# lab1 report

**191220138 杨飞洋**

## 实验目的

学会利用switchyard和mininet在linux环境下搭建一个随机虚拟网络

学会利用python语言结合switchy库来进行一些简单的网络编程

## 实验过程

### step1：Modify the Mininet topology

Delete `server2` in the topology，只需要在nodes中注释掉关于server2的信息就好了

```
nodes = {
    "server1": {
        "mac": "10:00:00:00:00:{:02x}",
        "ip": "192.168.100.1/24"
    },
#    "server2": {
 #       "mac": "20:00:00:00:00:{:02x}",
  #      "ip": "192.168.100.2/24"
   # },
    "client": {
        "mac": "30:00:00:00:00:{:02x}",
        "ip": "192.168.100.3/24"
    },
    "hub": {
        "mac": "40:00:00:00:00:{:02x}",
    }
}
```

`sudo python3 start_mininet.py`，再`pingall`看下效果：

```
mininet> pingall
*** Ping: testing ping reachability
client -> X X
hub -> X X
server1 -> X X
*** Results: 100% dropped (0/6 received)
```

### step2：Modify the logic of a device

count how many packets pass through a hub in and out

用`InNum`和`OutNum`来记录入包和出包的个数

如果`try`过了，则从函数名可以推测，receive了一个包，则`InNum += 1`

```
try:
    _, fromIface, packet = net.recv_packet()
    InNum += 1
```

`fromIface`指代这个包从哪来，在`my_interfaces`接口中，只要和`fromIface`不一样，就会调用`send_packet`发出一个包给`intf`，相应的，`OutNum += 1`

```
for intf in my_interfaces:
    if fromIface!= intf.name:
        log_info (f"Flooding packet {packet} to {intf.name}")
        net.send_packet(intf, packet)
        OutNum += 1
```

还要记录下来in和out的情况，可以在每条分支语句下加一条log信息：

```
log_info(f"in:{InNum} out:{OutNum}")
```

在hub中看下效果，可以用`hub gnome-terminal`替代`hub xterm`可以让界面好看一些

在hub中键入`source /home/rafael/switchyard/syenv/bin/activate`开启虚拟环境，再键入`swyard myhub.py`，在mininet中

`pingall`，查看效果如下：

```
11:36:59 2021/03/19     INFO in:3 out:3
11:37:00 2021/03/19     INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:0
0:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 3030 1 (
56 data bytes) to hub-eth1
11:37:00 2021/03/19     INFO in:4 out:4
11:37:00 2021/03/19     INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:0
0:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoRequest 3033 1
 (56 data bytes) to hub-eth1
11:37:00 2021/03/19     INFO in:5 out:5
11:37:00 2021/03/19     INFO Flooding packet Ethernet 30:00:00:00:00:01->10:00:0
0:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoReply 3033 1 (
56 data bytes) to hub-eth0
11:37:00 2021/03/19     INFO in:6 out:6
11:37:05 2021/03/19     INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:0
0:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:00:192.168.1
00.3 to hub-eth1
11:37:05 2021/03/19     INFO in:7 out:7
11:37:05 2021/03/19     INFO Flooding packet Ethernet 30:00:00:00:00:01->10:00:0
0:00:00:01 ARP | Arp 30:00:00:00:00:01:192.168.100.3 10:00:00:00:00:01:192.168.1
00.1 to hub-eth0
```

## step3：Modify the test scenario of a device

i choose to create one test case by using the given function `new_packet` with different arguments

仿照testcase1，将hwsrc改为`"10:00:00:00:00:03"`,即对应`eth2`

```
mypkt = new_packet(
    "10:00:00:00:00:03",
    "ff:ff:ff:ff:ff:ff",

    "192.168.1.100",
    "255.255.255.255"
)
s.expect(
```

```
    PacketInputEvent("eth2", mypkt, display=Ethernet),
    ("An Ethernet frame with a broadcast destination address "
     "should arrive on eth2")
)
s.expect(
    PacketOutputEvent("eth0", mypkt, "eth1", mypkt, display=Ethernet),
    ("The Ethernet frame with a broadcast destination address should be "
     "forwarded out ports eth0 and eth1")
)
```

在hub中键入`swyard -t testcases/myhub_testscenario.py myhub.py`查看效果如下：

```
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 outeth0 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 with a broadcast destination address
    should arrive on eth2
10  The Ethernet frame with a broadcast destination address
    should be forwarded out ports eth0 and eth1


All tests passed!
```

line 9和line 10对应自己加的testcase

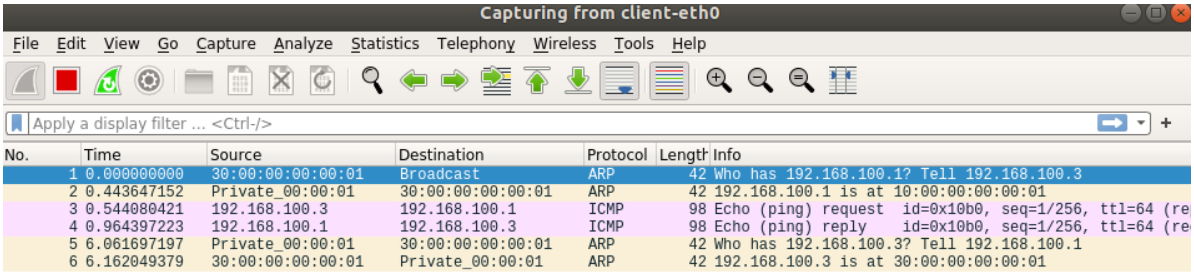
### step4: Run your device in Mininet

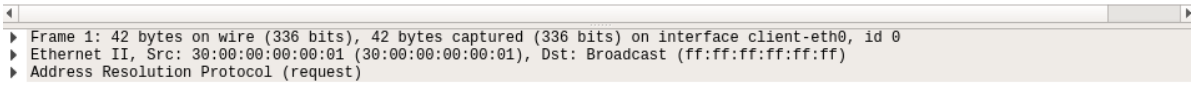
可以对应step3和step2中的内容

### step5: Capture using Wireshark

在mininet下键入`client wireshark`开启wireshark，进入hub，在hub中键入`source/home/rafael/switchyard/syenv/bin/activate`开启虚拟环境，再键入`swyard myhub.py`开启进程，在mininet中

键入`ping client c1 server1`，查看wireshark效果如下：





## 实验心得

更加熟悉linux系统下的操作

学习了使用git工具

对python网络编程有了初步了解

不足之处：mininet,wireshark,switchyard之间的内在关联和实现原理还没有吃透，还需要一定的网络基础知识补充。