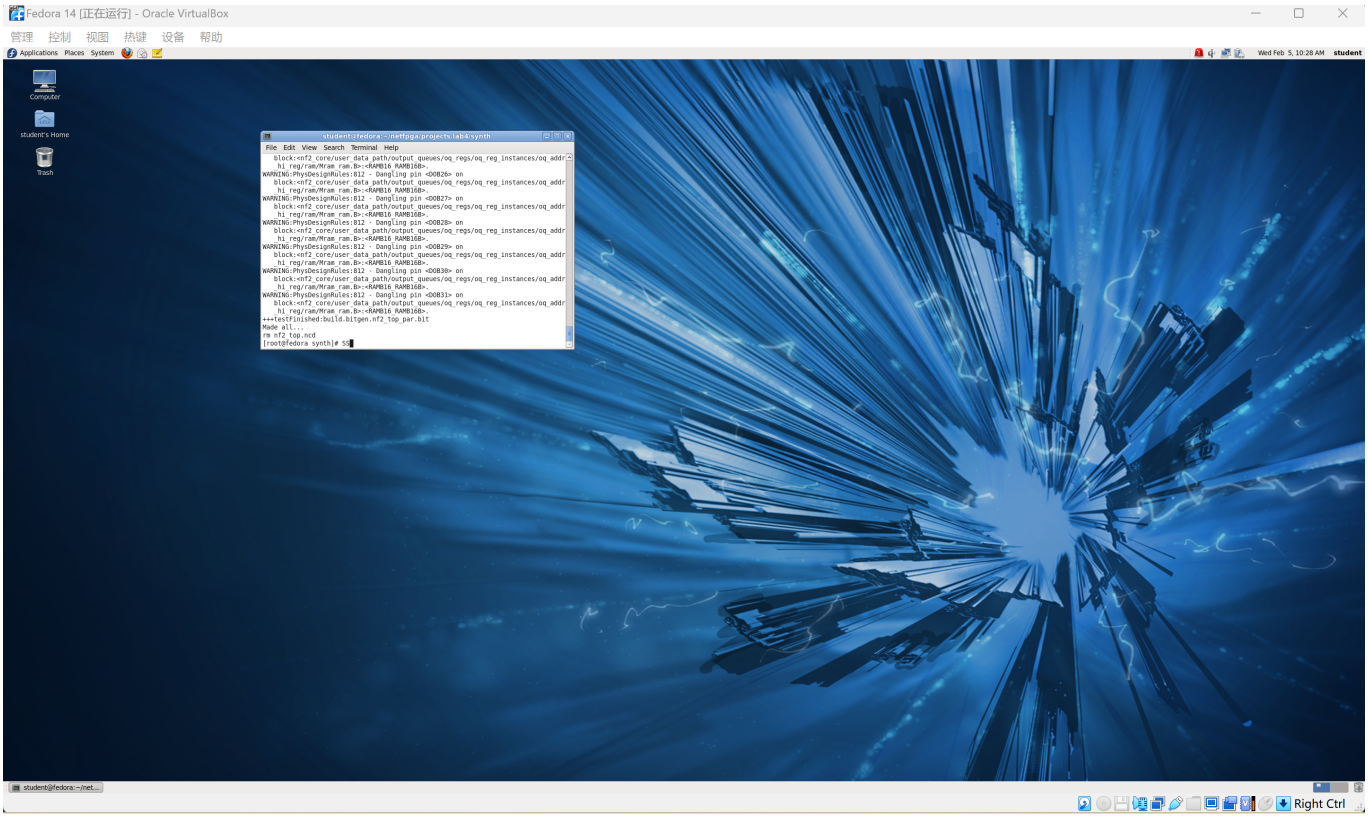
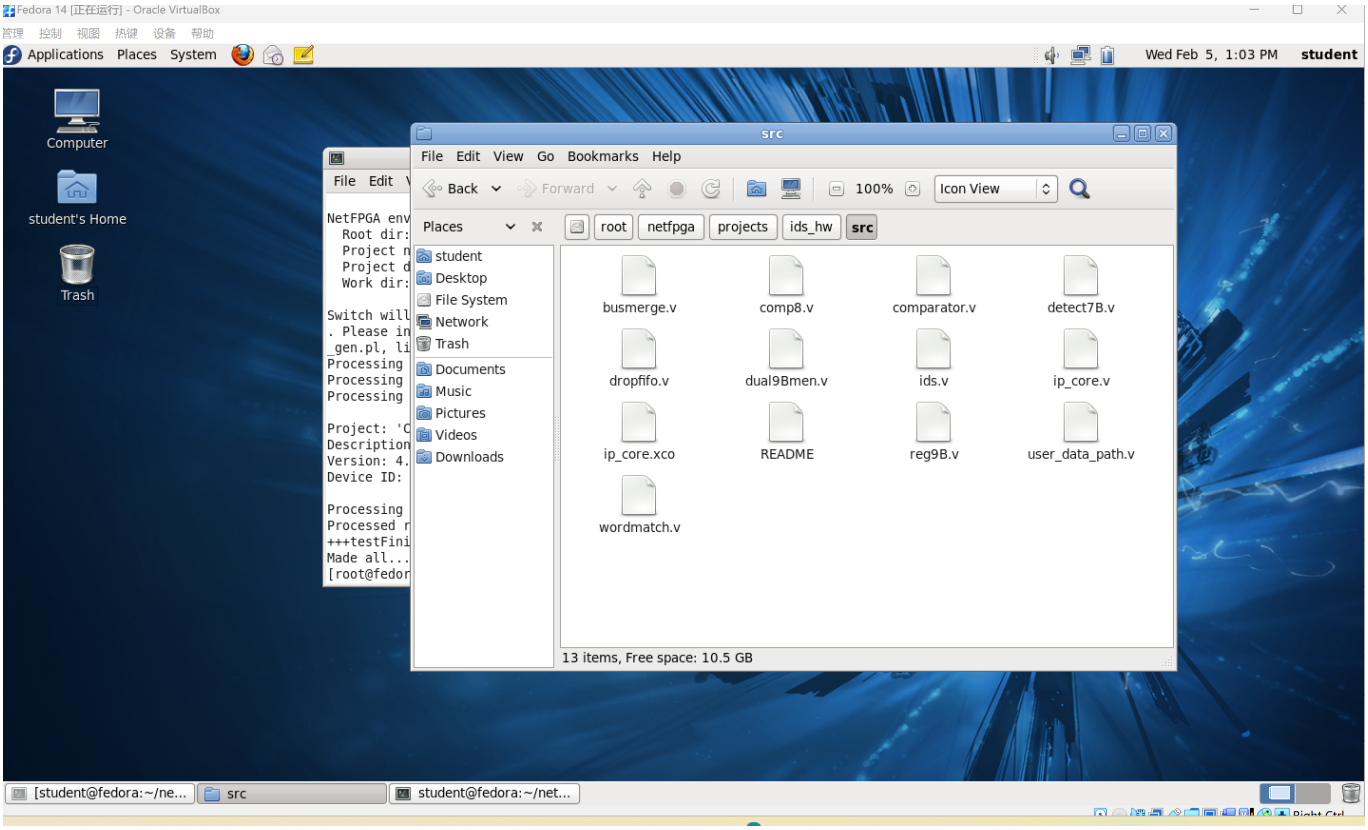
EE533_Lab4_Report

1. Download and Setup Fedora VM

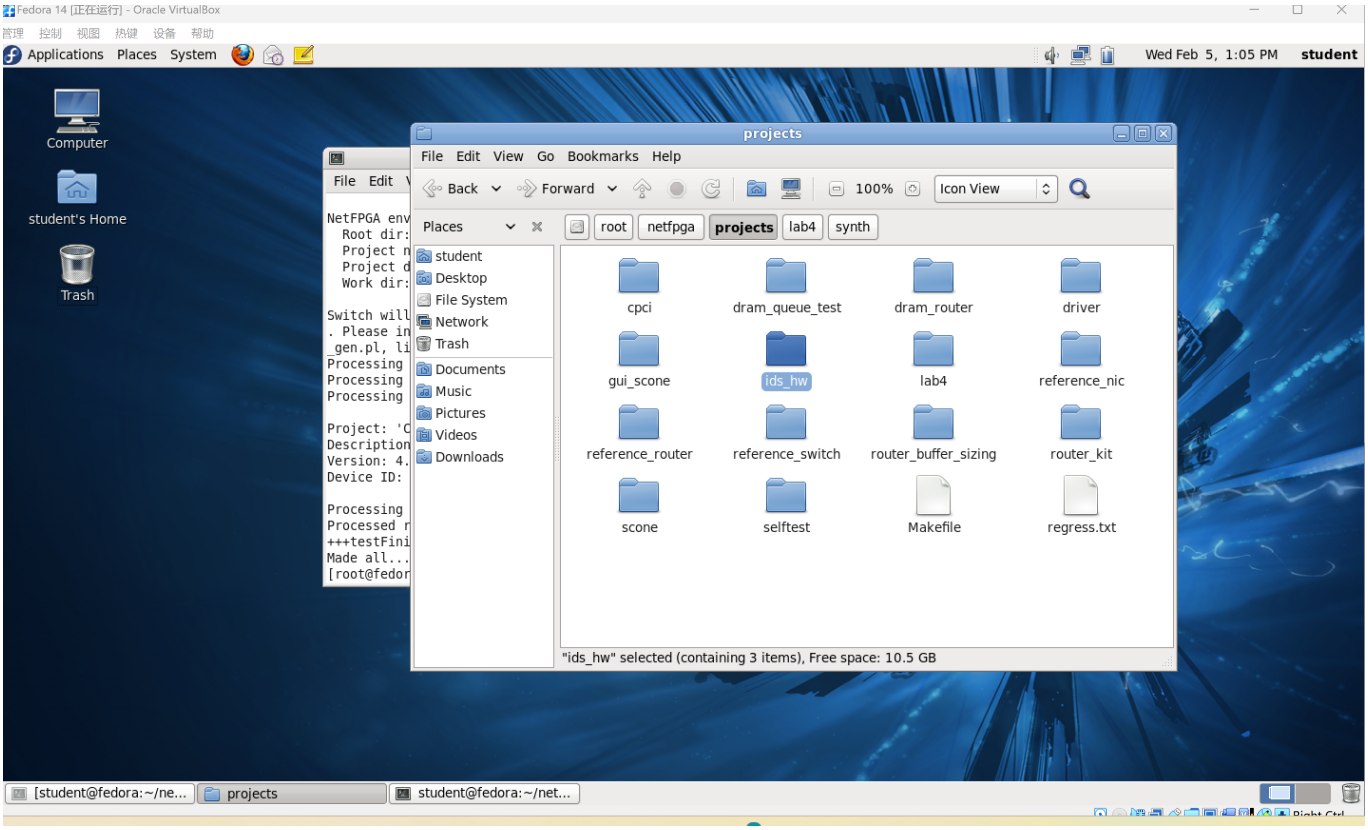


2. Compile and generate a design bitfile for NetFPGA

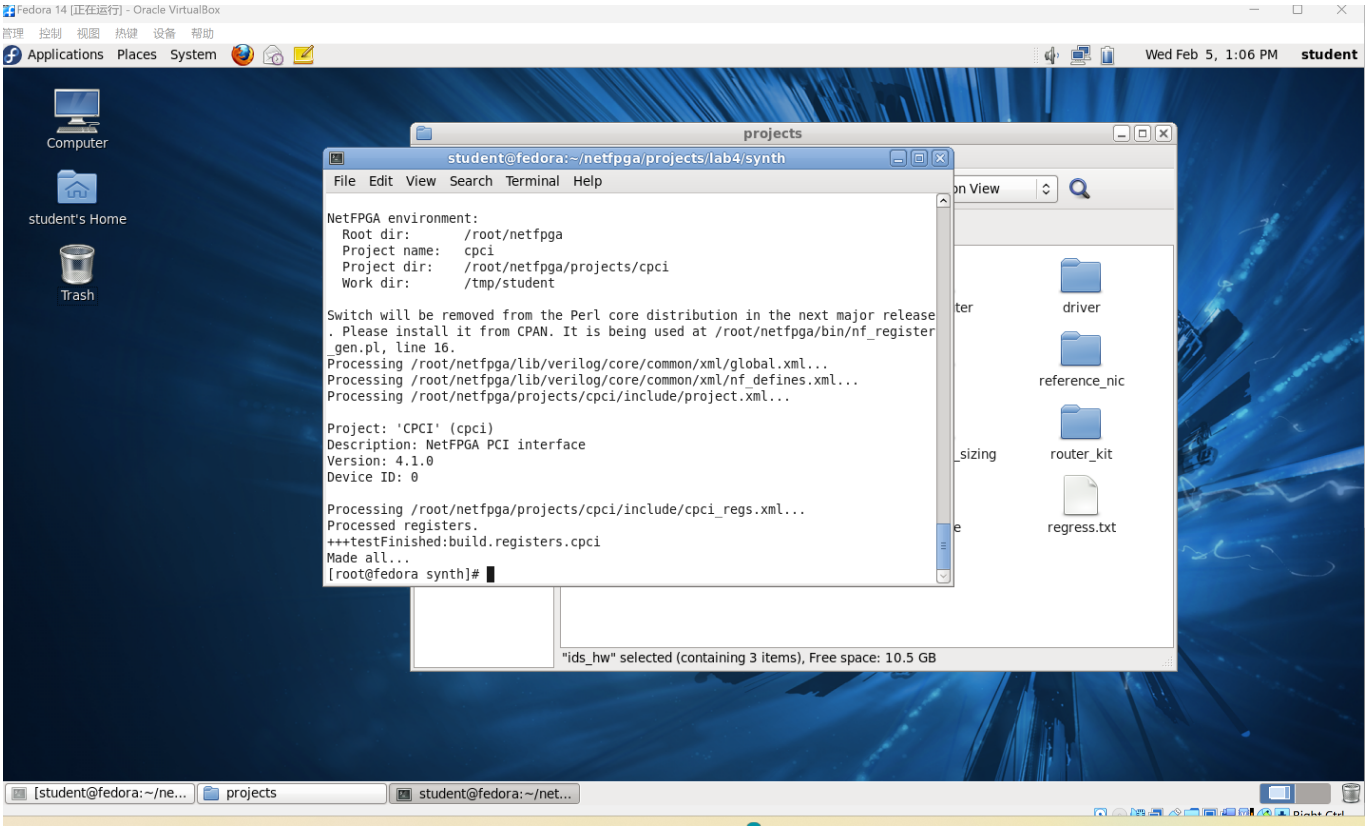
2.1 Extract ids_hw and copy all Verilog and ip core files into ids_hw/src



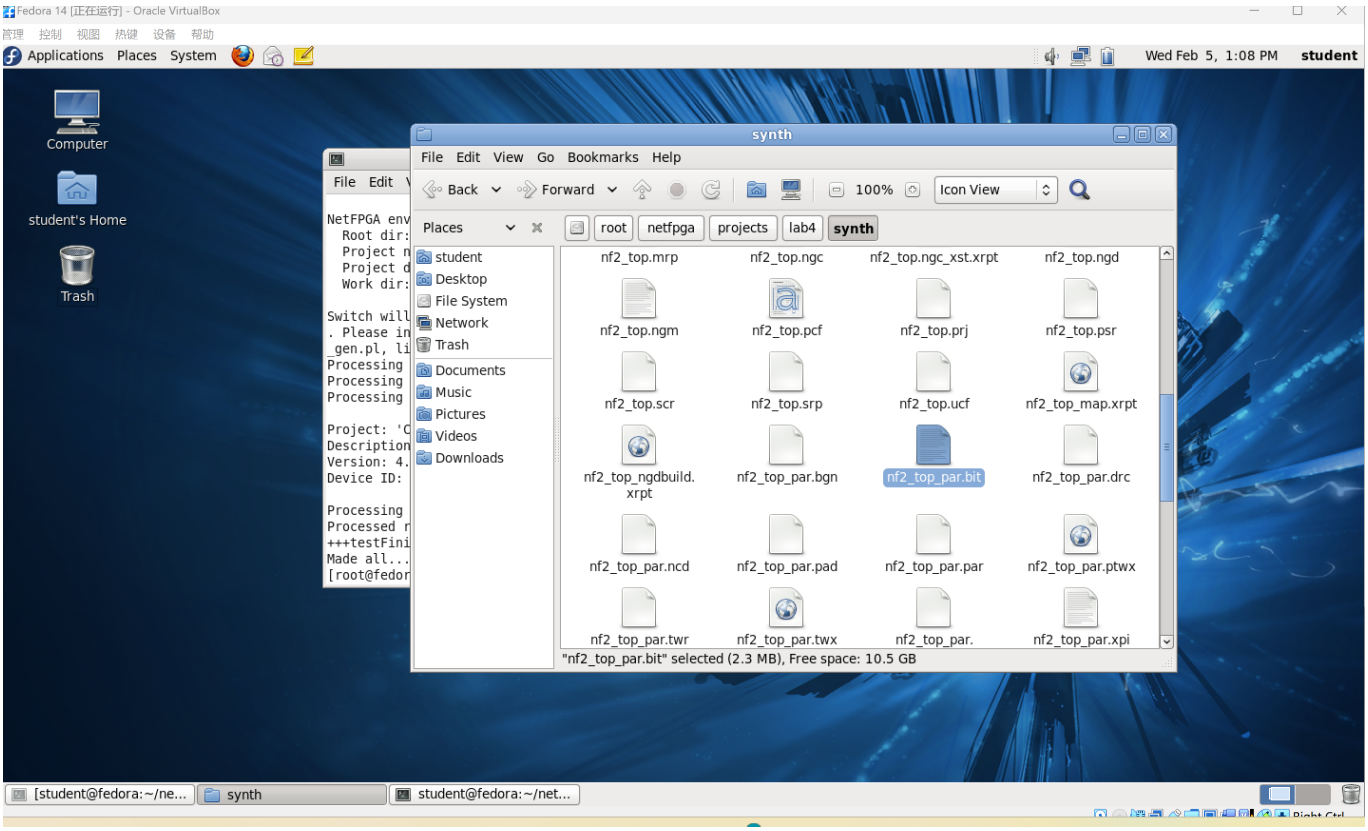
2.2 Copy all of source files within ids_hw into corresponding folders into NetFPGA project folder



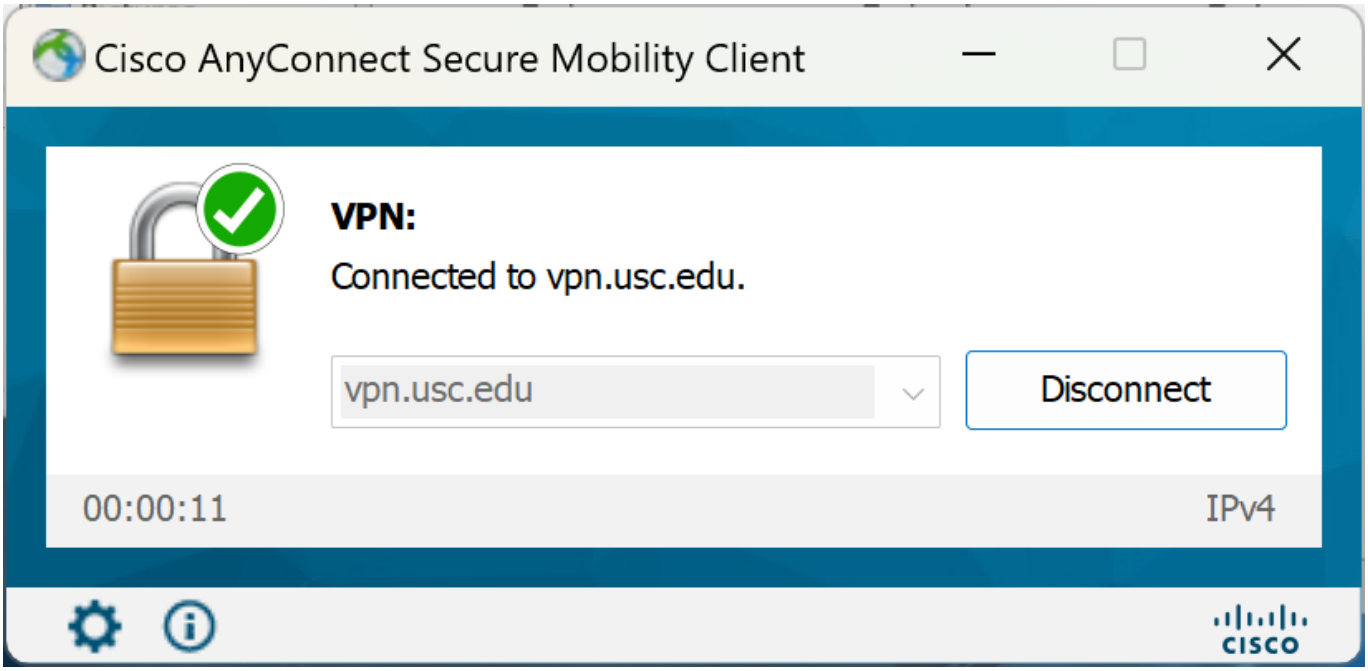
2.3 Compile the design by 'cd' into the synth folder type 'make'



2.4 '*.bit' file generated

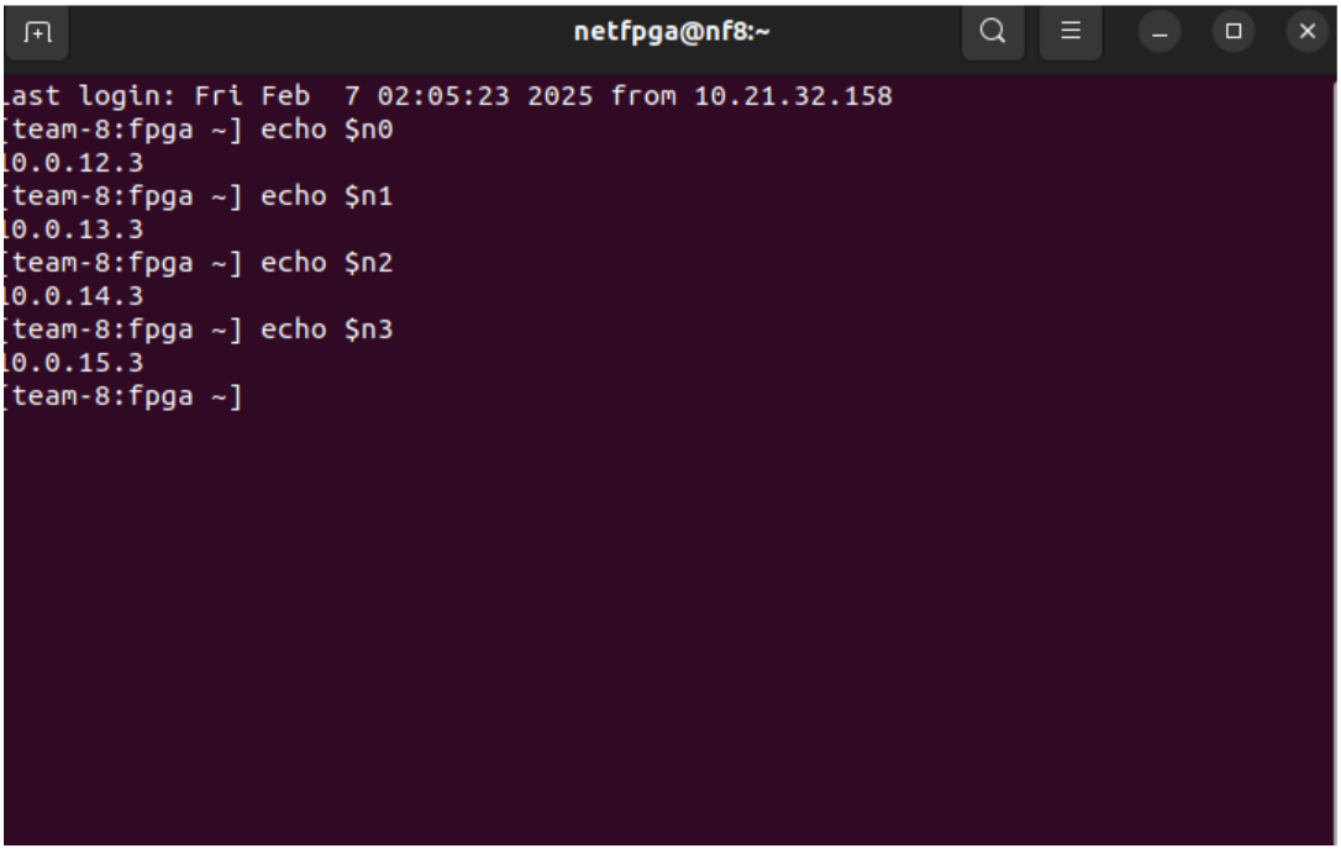


3. Set Up VPN to USC



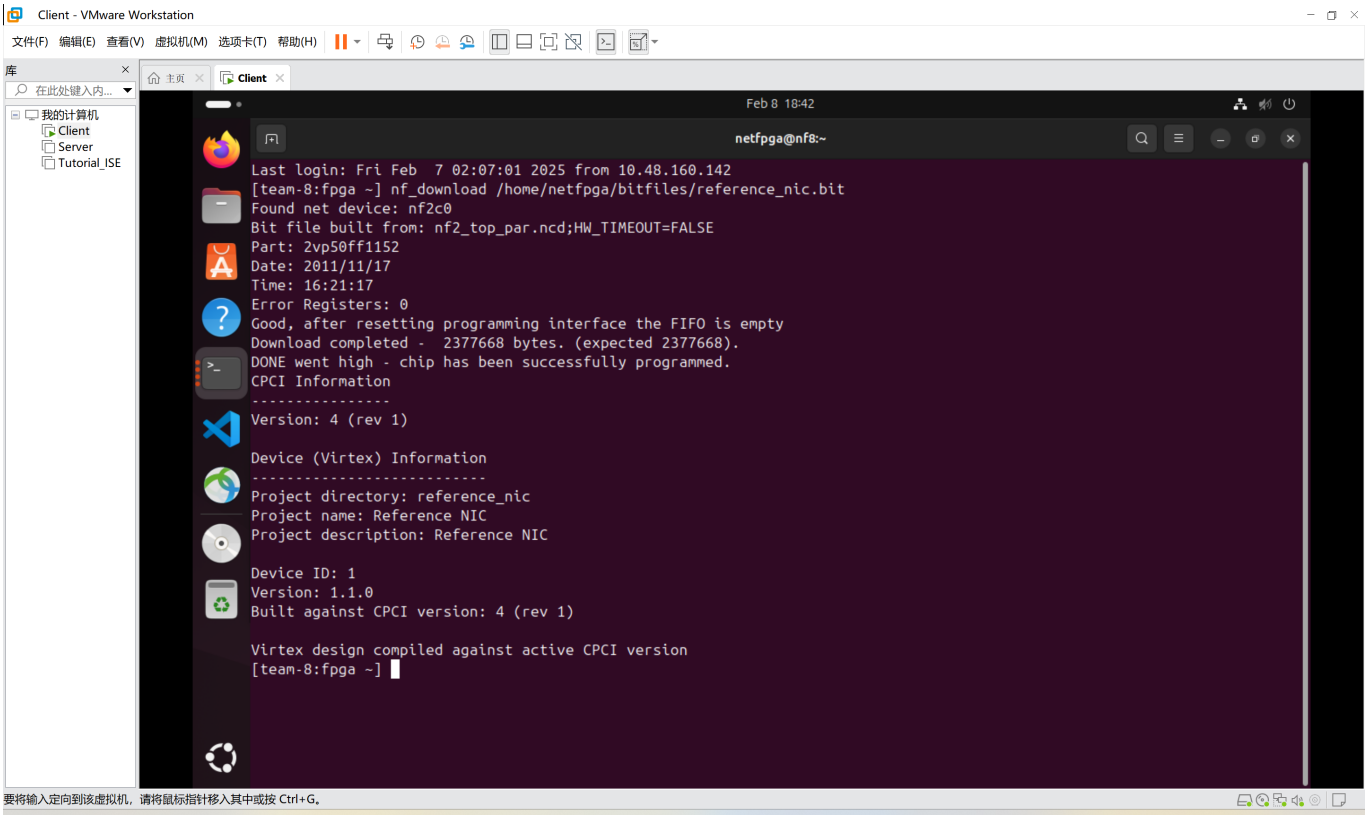
4. NetFPGA Environment

echo test results

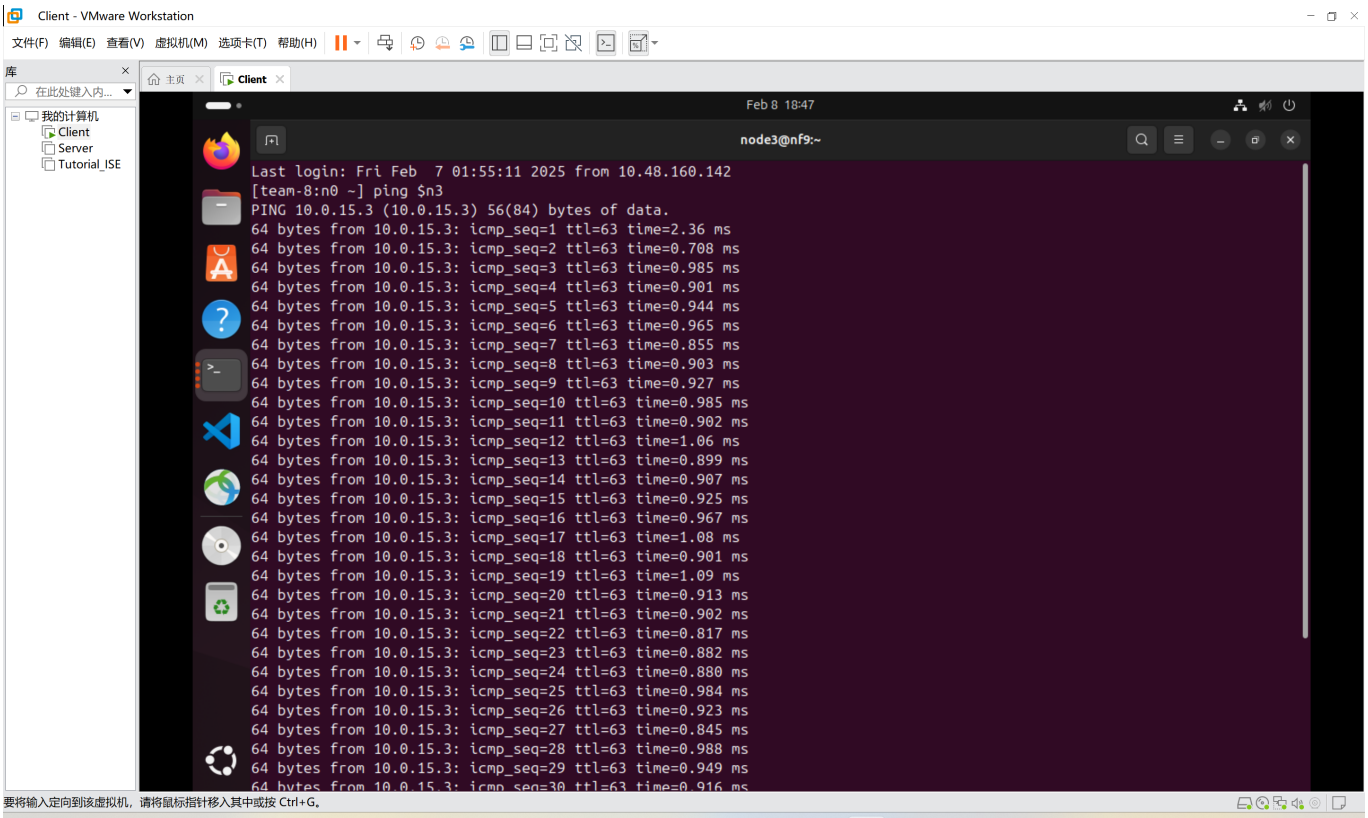


5. NetFPGA-based Linux Kernel IP Router

5.1 Test NetFPGA as network interface card

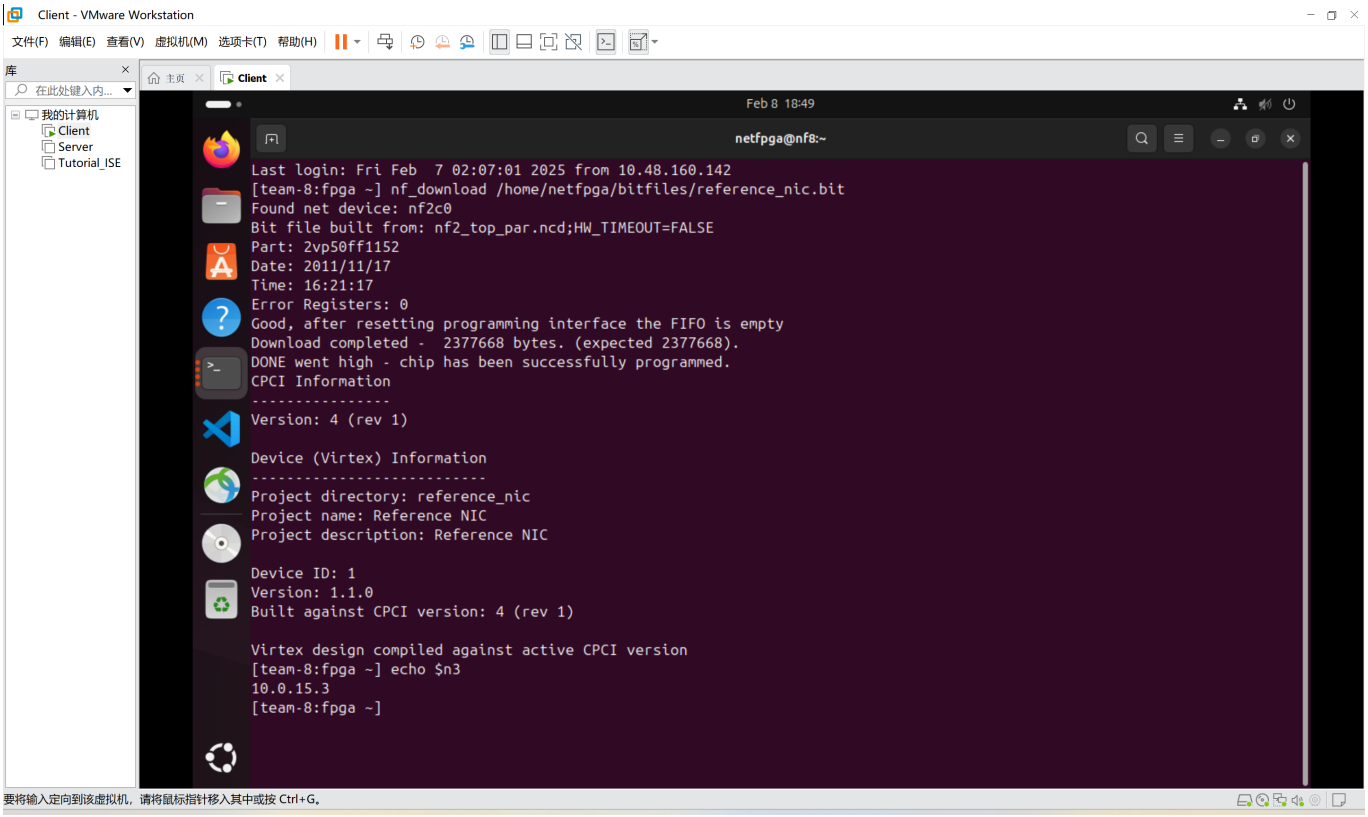


5.2 Test ping from n0 to n3

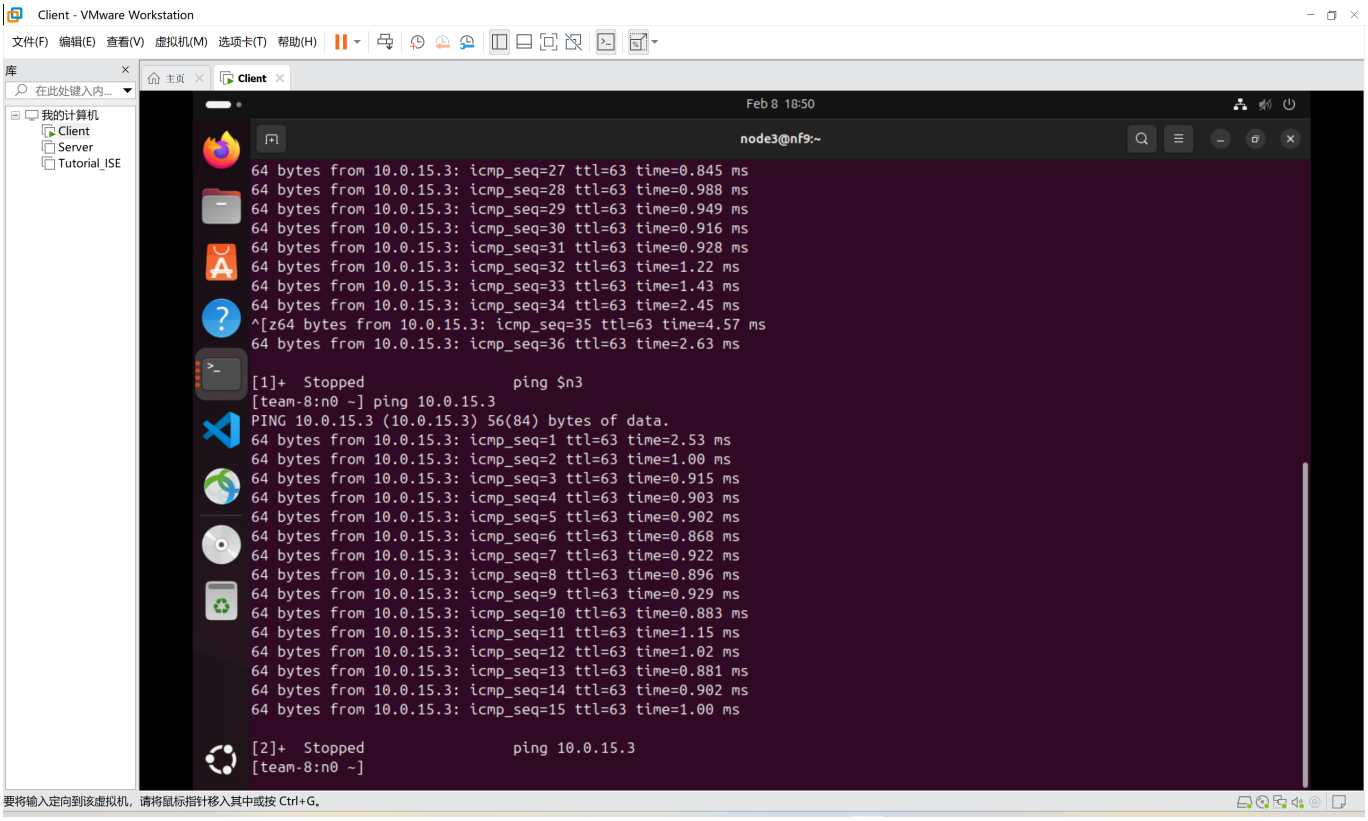


5.3 Test ping from n0 to n3 by typing IP Address

- Checking n3 IP Address
  - 10.0.15.3

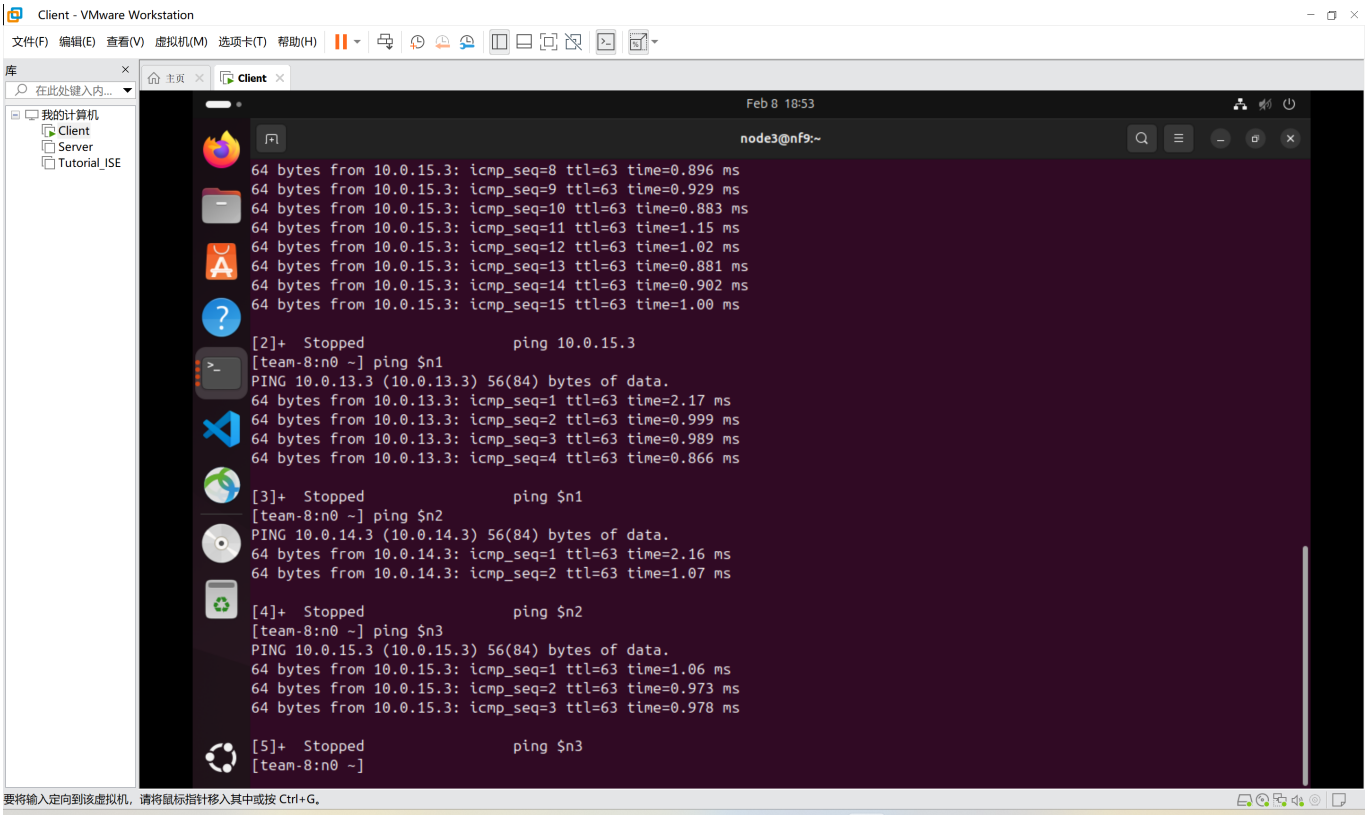


- Ping n3 by IP Address

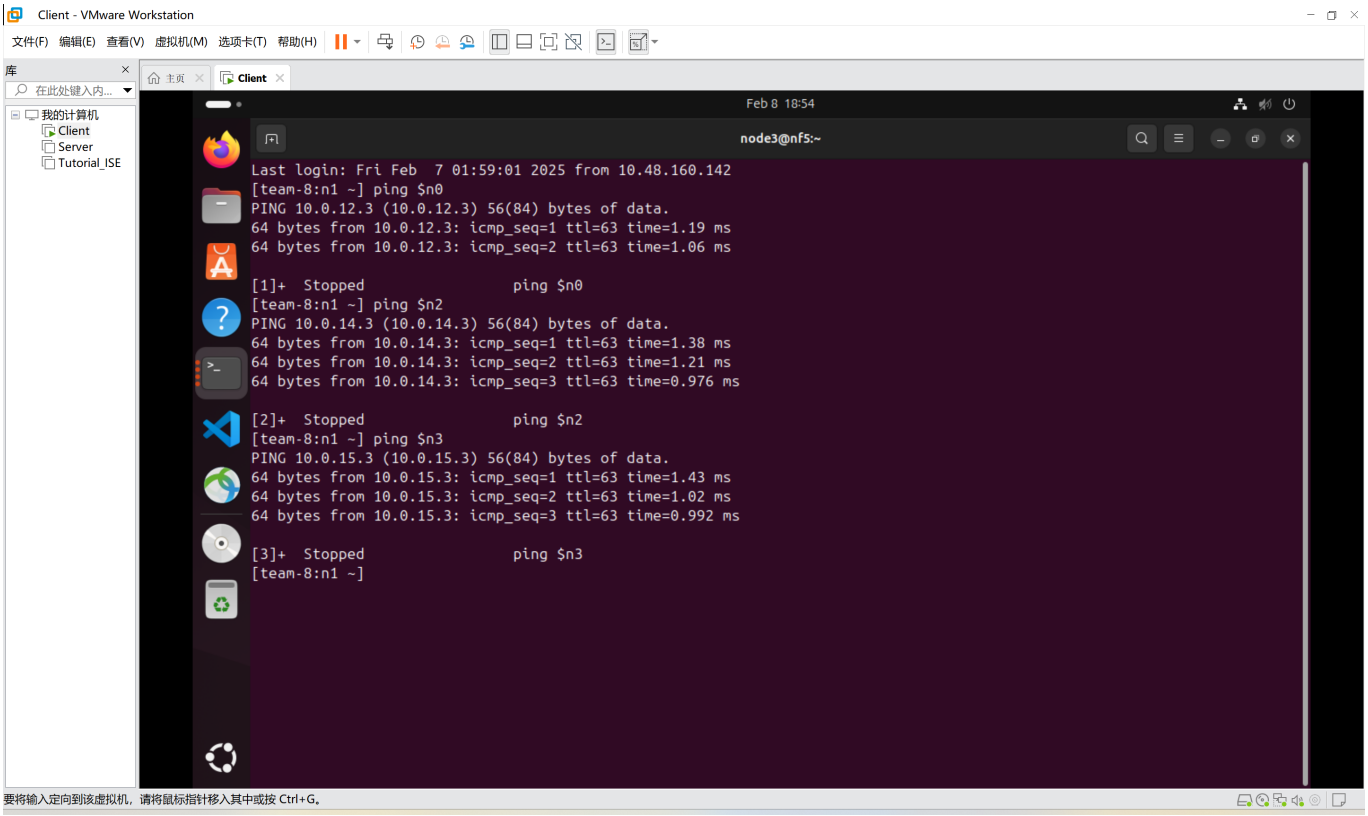


5.4 Test Ping all other nodes from one to another

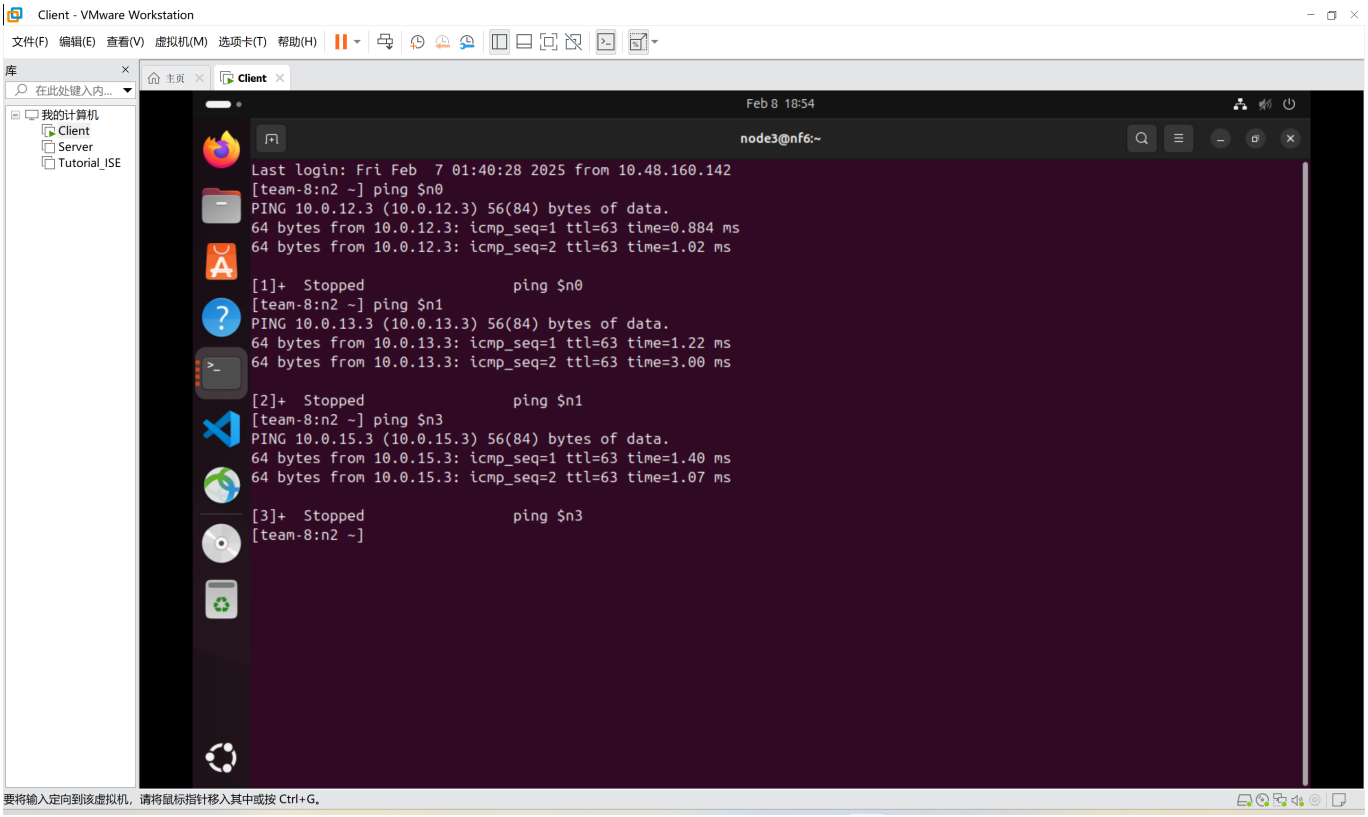
5.4.1 n0 -> n1, n2, n3



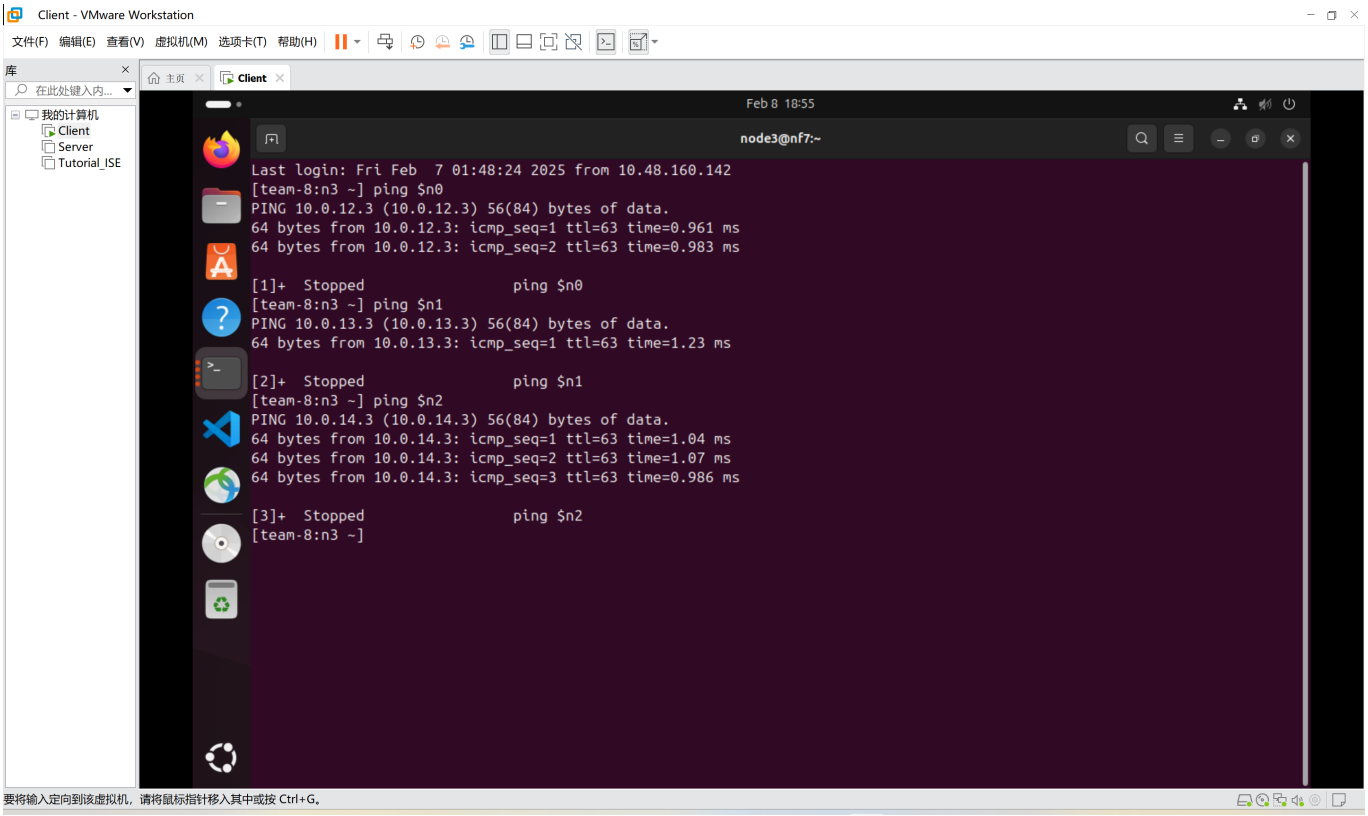
5.4.2 n1 -> n0, n2, n3



5.4.3 n2 -> n0, n1, n3

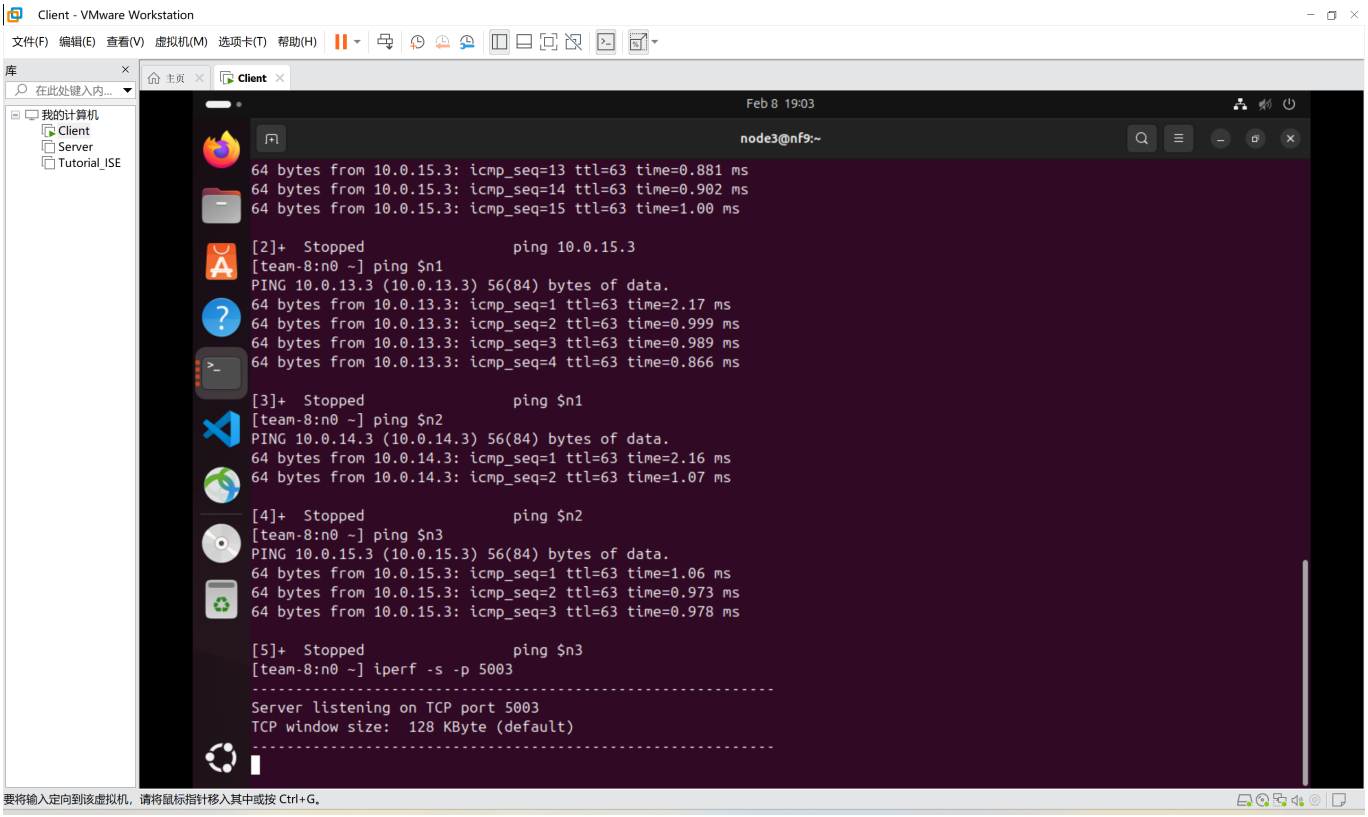


5.4.4 n3 -> n0, n1, n2

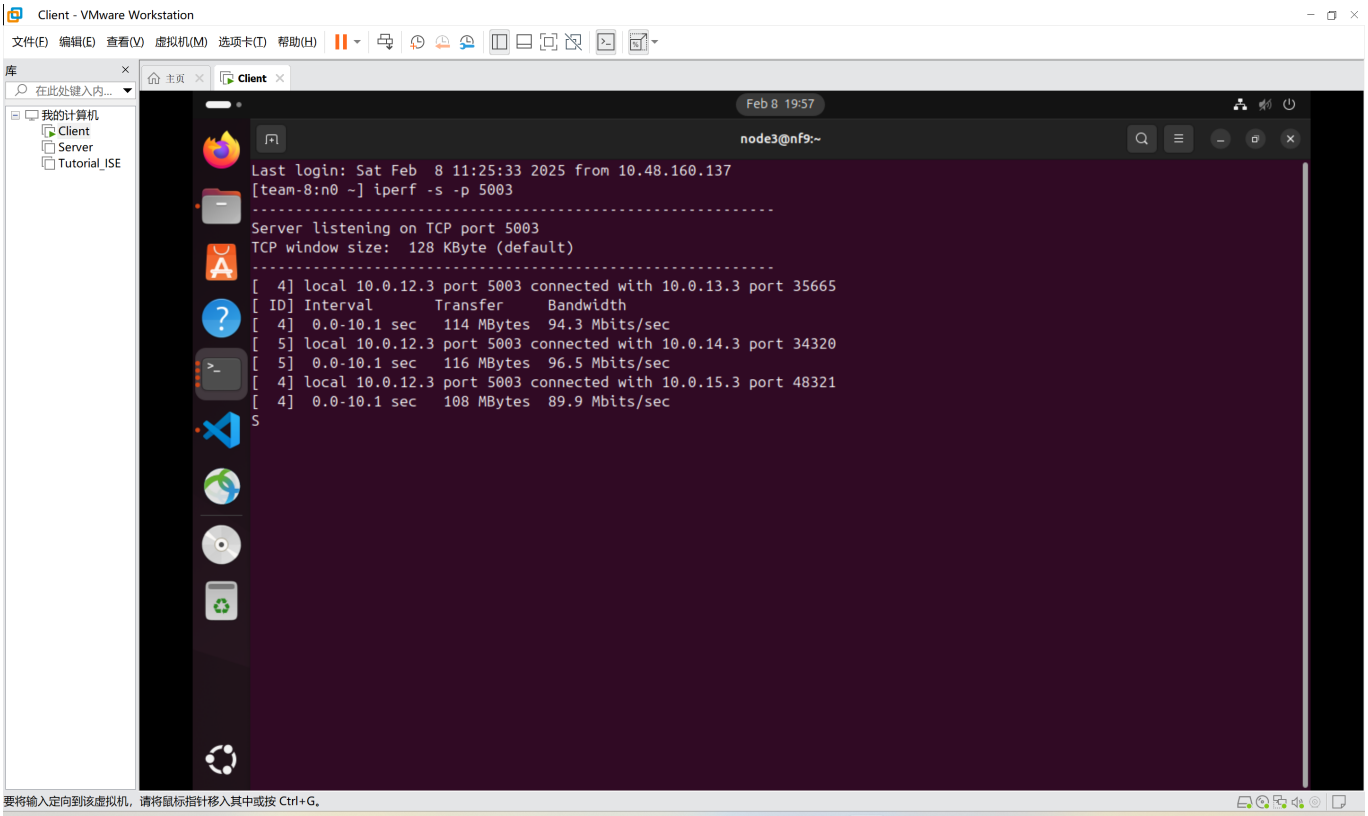


5.5 Test with iperf

- Set n0 as TCP Server, the port number is 5003



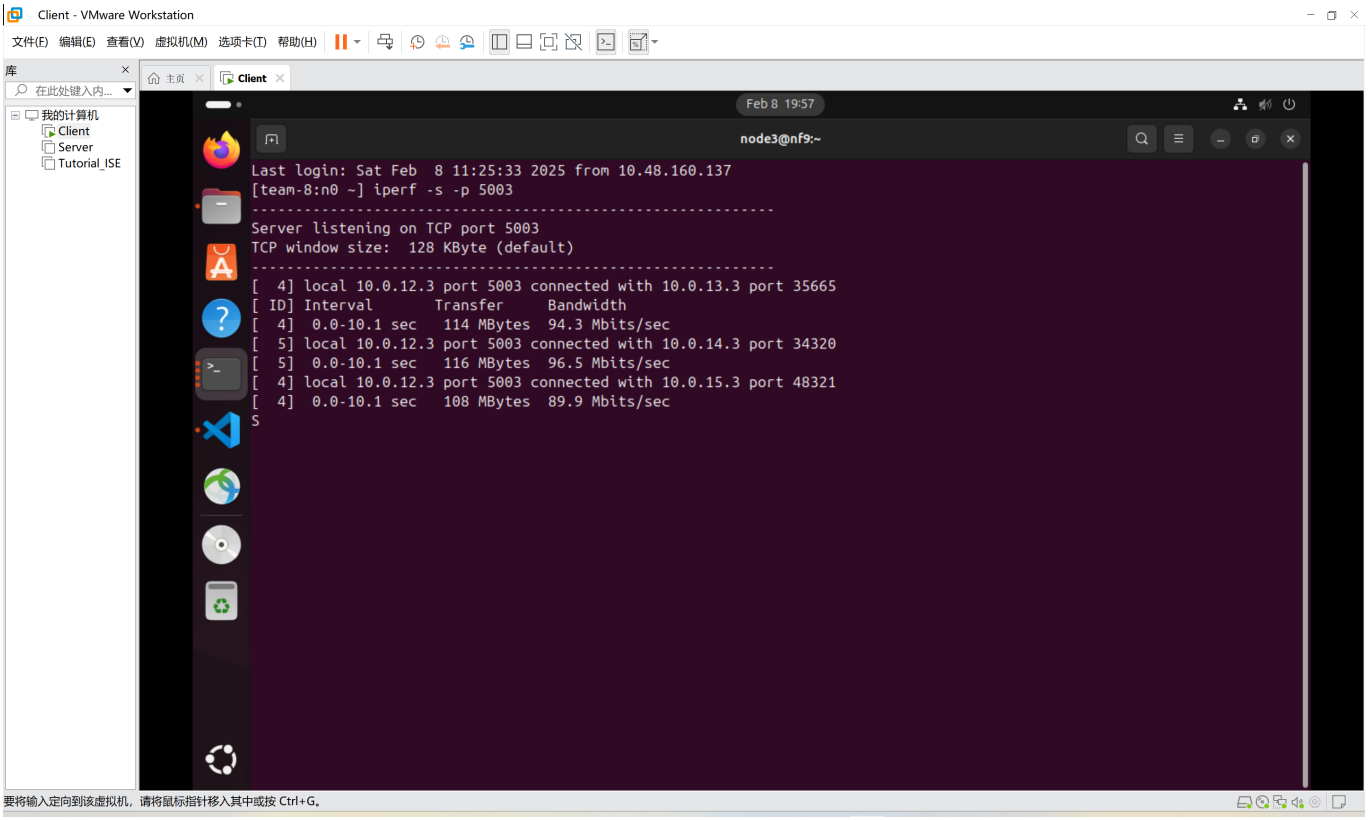
- Set n1, n2, n3 as TCP Client, and set connection with the server n0 via port 5003



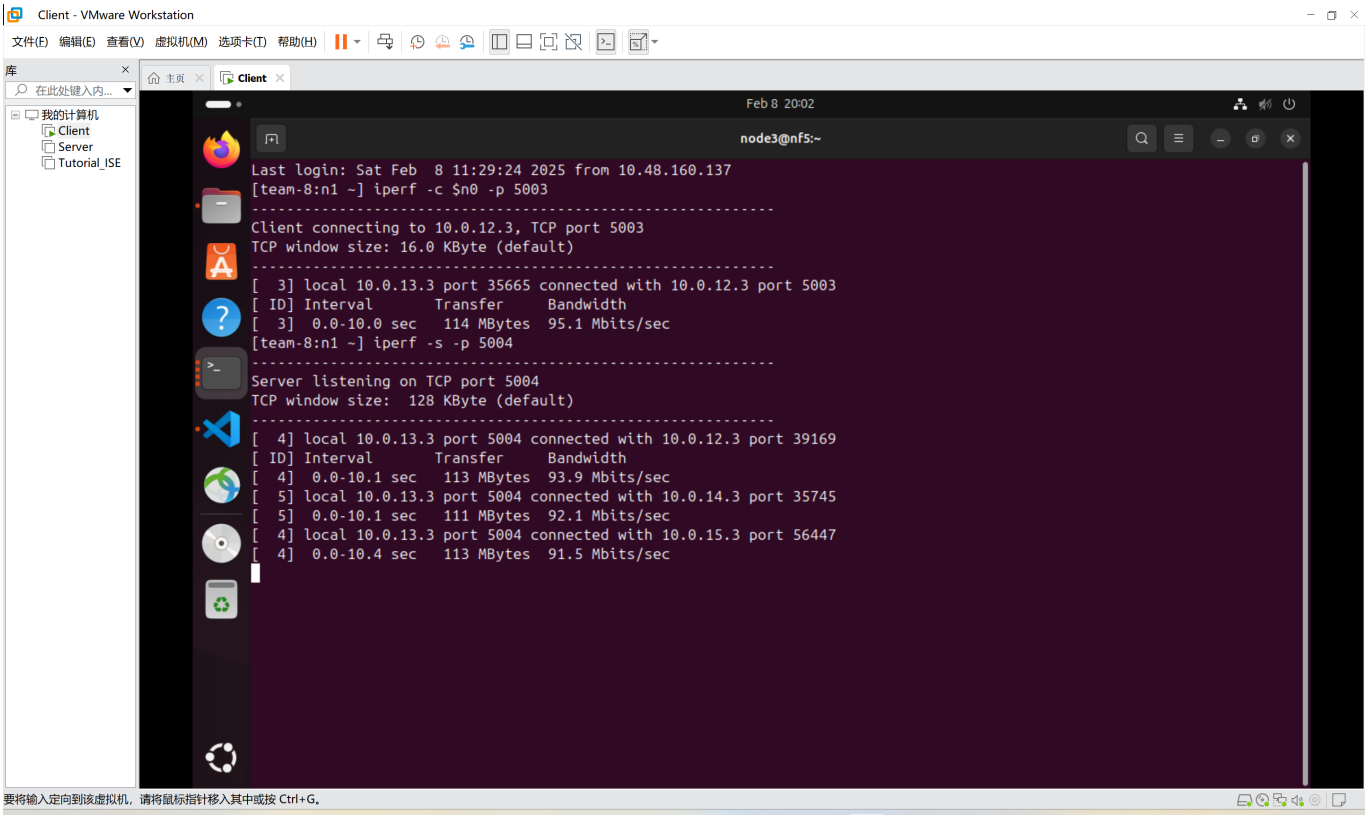
Server	Client	Protocol	Bandwidth (Mbits/sec)
n0	n1	TCP	94.3
n0	n2	TCP	96.5
n0	n3	TCP	89.9

5.6 Test iperf with script

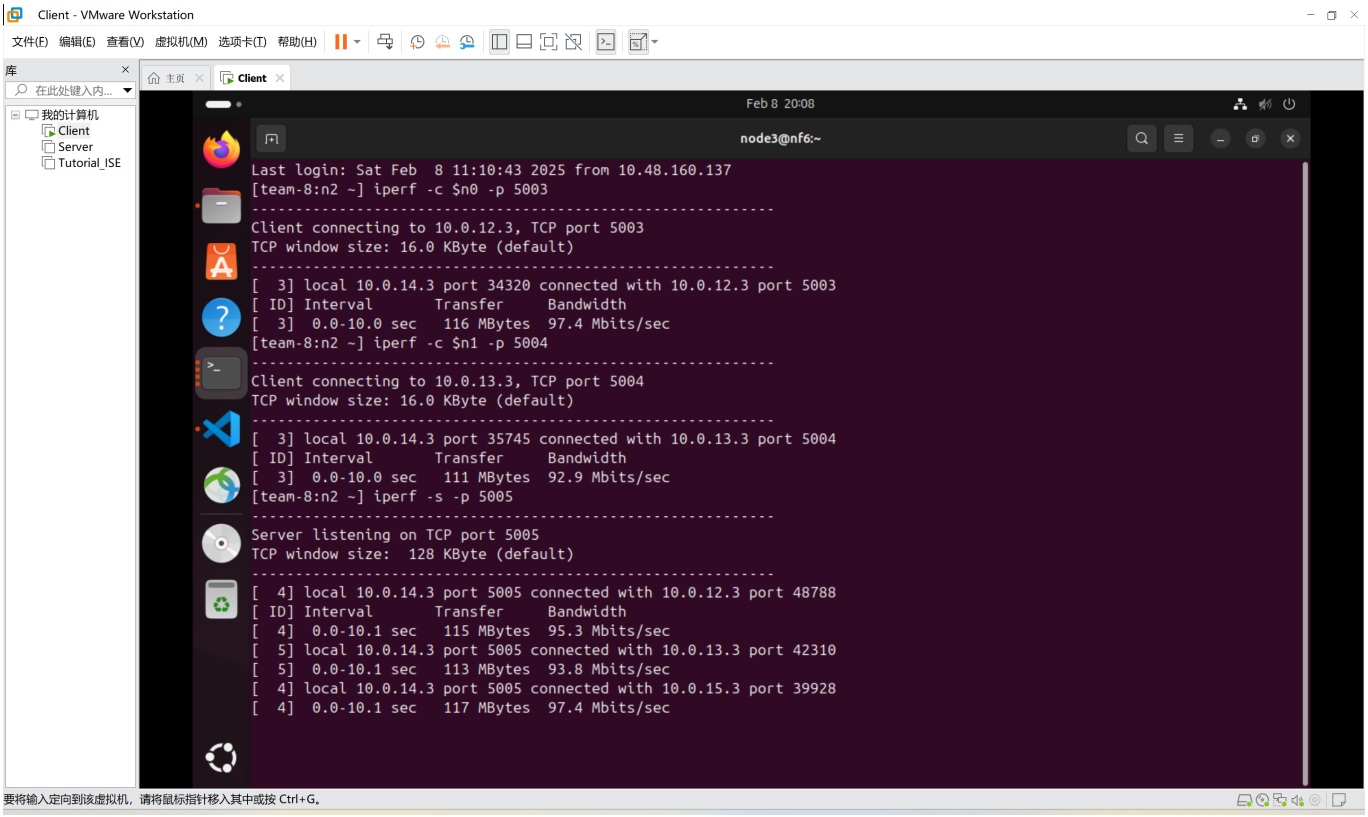
- n0 as TCP server



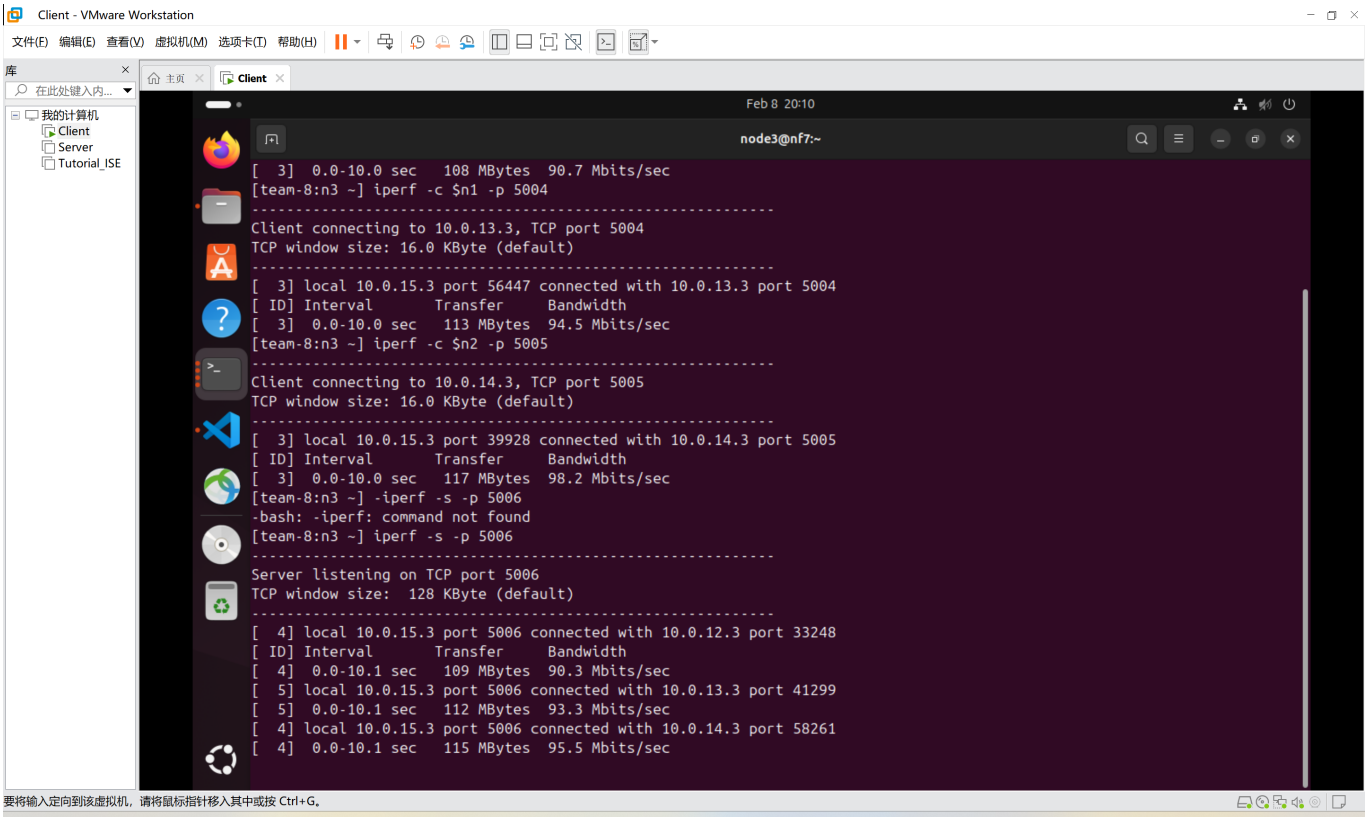
- n1 as TCP server



- n2 as TCP server



- n3 as TCP server

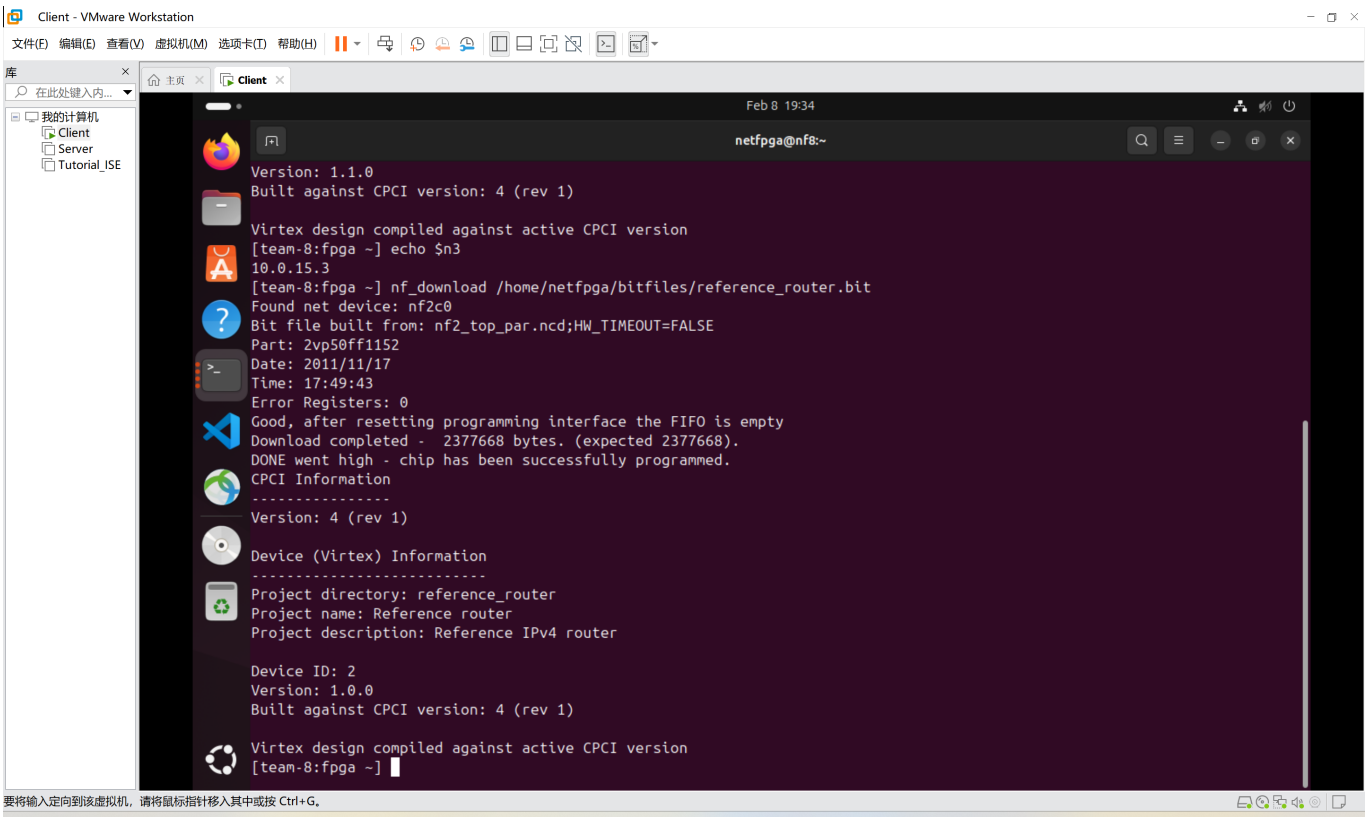


Server	Client	Protocol	Bandwidth (Mbits/sec)
n0	n1	TCP	94.3
n0	n2	TCP	96.5
n0	n3	TCP	89.9

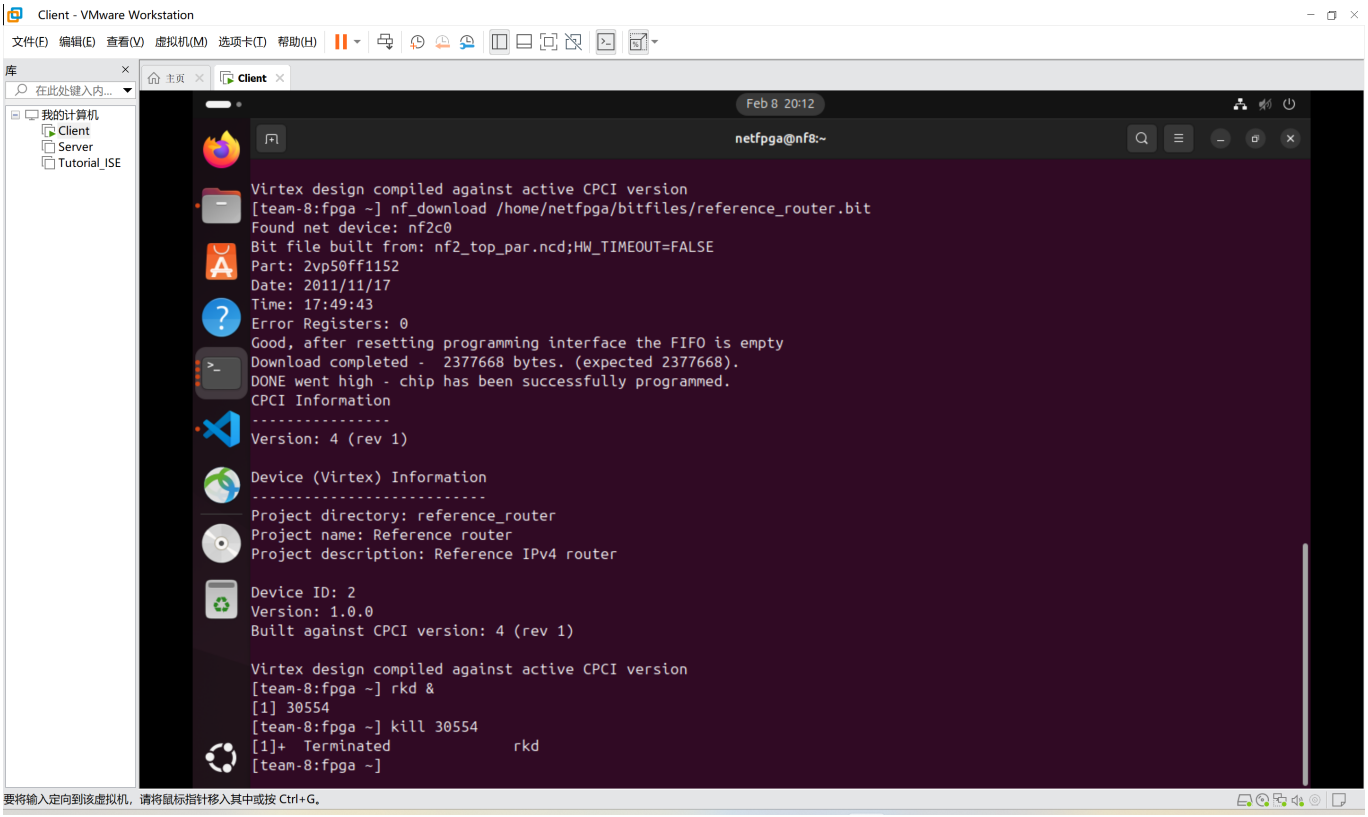
Server	Client	Protocol	Bandwidth (Mbits/sec)
n1	n0	TCP	93.9
n1	n2	TCP	92.1
n1	n3	TCP	91.5
n2	n0	TCP	95.3
n2	n1	TCP	93.8
n2	n3	TCP	97.4
n3	n0	TCP	90.3
n3	n1	TCP	93.3
n3	n2	TCP	95.5

6. NetFPGA Hardware IP Router

6.1 Download reference router into NetFPGA

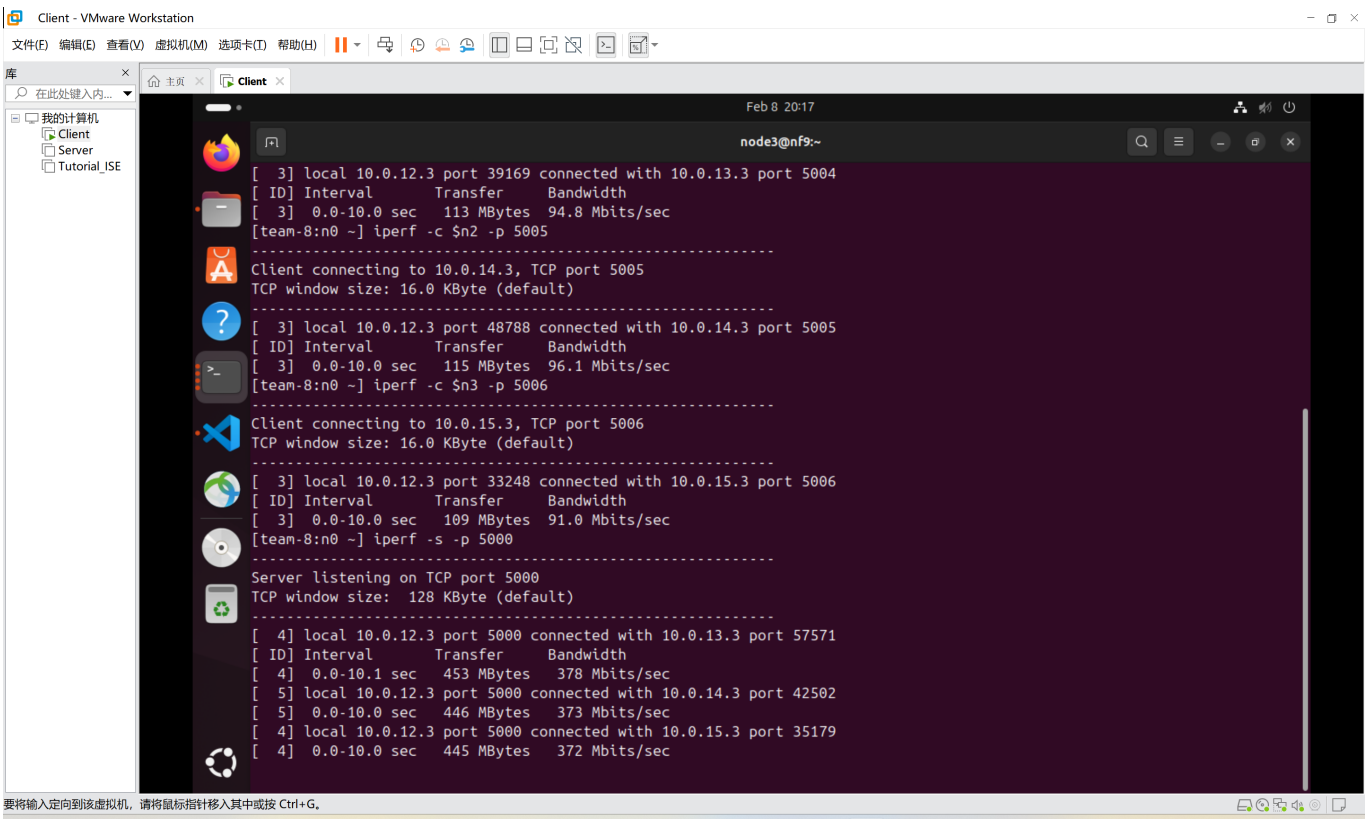


6.2 Check PID and Kill

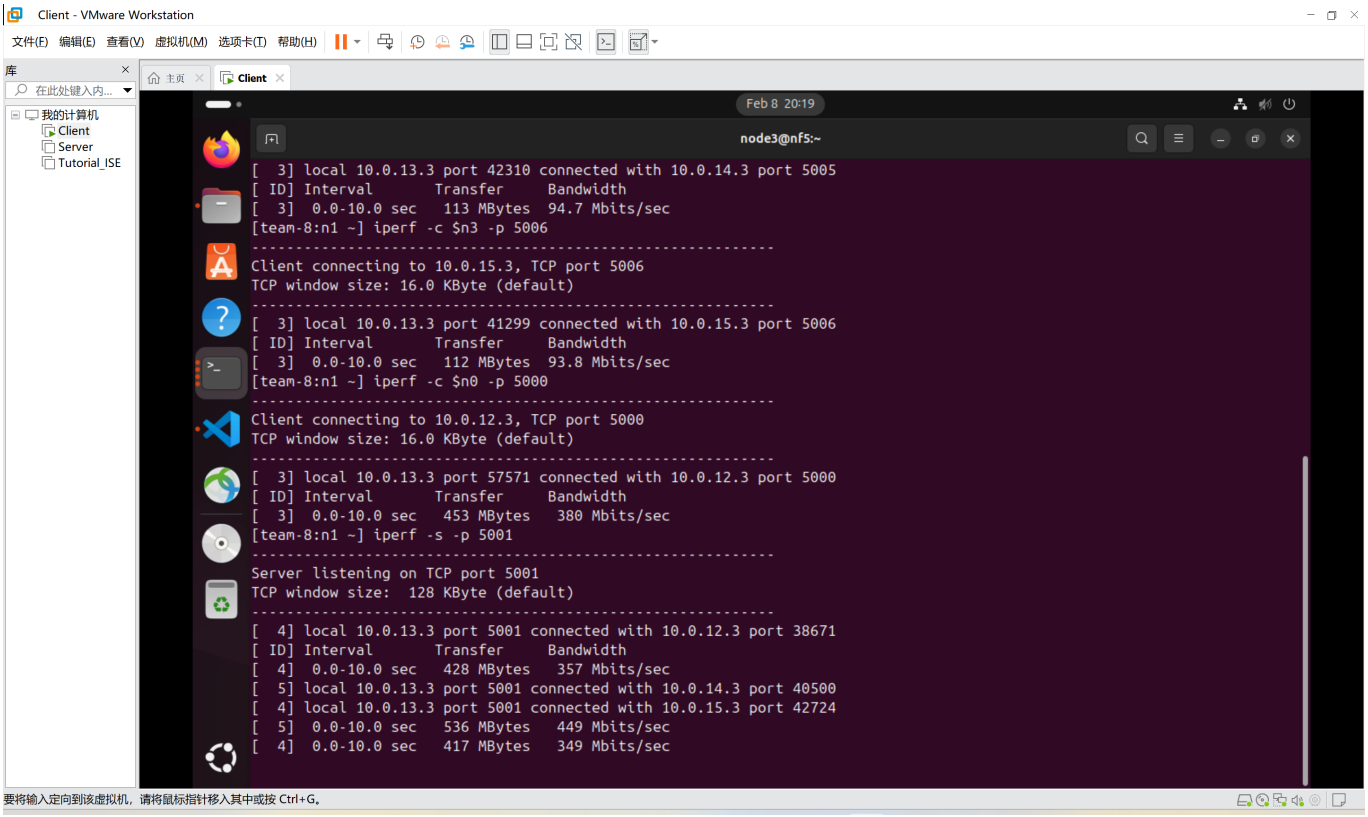


6.3 Re-Run the iperf with reference_router.bit and check Bandwidth

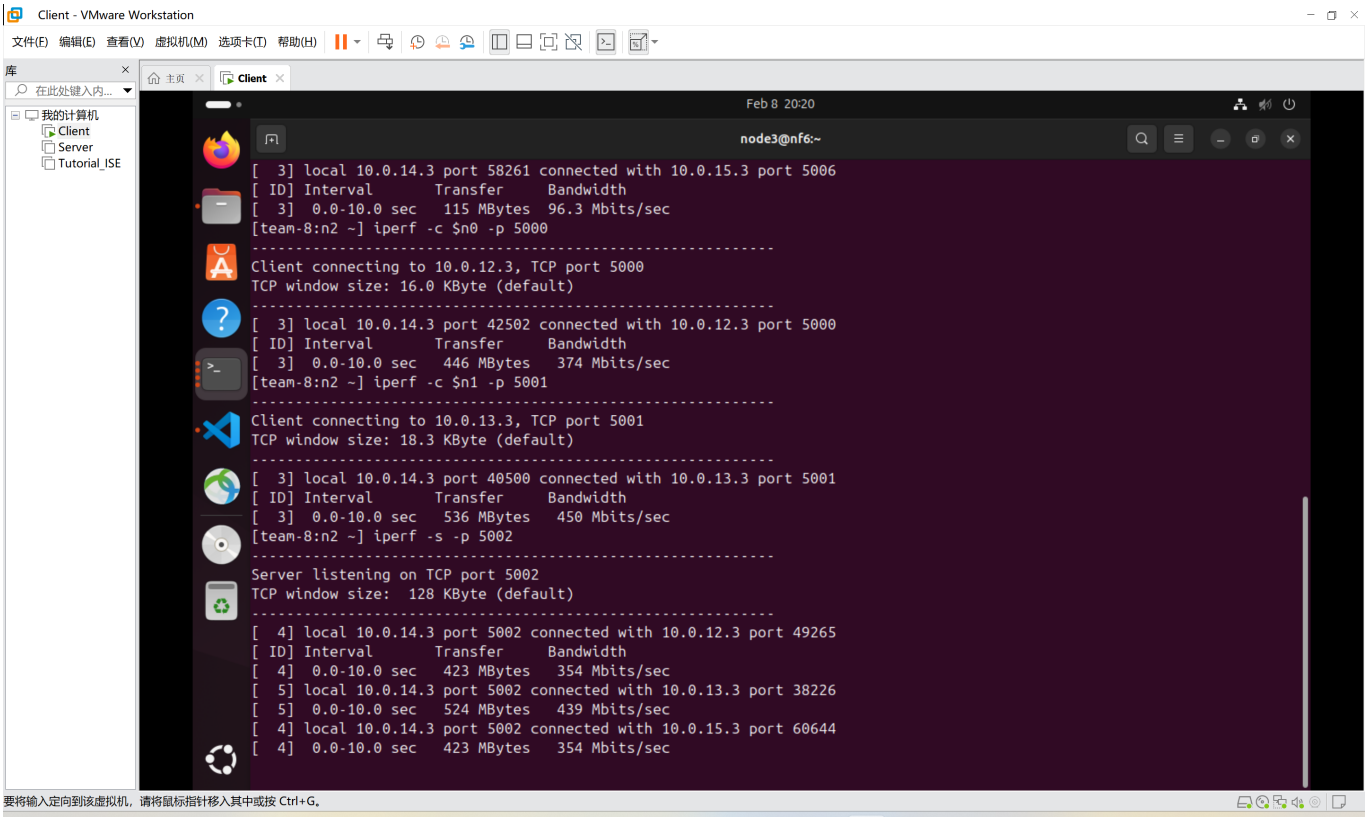
- n0 as TCP server



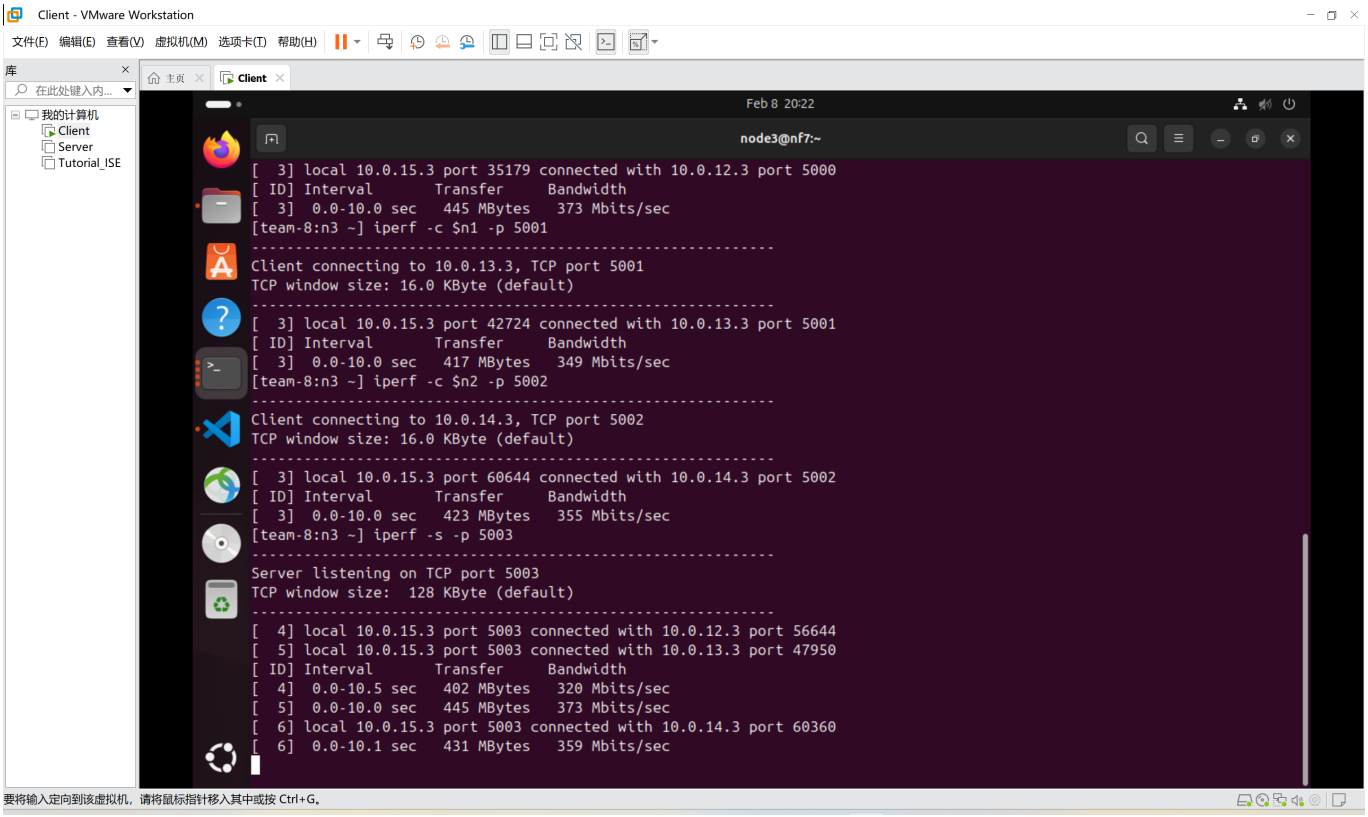
- n1 as TCP server



- n2 as TCP server



- n3 as TCP server



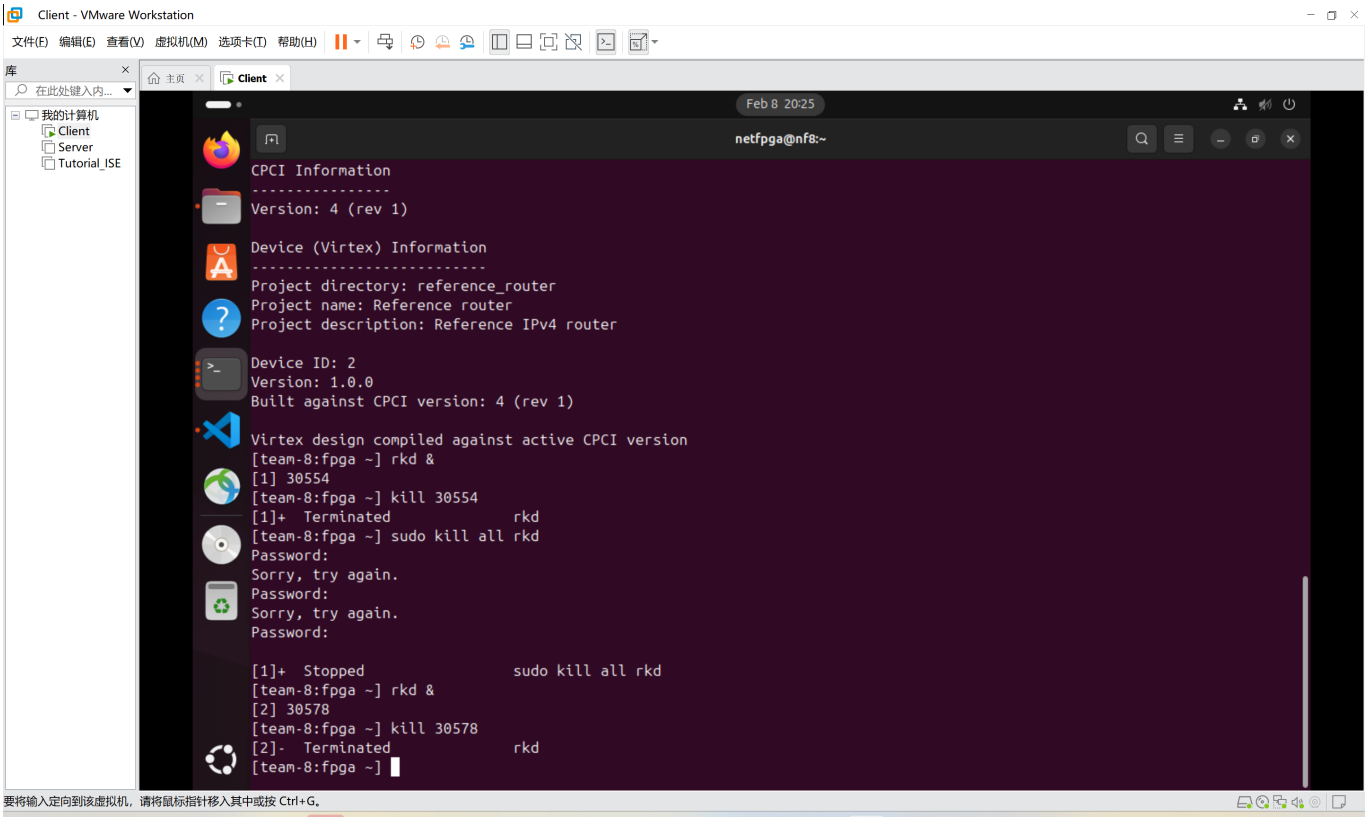
Server	Client	Protocol	Bandwidth (Mbits/sec)
n0	n1	TCP	378
n0	n2	TCP	373
n0	n3	TCP	372
n1	n0	TCP	357
n1	n2	TCP	449
n1	n3	TCP	349
n2	n0	TCP	354
n2	n1	TCP	439
n2	n3	TCP	354
n3	n0	TCP	320
n3	n1	TCP	373
n3	n2	TCP	359

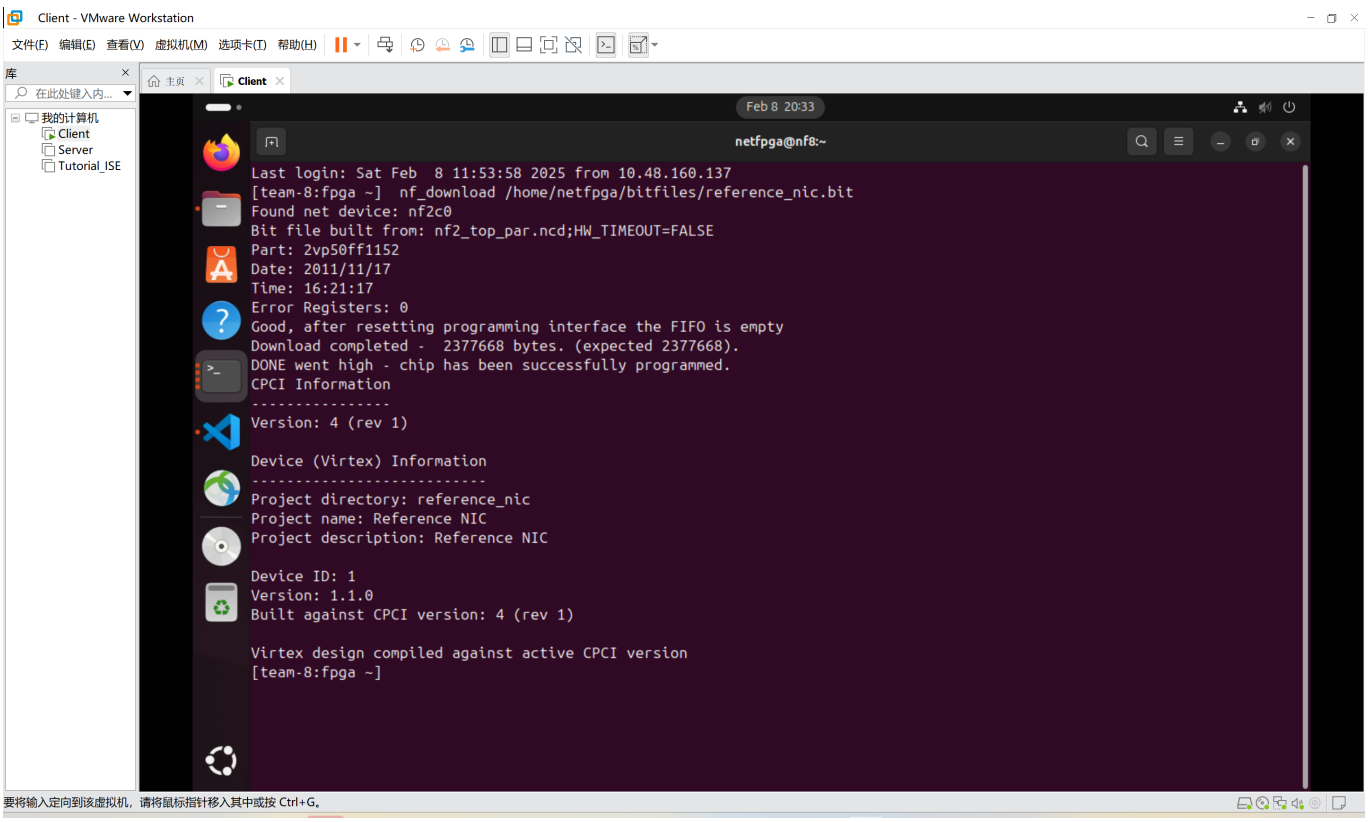
- Has the bandwidth changed? If so, why?

we can see that the bandwith went up by a lot, and that we guess the reason could be that since the netfpga now already knows about the routes and configurations, it would make sending packages between each nodes much quicker since it doesn't have to sent the header to all the nodes and wait for the right nodes to response, hence the latency and bandwith could improve, and therefore the speed improvements

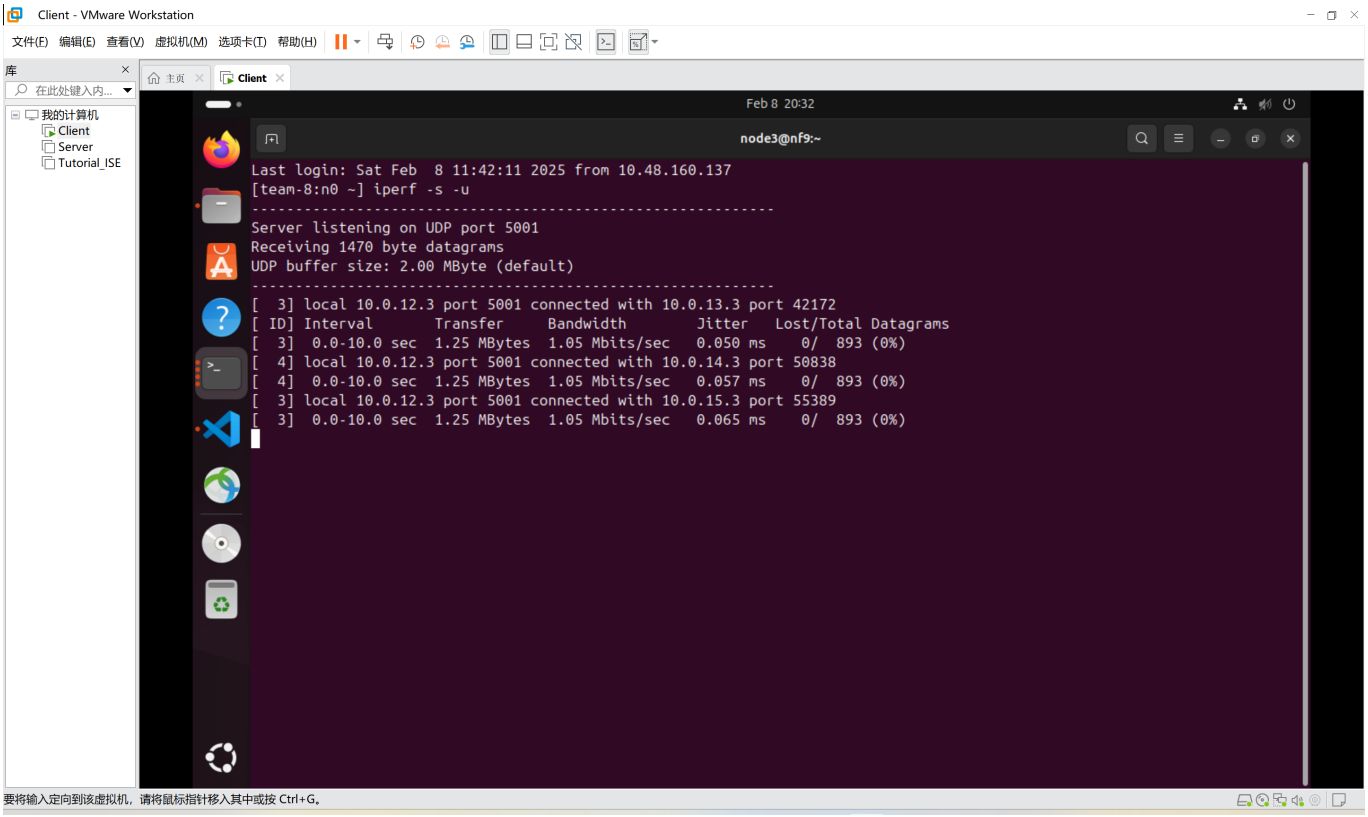
6.4 Re-Run the iperf with reference_nic.bit and check Bandwidth

- Kill rkd & download reference_nic.bit

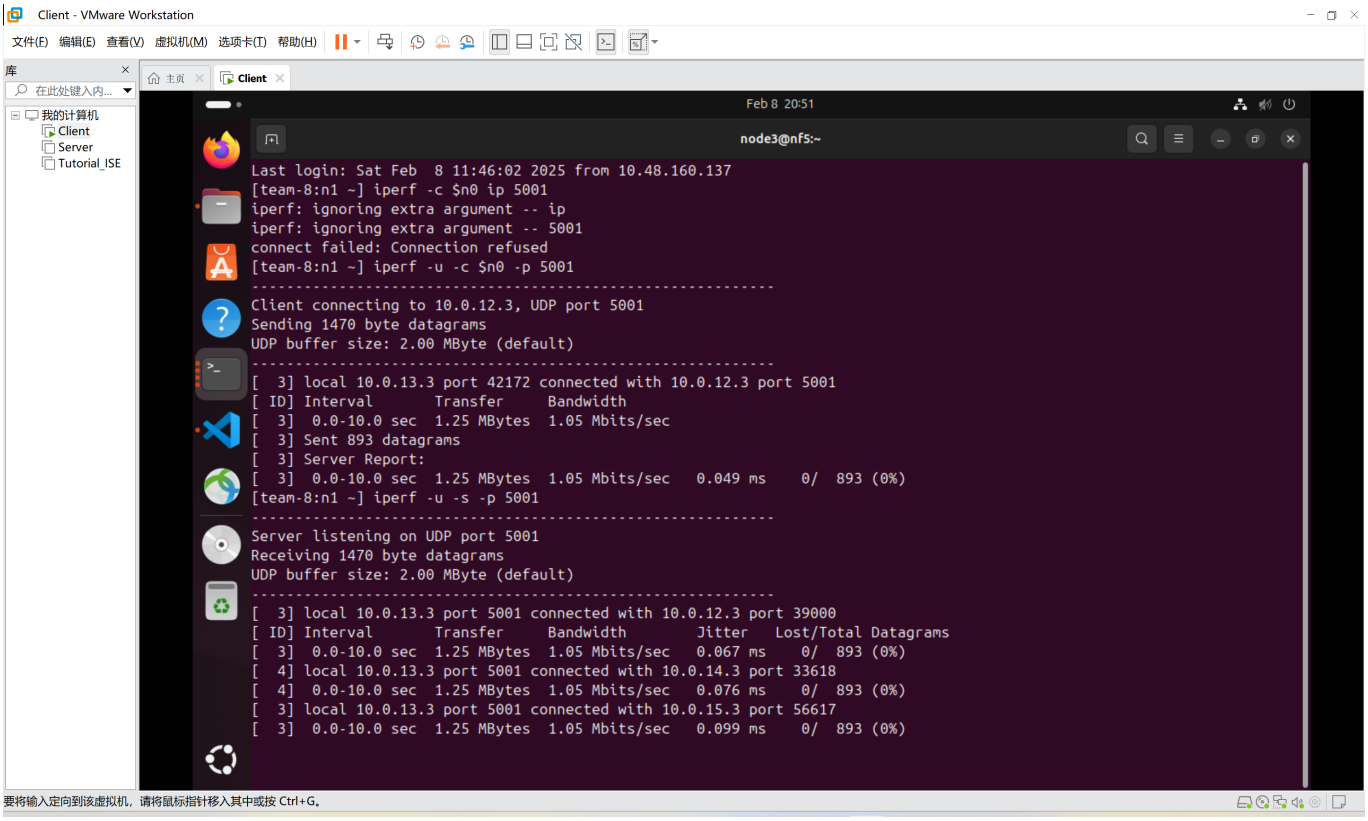




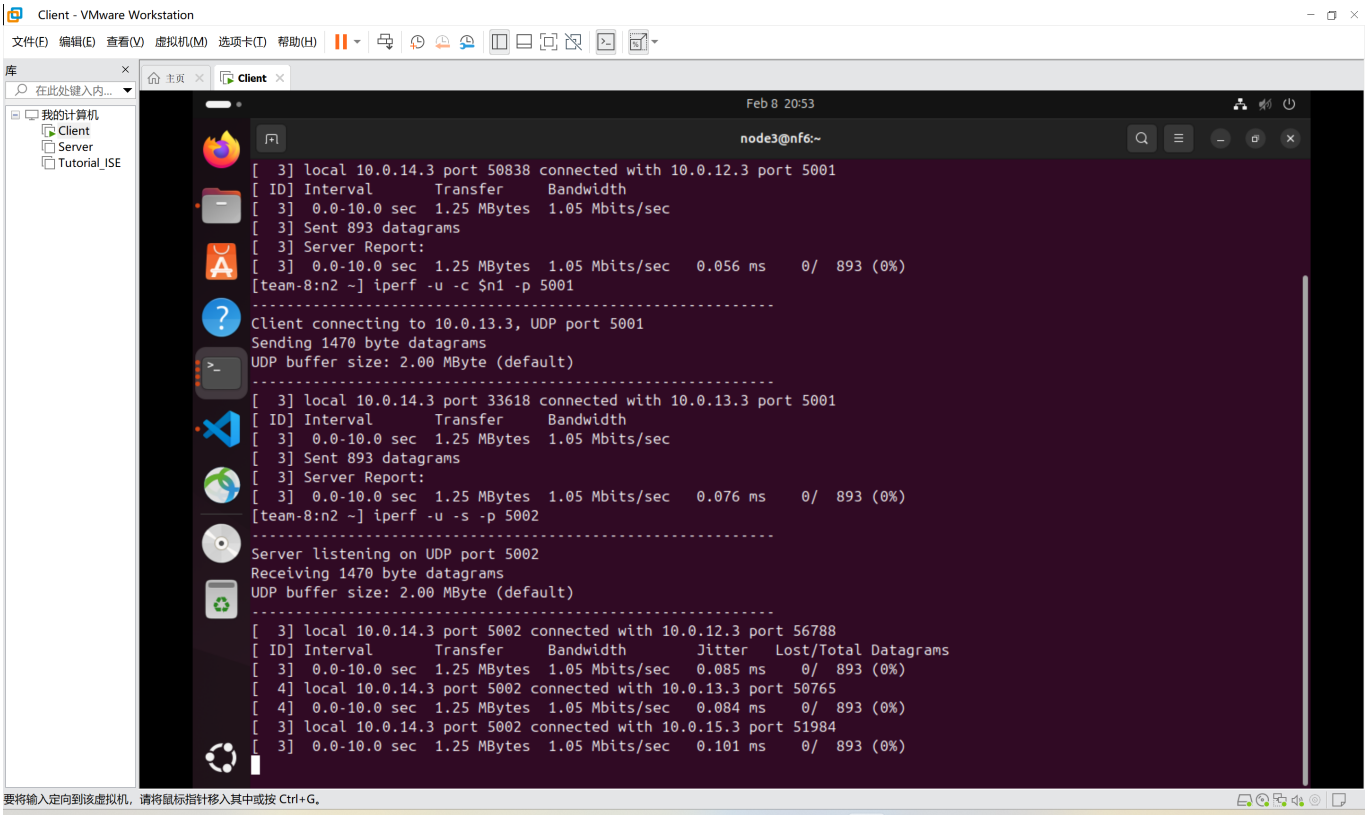
- n0 as UDP server



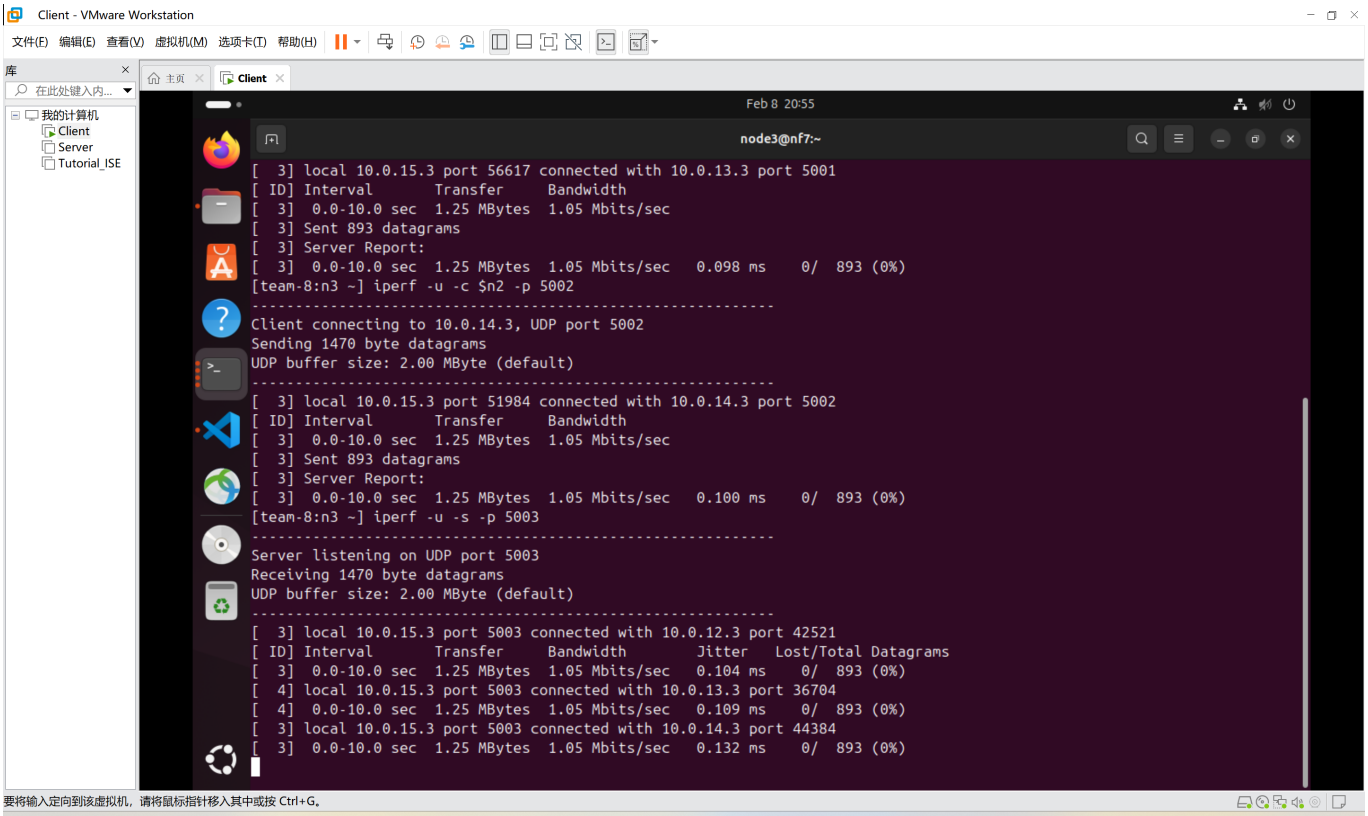
- n1 as UDP server



- n2 as UDP server



- n3 as UDP server



Server	Client	Protocol	Bandwidth (Mbits/sec)
n0	n1	UDP	1.05
n0	n2	UDP	1.05
n0	n3	UDP	1.05

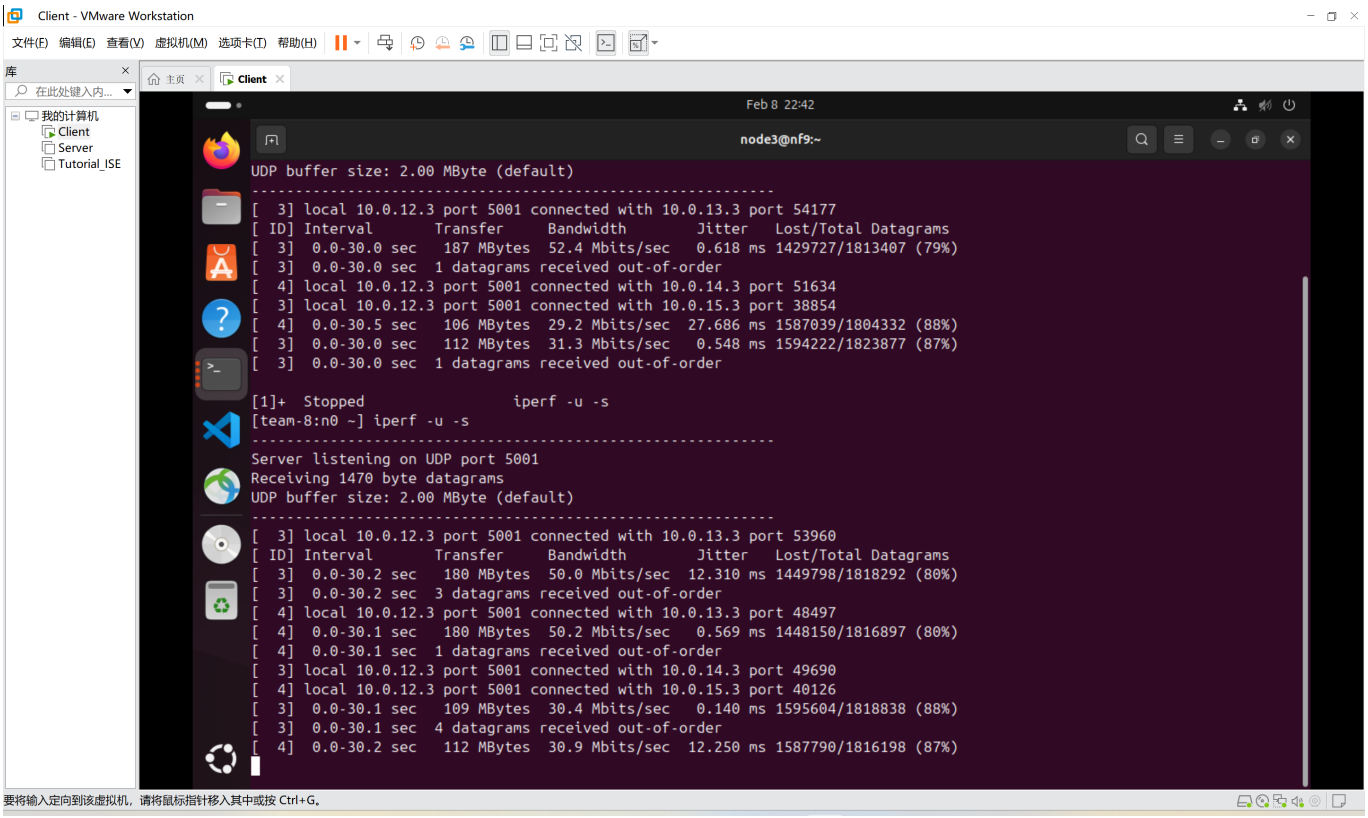
Server	Client	Protocol	Bandwidth (Mbits/sec)
n1	n0	UDP	1.05
n1	n2	UDP	1.05
n1	n3	UDP	1.05
n2	n0	UDP	1.05
n2	n1	UDP	1.05
n2	n3	UDP	1.05
n3	n0	UDP	1.05
n3	n1	UDP	1.05
n3	n2	UDP	1.05

- Why does using small packets stress the system?

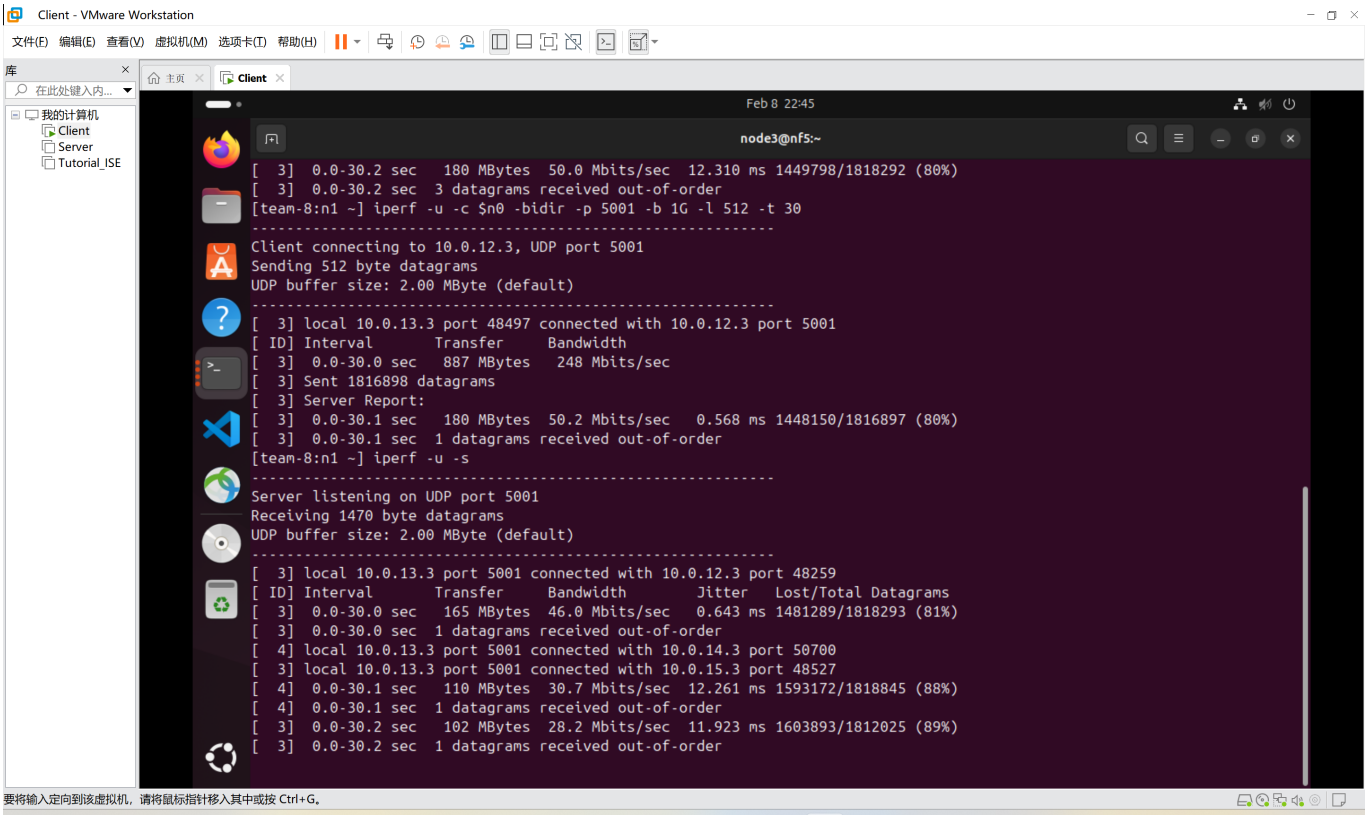
Using small packets means that the router has to put more efferts in analysising each packet's header to decide where to sent them, and since every packet is very small in size, the total bandwidth would be limited by the amount of packet header the router could process every second.

6.5 Testing bidir UDP on each node

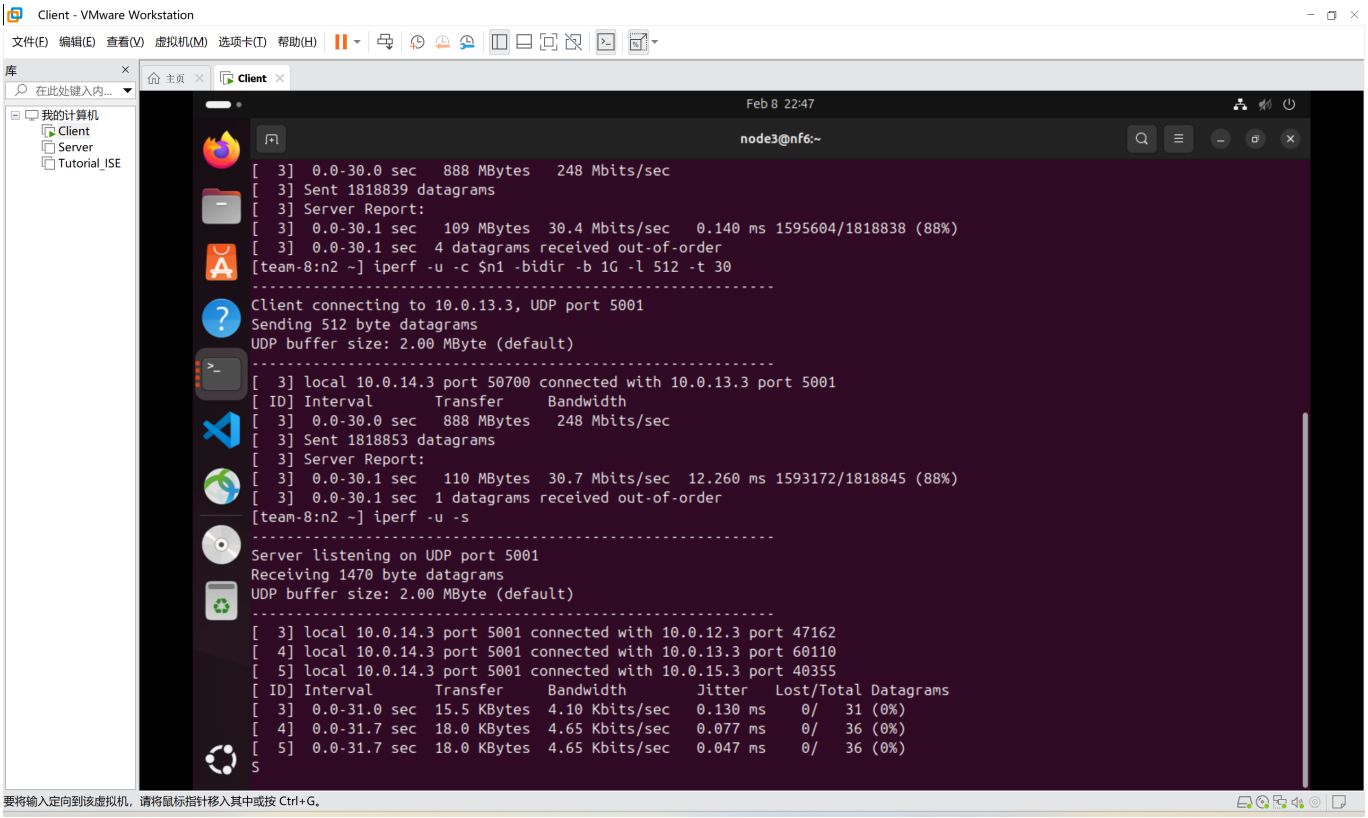
- n0 as UDP server



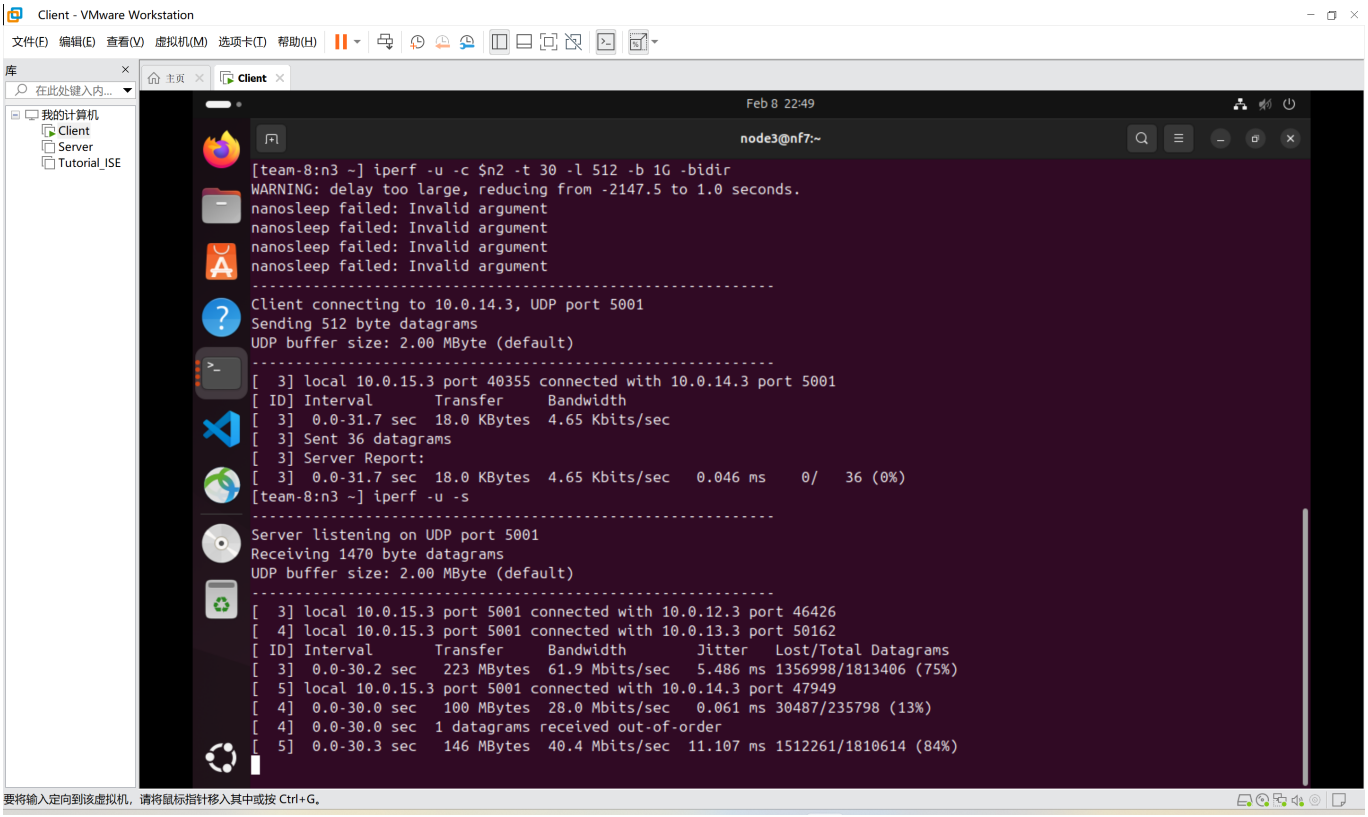
- n1 as UDP server



- n2 as UDP server



- n3 as UDP server



We can see that the speed of the test did improve but did not reach a point that it uses all 4Gbps of bandwidth, so we don't think that its fair to say that the NetFPGA can route IP trafffic bi-directionally at line speed for a total of 4Gbps of cross-wise bandwidth

Extra. GitHub Link

- This Lab's update and commit history could be checked by the below link:
  - https://github.com/yuezhenglingluan/USC_EE533_lab4.git