环境说明

节点规划

节点	IP	角色	备注
k8s-master	192.168.3.94	master,harbor仓库	
k8s-node1	192.168.3.124	node	
edge-node1	192.168.3.9	edgenode	

软件版本

- 操作系统版本: cetos7.5
- 内核版本: 3.10.0-862.el7.x86_64
- 软件安装目录: /usr/local/src
- KubeEdge 版本: 0.2.1
- kubernetes 版本: 1.14.1

一、执行master节点的安装

注意,本文档中 192.168.3.94 为 k8s master节点 192.168.3.124为 k8s node 节点

环境为centos7

1.解压安装包

```
tar -zxf k8s-deploy.tar.gz
cd k8s-deploy/
chmod +x installk8s.sh
```

2.修改kubeedge.ini

```
[root@host-192-168-3-94 k8s-deploy]# cat kubeedge.ini
POD_NETWORK_CIDR=10.244.0.0/16
SERVICE_CIDR=10.2.0.0/16
APISERVER_ADVERTISE_ADDRESS=192.168.3.94
```

3.修改或者检查 `/etc/hosts`

4.清理环境

```
[root@host-192-168-3-94 k8s-deploy]# sh installk8s.sh
-----------------------------------Kubernetes Install Menu-----------------------------------------
| Choose your option                                                                              |
|                                                                                                 |
|                         1.Install K8s On Master                                                 |
|                         2.Init Env For All (OS/docker/images/kubelet)                           |
|                         3.Init Env (OS/kubelet)                                                 |
|                         4.Install Docker Only                                                   |
|                         5.Load Docker Images Only                                               |
|                         6.Install Docker And Load Docker Images                                 |
|                         7.Install Master Only                                                   |
|                         8.Uninstall K8s Config                                                  |
|                         9.Uninstall All Config                                                  |
|                         10.Exit                                                                 |
|                                                                                                 |
-------------------------------------------------------------------------------------------------
Choose your option (1-10):
```

如果以前有安装过k8s的集群,需要执行卸载环境,保证环境是干净的

5.执行安装master

```
[root@host-192-168-3-94 k8s-deploy]# sh installk8s.sh
-----------------------------------Kubernetes Install Menu-----------------------------------------
| Choose your option                                                                              |
|                                                                                                 |
|                         1.Install K8s On Master                                                 |
|                         2.Init Env For All (OS/docker/images/kubelet)                           |
|                         3.Init Env (OS/kubelet)                                                 |
|                         4.Install Docker Only                                                   |
|                         5.Load Docker Images Only                                               |
|                         6.Install Docker And Load Docker Images                                 |
|                         7.Install Master Only                                                   |
|                         8.Uninstall K8s Config                                                  |
|                         9.Uninstall All Config                                                  |
|                         10.Exit                                                                 |
|                                                                                                 |
-------------------------------------------------------------------------------------------------
Choose your option (1-10):
```

```
Choose your option (1-10):
1
*************************************************************************************************************
*   NOTE:                                                                                                   *
*        Your /etc/hosts and apiserver-address(kubeedge.ini) as shown below :                               *
*                                                                                                           *
*************************************************************************************************************
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.3.94 host-192-168-3-94
192.168.3.94 edgehub.acedge.cn
192.168.3.124  static.acedge.cn

-------------------------------------------------------------------------------------------------------------
192.168.3.94
*************************************************************************************************************
*   NOTE:                                                                                                   *
*       Please make sure your config is right !                                                             *
*                                                                                                           *
*************************************************************************************************************
Are you sure?  (y/n):
```

会提示/ect/hosts的检查和kubeedge.ini中apiserver的检查

确实安装后,就开始正式安装master节点了. 安装过程中,会做如下的操作:

- 关闭操作系统的防火墙
- 添加bridgesupport
- 关闭selinux
- 二进制方式安装docker

- 导入k8s集群需要的镜像
- 安装kubelet
- 安装kubernetes master节点
- 配置kube-config
- 安装网络插件flannel

6.安装结束的提示

很快就会安装结束,除非有问题.. 安装完成后,会有下面所示的提示:

```
**********************************************************************************************************
*   NOTE:                                                                                                *
*   Then you can join any number of worker nodes by running the following on each as root:               *
*   kubeadm join 192.168.3.94:6443 --token ezgyei.st500hn6bmneez2a  \                                    *
--discovery-token-ca-cert-hash sha256:f9f23f026ece8d4a995528a24efcc95b28eaa25ca8b0c07b2e49dee91b32d6bc   *
*                                                                                                        *
**********************************************************************************************************
```

在其他node节点,在完成初始化环境后,只需要执行以上的命令即可加入到k8s集群中去.

二、node节点的安装

1.清理环境

```
[root@host-192-168-3-94 k8s-deploy]# sh installk8s.sh
-----------------------------------Kubernetes Install Menu-----------------------------------------
| Choose your option                                                                              |
|                                                                                                 |
|                        1.Install K8s On Master                                                  |
|                        2.Init Env For All (OS/docker/images/kubelet)                            |
|                        3.Init Env (OS/kubelet)                                                  |
|                        4.Install Docker Only                                                    |
|                        5.Load Docker Images Only                                                |
|                        6.Install Docker And Load Docker Images                                  |
|                        7.Install Master Only                                                    |
|                        8.Uninstall K8s Config                                                   |
|                        9.Uninstall All Config                                                   |
|                        10.Exit                                                                  |
|                                                                                                 |
-------------------------------------------------------------------------------------------------
Choose your option (1-10):
```

如果以前有安装过k8s的集群,需要执行卸载环境,保证环境是干净的

2.配置环境

```
-----------------------------------Kubernetes Install Menu-----------------------------------------
| Choose your option                                                                              |
|                                                                                                 |
|                        1.Install K8s On Master                                                  |
|                        2.Init Env For All (OS/docker/images/kubelet)                            |
|                        3.Init Env (OS/kubelet)                                                  |
|                        4.Install Docker Only                                                    |
|                        5.Load Docker Images Only                                                |
|                        6.Install Docker And Load Docker Images                                  |
|                        7.Install Master Only                                                    |
|                        8.Uninstall K8s Config                                                   |
|                        9.Uninstall All Config                                                   |
|                        10.Exit                                                                  |
|                                                                                                 |
-------------------------------------------------------------------------------------------------
```

3.添加node到k8s集群中

```
*   NOTE:                                                                                                *
*   Then you can join any number of worker nodes by running the following on each as root:               *
*   kubeadm join 192.168.3.94:6443 --token ezgyei.st500hn6bmneez2a  \                                    *
--discovery-token-ca-cert-hash sha256:f9f23f026ece8d4a995528a24efcc95b28eaa25ca8b0c07b2e49dee91b32d6bc   *
*                                                                                                        *
```

根据在master节点最后的提示,执行命令

```
kubeadm join 192.168.3.94:6443 --token ezgyei.st500hn6bmneez2a  \
--discovery-token-ca-cert-hash
sha256:f9f23f026ece8d4a995528a24efcc95b28eaa25ca8b0c07b2e49dee91b32d6bc
```

```
[root@host-192-168-3-124 k8s-deploy]# kubeadm join 192.168.3.94:6443 --token ezgyei.st500hn6bmneez2a  \
> --discovery-token-ca-cert-hash sha256:f9f23f026ece8d4a995528a24efcc95b28eaa25ca8b0c07b2e49dee91b32d6bc
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please
kubernetes.io/docs/setup/cri/
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.14" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Activating the kubelet service
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.
```

安装完成后有如上提示:

4.如果还要添加其他节点,则需要再重复执行以上即可.

三、k8s集群检查及其他配置

1.节点状态检查

安装完master节点和node节点后,检查节点的状态

```
[root@host-192-168-3-94 k8s-deploy]# kubectl get cs
NAME                 STATUS    MESSAGE             ERROR
controller-manager   Healthy   ok
scheduler            Healthy   ok
etcd-0               Healthy   {"health":"true"}
[root@host-192-168-3-94 k8s-deploy]# kubectl get node
NAME                 STATUS   ROLES    AGE    VERSION
host-192-168-3-124   Ready    <none>   2m3s   v1.14.1
host-192-168-3-94    Ready    master   10m    v1.14.1
```

2.已经安装的pod如下:

```
[root@host-192-168-3-94 k8s-deploy]# kubectl get pod --all-namespaces
NAMESPACE     NAME                                         READY   STATUS
RESTARTS   AGE
```

```
kube-system   coredns-fb8b8dccf-42lhz                      1/1     Running   0
11m
kube-system   coredns-fb8b8dccf-lv97r                      1/1     Running   0
11m
kube-system   etcd-host-192-168-3-94                       1/1     Running   0
10m
kube-system   kube-apiserver-host-192-168-3-94             1/1     Running   0
10m
kube-system   kube-controller-manager-host-192-168-3-94    1/1     Running   0
10m
kube-system   kube-flannel-ds-amd64-6gcjj                  1/1     Running   0
11m
kube-system   kube-flannel-ds-amd64-cw8b6                  1/1     Running   0
2m42s
kube-system   kube-proxy-jxc8q                             1/1     Running   0
2m42s
kube-system   kube-proxy-mnnv5                             1/1     Running   0
11m
kube-system   kube-scheduler-host-192-168-3-94             1/1     Running   0
10m
```

3.修改apiserver的配置

```
cd /etc/kubernetes/manifests
vim kube-apiserver.yaml
```

添加:

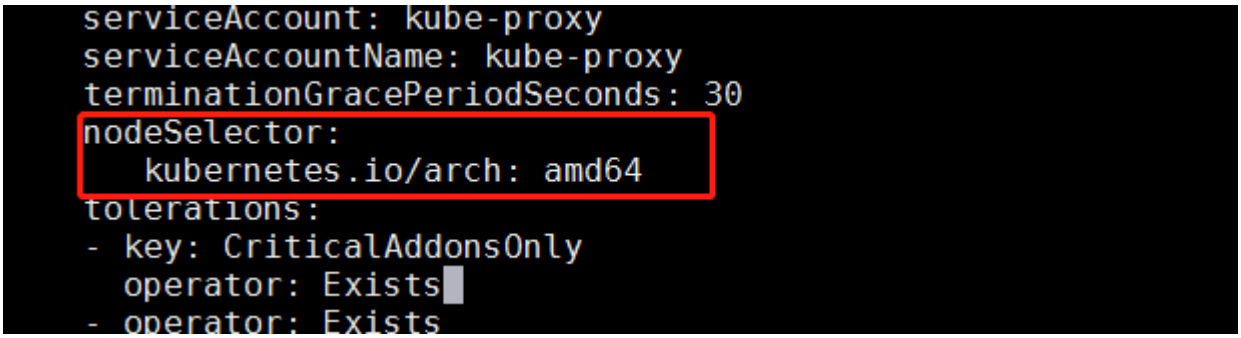
```
- --service-node-port-range=8000-40000
- --insecure-port=8080
- --insecure-bind-address=0.0.0.0
```

修改完成保存后,会自动重启

4.修改kube-proxy的配置

给kube-proxy加上nodeselector,防止边缘节点上自动启动kube-proxy

```
nodeSelector:
  kubernetes.io/arch: amd64
```



修改完成后查询

```
[root@host-192-168-3-94 manifests]# kubectl get daemonsets. -n kube-system
NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE
NODE SELECTOR                     AGE
kube-flannel-ds-amd64     2         2         2       2            2
beta.kubernetes.io/arch=amd64     22m
kube-flannel-ds-arm       0         0         0       0            0
beta.kubernetes.io/arch=arm       22m
kube-flannel-ds-arm64     0         0         0       0            0
beta.kubernetes.io/arch=arm64     22m
kube-flannel-ds-ppc64le   0         0         0       0            0
beta.kubernetes.io/arch=ppc64le   22m
kube-flannel-ds-s390x     0         0         0       0            0
beta.kubernetes.io/arch=s390x     22m
kube-proxy                2         2         2       2            2
<none>                            22m
[root@host-192-168-3-94 manifests]# kubectl edit daemonsets kube-proxy -n kube-
system
daemonset.extensions/kube-proxy edited
[root@host-192-168-3-94 manifests]#
[root@host-192-168-3-94 manifests]# kubectl get daemonsets. -n kube-system
NAME                      DESIRED   CURRENT   READY   UP-TO-DATE   AVAILABLE
NODE SELECTOR                     AGE
kube-flannel-ds-amd64     2         2         2       2            2
beta.kubernetes.io/arch=amd64     25m
kube-flannel-ds-arm       0         0         0       0            0
beta.kubernetes.io/arch=arm       25m
kube-flannel-ds-arm64     0         0         0       0            0
beta.kubernetes.io/arch=arm64     25m
kube-flannel-ds-ppc64le   0         0         0       0            0
beta.kubernetes.io/arch=ppc64le   25m
kube-flannel-ds-s390x     0         0         0       0            0
beta.kubernetes.io/arch=s390x     25m
kube-proxy                2         2         2       2            2
kubernetes.io/arch=amd64          25m
```

查看NODE SELECTOR这一列是否已经有了加上的nodeselector

5.添加用户admin/admin可以访问k8s的权限的配置

5.1添加basic-auth.csv到/etc/kubernetes/pki/下

```
[root@host-192-168-3-94 src]# cat /etc/kubernetes/pki/basic-auth.csv
admin,admin,1
```

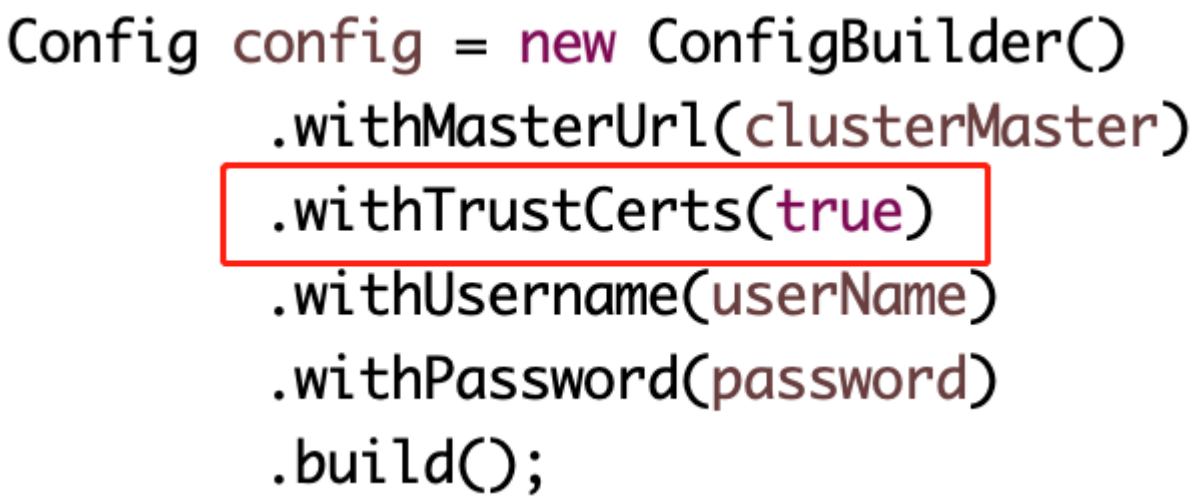
5.2 在kube-apiserver中加入如下配置

```
- --basic-auth-file=/etc/kubernetes/pki/basic-auth.csv
```

5.3 创建clusterrole与user的绑定

```
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-crb
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: cluster-admin
subjects:
-
  name: admin
  apiGroup: rbac.authorization.k8s.io
```

5.4 java代码中要注意的地方:



四、部署其他组件

1.部署harbor

1.1进入目录将docker-composer复制到/usr/bin目录下

```
# cd /usr/local/src
# chmod a+x docker-compose-Linux-x86_64
# mv docker-compose-Linux-x86_64 /usr/local/bin/docker-compose
# docker-compose -version
docker-compose version 1.24.0, build 0aa59064
```

1.2解压harbor的离线安装包

```
[root@host-192-168-3-94 src]# tar -zxf harbor-offline-installer-v1.7.5.tgz
[root@host-192-168-3-94 src]#
```

1.3进入harbor目录,修改配置文件

注意:最好把harbor对应的admin的默认密码修改了!

```
[root@host-192-168-3-94 src]# cd harbor/
[root@host-192-168-3-94 harbor]# cp harbor.cfg harbor.cfg.bak
[root@host-192-168-3-94 harbor]# vim harbor.cfg
[root@host-192-168-3-94 harbor]# diff harbor.cfg harbor.cfg.bak
8c8
< hostname = edgehub.acedge.cn:8888
---
> hostname = reg.mydomain.com
[root@host-192-168-3-94 harbor]#
```

1.4执行安装

```
[root@host-192-168-3-94 harbor]# ./install.sh

...
...
...

✓ ----Harbor has been installed and started successfully.----

Now you should be able to visit the admin portal at http://edgehub.acedge.cn:8888.
For more details, please visit https://github.com/goharbor/harbor .
```

1.5 配置harbor仓库

通过浏览器上登录harbor仓库,做如下操作:

1. 创建用户
2. 修改用户权限
3. 创建项目
4. 修改项目所有者

1.6docker的相关配置

修改/etc/docker/daemon.json添加

```
# cat /etc/docker/daemon.json
{
          "insecure-registries" :
["edgehub.acedge.cn:8888","192.168.3.XXXXXXXXX:8888"]
}
```

1.7 和k8s的结合使用

docker登录harbor的地址生成相关信息

docker先登录harbor仓库后,会在/root/.docker/config.json自动生成登录的信息,类似:

```
[root@master ~]# cat /root/.docker/config.json
{
        "auths": {
                "192.168.3.27:8888": {
                        "auth": "eWZ6eDoxcWF6QFdTWA=="
                },
                "edgehub.acedge.cn:8888": {
                        "auth": "eWZ6eDoxcWF6QFdTWA=="
                }
        },
        "HttpHeaders": {
                "User-Agent": "Docker-Client/18.09.0 (linux)"
        }
}[root@master ~]#
```

将这个密码做base64转换

```
# cat /root/.docker/config.json | base64 -w 0
ewoJImF1dGhzIjogewoJCSIxOTIuMTY4LjMuNiI6IHsKCQkJImF1dGgiOiAiWVdSdGFXNDZVM1JoY2lveU
1ERTAiCgkJfQoJfSwKCSJIdHRwSGVhZGVycyI6IHsKCQkiVXNlci1BZ2VudCI6ICJEb2NrZXItQ2xpZW50
LzE4LjA2LjEtY2UgKGxpbnV4KSIKCX0KfQ==
```

生成secret

```
apiVersion: v1
kind: Secret
metadata:
  name: harborsecret
data:
  .dockerconfigjson:
ewoJImF1dGhzIjogewoJCSIxOTIuMTY4LjMuNiI6IHsKCQkJImF1dGgiOiAiWVdSdGFXNDZVM1JoY2lveU
1ERTAiCgkJfQoJfSwKCSJIdHRwSGVhZGVycyI6IHsKCQkiVXNlci1BZ2VudCI6ICJEb2NrZXItQ2xpZW50
Lz
E4LjA2LjEtY2UgKGxpbnV4KSIKCX0KfQ==
type: kubernetes.io/dockerconfigjson
```

在deployment或者pod中配置拉取镜像的的imagePullSecrets

```
imagePullSecrets:
- name: harborsecret
```

```
spec:
  containers:
  - image: 192.168.3.6/k8spublic/nginx
    imagePullPolicy: Always
    name: mynginx
    resources:
      requests:
        cpu: 100m
    terminationMessagePath: /dev/termination-log
    terminationMessagePolicy: File
  dnsPolicy: ClusterFirst
  restartPolicy: Always
  schedulerName: default-scheduler
  securityContext: {}
  terminationGracePeriodSeconds: 30
  imagePullSecrets:
  - name: harborsecret
```

2.部署ingress

2.1创建

```
[root@host-192-168-3-94 src]# tar -zxf addons.tar.gz
[root@host-192-168-3-94 src]# cd addons/
[root@host-192-168-3-94 addons]# pwd
/usr/local/src/addons
```

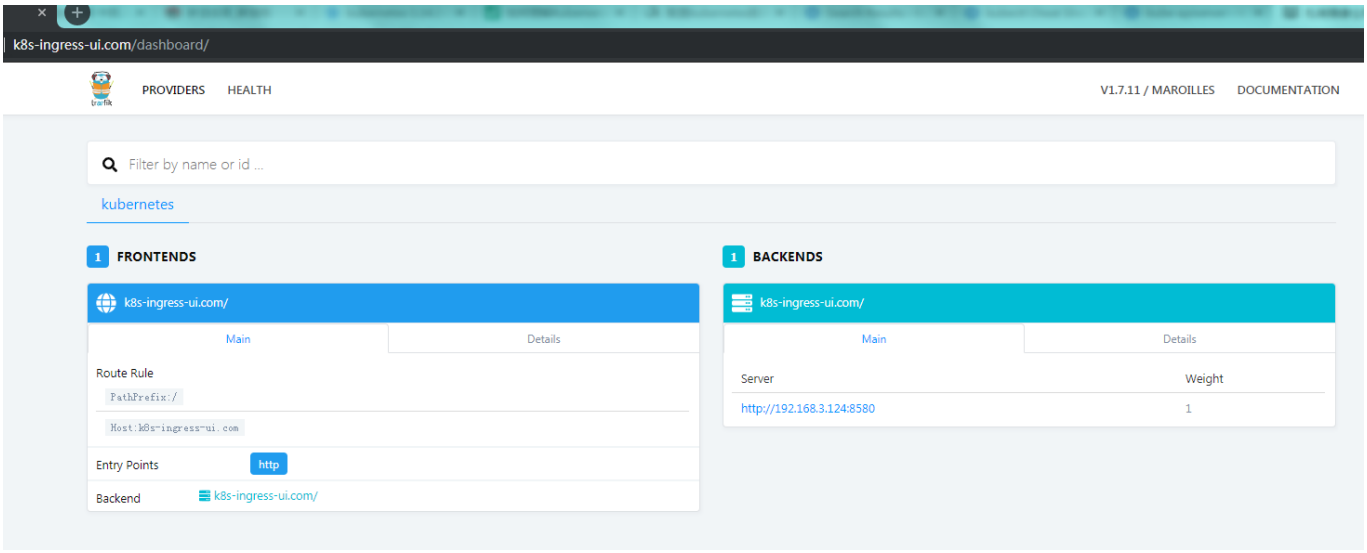
```
[root@host-192-168-3-94 addons]# ls
dl  edgecontroller  harbor_soft  ingress  storageclass
[root@host-192-168-3-94 addons]# kubectl create -f ingress/
daemonset.extensions/traefik-ingress-lb created
serviceaccount/ingress created
clusterrolebinding.rbac.authorization.k8s.io/ingress created
service/traefik-web-ui created
ingress.extensions/traefik-web-ui created
[root@host-192-168-3-94 addons]#
```

2.2验证

```
[root@host-192-168-3-94 addons]# kubectl get ingresses -n kube-system
NAME             HOSTS                 ADDRESS   PORTS   AGE
traefik-web-ui   k8s-ingress-ui.com
```

在访问的主机hosts里面添加`192.168.3.124 k8s-ingress-ui.com`

通过浏览器访问



3.部署nginx用于文件的下载

3.1创建

```
[root@host-192-168-3-94 addons]# pwd
/usr/local/src/addons
[root@host-192-168-3-94 addons]# ls
dl  edgecontroller  harbor_soft  ingress  storageclass
[root@host-192-168-3-94 addons]# kubectl create -f dl
ingress.extensions/dl-file-url created
deployment.extensions/nginx-test created
service/nginx-test created
persistentvolume/dl-url created
```

```
persistentvolumeclaim/dl-url-pvc created
[root@host-192-168-3-94 addons]#
```

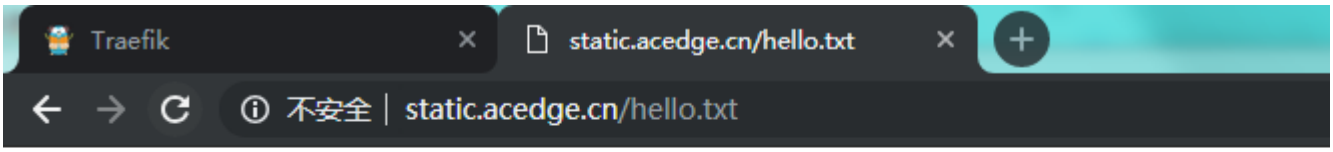
3.2验证

```
[root@host-192-168-3-94 addons]# kubectl get ingresses
NAME          HOSTS              ADDRESS   PORTS   AGE
dl-file-url   static.acedge.cn             80      30s
```

在访问的主机hosts里面添加`192.168.3.124 static.acedge.cn` 在节点的 /data4dlurl 任意添加一个文件

```
[root@host-192-168-3-124 data4dlurl]# cd /data4dlurl/
[root@host-192-168-3-124 data4dlurl]# echo "just 4 download! " > hello.txt
[root@host-192-168-3-124 data4dlurl]#
```

通过浏览器访问




```
just 4 download!
```

五、部署edgecontroller

1.进入相关目录

```
[root@host-192-168-3-94 edgecontroller]# pwd
/usr/local/src/addons/edgecontroller
[root@host-192-168-3-94 edgecontroller]# ll
总用量 64
-rw-r--r-- 1 root root   58 5月   9 21:38 01-namespace.yml
-rw-r--r-- 1 root root   91 5月   9 21:38 02-serviceaccount.yaml
-rw-r--r-- 1 root root  381 5月   9 21:38 03-clusterrole.yaml
-rw-r--r-- 1 root root  333 5月   9 21:38 04-clusterrolebinding.yaml
-rw-r--r-- 1 root root  882 5月   9 21:38 05-configmap.yaml
-rw-r--r-- 1 root root  906 5月   9 21:38 05-configmap.yaml.bak
-rw-r--r-- 1 root root 2195 5月   9 21:38 07-deployment.yaml
-rw-r--r-- 1 root root  297 5月   9 21:38 08-service.yaml
-rw-r--r-- 1 root root  258 5月   9 21:38 08-service.yaml.example
-rwxr-xr-x 1 root root 1597 5月   9 21:38 certgen.sh
```

```
-rw-r--r-- 1 root root 1140 5月   9 21:38 README.md
-rw-r--r-- 1 root root 1255 5月   9 21:38 test-nginx.yml
-rwxr-xr-x 1 root root   32 5月   9 21:38 x-02-certgen.sh
-rwxr-xr-x 1 root root   48 5月   9 21:38 x-03-create-06-secret.sh
-rwxr-xr-x 1 root root  171 5月   9 21:38 x-04-doit.sh
-rwxr-xr-x 1 root root  350 5月   9 21:38 x-05-clean.sh
```

2.修改05-configmap.yaml

修改05-configmap.yaml中的master对应的地址,其他不用改

```
[root@host-192-168-3-94 edgecontroller]# vi 05-configmap.yaml
[root@host-192-168-3-94 edgecontroller]# cat 05-configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: edgecontroller
  namespace: kubeedge
  labels:
    k8s-app: kubeedge
    kubeedge: edgecontroller
data:
  controller.yaml: |
    controller:
      kube:
        master: http://192.168.3.94:8080
        kubeconfig: /etc/kubeedge/cloud/kubeconfig.yaml
        namespace: ""
        content_type: "application/vnd.kubernetes.protobuf"
        qps: 5
        burst: 10
        node_update_frequency: 10
    cloudhub:
      address: 0.0.0.0
      port: 10000
      ca: /etc/kubeedge/certs/ca.crt
      cert: /etc/kubeedge/certs/cloud.crt
      key: /etc/kubeedge/certs/cloud.key
      keepalive-interval: 30
      write-timeout: 30
      node-limit: 10
  logging.yaml: |
    loggerLevel: "INFO"
    enableRsyslog: false
    logFormatText: true
    writers: [stdout]
  modules.yaml: |
    modules:
      enabled: [controller, cloudhub]
```

3.依次执行:

1. `x-02-certgen.sh` ：用于生产密钥.路径在`/etc/kubeedge/ca`和`/etc/kubeedge/certs`

2. x-03-create-06-secret.sh ：用于生成edgecontroller的secret.yaml
3. x-04-doit.sh ：用于创建kubeedge对应的各种资源

3.1 执行x-02-certgen.sh

```
[root@host-192-168-3-94 edgecontroller]# sh x-02-certgen.sh
Generating RSA private key, 2048 bit long modulus
.........................+++
.....................................................................................
.....................................................+++
e is 65537 (0x10001)
Signature ok
subject=/C=CN/ST=Sichuan/L=Chengdu/O=KubeEdge/CN=kubeedge.io
Getting CA Private Key
[root@host-192-168-3-94 edgecontroller]#
[root@host-192-168-3-94 edgecontroller]# ls -l /etc/kubeedge/ca
-rw-r--r-- 1 root root 1976 5月   8 20:14 ca.crt
-rw-r--r-- 1 root root 3311 5月   8 20:14 ca.key
-rw-r--r-- 1 root root   17 5月  10 11:39 ca.srl
[root@host-192-168-3-94 edgecontroller]# ls -l /etc/kubeedge/certs/
-rw-r--r-- 1 root root 1513 5月  10 11:39 edge.crt
-rw-r--r-- 1 root root  985 5月  10 11:39 edge.csr
-rw-r--r-- 1 root root 1675 5月  10 11:39 edge.key
```

3.2 执行x-03-create-06-secret.sh

```
[root@host-192-168-3-94 edgecontroller]# sh x-03-create-06-secret.sh
apiVersion: v1
kind: Secret
metadata:
  name: edgecontroller
  namespace: kubeedge
  labels:
    k8s-app: kubeedge
    kubeedge: edgecontroller
stringData:
  ca.crt: |
    -----BEGIN CERTIFICATE-----
    xxxxxxpapapa
    -----END CERTIFICATE-----
  cloud.crt: |
    -----BEGIN CERTIFICATE-----
    xxxxxxpapapa
    -----END CERTIFICATE-----
  cloud.key: |
    -----BEGIN RSA PRIVATE KEY-----
      xxxxxxpapapa
    -----END RSA PRIVATE KEY-----
```

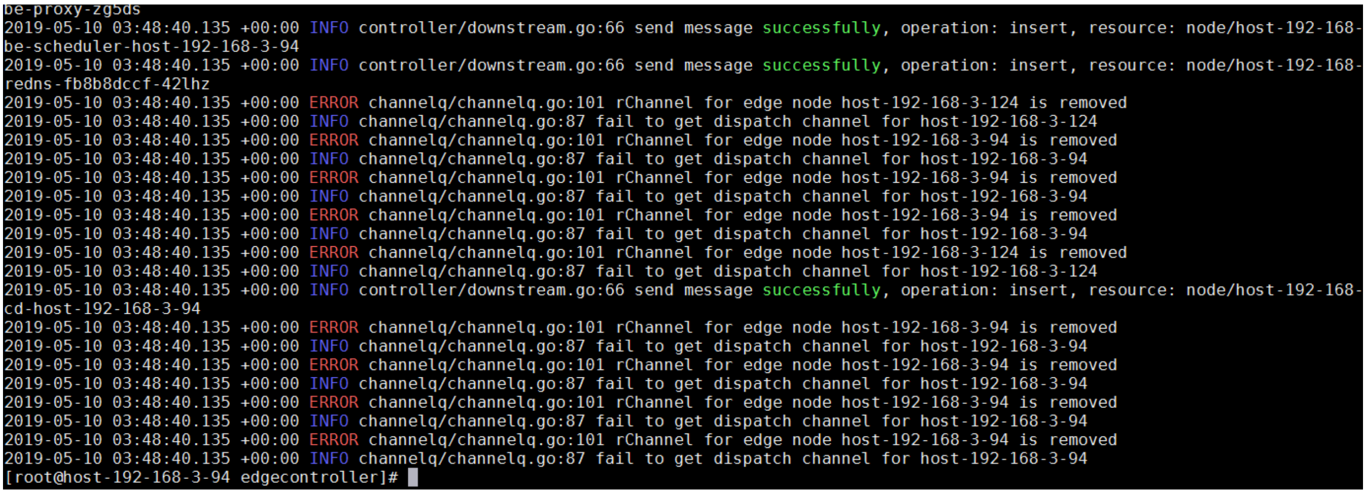
执行完这个会生成一个文件 `06-secret.yaml`

3.3 执行`x-04-doit.sh`

```
[root@host-192-168-3-94 edgecontroller]# sh x-04-doit.sh
namespace/kubeedge created
serviceaccount/edgecontroller created
clusterrole.rbac.authorization.k8s.io/edgecontroller created
clusterrolebinding.rbac.authorization.k8s.io/edgecontroller created
configmap/edgecontroller created
secret/edgecontroller created
deployment.apps/edgecontroller created
service/edgecontroller created
```

4.检查edgecontroller pod的状态

```
[root@host-192-168-3-94 edgecontroller]# kubectl get pod -n kubeedge
NAME                              READY   STATUS    RESTARTS   AGE
edgecontroller-7c894ddf45-zrvg2   1/1     Running   0          85s
[root@host-192-168-3-94 edgecontroller]# kubectl logs edgecontroller-7c894ddf45-
zrvg2  -n kubeedge
```



没有出现网络的 i/o time out的错误就说明对了.

用telnet也可以检查

```
[root@host-192-168-3-94 src]# telnet 192.168.3.124 10000
Trying 192.168.3.124...
Connected to 192.168.3.124.
Escape character is '^]'.
```

六.边缘节点的配置(测试用)

在这以 192.168.3.9这个服务器作为edgenode作为示例

1.在k8s master创建node

```
[root@host-192-168-3-94 src]# cat node.json
{
  "kind": "Node",
  "apiVersion": "v1",
  "metadata": {
    "name": "host-192-168-3-9",
    "labels": {
      "name": "edge-node"
    }
  }
}
```

2.在边缘节点上拉取edge_code和conf和拉取密钥

将master节点上通过`sh x-02-certgen.sh`生成的密钥拉取过来即可

```
[root@host-192-168-3-94 edgecontroller]# ls -l /etc/kubeedge/certs/
-rw-r--r-- 1 root root 1513 5月  10 11:39 edge.crt
-rw-r--r-- 1 root root  985 5月  10 11:39 edge.csr
-rw-r--r-- 1 root root 1675 5月  10 11:39 edge.key
```

最终准备的文件如下:

```
[root@host-192-168-3-9 running]# pwd
/root/running
[root@host-192-168-3-9 running]# ll
总用量 98560
drwxr-xr-x 2 root root        84 5月  10 12:16 conf
-rwxr-xr-x 1 root root 100910464 5月  10 11:56 edge_core
-rw-r--r-- 1 root root      1513 5月  10 12:17 edge.crt
-rw-r--r-- 1 root root       985 5月  10 12:17 edge.csr
-rw-r--r-- 1 root root      1675 5月  10 12:17 edge.key
```

3.修改 conf/edge.yaml

需要修改的地方已经用框标识

示例:

```
[root@host-192-168-3-9 conf]# diff edge.yaml edge.yaml.bak
11,13c11,13
<         url: wss://192.168.3.124:10000/e632aba927ea4ac2b575ec1603d56f10/host-
192-168-3-9/events
<         certfile: /root/running/edge.crt
<         keyfile: /root/running/edge.key
---
>         url: wss://192.168.3.140:10000/e632aba927ea4ac2b575ec1603d56f10/host-
192-168-3-107/events
>         certfile: /etc/kubeedge/edge/certs/edge.crt
>         keyfile: /etc/kubeedge/edge/certs/edge.key
24c24
<         node-id: host-192-168-3-9
---
>         node-id: host-192-168-3-107
28c28
<     hostname-override: host-192-168-3-9
---
>     hostname-override: host-192-168-3-107
```

```
mqtt:
    server: tcp://127.0.0.1:1883 # external mqtt broker url.
    internal-server: tcp://127.0.0.1:1884 # internal mqtt broker url.
    mode: 0 # 0: internal mqtt broker enable only. 1: internal and external mqtt broker enable. 2: external mqtt broker enable only.
    qos: 0 # 0: QOSAtMostOnce, 1: QOSAtLeastOnce, 2: QOSExactlyOnce.
    retain: false # if the flag set true, server will store the message and can be delivered to future subscribers.
    session-queue-size: 100 # A size of how many sessions will be handled. default to 100.

edgehub:
    websocket:
        url: wss://192.168.3.124:10000/e632aba927ea4ac2b575ec1603d56f10/host-192-168-3-9/events
        certfile: /root/running/edge.crt
        keyfile: /root/running/edge.key
        handshake-timeout: 30 #second
        write-deadline: 15 # second
        read-deadline: 15 # second
    controller:
        placement: false
        heartbeat: 15  # second
        refresh-ak-sk-interval: 10 # minute
        auth-info-files-path: /var/IEF/secret
        placement-url: https://10.154.193.32:7444/v1/placement_external/message_queue
        project-id: e632aba927ea4ac2b575ec1603d56f10
        node-id: host-192-168-3-9

edged:
    register-node-namespace: default
    hostname-override: host-192-168-3-9
    interface-name: eth0
    node-status-update-frequency: 10 # second
    device-plugin-enabled: false
    gpu-plugin-enabled: false
    image-gc-high-threshold: 80 # percent
    image-gc-low-threshold: 40 # percent
    maximum-dead-containers-per-container: 1
    version: 2.0.0
```

4.边缘节点后台运行edge_core

```
nohup ./edge_core &
```

注意,在生产环境建议修改log的等级,或者重定向日志的输出到 /dev/null

5.在master查看节点状态

```
[root@host-192-168-3-94 tmp]# kubectl get node
NAME                  STATUS   ROLES    AGE    VERSION
host-192-168-3-124    Ready    <none>   116m   v1.14.1
host-192-168-3-9      Ready    <none>   10s    2.0.0
host-192-168-3-94     Ready    master   125m   v1.14.1
[root@host-192-168-3-94 tmp]#
```

七、注意事项

1.docker根目录的位置与大小

默认的目录为`/var/lib/docker`,这个路径在根目录上,可以考虑先创建一个lvm来挂载这个目录,方便今后的扩容

```
systemctl stop docker
cd /var/lib
cp -rf docker docker.bak
cp -rf docker /xxx/
rm -rf docker
```

```
ln -s /xxx/docker docker
systemctl start docker
docker info
```

2.harbor的默认安装目录

harbor的默认安装目录为/data 可以考虑先创建一个lvm来挂载这个目录,方便今后的扩容

3.kubectl 自动补全

```
yum install -y bash-completion
source /usr/share/bash-completion/bash_completion
source <(kubectl completion bash)
```