

22.ROS distributed communication

A running ROS system can contain multiple nodes distributed across multiple computers. Depending on the system configuration, any node may need to communicate with any other node at any time. However, a necessary condition for implementing this distributed communication is that each machine must be on the same local area network. In this lesson, we will use Raspberry Pi and Virtual Machine as an example to illustrate how to configure and implement distributed communication.

22.1 Preparation work

22.1.1 Configuration

Require:

- The virtual machine and Raspberry Pi are connected to the same LAN;
- Select one as the host. The host needs to start roscore. Here, select Raspberry Pi as the host and the virtual machine as the slave. **The Raspberry Pi here needs to have ros-noetic installed;**
- Install the ssh and chrony packages on each device for synchronization. You can enter the following instructions to install.

```
sudo apt-get install chrony openssh-server -y
```

22.1.2 Check ip address

Both are connected to the same LAN, query the network IP address through the following command,

```
ifconfig
```

Virtual machine side,

```
yahboom@yahboom-virtual-machine:~$ ifconfig
ens33: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 192.168.2.131  netmask 255.255.255.0  broadcast 192.168.2.255
    inet6 fe80::3c01:59d6:de4d:ad5d  prefixlen 64  scopeid 0x20<link>
    ether 00:0c:29:77:af:63  txqueuelen 1000  (Ethernet)
    RX packets 416270  bytes 47735068 (47.7 MB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 3033  bytes 506829 (506.8 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 678251  bytes 27415843226 (27.4 GB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 678251  bytes 27415843226 (27.4 GB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

The address found is 192.168.2.131

Raspberry Pi side,

```
ifconfig
```

```
dofbot@dofbot:~$ ifconfig
eth0: flags=4099<UP,BROADCAST,MULTICAST> mtu 1500
    ether e4:5f:01:7f:a7:a5 txqueuelen 1000 (Ethernet)
    RX packets 0 bytes 0 (0.0 B)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 0 bytes 0 (0.0 B)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 1386 bytes 113422 (113.4 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 1386 bytes 113422 (113.4 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

wlan0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.2.117 netmask 255.255.255.0 broadcast 192.168.2.255
    inet6 fe80::d6fc:d8f5:7eb2:e07e prefixlen 64 scopeid 0x20<link>
    ether e4:5f:01:7f:a7:a6 txqueuelen 1000 (Ethernet)
    RX packets 190602 bytes 21847906 (21.8 MB)
    RX errors 0 dropped 124 overruns 0 frame 0
    TX packets 90605 bytes 122265117 (122.2 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
```

The found address is 192.168.2.117

22.2 Modify configuration file

22.2.1 Virtual machine side

Taking the virtual machine as a slave machine as an example, you need to modify the ROS_MASTER_URI of the virtual machine to the Raspberry Pi IP. Here is the 192.168.2.117 you just queried, enter it in the terminal,

```
sudo gedit ~/.bashrc
```

After entering the password, add it at the bottom of the file,

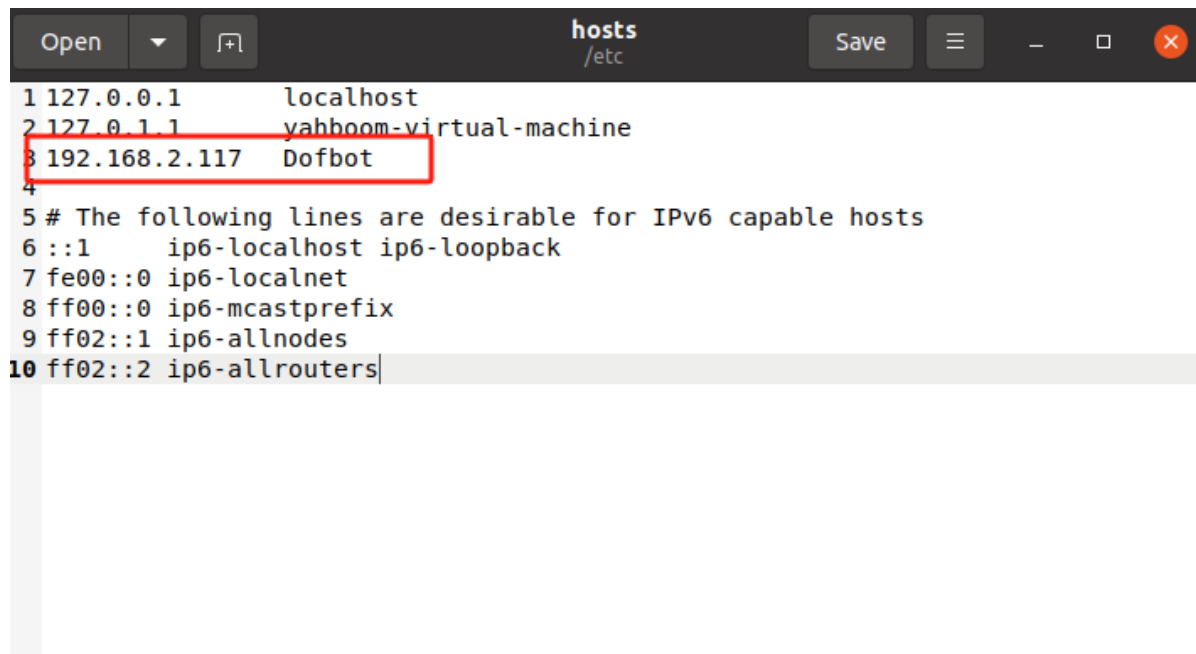
```
export ROS_MASTER_URI=http://192.168.2.117:11311
```

```
1
2
3 source /opt/ros/noetic/setup.bash
4 #source /home/yahboom/ros_test/devel/setup.bash
5 export PATH
6 #export ROS_HOSTNAME=localhost
7 export ROS_MASTER_URI=http://192.168.2.117:11311
8 #export ROS_IP=$ip
9 #export ROS_MASTER_URI=http://$ROS_IP:11311
10 #export ROS_MASTER_URI=http://192.168.2.185:11311
11
```

Exit after saving and reopen the terminal to refresh the environment variables to take effect.

Then, modify the/etc/hosts file,

```
sudo gedit /etc/hosts
```



```
Open hosts Save
1 127.0.0.1 localhost
2 127.0.1.1 yahboom-virtual-machine
3 192.168.2.117 Dofbot
4
5 # The following lines are desirable for IPv6 capable hosts
6 ::1 ip6-localhost ip6-loopback
7 fe00::0 ip6-localnet
8 ff00::0 ip6-mcastprefix
9 ff02::1 ip6-allnodes
10 ff02::2 ip6-allrouters
```

As shown in the picture above, add the Raspberry Pi's IP and machine name, save and exit.

22.2.2 Raspberry Pi terminal

The modified files and steps are almost the same as above. The only difference is that the ROS_MASTER_URI of the Raspberry Pi needs to be replaced with the local IP, which is 192.168.2.131. Just add the ip and machine name of the virtual machine at the end of the /etc/hosts file. The two files are as follows:

~/bashrc

```
sudo vim ~/.bashrc
```

```
18 alias python=python3
19 ip=$(ip addr show eth0 | grep -o 'inet [0-9]\+\.[0-9]\+\.[0-9]\+' | grep -o [0-9].*)
20 if [ -z $ip ]; then
21   ip=$(ip addr show wlan0 | grep -o 'inet [0-9]\+\.[0-9]\+\.[0-9]\+' | grep -o
    [0-9].*)
22 fi
23 if [ -z $ip ]; then
24   ip=$(ip addr show lo | grep -o 'inet [0-9]\+\.[0-9]\+\.[0-9]\+' | grep -o [0-9].*)
25 fi
26 echo "-----"
27 echo -e "MY_IP: \033[32m$ip\033[0m"
28 echo "-----"
29
30 export ROS_MASTER_URI=http://$ip:11311
31
```

/etc/hosts

```
sudo vim /etc/hosts
```

```
1 |127.0.0.1      localhost
2 |127.0.1.1      Dofbot
3 |192.168.2.131  yahboom-virtual-machine
4
5
6 # The following lines are desirable for IPv6 capable hosts
7 ::1          ip6-localhost ip6-loopback
8 fe00::0      ip6-localnet
9 ff00::0      ip6-mcastprefix
10 ff02::1      ip6-allnodes
11 ff02::2      ip6-allrouters
```

Similarly, exit after saving, reopen the terminal or source the terminal to refresh the environment variables.

22.3 Enable distributed communication

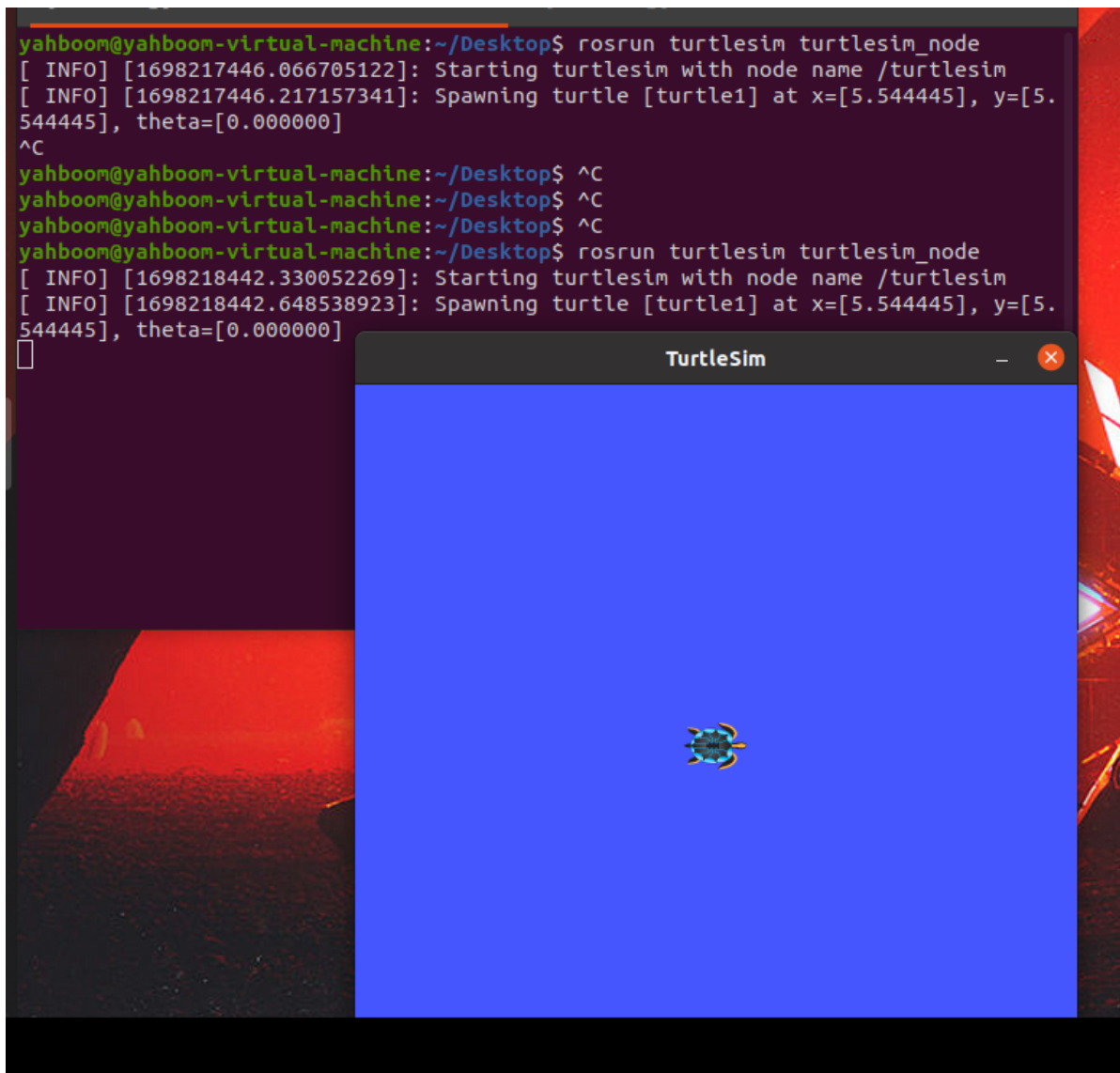
Let's verify whether the configuration is successful. First, on the host (here is the Raspberry Pi), start roscore and enter in the host terminal,

```
roscore
```

Then, run the little turtle node on the slave machine (virtual machine) and enter from the slave machine terminal,

```
roslaunch turtlesim turtlesim_node
```

A little turtle appears from the machine,



Then, we start the little turtle keyboard control node on the host (Raspberry Pi), and input at the host terminal,

```
roslaunch turtlesim turtle_teleop_key
```

Click the keyboard to control the terminal, and then use the up, down, left and right buttons to control the movement of the little turtle on the virtual machine.

