安装部署（二）

~~做到这里发现坑了 阿里云的机器无法使用nat表~~
不过！ 用vxlan模型可以搞定（普天同庆）

flannel

flannel的三种网络模型

- host-gw模型：所有node ip必须在同一个物理网关设备下才能使用 它的原理就是：给宿主机添加一个静态路由，指明到达pod之前要经过的宿主机
- Vxlan模型：
- 直接路由模型：当node不在同一个物理网关下，走vxaln模型,在同一个网关下，走host-gw模型

host-gw模型

下载二进制包

Plain Text 复制代码

```
[root@alice002 ~]# cd /opt/src/
[root@alice002 src]# wget "https://github.com/coreos/flannel/releases/download/v0.11.0/flannel-v0.11.0-linux-amd64.tar.gz"
[root@alice002 opt]# mkdir flannel-v0.11.0
[root@alice002 src]# tar xf flannel-v0.11.0-linux-amd64.tar.gz -C /opt/flannel-v0.11.0
[root@alice002 src]# cd /opt/
[root@alice002 opt]# ln -s /opt/flannel-v0.11.0  flannel
```

拷贝证书到flannel目录

Plain Text 复制代码

```
[root@alice002 opt]# cd flannel
[root@alice002 flannel]# mkdir cert
[root@alice002 flannel]# cd cert/
[root@alice002 cert]# scp alice001:/opt/certs/ca.pem .
[root@alice002 cert]# scp alice001:/opt/certs/client.pem .
[root@alice002 cert]# scp alice001:/opt/certs/client-key.pem .
[root@alice002 cert]# ll
total 12
-rw-r--r-- 1 root root 1346 Feb  8 20:49 ca.pem
-rw------- 1 root root 1679 Feb  8 20:50 client-key.pem
-rw-r--r-- 1 root root 1363 Feb  8 20:50 client.pem
```

配置文件&启动脚本

Plain Text 复制代码

```
[root@alice002 cert]# cd ..
[root@alice002 flannel]# vim subnet.env
[root@alice002 flannel]# cat subnet.env # 注意网段要修改
FLANNEL_NETWORK=172.187.0.0/16
FLANNEL_SUBNET=172.187.173.1/24
FLANNEL_MTU=1500
FLANNEL_IPMASQ=false
[root@alice002 flannel]# vi flanneld.sh
[root@alice002 flannel]# cat flanneld.sh #注意IP和网卡名称要修改
#!/bin/sh
./flanneld
  --public-ip=172.23.187.173 \
  --etcd-endpoints=https://172.23.187.175:2379,https://172.23.187.173:2379,https://172.23.187.174:2379 \
  --etcd-keyfile=./cert/client-key.pem
  --etcd-certfile=./cert/client.pem
  --etcd-cafile=./cert/ca.pem
  --iface=eth0 \
  --subnet-file=./subnet.env
  --healthz-port=2401
[root@alice002 flannel]# chmod u+x flanneld.sh
```

etcd增加host-gw模型

Plain Text 复制代码

```
这里是etcd集群所以在任意一台操作即可
[root@alice002 flannel]# cd /opt/etcd/
[root@alice002 etcd]# ./etcdctl set /coreos.com/network/config '{"Network": "172.187.0.0/16", "Backend": {"Type": "host-gw"}}'
{"Network": "172.187.0.0/16", "Backend": {"Type": "host-gw"}}
[root@alice002 etcd]# ./etcdctl get /coreos.com/network/config
{"Network": "172.187.0.0/16", "Backend": {"Type": "host-gw"}}
```

启动flannel

Plain Text 复制代码

```

[root@alice002 etcd]# vim /etc/supervisord.d/flannel.ini
[root@alice002 etcd]# cat /etc/supervisord.d/flannel.ini
[program:flanneld-alice002]
command=/opt/flannel/flanneld.sh                             ; the program (relative uses PATH, can take args)
numprocs=1                                                   ; number of processes copies to start (def 1)
directory=/opt/flannel                                       ; directory to cwd to before exec (def no cwd)
autostart=true                                               ; start at supervisord start (default: true)
autorestart=true                                             ; retstart at unexpected quit (default: true)
startsecs=30                                                 ; number of secs prog must stay running (def. 1)
startretries=3                                               ; max # of serial start failures (default 3)
exitcodes=0,2                                                ; 'expected' exit codes for process (default 0,2)
stopsignal=QUIT                                              ; signal used to kill process (default TERM)
stopwaitsecs=10                                              ; max num secs to wait b4 SIGKILL (default 10)
user=root                                                    ; setuid to this UNIX account to run the program
redirect_stderr=true                                         ; redirect proc stderr to stdout (default false)
stdout_logfile=/data/logs/flanneld/flanneld.stdout.log       ; stderr log path, NONE for none; default AUTO
stdout_logfile_maxbytes=64MB                                 ; max # logfile bytes b4 rotation (default 50MB)
stdout_logfile_backups=4                                     ; # of stdout logfile backups (default 10)
stdout_capture_maxbytes=1MB                                  ; number of bytes in 'capturemode' (default 0)
stdout_events_enabled=false                                  ; emit events on stdout writes (default false)
[root@alice002 etcd]# mkdir -p /data/logs/flanneld/
[root@alice002 etcd]# supervisorctl update
[root@alice002 etcd]# supervisorctl  status
etcd-server-alice002              RUNNING   pid 13813, uptime 9 days, 5:03:14
flanneld-alice002                 RUNNING   pid 10259, uptime 0:00:37
kube-apiserver-alice002           RUNNING   pid 14745, uptime 9 days, 2:03:16
kube-controller-manager-alice002  RUNNING   pid 14731, uptime 9 days, 2:05:00
kube-kubelet-alice002             RUNNING   pid 14901, uptime 9 days, 0:51:26
kube-proxy-alice002               RUNNING   pid 26020, uptime 8 days, 11:40:05
kube-scheduler-alice002           RUNNING   pid 14344, uptime 9 days, 2:35:35
```

查看路由

```
[root@alice002 flannel]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.23.255.253  0.0.0.0         UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
172.23.0.0      0.0.0.0         255.255.0.0     U     0      0        0 eth0
172.187.173.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
172.187.174.0   172.23.187.174  255.255.255.0   UG    0      0        0 eth0
# 要取到172.187.174.0/24 网络的包网关地址是172.23.187.174
```

然而当你设置完发现 还是不通
这是因为阿里云不支持iptables的nat表
所以需要用到VxLAN模型

VxLAN模型

停止进程

```
[root@alice002 flannel]# supervisorctl status
etcd-server-alice002              RUNNING   pid 13813, uptime 11 days, 4:31:41
flanneld-alice002                 RUNNING   pid 30447, uptime 23:07:38
kube-apiserver-alice002           RUNNING   pid 14745, uptime 11 days, 1:31:43
kube-controller-manager-alice002  RUNNING   pid 14731, uptime 11 days, 1:33:27
kube-kubelet-alice002             RUNNING   pid 14901, uptime 11 days, 0:19:53
kube-proxy-alice002               RUNNING   pid 26020, uptime 10 days, 11:08:32
kube-scheduler-alice002           RUNNING   pid 14344, uptime 11 days, 2:04:02
[root@alice002 flannel]# supervisorctl stop flanneld-alice002
[root@alice002 flannel]# ps -ef |grep flannel
root      5571 29646  0 20:50 pts/0    00:00:00 grep --color=auto flannel
root     30448     1  0 Feb09 ?        00:00:11 ./flanneld --public-ip=172.23.187.173 --etcd-endpoints=https://172.23.187.175:2379,https://172.23.187.173:2379,https://172.23.187.174:2379 --etcd-keyfile=./cert/client-key.pem --etcd-certfile=./cert/client.pem --etcd-cafile=./cert/ca.pem --iface=eth0 --subnet-file=./subnet.env --healthz-port=2401
[root@alice002 flannel]# bash
[root@alice002 flannel]# exit
[root@alice002 flannel]# kill 30448
[root@alice002 flannel]# kill 30448
-bash: kill: (30448) - No such process
[root@alice002 flannel]# ps -ef |grep flannel
root      5695 29646  0 20:51 pts/0    00:00:00 grep --color=auto flannel
```

删除路由

```
[root@alice002 flannel]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.23.255.253  0.0.0.0         UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
172.23.0.0      0.0.0.0         255.255.0.0     U     0      0        0 eth0
172.187.173.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
172.187.174.0   172.23.187.174  255.255.255.0   UG    0      0        0 eth0
[root@alice002 flannel]# route del -net 172.187.174.0/24 gw 172.23.187.174
[root@alice002 flannel]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.23.255.253  0.0.0.0         UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
172.23.0.0      0.0.0.0         255.255.0.0     U     0      0        0 eth0
172.187.173.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

删除设置vxlan模型

```
[root@alice001 ~]# etcdctl ls coreos.com/network/config/
/coreos.com/network/config
[root@alice001 ~]# etcdctl rm /coreos.com/network/config
PrevNode.Value: {"Network": "172.187.0.0/16", "Backend": {"Type": "host-gw"}}

[root@alice001 ~]# etcdctl ls coreos.com/network/subnets/
/coreos.com/network/subnets/172.187.174.0-24
/coreos.com/network/subnets/172.187.173.0-24
[root@alice001 ~]# etcdctl rm /coreos.com/network/subnets/172.187.174.0-24
Error:  x509: certificate signed by unknown authority
[root@alice001 ~]# etcdctl rm /coreos.com/network/subnets/172.187.174.0-24
PrevNode.Value: {"PublicIP":"172.23.187.174","BackendType":"host-gw"}
[root@alice001 ~]# etcdctl rm /coreos.com/network/subnets/172.187.173.0-24
PrevNode.Value: {"PublicIP":"172.23.187.173","BackendType":"host-gw"}
[root@alice001 ~]# etcdctl ls coreos.com/network/subnets/


[root@alice001 ~]# etcdctl set /coreos.com/network/config '{"Network": "172.187.0.0/16", "Backend": {"Type": "VxLAN"}}'
{"Network": "172.187.0.0/16", "Backend": {"Type": "VxLAN"}}
[root@alice001 ~]# etcdctl get /coreos.com/network/config
{"Network": "172.187.0.0/16", "Backend": {"Type": "VxLAN"}}
```

重新启动flannel

```
[root@alice002 flannel]# supervisorctl start flanneld-alice002
flanneld-alice002: started
[root@alice002 flannel]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         172.23.255.253  0.0.0.0         UG    0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
172.23.0.0      0.0.0.0         255.255.0.0     U     0      0        0 eth0
172.187.173.0   0.0.0.0         255.255.255.0   U     0      0        0 docker0
172.187.174.0   172.187.174.0   255.255.255.0   UG    0      0        0 flannel.1
[root@alice002 flannel]#
[root@alice002 flannel]# kubectl get pod -o wide
NAME             READY   STATUS    RESTARTS   AGE     IP              NODE                NOMINATED NODE   READINESS GATES
nginx-ds-2wgnl   1/1     Running   0          4h51m   172.187.173.2   alice002.host.com   <none>           <none>
nginx-ds-tph79   1/1     Running   0          4h50m   172.187.174.2   alice003.host.com   <none>           <none>
[root@alice002 flannel]# ping 172.187.174.2
PING 172.187.174.2 (172.187.174.2) 56(84) bytes of data.
64 bytes from 172.187.174.2: icmp_seq=1 ttl=63 time=0.339 ms
64 bytes from 172.187.174.2: icmp_seq=2 ttl=63 time=0.296 ms
64 bytes from 172.187.174.2: icmp_seq=3 ttl=63 time=0.308 ms
^C
--- 172.187.174.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 0.296/0.314/0.339/0.023 ms
[root@alice002 flannel]#
```

直接路由模型

```
etcdctl set /coreos.com/network/config '{"Network": "172.7.0.0/16", "Backend": {"Type": "VxLAN","Directrouting": true}}'
```

snat优化

```
 ~]# docker run -it --rm nginx:1.7.9 bash
root@ad83cb1da6f6:/# tee /etc/apt/sources.list << EOF
> deb http://mirrors.163.com/debian/ jessie main non-free contrib
> deb http://mirrors.163.com/debian/ jessie-updates main non-free contrib
> EOF
deb http://mirrors.163.com/debian/ jessie main non-free contrib
deb http://mirrors.163.com/debian/ jessie-updates main non-free contrib
root@ad83cb1da6f6:/# apt-get update && apt-get install curl -y
root@ad83cb1da6f6:/# curl -k https://www.baidu.com
```

Plain Text 复制代码

```
[root@alice001 ~]# docker  ps
CONTAINER ID    IMAGE        COMMAND    CREATED        STATUS       PORTS                        NAMES
ad83cb1da6f6    nginx:1.7.9 "bash"     4 minutes ago  Up 4 minutes  80/tcp, 443/tcp             affectionate_shtern
[root@alice001 ~]# docker commit -p ad83cb1da6f6 harbor.od.com/public/nginx:curl
sha256:a3cead594cb0674fc8217f9274066d634e07ed85eebed1e754d17d39a819e9d3
[root@alice001 ~]# docker push harbor.od.com/public/nginx:curl
```

Plain Text 复制代码

```
~]# yum install iptables-services -y
[root@alice002 ~]# systemctl start iptables
[root@alice002 ~]# systemctl enable iptables
Created symlink from /etc/systemd/system/basic.target.wants/iptables.service to /usr/lib/systemd/system/iptables.service.
[root@alice002 ~]# iptables-save | grep -i postrouting
:POSTROUTING ACCEPT [8:486]
:KUBE-POSTROUTING - [0:0]
-A POSTROUTING -m comment --comment "kubernetes postrouting rules" -j KUBE-POSTROUTING
-A POSTROUTING -s 172.187.173.0/24 ! -o docker0 -j MASQUERADE
-A KUBE-POSTROUTING -m comment --comment "kubernetes service traffic requiring SNAT" -m mark --mark 0x4000/0x4000 -j MASQUERADE
[root@alice002 ~]# iptables -t nat -D POSTROUTING -s 172.187.173.0/24 ! -o docker0 -j MASQUERADE
[root@alice002 ~]# iptables -t nat -I POSTROUTING -s 172.187.173.0/24 ! -d 172.187.0.0/16 ! -o docker0 -j MASQUERADE
[root@alice002 ~]# iptables-save | grep -i postrouting
:POSTROUTING ACCEPT [2:120]
:KUBE-POSTROUTING - [0:0]
-A POSTROUTING -s 172.187.173.0/24 ! -d 172.187.0.0/16 ! -o docker0 -j MASQUERADE
-A POSTROUTING -m comment --comment "kubernetes postrouting rules" -j KUBE-POSTROUTING
-A KUBE-POSTROUTING -m comment --comment "kubernetes service traffic requiring SNAT" -m mark --mark 0x4000/0x4000 -j MASQUERADE
[root@alice002 ~]# iptables-save | grep -i reject
-A INPUT -j REJECT --reject-with icmp-host-prohibited
-A FORWARD -j REJECT --reject-with icmp-host-prohibited
[root@alice002 ~]# iptables -t filter -D INPUT -j REJECT --reject-with icmp-host-prohibited
[root@alice002 ~]# iptables -t filter -D FORWARD -j REJECT --reject-with icmp-host-prohibited
[root@alice002 ~]# iptables-save > /etc/sysconfig/iptables
[root@alice002 ~]# service iptables save
iptables: Saving firewall rules to /etc/sysconfig/iptables:[  OK  ]
[root@alice002 ~]#
```

注意docker重启后操作

docker服务重启后,会再次增加该规则,要注意在每次重启docker服务后,删除该规则

验证:

修改后会影响到docker原本的iptables链的规则，所以需要重启docker服务

Plain Text 复制代码

```
[root@hdss7-21 ~]# systemctl restart docker
[root@hdss7-21 ~]# iptables-save |grep -i postrouting|grep docker0
-A POSTROUTING -s 172.7.21.0/24 ! -o docker0 -j MASQUERADE
-A POSTROUTING -s 172.7.21.0/24 ! -d 172.7.0.0/16 ! -o docker0 -j MASQUERADE
# 可以用iptables-restore重新应用iptables规则,也可以直接再删
[root@hdss7-21 ~]# iptables-restore /etc/sysconfig/iptables
[root@hdss7-21 ~]# iptables-save |grep -i postrouting|grep docker0
-A POSTROUTING -s 172.7.21.0/24 ! -d 172.7.0.0/16 ! -o docker0 -j MASQUERADE
```

Plain Text 复制代码

```
[root@alice002 ~]# kubectl get pod -o wide
NAME             READY   STATUS    RESTARTS   AGE   IP              NODE                NOMINATED NODE   READINESS GATES
nginx-ds-v96dl   1/1     Running   0          21h   172.187.174.2   alice003.host.com   <none>           <none>
nginx-ds-wn25q   1/1     Running   0          21h   172.187.173.2   alice002.host.com   <none>           <none>
[root@alice002 ~]# kubectl exec -it nginx-ds-wn25q bash
root@nginx-ds-wn25q:/# curl 172.187.174.2
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
root@nginx-ds-wn25q:/#
[root@alice002 ~]# curl 172.187.174.2 宿主机也curl一下


[root@alice003 ~]# kubectl logs -f nginx-ds-v96dl
172.187.173.0 - - [19/Feb/2021:12:37:57 +0000] "GET / HTTP/1.1" 200 612 "-" "curl/7.38.0" "-"



---
172.187.173.2 - - [19/Feb/2021:13:02:26 +0000] "GET / HTTP/1.1" 200 612 "-" "curl/7.38.0" "-"
172.187.173.0 - - [19/Feb/2021:13:03:36 +0000] "GET / HTTP/1.1" 200 612 "-" "curl/7.29.0" "-"
```

coredns

创建在线yaml

这里我为了可以公网访问 顺便把真实域名解析了一下

以下内容中的grep.pro都是可以真实访问的

后续yaml以及dockerfile地址http://k8s-yaml.grep.pro <http://k8s-yaml.grep.pro/>

Plain Text 复制代码

```
[root@alice001 ~]# vim /etc/nginx/conf.d/k8s-yaml.od.com.conf
[root@alice001 ~]# cat /etc/nginx/conf.d/k8s-yaml.od.com.conf
server {
    listen       80;
    server_name  k8s-yaml.od.com k8s-yaml.grep.pro;

    location / {
        autoindex on;
        default_type text/plain;
        root /data/k8s-yaml;
    }
}

[root@alice001 ~]# vim /var/named/od.com.zone
[root@alice001 coredns]# cat /var/named/od.com.zone
$ORIGIN od.com.
$TTL 600    ; 10 minutes
@     IN SOA  dns.od.com. dnsadmin.od.com. (
        2021012906 ; serial
        10800      ; refresh (3 hours)
        900        ; retry (15 minutes)
        604800     ; expire (1 week)
        86400      ; minimum (1 day)
        )
        NS   dns.od.com.
$TTL 60 ; 1 minute
dns                A    47.243.20.250
harbor             A    47.243.20.250
k8s-yaml     A    47.243.20.250
[root@alice001 ~]# systemctl restart named
[root@alice001 ~]# dig -t A k8s-yaml.od.com @172.23.187.175 +short
47.243.20.250
[root@alice001 ~]# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[root@alice001 ~]# systemctl restart nginx
[root@alice001 ~]# docker pull coredns/coredns:1.6.1
1.6.1: Pulling from coredns/coredns
c6568d217a00: Pull complete
d7ef34146932: Pull complete
Digest: sha256:9ae3b6fcac4ee821362277de6bd8fd2236fa7d3e19af2ef0406d80b595620a7a
Status: Downloaded newer image for coredns/coredns:1.6.1
[root@alice001 ~]# docker images |grep coredns
coredns/coredns                 1.6.1                       c0f6e815079e        18 months ago       42.2MB
[root@alice001 ~]# docker tag c0f6e815079e harbor.od.com/public/coredns:v1.6.1
[root@alice001 ~]# docker push !$
docker push harbor.od.com/public/coredns:v1.6.1
The push refers to repository [harbor.od.com/public/coredns]
da1ec456edc8: Pushed
225df95e717c: Pushed
v1.6.1: digest: sha256:c7bf0ce4123212c87db74050d4cbab77d8f7e0b49c041e894a35ef15827cf938 size: 739
[root@alice001 ~]# cd /data/
[root@alice001 data]# ls
docker  etcd  harbor  logs
[root@alice001 data]# mkdir /data/k8s-yaml
[root@alice001 data]# cd k8s-yaml/
[root@alice001 k8s-yaml]# cat /etc/nginx/conf.d/k8s-yaml.od.com.conf
server {
    listen       80;
    server_name  k8s-yaml.od.com;

    location / {
        autoindex on;
        default_type text/plain;
        root /data/k8s-yaml;
    }
}
[root@alice001 k8s-yaml]#
[root@alice001 k8s-yaml]# mkdir coredns
```

cm.yaml中 forward . 172.23.187.175为 集群的dns地址也是coredns的上游dns地址

svc.yaml中 clusterIP: 192.168.0.2地址为集群中kubelet定义好的集群dns地址

其余的没动 还没弄懂 有同学懂了可以在这里评论一下

Plain Text 复制代码

```
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/coredns/rbac.yaml
serviceaccount/coredns created
clusterrole.rbac.authorization.k8s.io/system:coredns created
clusterrolebinding.rbac.authorization.k8s.io/system:coredns created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/coredns/cm.yaml
configmap/coredns created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/coredns/dp.yaml
deployment.apps/coredns created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/coredns/svc.yaml
service/coredns created
```

验证

Plain Text 复制代码

```
[root@alice002 ~]# dig -t A www.baidu.com @172.23.187.175 +short
www.a.shifen.com.
www.wshifen.com.
104.193.88.123
104.193.88.77
[root@alice002 ~]# dig -t A alice001.host.com  @172.23.187.175 +short
47.243.20.250
[root@alice002 ~]# dig -t A alice001.host.com  @192.168.0.2 +short
47.243.20.250
[root@alice002 ~]# dig -t A www.baidu.com @192.168.0.2 +short
www.a.shifen.com.
www.wshifen.com.
104.193.88.123
104.193.88.77


[root@alice002 ~]# kubectl create deployment nginx-web --image=harbor.od.com/public/nginx:curl_ps
deployment.apps/nginx-web created
[root@alice002 ~]# kubectl expose deployment nginx-web --port=80 --target-port=80
service/nginx-web exposed
[root@alice002 ~]# kubectl get svc
NAME         TYPE        CLUSTER-IP        EXTERNAL-IP   PORT(S)   AGE
kubernetes   ClusterIP   192.168.0.1       <none>        443/TCP   20d
nginx-web    ClusterIP   192.168.208.221   <none>        80/TCP    4s
[root@alice002 ~]# kubectl get svc -n kube-system
NAME      TYPE        CLUSTER-IP    EXTERNAL-IP   PORT(S)                  AGE
coredns   ClusterIP   192.168.0.2   <none>        53/UDP,53/TCP,9153/TCP   14m
[root@alice002 ~]# dig -t A nginx-web.default.svc.cluster.local @192.168.0.2 +short
192.168.208.221
```

traefik

Plain Text 复制代码

```
[root@alice001 coredns]# cd /data/k8s-yaml/
[root@alice001 k8s-yaml]# mkdir traefik
[root@alice001 k8s-yaml]# cd traefik/
[root@alice001 traefik]# docker pull traefik:v1.7.2-alpine
v1.7.2-alpine: Pulling from library/traefik
4fe2ade4980c: Pull complete
8d9593d002f4: Pull complete
5d09ab10efbd: Pull complete
37b796c58adc: Pull complete
Digest: sha256:cf30141936f73599e1a46355592d08c88d74bd291f05104fe11a8bcce447c044
Status: Downloaded newer image for traefik:v1.7.2-alpine
[root@alice001 traefik]# docker tag traefik:v1.7.2-alpine harbor.od.com/public/traefik:v1.7.2
[root@alice001 traefik]# docker push !$
docker push harbor.od.com/public/traefik:v1.7.2
The push refers to repository [harbor.od.com/public/traefik]
a02beb48577f: Pushed
ca22117205f4: Pushed
3563c211d861: Pushed
df64d3292fd6: Pushed
v1.7.2: digest: sha256:6115155b261707b642341b065cd3fac2b546559ba035d0262650b3b3bbdd10ea size: 1157
[root@alice001 traefik]# ll
-rw-r--r-- 1 root root 1100 Jan  9 16:36 ds.yaml
-rw-r--r-- 1 root root  330 Jan  9 16:36 ingress.yaml
-rw-r--r-- 1 root root  800 Jan  9 16:36 rbac.yaml
-rw-r--r-- 1 root root  269 Jan  9 16:36 svc.yaml
[root@alice001 traefik]# vi /etc/nginx/conf.d/od.com.conf
[root@alice001 traefik]# cat /etc/nginx/conf.d/od.com.conf
upstream default_backend_traefik {
    server 172.23.187.173:81    max_fails=3 fail_timeout=10s;
    server 172.23.187.174:81    max_fails=3 fail_timeout=10s;
}
server {
    server_name *.od.com *.grep.pro;
    location / {
        proxy_pass http://default_backend_traefik;
        proxy_set_header Host       $http_host;
        proxy_set_header x-forwarded-for $proxy_add_x_forwarded_for;
    }
}

[root@alice001 traefik]# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[root@alice001 traefik]# systemctl reload nginx
[root@alice001 traefik]# vim /var/named/od.com.zone
[root@alice001 traefik]# cat /var/named/od.com.zone
$ORIGIN od.com.
$TTL 600    ; 10 minutes
@     IN SOA  dns.od.com. dnsadmin.od.com. (
        2021012907 ; serial
        10800      ; refresh (3 hours)
        900        ; retry (15 minutes)
        604800     ; expire (1 week)
        86400      ; minimum (1 day)
        )
        NS   dns.od.com.
$TTL 60 ; 1 minute
dns                A    47.243.20.250
harbor             A    47.243.20.250
k8s-yaml     A    47.243.20.250
traefik      A    47.243.20.250
[root@alice001 traefik]# systemctl restart named
[root@alice001 traefik]# dig -t A traefik.od.com @172.23.187.175 +short
47.243.20.250
```

在任意节点上

Plain Text 复制代码

```
[root@alice002 ~]# #----traefik
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/traefik/rbac.yaml
serviceaccount/traefik-ingress-controller created
clusterrole.rbac.authorization.k8s.io/traefik-ingress-controller created
clusterrolebinding.rbac.authorization.k8s.io/traefik-ingress-controller created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/traefik/ds.yaml
daemonset.extensions/traefik-ingress created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/traefik/svc.yaml
service/traefik-ingress-service created
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.od.com/traefik/ingress.yaml
ingress.extensions/traefik-web-ui created
[root@alice002 ~]#
[root@alice002 ~]# kubectl get pod -n kube-public
No resources found.
[root@alice002 ~]# kubectl get pod -n kube-system
NAME                        READY   STATUS              RESTARTS   AGE
coredns-6b6c4f9648-g6btt    1/1     Running             0          12h
traefik-ingress-62gb2       0/1     ContainerCreating   0          35s
traefik-ingress-wz2r7       0/1     ContainerCreating   0          35s
重启两台机器的docker
[root@alice002 ~]# systemctl restart docker.service
[root@alice003 ~]# systemctl restart docker.service
[root@alice002 ~]# kubectl get pod -n kube-system -o wide
NAME                        READY   STATUS    RESTARTS   AGE    IP              NODE                NOMINATED NODE   READINESS GATES
coredns-6b6c4f9648-g6btt    1/1     Running   0          12h    172.187.174.3   alice003.host.com   <none>           <none>
traefik-ingress-62gb2       1/1     Running   0          2m1s   172.187.173.4   alice002.host.com   <none>           <none>
traefik-ingress-wz2r7       1/1     Running   0          2m1s   172.187.174.4   alice003.host.com   <none>           <none>
```

dashboard

Plain Text 复制代码

```
[root@alice001 traefik]# docker pull k8scn/kubernetes-dashboard-amd64:v1.8.3
v1.8.3: Pulling from k8scn/kubernetes-dashboard-amd64
a4026007c47e: Pull complete
Digest: sha256:ebc993303f8a42c301592639770bd1944d80c88be8036e2d4d0aa116148264ff
Status: Downloaded newer image for k8scn/kubernetes-dashboard-amd64:v1.8.3
[root@alice001 traefik]# docker tag k8scn/kubernetes-dashboard-amd64:v1.8.3 harbor.od.com/public/dashboard:v1.8.3
[root@alice001 traefik]# docker push !$
docker push harbor.od.com/public/dashboard:v1.8.3
The push refers to repository [harbor.od.com/public/dashboard]
23ddb8cbb75a: Pushed
v1.8.3: digest: sha256:ebc993303f8a42c301592639770bd1944d80c88be8036e2d4d0aa116148264ff size: 529
[root@alice001 traefik]# mkdir -p /data/k8s-yaml/dashboard && cd /data/k8s-yaml/dashboard
[root@alice001 dashboard]# ll
total 16
-rw-r--r-- 1 root root 1427 Feb 20 19:11 deployment.yaml
-rw-r--r-- 1 root root  347 Feb 20 16:23 ingress.yaml
-rw-r--r-- 1 root root  610 Feb 20 18:23 rbac.yaml
-rw-r--r-- 1 root root  322 Feb 20 16:22 svc.yaml
[root@alice001 dashboard]# vim /var/named/od.com.zone
[root@alice001 dashboard]# cat /var/named/od.com.zone
$ORIGIN od.com.
$TTL 600    ; 10 minutes
@     IN SOA  dns.od.com. dnsadmin.od.com. (
        2021012909 ; serial
        10800      ; refresh (3 hours)
        900        ; retry (15 minutes)
        604800     ; expire (1 week)
        86400      ; minimum (1 day)
        )
        NS   dns.od.com.
$TTL 60 ; 1 minute
dns                A    47.243.20.250
harbor             A    172.23.187.175
k8s-yaml     A    47.243.20.250
traefik      A    47.243.20.250
dashboard      A    47.243.20.250
[root@alice001 dashboard]# systemctl restart named
[root@alice001 dashboard]# dig  -t A dashboard.od.com @172.23.187.175 +short
47.243.20.250
[root@alice001 dashboard]# cd /opt/certs/
[root@alice001 certs]# openssl req -new -key dashboard.od.com.key -out dashboard.od.com.csr -subj
"/CN=dashboard.od.com/C=CN/ST=BJ/L=Beijing/O=Oldb
[root@alice001 certs]# openssl x509 -req -in dashboard.od.com.csr -CA ca.pem -CAkey ca-key.pem -CAcreateserial -out dashboard.od.com.crt -
days 3650
Signature ok
subject=/CN=dashboard.od.com/C=CN/ST=BJ/L=Beijing/O=OldboyEdu/OU=ops
Getting CA Private Key
[root@alice001 certs]# ll dashboard.od.com.*
-rw-r--r-- 1 root root 1196 Feb 20 18:53 dashboard.od.com.crt
-rw-r--r-- 1 root root 1005 Feb 20 18:53 dashboard.od.com.csr
-rw------- 1 root root 1679 Feb 20 18:53 dashboard.od.com.key
[root@alice001 certs]# cd /etc/nginx/
[root@alice001 nginx]# mkdir certs
[root@alice001 nginx]# cd certs/
[root@alice001 certs]# ls
[root@alice001 certs]# cp /opt/certs/dashboard.od.com.key .
[root@alice001 certs]# cp /opt/certs/dashboard.od.com.crt .
[root@alice001 certs]# ll
total 8
-rw-r--r-- 1 root root 1196 Feb 20 18:57 dashboard.od.com.crt
-rw------- 1 root root 1679 Feb 20 18:57 dashboard.od.com.key
[root@alice001 certs]# vim /etc/nginx/conf.d/dashborad.conf
[root@alice001 dashboard]# cat /etc/nginx/conf.d/dashborad.conf
server {
    listen       80;
    server_name  dashboard.od.com dashboard.grep.pro;
    rewrite ^(.*)$ https://${server_name}$1 permanent;
}

server {
    listen       443 ssl;
    server_name  dashboard.od.com dashboard.grep.pro;

    ssl_certificate "certs/dashboard.od.com.crt";
    ssl_certificate_key "certs/dashboard.od.com.key";
    ssl_session_cache shared:SSL:1m;
    ssl_session_timeout  10m;
    ssl_ciphers HIGH:!aNULL:!MD5;
    ssl_prefer_server_ciphers on;

    location / {
```

```
        proxy_pass http://default_backend_traefik;
        proxy_set_header Host       $http_host;
        proxy_set_header x-forwarded-for $proxy_add_x_forwarded_for;
    }
}
[root@alice001 dashboard]#
[root@alice001 dashboard]# nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[root@alice001 dashboard]# systemctl restart nginx
```

这里部署完后我的docker pull不到镜像了 报错

Plain Text 复制代码

```
[root@alice002 ~]# docker pull harbor.od.com/public/dashboard:v1.10.1
Error response from daemon: error parsing HTTP 404 response body: invalid character 'p' after top-level value: "404 page not found\n"
```

把nginx的dashboard.conf删掉后就可以恢复 没有搞明白是什么问题 如果有知道的朋友麻烦评论告诉我一下

节点上

Plain Text 复制代码

```
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.grep.pro/dashboard/rbac.yaml
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.grep.pro/dashboard/deployment.yaml
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.grep.pro/dashboard/svc.yaml
[root@alice002 ~]# kubectl apply -f http://k8s-yaml.grep.pro/dashboard/ingress.yaml
```

登陆dashboard

Plain Text 复制代码

```
[root@alice002 ~]# kubectl get secert -n kube-system
error: the server doesn't have a resource type "secert"
[root@alice002 ~]# kubectl get secret -n kube-system
NAME                                 TYPE                                  DATA   AGE
coredns-token-snpx8                  kubernetes.io/service-account-token   3      20h
default-token-z6pmn                  kubernetes.io/service-account-token   3      21d
kubernetes-dashboard-admin-token-pbr2v   kubernetes.io/service-account-token   3      16m
kubernetes-dashboard-key-holder          Opaque                                2      6m19s
traefik-ingress-controller-token-t27zn   kubernetes.io/service-account-token   3      8h
[root@alice002 ~]# kubectl describe secret kubernetes-dashboard-admin-token-pbr2v  -n kube-system
Name:         kubernetes-dashboard-admin-token-pbr2v
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name: kubernetes-dashboard-admin
              kubernetes.io/service-account.uid: 1f03a210-3dae-4b10-9a19-1c5b6679edd4

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1346 bytes
namespace:  11 bytes
token:
eyJhbGciOiJSUzI1NJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uYW1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImtYmVybmV0ZXMuaW8vc2VydmljZWFY291bnQvc2VjcmV0Lm5hbWUiOiJr
y23DLvcekzs_wKk7D1KSUDTF_yGF9GnQZ_ECA_4d8yH2q3l0vwpCcitXw0H_YsOaGw5t8wZbATSUKEEZfjAULXXnZREP9Aa8as14i1tcgw2DGcHxyBCcP9bvhZcj3INsat3lBcmotr3Y3
[root@alice002 ~]# 用这里的token登陆就可以
```