

# 南京大学本科生实验报告

课程名称：计算机网络

任课教师：田臣/李文中

助教：

学院	计算机科学与技术系	专业（方向）	计算机科学与技术
学号	202220013	姓名	徐简
Email	<a href="mailto:161200063@smail.nju.edu.cn">161200063@smail.nju.edu.cn</a>	开始/完成日期	2021.3.18-2021.3.21

## 1. 实验名称

Lab 1: Switchyard&mininet

## 2. 实验目的

1. get all preparations for the network experiments.
2. modify the examples provided according to the requirements.

## 3. 实验内容、代码与结果

### Step 1: Modify the mininet topology

Option 2: create a different topology containing 6 nodes using hosts and hubs

1. Mininet topology after modification

```
# Host and link configuration
#
# server1          clinet1
#          \      /
# server2 - hub
#          /      \
# server3          client2
```

2. Add 3 servers, 2 clients and 1 hub

```

self.addHost('server1',**nodeconfig)
self.addHost('server2',**nodeconfig)
self.addHost('server3',**nodeconfig)
self.addHost('hub',**nodeconfig)
self.addHost('client1',**nodeconfig)
self.addHost('client2',**nodeconfig)

```

### 3. Create links between nodes

```

for node in ['server1','server2','server3','client1','client2']:
    self.addLink(node,'hub',bw=10,delay='10ms')

```

### 4. Set MAC addresses and IP addresses for 6 nodes

```

reset_macs(net,'server1','10:00:00:00:00:{:02x}')
reset_macs(net,'server2','20:00:00:00:00:{:02x}')
reset_macs(net,'server3','30:00:00:00:00:{:02x}')
reset_macs(net,'hub','60:00:00:00:00:{:02x}')
reset_macs(net,'client1','40:00:00:00:00:{:02x}')
reset_macs(net,'client2','50:00:00:00:00:{:02x}')
set_ip(net,'server1','hub','192.168.100.1/24')
set_ip(net,'server2','hub','192.168.100.2/24')
set_ip(net,'server3','hub','192.168.100.3/24')
set_ip(net,'client1','hub','192.168.100.4/24')
set_ip(net,'client2','hub','192.168.100.5/24')

```

## Step 2: Modify the logic of a device

1. At the beginning of main function, define and initialize incnt to count the number of packets entering the hub and outcnt to count the number of packets leaving the hub.

```

incnt=0
outcnt=0

```

2. In the while true loop, the first except means there is no packet and the second except means the hub is shut down. Both two indicate that there is no packet entering the hub, so adding `incnt+=1` after the 2 `except` conditions should count how many packets pass through a hub in.

```

while True:
    try:
        _, fromIface, packet = net.recv_packet()
    except NoPackets:
        continue
    except Shutdown:
        break
    incnt+=1

```

3. Hub can't receive packets, which means if the destination of a packet is not the hub itself, it should pass through the hub out.

```

if fromIface!= intf.name:
    outcnt+=1

```

4. At the end of the while true loop, print the information of packets passing through the hub in and out.

```

while True:
    '''
    '''
    log_info("in:{} out:{}".format(incnt,outcnt))

```

5. `pingall` and the terminal shows the result of packets passing in and out.

```

*** Starting CLI:
mininet> xterm hub
mininet> pingall
*** Ping: testing ping reachability
client1 -> client2 X server1 server2 server3
client2 -> client1 X server1 server2 server3
hub -> X X X X X
server1 -> client1 client2 X server2 server3
server2 -> client1 client2 X server1 server3
server3 -> client1 client2 X server1 server2
*** Results: 33% dropped (20/30 received)
mininet>

```

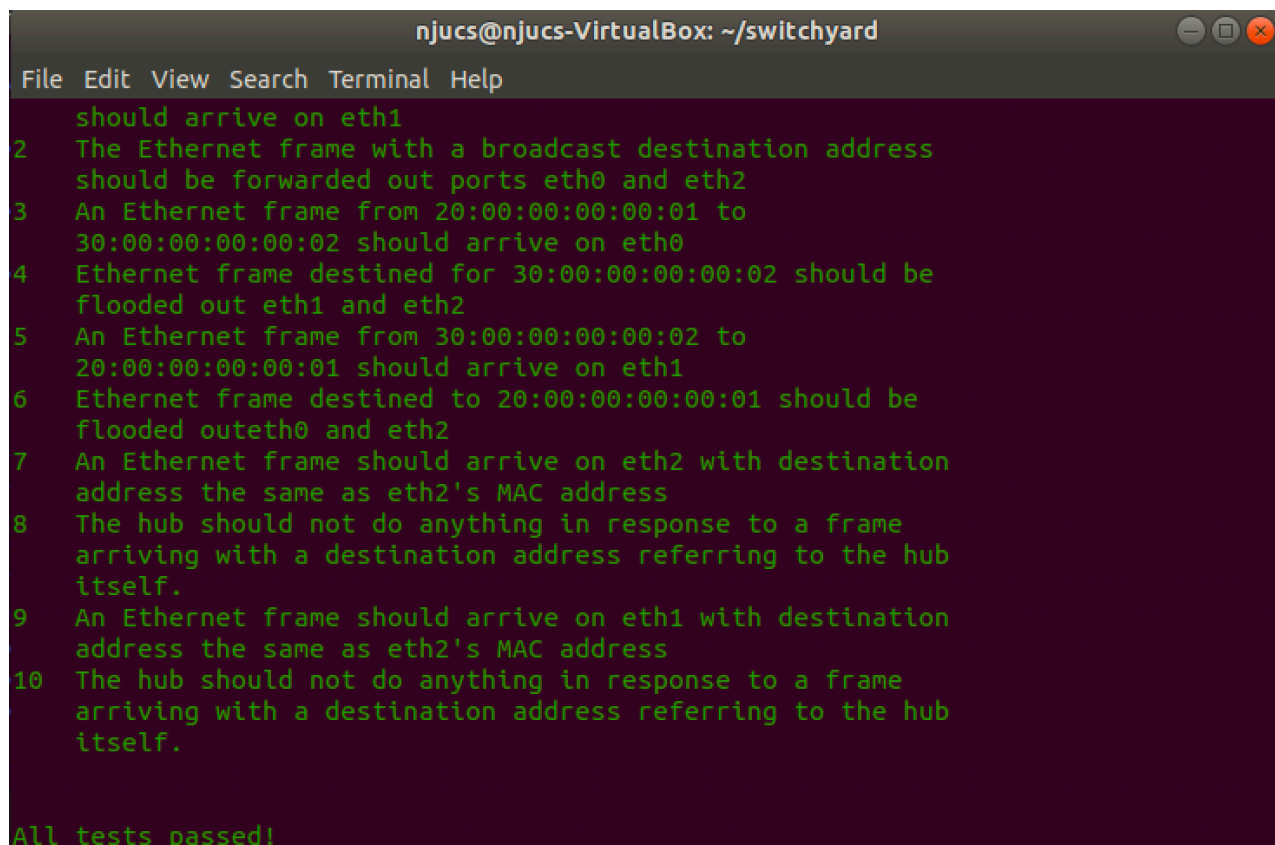
## Step 3: Modify the test scenario of a device

Option 1: Create one test case by using the given function `new_packet` with different arguments

1. Read testcase3 and make some modifications. Use `new_packet` to create a packet arriving at eth1 port, whose destination is also eth1. Therefore, if the hub functions well, it should do nothing.

```
reqpkt=new_packet(  
    "20:00:00:00:00:01",  
    "10:00:00:00:00:02",  
    "192.168.1.100",  
    "172.16.42.2"  
)  
s.expect(  
    PacketInputEvent("eth1",reqpkt,display=Ethernet),  
    ("An Ethernet frame should arrive on eth1 with destination address "  
     "the same as eth2's MAC address")  
)  
s.expect(  
    PacketInputTimeoutEvent(1.0),  
    ("The hub should not do anything in response to a frame arriving with"  
     " a destination address referring to the hub itself.")  
)
```

2. run the tests and all tests passed!



The screenshot shows a terminal window titled "njucs@njucs-VirtualBox: ~/switchyard". The terminal displays a list of 10 test cases, each preceded by a number. The tests describe various Ethernet frame forwarding scenarios between ports eth0, eth1, and eth2. At the bottom of the terminal, the text "All tests passed!" is displayed in green.

```
njucs@njucs-VirtualBox: ~/switchyard  
File Edit View Search Terminal Help  
1 should arrive on eth1  
2 The Ethernet frame with a broadcast destination address  
  should be forwarded out ports eth0 and eth2  
3 An Ethernet frame from 20:00:00:00:00:01 to  
  30:00:00:00:00:02 should arrive on eth0  
4 Ethernet frame destined for 30:00:00:00:00:02 should be  
  flooded out eth1 and eth2  
5 An Ethernet frame from 30:00:00:00:00:02 to  
  20:00:00:00:00:01 should arrive on eth1  
6 Ethernet frame destined to 20:00:00:00:00:01 should be  
  flooded out eth0 and eth2  
7 An Ethernet frame should arrive on eth2 with destination  
  address the same as eth2's MAC address  
8 The hub should not do anything in response to a frame  
  arriving with a destination address referring to the hub  
  itself.  
9 An Ethernet frame should arrive on eth1 with destination  
  address the same as eth2's MAC address  
10 The hub should not do anything in response to a frame  
    arriving with a destination address referring to the hub  
    itself.  
All tests passed!
```

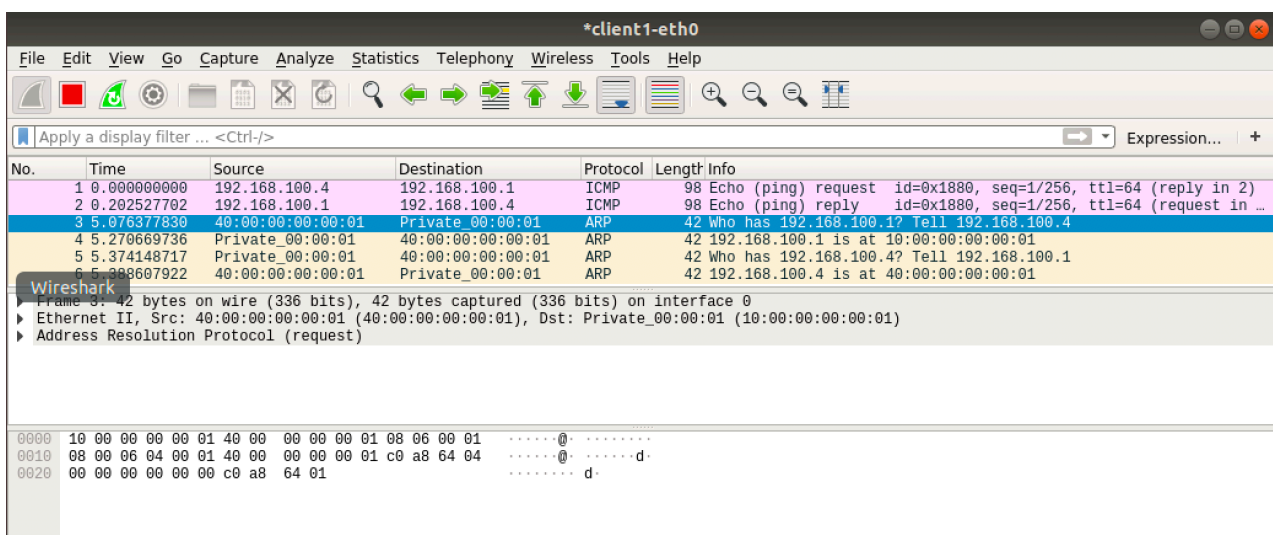
## Step 4: Run your device in Mininet

1. Open `start_mininet.py` to start Mininet.
2. run `xterm hub` , then activate your Python virtual environment first. After activation, run `myhub.py` on it.
3. In Mininet CLI, type `pingall` and return.

```
*** Starting CLI:
mininet> xterm hub
mininet> pingall
*** Ping: testing ping reachability
client1 -> client2 X server1 server2 server3
client2 -> client1 X server1 server2 server3
hub -> X X X X X
server1 -> client1 client2 X server2 server3
server2 -> client1 client2 X server1 server3
server3 -> client1 client2 X server1 server2
*** Results: 33% dropped (20/30 received)
mininet> 
```

## Step 5: Capture using Wireshark

1. In Mininet CLI, type `client1 wireshark &` and return.
2. Type `client1 ping -c 1 server1` to make client1 communicate with server1. After that, the process of communication pops out.



3. First, client1 sent a request to server1 via ICMP protocol. Then, server1 replied back also via ICMP protocol after 0.2025 seconds. As the info column shows, in the next 4 packets, client1 and server1 successfully exchanged their MAC address via ARP protocol.

## 4. 总结与感想

Overall, code need to write in lab1 is not hard. However, to finish the tasks efficiently, the manual and the skeleton code should be read and understand carefully. At first, I was anxious to implement functions without a comprehensive of the lab structure, thus I made a lot of mistakes and wasted a lot of time checking the manual repeatedly.

To sum up, the first lab is well-designed to guided me through the whole workflow well and it is really interesting to understand the skeleton code and experiment on it.