Kubernetes1.9集群，增加对Node授权的模式，否则将无法注册node。
- kube-scheduler、kube-controller-manager 一般和 kube-apiserver 部署在同一台机器上，它们使用**非安全端口**和 kube-apiserver通信;
- kubelet、kube-proxy、kubectl 部署在其它 Node 节点上，如果通过**安全端口**访问 kube-apiserver，则必须先通过 TLS 证书认证，再通过 RBAC 授权；
- kube-proxy、kubectl 通过在使用的证书里指定相关的 User、Group 来达到通过 RBAC 授权的目的；
- 如果使用了 kubelet TLS Boostrap 机制，则不能再指定 --kubelet-certificate-authority、--kubelet-client-certificate 和 --kubelet-client-key 选项，否则后续 kube-apiserver 校验 kubelet 证书时出现 ”x509: certificate signed by unknown authority“ 错误；
- --admission-control 值必须包含 ServiceAccount；
- --bind-address 不能为 127.0.0.1；
- runtime-config配置为rbac.authorization.k8s.io/v1beta1，表示运行时的apiVersion；
- --service-cluster-ip-range 指定 Service Cluster IP 地址段，该地址段不能路由可达；
- 缺省情况下 kubernetes 对象保存在 etcd /registry 路径下，可以通过 --etcd-prefix 参数进行调整；
- 如果需要开通http的无认证的接口，则可以增加以下两个参数：--insecure-port=8080 --insecure-bind-address=127.0.0.1。注意，生产上不要绑定到非127.0.0.1的地址上

启动kube-apiserver

```
[root@vlnx251101 kubernetes]# systemctl daemon-reload ; systemctl enable kube-apiserver
; systemctl start kube-apiserver ; systemctl status kube-apiserver
```

配置和启动kube-controller-manager

创建kube-controller-manager的 serivce配置文件

文件路径/usr/lib/systemd/system/kube-controller-manager.service

```
[root@vlnx251101 kubernetes]# vim /usr/lib/systemd/system/kube-controller-
manager.service
[Unit]
Description=Kubernetes Controller Manager
Documentation=https://github.com/GoogleCloudPlatform/kubernetes

[Service]
EnvironmentFile=-/etc/kubernetes/config
EnvironmentFile=-/etc/kubernetes/controller-manager
ExecStart=/usr/local/bin/kube-controller-manager \
         $KUBE_LOGTOSTDERR \
         $KUBE_LOG_LEVEL \
         $KUBE_MASTER \
         $KUBE_CONTROLLER_MANAGER_ARGS
Restart=on-failure
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

配置文件 /etc/kubernetes/controller-manager

```
[root@vlnx251101 kubernetes]# vim /etc/kubernetes/controller-manager

###
# The following values are used to configure the kubernetes controller-manager
# defaults from config and apiserver should be adequate
# Add your own!
KUBE_CONTROLLER_MANAGER_ARGS="--address=127.0.0.1 --service-cluster-ip-
range=10.254.0.0/16 --cluster-name=kubernetes --cluster-signing-cert-
file=/etc/kubernetes/ssl/ca.pem --cluster-signing-key-file=/etc/kubernetes/ssl/ca-
key.pem  --service-account-private-key-file=/etc/kubernetes/ssl/ca-key.pem --root-ca-
file=/etc/kubernetes/ssl/ca.pem --leader-elect=true"
```

- --service-cluster-ip-range参数指定 Cluster 中Service 的CIDR 范围，该网络在各Node间必须路由不可达，必须和 kube-apiserver中的参数一致；
- --cluster-signing-* 指定的证书和私钥文件用来签名为TLS BootStrap 创建的证书和私钥；
- --root-ca-file 用来对kube-apiserver 证书进行校验，**指定该参数后，才会在Pod容器的ServiceAccount中放置该 CA 证书文件；**
- --address值必修为 127.0.0.1，应为当前 kube-apiserver 期望scheduler和 controller-manager在同一台机器，否则：

启动kube-controller-manager

```
[root@vlnx251101 kubernetes]# systemctl daemon-reload ; systemctl enable kube-controller-manager ; systemctl start kube-controller-manager ; systemctl status kube-controller-manager
```

配置和启动kube-scheduler

创建kube-scheduler 的serivce配置文件

文件路径/usr/lib/systemd/system/kube-scheduler.service。

```
[root@vlnx251101 kubernetes]# vim /usr/lib/systemd/system/kube-scheduler.service

[Unit]
Description=Kubernetes Scheduler Plugin
Documentation=https://github.com/GoogleCloudPlatform/kubernetes

[Service]
EnvironmentFile=-/etc/kubernetes/config
EnvironmentFile=-/etc/kubernetes/scheduler
ExecStart=/usr/local/bin/kube-scheduler \
          $KUBE_LOGTOSTDERR \
          $KUBE_LOG_LEVEL \
          $KUBE_MASTER \
          $KUBE_SCHEDULER_ARGS
Restart=on-failure
LimitNOFILE=65536

[Install]
WantedBy=multi-user.target
```

配置文件/etc/kubernetes/scheduler

```
[root@vlnx251101 kubernetes]# vim /etc/kubernetes/scheduler

###
# kubernetes scheduler config
# default config should be adequate
# Add your own!
KUBE_SCHEDULER_ARGS="--leader-elect=true --address=127.0.0.1"
```

- --address 值必须为127.0.0.1，因为当前 kube-apiserver 期望scheduler 和controller-manager在同一台机器

启动kube-scheduler

```
[root@vlnx251101 kubernetes]# systemctl daemon-reload ; systemctl enable kube-scheduler
; systemctl start kube-scheduler ; systemctl status kube-scheduler
```

验证master节点功能

```
[root@vlnx251101 kubernetes]# kubectl get componentstatuses --
server=192.168.251.101:8080
NAME                 STATUS    MESSAGE              ERROR
controller-manager   Healthy   ok
scheduler            Healthy   ok
etcd-0               Healthy   {"health": "true"}
etcd-2               Healthy   {"health": "true"}
etcd-1               Healthy   {"health": "true"}
```