linux安装flannel实现docker跨主机通信

ip	安装内容	节点名称
10.2.40.223	flannel,etcd,docker	node1
10.2.40.224	flannel,etcd,docker	node2

原理图点击这儿查看

- 数据从源容器中发出后，经由所在主机的docker0虚拟网卡转发到flannel0虚拟网卡，这是个P2P的虚拟网卡，flanneld服务监听在网卡的另外一端。
- Flannel通过Etcd服务维护了一张节点间的路由表，详细记录了各节点子网网段 。
- 源主机的flanneld服务将原本的数据内容UDP封装后根据自己的路由表投递给目的节点的flanneld服务，数据到达以后被解包，然后直接进入目的节点的flannel0虚拟网卡，然后被转发到目的主机的docker0虚拟网卡，最后就像本机容器通信一下的有docker0路由到达目标容器。

安装部署

1、etcd安装

1) 离线下载etcd安装包

```
wget https://github.com/etcd-io/etcd/releases/download/v3.3.10/etcd-v3.3.10-linux-amd64.tar.gz
tar -xf etcd-v3.3.10-linux-amd64.tar.gz
cd etcd-v3.3.10-linux-amd64
mv etcd* /usr/bin/
```

2) 在node1上创建文件`/usr/lib/systemd/system/etcd.service`

```
cat > /usr/lib/systemd/system/etcd.service << EOF
[Unit]
Description=etcd
After=network.target

[Service]
Environment=ETCD_NAME=etcd-1
Environment=ETCD_DATA_DIR=/var/lib/etcd
Environment=ETCD_LISTEN_CLIENT_URLS=http://10.2.40.223:2379,http://127.0.0.1:2379
Environment=ETCD_LISTEN_PEER_URLS=http://10.2.40.223:2380
Environment=ETCD_ADVERTISE_CLIENT_URLS=http://10.2.40.223:2379,http://127.0.0.1:2379
Environment=ETCD_INITIAL_ADVERTISE_PEER_URLS=http://10.2.40.223:2380
Environment=ETCD_INITIAL_CLUSTER_STATE=new
Environment=ETCD_INITIAL_CLUSTER_TOKEN=etcd-cluster-token
```

```
Environment=ETCD_INITIAL_CLUSTER=etcd-1=http://10.2.40.223:2380,etcd-2=http://10.2.40.224:2380
ExecStart=/usr/bin/etcd
[Install]
WantedBy=multi-user.target
EOF
```

> - name：节点名称 data-dir 指定节点的数据存储目录
> - listen-peer-urls： 监听URL，用于与其他节点通讯
> - listen-client-urls： 对外提供服务的地址：比如 http://ip:2379,http://127.0.0.1:2379 ，客户端会连接到这里和 etcd 交互
> - initial-advertise-peer-urls： 该节点同伴监听地址，这个值会告诉集群中其他节点
> - initial-cluster 集群中所有节点的信息，格式为 node1=http://ip1:2380,node2=http://ip2:2380,... 。注意：这里的 node1 是节点的 --name 指定的名字；后面的 ip1:2380 是 --initial-advertise-peer-urls 指定的值
> - initial-cluster-state： 新建集群的时候，这个值为 new ；假如已经存在的集群，这个值为 existing
> - initial-cluster-token ： 创建集群的 token，这个值每个集群保持唯一。这样的话，如果你要重新创建集群，即使配置和之前一样，也会再次生成新的集群和节点 uuid；否则会导致多个集群之间的冲突，造成未知的错误
> - advertise-client-urls：对外公告的该节点客户端监听地址，这个值会告诉集群中其他节点

3) 在node2上创建文件`/usr/lib/systemd/system/etcd.service`

```
cat > /usr/lib/systemd/system/etcd.service << EOF
[Unit]
Description=etcd
After=network.target

[Service]
Environment=ETCD_NAME=etcd-2
Environment=ETCD_DATA_DIR=/var/lib/etcd
Environment=ETCD_LISTEN_CLIENT_URLS=http://10.2.40.224:2379,http://127.0.0.1:2379
Environment=ETCD_LISTEN_PEER_URLS=http://10.2.40.224:2380
Environment=ETCD_ADVERTISE_CLIENT_URLS=http://10.2.40.224:2379,http://127.0.0.1:2379
Environment=ETCD_INITIAL_ADVERTISE_PEER_URLS=http://10.2.40.224:2380
Environment=ETCD_INITIAL_CLUSTER_STATE=new
Environment=ETCD_INITIAL_CLUSTER_TOKEN=etcd-cluster-token
Environment=ETCD_INITIAL_CLUSTER=etcd-1=http://10.2.40.223:2380,etcd-2=http://10.2.40.224:2380
ExecStart=/usr/bin/etcd
[Install]
WantedBy=multi-user.target
EOF
```

2 、启动etcd服务

```
systemctl enable etcd && systemctl start etcd
```

3、测试集群

```
#node1上执行以下命令
etcdctl mkdir /boshen
#在node2上执行以下命令,查看是否已经创建成功文件夹
etcd ls
```

4、安装flannel

1) 下载离线包

```
wget https://github.com/coreos/flannel/releases/download/v0.10.0/flannel-v0.10.0-linux-amd64.tar.gz
tar -xf flannel-v0.10.0-linux-amd64.tar.gz
cp flanneld /usr/bin/
```

2) 定义flannel网络ip池(在node1上执行)

```
cat > flannel-config.json << EOF
{
  "Network": "10.3.0.0/16",
  "SubnetLen": 24,
  "Backend": {
    "Type": "vxlan"
  }
}
EOF

etcdctl set /docker-flannel/network/config < flannel-config.json
```

> 1. `Network` 定义该网络的 IP 池为 `10.3.0.0/16`。
> 2. `SubnetLen` 指定每个主机分配到的 subnet 大小为 24 位，即`10.3.X.0/24`。
> 3. `Backend` 为 `vxlan`，即主机间通过 vxlan 通信。

3) 新建启动文件: `/usr/lib/systemd/system/flanneld.service` 所有节点执行

```
cat > /usr/lib/systemd/system/flanneld.service << EOF
[Unit]
Description=flannel
After=etcd.service network.target

[Service]
ExecStart=/usr/bin/flanneld --etcd-endpoints=http://10.2.40.223:2379,http://10.2.40.224:2379 -etcd-prefix=/docker-flannel/network  --iface=eth0
[Install]
WantedBy=multi-user.target
EOF

```

> - `--etcd-endpoints` 指定 etcd url。
> - `--iface` 指定主机间数据传输使用的 interface。
> - `--etcd-prefix` 指定 etcd 存放 flannel 网络配置信息的 key。只能指到目录级别，比如上面的key为：/docker-flannel/network/config ，只能指到/docker-flannel/network

4) 启动flannel

```
systemctl enable flanneld && systemctl start flanneld
```

5) 查看状态

> ```
> [root@localhost flannel]# systemctl status flanneld
> ● flanneld.service - flannel
> Loaded: loaded (/usr/lib/systemd/system/flanneld.service; enabled; vendor
> preset: disabled)
> Active: active (running) since 二 2019-10-01 18:04:11 CST; 14s ago
> Main PID: 36439 (flanneld)
>  Tasks: 19
> Memory: 13.6M
> CGroup: /system.slice/flanneld.service
>         └─36439 /usr/bin/flanneld --etcd-
> endpoints=http://10.2.40.223:2379,http://10.2.40.224:2379 -etcd-
> prefix=/docker-flannel/network --iface=eth0
>
> #10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.133043   36439
> local_manager.go:220] Allocated lease (10.3.98.0/24) to current node
> (10.2.40.224)
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.133043   36439
> local_manager.go:220] Allocated lease (10.3.98.0/24) to current node
> (10.2.40.224)
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.138798   36439
> main.go:300] Wrote subnet file to /run/flannel/subnet.env
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.138811   36439
> main.go:304] Running backend.
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.138896   36439
> vxlan_network.go:60] watching for new subnet leases
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.143955   36439
> main.go:396] Waiting for 22h59m59.96919843s to renew lease
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.184424   36439
> iptables.go:115] Some iptables rules are missing; deleting and recreating
> rules
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.184451   36439
> iptables.go:137] Deleting iptables rule: -s 10.3.0.0/16 -j ACCEPT
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.186413   36439
> iptables.go:137] Deleting iptables rule: -d 10.3.0.0/16 -j ACCEPT
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.187780   36439
> iptables.go:125] Adding iptables rule: -s 10.3.0.0/16 -j ACCEPT
> 10月 01 18:04:12 bogon flanneld[36439]: I1001 18:04:12.191694   36439
> iptables.go:125] Adding iptables rule: -d 10.3.0.0/16 -j ACCEPT
> ```
>
> 查看ip

> ```
> [root@localhost flannel]# ip r
> default via 10.2.8.254 dev eth0 proto static metric 100
> 10.2.8.254 dev eth0 proto static scope link metric 100
> 10.2.40.0/24 dev eth0 proto kernel scope link src 10.2.40.224 metric 100
> #10.3.96.0/24 via 10.3.96.0 dev flannel.1 onlink
> 10.3.96.0/24 via 10.3.96.0 dev flannel.1 onlink
> 172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
> 192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
> ```
>
> 可以看到分配子网为`10.3.98.0/24`
>
> 新建了一个interface：flannel.1
>
> 添加了一条路由：目的地址为 flannel 网络 10.3.96.0/24 的数据包都由 flannel.1 转发

6）修改docker配置文件连接flannel（所有节点都需要修改）

```
# 查看flannel配置文件/run/flannel/subnet.env
cat /run/flannel/subnet.env
[root@localhost flannel]# cat /run/flannel/subnet.env
FLANNEL_NETWORK=10.3.0.0/16
FLANNEL_SUBNET=10.3.98.1/24
FLANNEL_MTU=1450
FLANNEL_IPMASQ=false

# 修改docker配置文件使docker默认网络为flannel
#在 /usr/lib/systemd/system/docker.service中给ExecStart命令的参数中添加 --bip和--mtu
分别对应subnet.env的 FLANNEL_SUBNET 与FLANNEL_MTU
ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock
--bip=10.3.98.1/24 --mtu=1450
```

7）重启docker

```
systemctl daemon-reload && systemctl restart docker.service
```

> ```
> [root@localhost flannel]# ip r
> default via 10.2.8.254 dev eth0 proto static metric 100
> 10.2.8.254 dev eth0 proto static scope link metric 100
> 10.2.40.0/24 dev eth0 proto kernel scope link src 10.2.40.224 metric 100
> 10.3.96.0/24 via 10.3.96.0 dev flannel.1 onlink
> # 10.3.98.0/24 dev docker0 proto kernel scope link src 10.3.98.1
> 10.3.98.0/24 dev docker0 proto kernel scope link src 10.3.98.1
> 192.168.122.0/24 dev virbr0 proto kernel scope link src 192.168.122.1
> ```
>
> 此时已经将 10.3.98.1配置到 linux bridge docker0 上，路由规则为 10.3.98.0/24

8）测试是否连通

所有节点运行`docker run -itd --name=box1 busybox`

```
[root@localhost flannel]# docker run -itd --name=box1 busybox
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox
7c9d20b9b6cd: Pull complete
Digest: sha256:fe301db49df08c384001ed752dff6d52b4305a73a7f608f21528048e8a08b51e
Status: Downloaded newer image for busybox:latest
73da714580339e3da4ce841d4d8c6285f2097e5c1f15cb1a23c88298d4a3fb5e
```

node1节点运行`docker exec box1 ip r`

```
[root@localhost flannel]# docker exec box1 ip r
default via 10.3.96.1 dev eth0
10.3.96.0/24 dev eth0 scope link  src 10.3.96.2
```

node2节点运行`docker exec box1 ip r`

```
[root@localhost flannel]# docker exec box1 ip r
default via 10.3.98.1 dev eth0
10.3.98.0/24 dev eth0 scope link  src 10.3.98.2
```

> 可以看到node1节点容器ip为 10.3.96.2 node2节点容器ip为 src 10.3.98.2

在node1运行`docker exec box1 ping 10.3.98.2`

```
[root@localhost flannel]# docker exec box1 ping 10.3.98.2
PING 10.3.98.2 (10.3.98.2): 56 data bytes
64 bytes from 10.3.98.2: seq=0 ttl=62 time=1.039 ms
64 bytes from 10.3.98.2: seq=1 ttl=62 time=0.488 ms
64 bytes from 10.3.98.2: seq=2 ttl=62 time=0.457 ms
64 bytes from 10.3.98.2: seq=3 ttl=62 time=0.420 ms
```

在node1运行`docker exec box1 traceroute 10.3.98.2`

```
[root@localhost flannel]# docker exec box1 traceroute 10.3.98.2
traceroute to 10.3.98.2 (10.3.98.2), 30 hops max, 46 byte packets
 1  bogon (10.3.96.1)  0.017 ms  0.012 ms  0.009 ms
 2  bogon (10.3.98.0)  0.288 ms  0.221 ms  0.196 ms
 3  bogon (10.3.98.2)  0.334 ms  0.248 ms  0.253 ms
```

> 首先数据包到了自己主机docker0 10.3.96.1 因为docker0是默认网关