NS3跨主机通信配置

1 软硬件需求

1.1 硬件需求

- 两台PC，操作系统为 Ubuntu 20.04 LTS
- 一条网线

1.2 软件需求

需要在两台PC中安装以下包：

- `Docker`
- `net-tools`(需要使用`ifconfig`)
- `vim`
  其中Docker的安装方式见TBD，`net-tools`和`vim`可以直接使用`apt`安装。
  在进行下一步配置之前请确保上述软件已经全部安装。

2 配置过程

2.1 网络连接设置

本小节对两台主机的网络进行初步设置，本质是通过设置静态IP使得两台主机通过网线直连之后可以直接通信。
首先使用网线直接将两台主机相连，方便起见，需要将两台主机所有其他的网络连接断开。
此时输入`ifconfig`命令，可以看到除`lo`本地环回外，还有一个设备，设备名以`en`开头，表示网卡类型为以太网，下面以`enp3s0`为例，演示配置过程。
注意：本小节配置过程需要用户具有`sudo`权限，不建议直接使用`root`用户配置！
输入以下指令打开网络配置文件

```
vim /etc/netplan/01-network-manager-all.yaml
```

添加以下内容，设置固定IP为`192.168.0.11/24`并设置相应的网关和DNS。

```
### ----- 原有内容
network:
  version: 2
  renderer: NetworkManager
```

```
### ----- 以下为新增内容
  ethernets:
    enp3s0:
      addresses: [192.168.0.11/24]
      dhcp4: no
      optional: true
      gateway4: 192.168.0.1
      nameservers:
        addresses: [8.8.8.8]
```

同理，另一台主机上进行同样的配置，该主机中对应设备名为`enp2s0`下面将其IP设置为`192.168.0.10/24`，注意两个IP应该位于同一子网下。

```
network:
  version: 2
  renderer: NetworkManager
  ethernets:
    enp2s0:
      addresses: [192.168.0.10/24]
      dhcp4: no
      optional: true
      gateway4: 192.168.0.1
      nameservers:
        addresses: [8.8.8.8]
```

完成配置后，分别在两台主机上执行以下命令，应用以上设置。

```
sudo netplan apply
```

执行后使用`ping`命令测试连通性，两主机可以通信。
下面的配置过程都在`192.168.0.10`主机上完成，另一台主机需要采用同样的配置方式，只需要修改`br0`对应的IP即可。

2.2 创建用户

不要使用`root`用户进行后续设置，此处创建一个名为`lab`的用户。

```
sudo adduser lab
```

将`lab`用户加入到`sudo`组和`docker`组中

```
sudo usermod -aG sudo lab
sudo usermod -aG docker lab
```

若没有`docker`组，则创建对应组再尝试添加

```
sudo groupadd docker
sudo usermod -aG docker lab
```

若仍然提示权限不足，重启`Docker`

```
sudo systemctl restart docker
```

并重复上述步骤。
随后，切换至该用户，开始以下步骤。

2.3 防火墙设置

设置`iptables`的默认表`filter`中的`FORWARD`链，并设置其目标值为`ACEEPT`

```
sudo iptables -t filter -P FORWARD ACCEPT
```

2.4 Docker容器创建

首先需要加载镜像：

```
docker load -i ns3.38-image.tar
```

查看本地镜像列表，确认是否加载成功：

```
docker images
# 预期输出
# REPOSITORY  TAG xxx xxx xxx
# bcng/ns3.38 dev xxx xxx xxx
```

从`bcng/ns3.38:dev`镜像创建容器：

- 设置容器名为`ns3`
- 通过`--privileged`选项赋予该容器特殊权限
- 设置其网络模式为`none`（因为后续需要自定义网络配置）

- 设置终端为`/bin/bash`

```
docker run -tid --name=ns3 --privileged --network=none --entrypoint=/bin/bash
bcng/ns3.38:dev
```

注意，其中选项`-tid`的意义如下：

- `-d`：后台运行容器并返回ID
- `-ti`：交互式运行容器，并分配一个伪终端
  因此该命令会返回一个字符串，即该容器对应的ID。

2.5 创建及配置`veth`对

获取名为`ns3`容器对应的主进程的`pid`，并设置相应的`network namespace`对象（即`netns`）

```
pid=$(docker inspect --format '{{ .State.Pid }}' ns3)
sudo mkdir -p /var/run/netns
sudo ln -s /proc/$pid/ns/net /var/run/netns/$pid
```

创建`veth`对

```
sudo ip link add veth0 type veth peer name veth1
```

将`veth0`配置到容器的`netns`内，并通过`netns`在`$pid`对应的`network namespace`中执行`ip link set veth0 up`命令，用于启动`veth0`

```
sudo ip link set dev veth0 netns $pid
sudo ip netns exec $pid ip link set veth0 up
```

2.6 在容器内创建相关设备

首先需要进入容器，本小节的后续操作均在容器内进行

```
docker exec -ti --privileged ns3 bash
```

创建`tap`设备，命名为`thetap`

```
ip tuntap add dev thetap mode tap
ip link set thetap up
```

```
ip link set thetap promisc on
```

在容器内创建名为`br0`的网桥

```
brctl addbr br0
brctl addif br0 veth0
brctl addif br0 thetap
ip link set br0 up
```

在网桥上配置虚拟子网地址，此处为`br0`分配`90.1.1.1/24`地址

```
ip addr add 90.1.1.1/24 dev br0
```

配置容器内网关

```
ip route add default via 90.1.1.1 dev br0
```

注意，若在另一台主机上配置，需要设置不同的IP及网关。
作者将`192.168.0.11`主机内容器的`br0`的IP设置为`100.1.1.1/24`，对应的网关设置为`100.1.1.1`

2.7 配置宿主机

本小节内容需要在宿主机内进行配置，建议另开一个终端执行命令，不要关闭或退出容器所在终端。
首先配置`veth1`设备

```
sudo ip link set dev veth1 up
sudo ip link set dev veth1 address ee:ee:ee:ee:ee:ee
```

配置宿主机`veth1`的`arp`代理。此处需要先切换至`root`用户写入，完成后再切换回`lab`用户

```
sudo su
echo 1 > /proc/sys/net/ipv4/conf/veth1/proxy_arp
su lab
```

配置路由

```
sudo ip route add 90.1.1.0/24 dev veth1
```

同理，另一台主机对应设置为`100.1.1.0/24`。

下面，在宿主机上设置节点路由。
此处需要重申两台主机对应的网络设置，以免混淆：

- `192.168.0.10`主机：网卡设备名为`enp2s0`，容器内部网关为`90.1.1.1`
- `192.168.0.11`主机：网卡设备名为`enp3s0`，容器内部网关为`100.1.1.1`
  在`192.168.0.10`主机上配置路由

```
sudo ip route add 100.1.1.0/24 via 192.168.0.11 dev enp2s0
```

对应地，`192.168.0.11`主机配置方式

```
sudo ip route add 90.1.1.0/24 via 192.168.0.10 dev enp3s0
```

至此，所有网络配置已经完成。

2.8 运行NS3测试程序

下面，在两台主机上分别进入容器，执行以下命令，运行NS3测试程序。
注意：

1. ns3程序运行时的参数需要进行调整，以下示例命令是在`192.168.0.10`主机上执行的，另一台主机需要调整对应的`netAddr`，`netGateway`以及`pingDest`。
2. 对于最后一条命令，在两台主机上的执行时刻需要尽可能靠近，即：建议先分别输入指令，不要执行，待两台主机均输入正确指令后，先后输入回车执行命令。否则可能导致大量丢包。

```
cd /ns3/ns-allinone-3.38
./build.py --enable-examples --enable-tests
/ns3/ns-allinone-3.38/ns-3.38/ns3 run tap-csma-ping --netAddr=90.1.1.0 --
netMask=255.255.255.0 --netGateway=90.1.1.1 --nodeNum=200 --pingDest=100.1.1.2
```