网络

Subtitle

2022/10/05

Table of Contents

网络

flannel host-gw 模式要求二层直连

> Use host-gw to create IP routes to subnets via remote machine IPs. Requires direct layer2 connectivity between hosts running flannel.

二层直连具体来说就是在一个vlan下，arp广播包可达，否则添加路由将失败

```
[root@k8s-node.10-12-3-10 ~]# ip route add 172.20.101.0/24 via 10.14.15.1
RTNETLINK answers: Network is unreachable
```

10.12.3.10在10.12.0.0/16网段，10.14.15.1在 10.14.15.0/24网段，并具有不同的vlan id，网关地址要求是arp广播可达的。

考虑改用 flannel vxlan 模式，或者使用 calico

容器互通的必要性

Ingress

nginx-ingress upstream是容器ip，因此容器网络未实现时将无法通过ingress访问容器

```
2018/08/25 13:43:31 [error] 4528#4528: *22 upstream timed out (110: Connection timed out) while connecting to upstream, client: 10.5.111.225, server: k8s.dev.k8sdemo.cn, request: "GET /favicon.ico HTTP/1.1", upstream: "http://172.20.45.3:80/favicon.ico", host: "k8s.dev.k8sdemo.cn", referrer: "http://k8s.dev.k8sdemo.cn/"
```

通过**nodePort**访问**pod**

mpaas没有实现容器网络，直接绑定的宿主机5位数端口，nginx upstream形式为 宿主机ip:54321

kubernetes若不实现容器网络，则 kube-proxy监听的nodePort只有容器部署的机器是可达的，即和上述mpaas是一样的。此时需要知道容器部署的机器以及nodePort，自行实现ingress

实验：

kubernetes-dashboard nodePort为30000，nmap所有kubernetes节点：

```
nmap -Pn -iL /tmp/k8s.list -p 30000 -oG /tmp/a.log
```

结果：只有容器实际部署的机器上端口是通的

```
[root@op server]# cat /tmp/a.log |grep Ports |grep open
Host: 10.11.3.10 (k8s-node.10-11-3-10)  Ports: 30000/open/tcp//unknown///
```

其他机器都是filter

```
Host: 10.14.15.5 (k8s-node.10-14-15-5)  Ports: 30000/filtered/tcp//unknown///
```

原因是在没有容器网络的情况下，只有本机有到达容器网段的路由

```
[root@k8s-node.10-11-3-10 ~]# route -n
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
0.0.0.0         10.11.0.1       0.0.0.0         UG    0      0        0 eth0
10.11.0.0       0.0.0.0         255.255.0.0     U     0      0        0 eth0
169.254.0.0     0.0.0.0         255.255.0.0     U     1002   0        0 eth0
172.20.95.0     0.0.0.0         255.255.255.0   U     0      0        0 docker0
```

DNS

默认配置的resolv是169.169.0.2，容器网络互通未实现时容器内不能解析域名（到dns服务器网络不通），但是到外网是通的

```
/ # ping baidu.com
ping: bad address 'baidu.com'
/ # ping 114.114.114.114
PING 114.114.114.114 (114.114.114.114): 56 data bytes
64 bytes from 114.114.114.114: seq=0 ttl=58 time=24.030 ms
64 bytes from 114.114.114.114: seq=1 ttl=79 time=23.236 ms
```

容器DNS可以通过kubelet参数指定多个，但是要求 **Note: all DNS servers appearing in the list MUST serve the same set of records**

> --cluster-dns strings Comma-separated list of DNS server IP address. This value is used for containers DNS server in case of Pods with "dnsPolicy=ClusterFirst". Note: all DNS servers appearing in the list MUST serve the same set of records otherwise name resolution within the cluster may not work correctly. There is no guarantee as to which DNS server may be contacted for name resolution. (DEPRECATED: This parameter should be set via the config file specified by the Kubelet's --config flag. See https://kubernetes.io/docs/tasks/administer-cluster/kubelet-config-file/ for more information.)

CNI

使用calico cni之后，nginx-ingress监听端口消失（ss -lnt看不到），但是可以访问。查询得知 CNI 中 hostPort 已经无效了。因此需要启用hostNetwork

```
 spec:
    # hostNetwork makes it possible to use ipv6 and to preserve the source IP correctly regardless of docker configuration
    # however, it is not a hard dependency of the nginx-ingress-controller itself and it may cause issues if port 10254 already is taken on the host
    # that said, since hostPort is broken on CNI (https://github.com/kubernetes/kubernetes/issues/31307) we have to use hostNetwork where CNI is used
    # like with kubeadm
    # hostNetwork: true
```

结论

容器网络不通的影响：

- ingress无法正常工作，所有服务挂掉
- 即使开发了自定义的使用nodePort的ingress，如果存在内部基础组件调用（redis□mysql等基于kubernetes实现），也会影响服务
- 影响内部域名解析

因此最好提前规划好网络方案，后期保持不变。

Printed on: 2022/10/05 17:37

Convert to img Failed!