南京大学本科生实验报告

1. 实验名称

Switchyard & Mininet

2. 实验目的

a. 了解计算机网络实验的环境
b. 了解和初步掌握各种工具的使用
c. 了解代码框架的逻辑后进行自己的修改和尝试

3. 实验过程

1) 修改 mininet 网络拓扑结构
   选择构建 6 个结点的拓扑结构（hub，client，server 各 2 个）。
   a. 核心代码

```python
def set_ip(net, node1, node2, ip):#设定结点的 IP 地址
def reset_macs(net, node, macbase):设定结点的 MAC 地址
```

首先利用所给两个接口设立每个结点的 IP 和 MAC 地址：

```python
def setup_addressing(net):
#分别设定 hub、server、client 各两个
    reset_macs(net, 'server1', '10:00:00:00:00:{:02x}')
    reset_macs(net, 'server2', '20:00:00:00:00:{:02x}')
    reset_macs(net, 'client1', '30:00:00:00:00:{:02x}')
    reset_macs(net, 'hub1', '40:00:00:00:00:{:02x}')
    reset_macs(net, 'hub2', '50:00:00:00:00:{:02x}')
    reset_macs(net, 'client2', '60:00:00:00:00:{:02x}')
    set_ip(net, 'server1', 'hub1', '192.168.100.1/24')
    set_ip(net, 'server2', 'hub2', '192.168.100.2/24')
    set_ip(net, 'client1', 'hub1', '192.168.100.3/24')
    set_ip(net, 'client2', 'hub2', '192.168.100.4/24')
```

之后利用 addHost 和 addLink 两个 API 建立拓扑结构：

```
    #
    #  server1              server2
    #        \            /
    #          hub1----hub2
    #        /            \
    #  client1              client2
```

```
        #
        self.addHost('server1', **nodeconfig)
        self.addHost('server2', **nodeconfig)
        self.addHost('hub1', **nodeconfig)
        self.addHost('client1', **nodeconfig)
        self.addHost('hub2', **nodeconfig)
        self.addHost('client2', **nodeconfig)
        for node in ['server1', 'client1']:
            # all links are 10Mb/s, 100 millisecond prop delay
            self.addLink(node, 'hub1', bw=10, delay='100ms')
        for node in ['server2', 'client2']:
            # all links are 10Mb/s, 100 millisecond prop delay
            self.addLink(node, 'hub2', bw=10, delay='100ms')
        self.addLink('hub1','hub2',bw=10,delay='100ms')
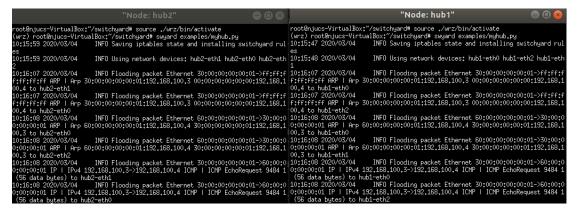```

b. 实验结果

运行 mininet，dump 指令查看各节点信息

```
mininet> dump
<Host client1: client1-eth0:192.168.100.3 pid=4486>
<Host client2: client2-eth0:192.168.100.4 pid=4488>
<Host hub1: hub1-eth0:10.0.0.3,hub1-eth1:None,hub1-eth2:None pid=4490>
<Host hub2: hub2-eth0:10.0.0.4,hub2-eth1:None,hub2-eth2:None pid=4492>
<Host server1: server1-eth0:192.168.100.1 pid=4494>
<Host server2: server2-eth0:192.168.100.2 pid=4496>
```

2) 修改 hub 网络设备
   a. 核心代码

   当 myhub 中收到一个包 in 计数加一，当成功转发出去 out 加一

```
while True:
    …
    in_count += 1
    eth = packet.get_header(Ethernet)
    …
    if eth.dst in mymacs:
        log_info("Received a packet intended for me")
    else:
        for intf in my_interfaces:
            if dev != intf.name:
                out_count+=1
                log_info('{} in: {}> out: {}>'.format(timestamp,in_c
ount,out_count))
                net.send_packet(intf, packet)
    net.shutdown()
```

   b. 实验结果

   在 mininet 中分别开启 hub1，hub2，并进行 pingall 测试

      a) 开启 hub

b) 进行 pingall 测试，查看结点输出

3) 修改 hub 测试文件

hubtest 测试文件用于测试 myhub 的逻辑正确性，接口的和包的编写都是任意的。使用 mk_pkt 生成一个包，并编写期望结果。

a. 核心代码

```
#tets case4: a frame with the same src and dest address should
    #result in nothing happening
    reqpkt = mk_pkt("20:00:00:00:00:01", "20:00:00:00:00:01", '172.16.4
2.2','172.16.42.2')
    s.expect(
        PacketInputEvent("eth0", reqpkt, display=Ethernet),
        "An Ethernet frame should arrive on eth0 with destination addre
ss the same as src address"
    )
    s.expect(
        PacketOutputEvent("eth1", reqpkt, "eth2", reqpkt, display=Ether
net),
        "Ethernet frame with destination address the same as src addres
s should be flooded out eth1 and eth2")
```

包具有发送和接受地址相同，假定从 eth0 出发，则会由 eth1 和 eth2 接收。

b. 实验结果

运行 swyard 进行测试，符合预期结果


```
(wrz) njucs@njucs-VirtualBox:~/switchyard$ swyard -t lab_1/hubtests.py lab_1/myhub.py
13:38:40 2020/03/06      INFO Starting test scenario lab_1/hubtests.py
13:38:40 2020/03/06      INFO 0.0 in: 1> out: 1>
13:38:40 2020/03/06      INFO 0.0 in: 1> out: 2>
13:38:40 2020/03/06      INFO 2.0 in: 2> out: 3>
13:38:40 2020/03/06      INFO 2.0 in: 2> out: 4>
13:38:40 2020/03/06      INFO 4.0 in: 3> out: 5>
13:38:40 2020/03/06      INFO 4.0 in: 3> out: 6>
13:38:40 2020/03/06      INFO Received a packet intended for me
13:38:41 2020/03/06      INFO 6.0 in: 5> out: 7>
13:38:41 2020/03/06      INFO 6.0 in: 5> out: 8>

Results for test scenario hub tests: 10 passed, 0 failed, 0 pending

Passed:
1    An Ethernet frame with a broadcast destination address
     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 eth0 with destination
     address the same as src address
10   Ethernet frame with destination address the same as src
     address should be flooded out eth1 and eth2


All tests passed!
```

4) 在 mininet 中运行拓扑网络

在 mininet 中分别开启 hub1，hub2，再次进行 pingall 测试。如下图
表明除 hub 外各节点均连通，hub 相当于网卡本身只能接受转发。


```
mininet> xterm hub1 hub2
mininet> pingall
*** Ping: testing ping reachability
client1 -> client2 X X server1 server2
client2 -> client1 X X server1 server2
hub1 -> X X X X X
hub2 -> X X X X X
server1 -> client1 client2 X X server2
server2 -> client1 client2 X X server1
*** Results: 60% dropped (12/30 received)
```

5) 使用 wireshark 进行抓包

抓包前同样要开启 hub 服务。

选取 client 进行抓包

```
mininet> client1 wireshark &
mininet> client1 ping -c 1 server2
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
PING 192.168.100.2 (192.168.100.2) 56(84) bytes of data.
64 bytes from 192.168.100.2: icmp_seq=1 ttl=64 time=798 ms

--- 192.168.100.2 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 798.356/798.356/798.356/0.000 ms
mininet> client1 ping -c 1 client2
PING 192.168.100.4 (192.168.100.4) 56(84) bytes of data.
64 bytes from 192.168.100.4: icmp_seq=1 ttl=64 time=744 ms

--- 192.168.100.4 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 744.881/744.881/744.881/0.000 ms
```

client1 与 server2 进行 ping，得到的抓包结果如下。

No.	Time	Source	Destination	Protocol	Length Info
1	0.000000000	30:00:00:00:00:01	Broadcast	ARP	42 Who has 192.168.100.2? Tell 192.168.100.3
2	0.850136938	20:00:00:00:00:01	30:00:00:00:00:01	ARP	42 192.168.100.2 is at 20:00:00:00:00:01
3	0.950810784	192.168.100.3	192.168.100.2	ICMP	98 Echo (ping) request id=0x16f0, seq=1/256, ttl=64 (rep
4	1.582111638	192.168.100.2	192.168.100.3	ICMP	98 Echo (ping) reply id=0x16f0, seq=1/256, ttl=64 (req

一、二包协议为 ARP，首先是 client1 在链路层发起广播，寻找 server2 对应的 mac 地址；之后为 server2 向 client1 回复自己的 mac 地址。

三、四包协议为 ICMP，为检测两个结点间能否正确通信，故采用 ICMP 协议在网络层进行通信，三四包为请求和回复。

4. 总结与感想

第一次计网实验，最开始读完手册要求依旧对实验的原理不明白，实验的要求不清楚，之后经过阅读手册中提供的工具说明文档，对整个实验工程的框架才有了大体的理解。

总的来讲，第一次实验感觉还是比较容易理解的，自己也觉得很有趣，希望可以保持这种热情吧。

5. 文档结构

```
njucs@njucs-VirtualBox:~/switchyard/lab_1$ tree
.
├── 181860109吴润泽_lab_1.pdf
├── hubtests.py
├── lab_1.pcapng
├── myhub.py
└── start_mininet.py
```