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：lzh、lsp、wcx

学院	**计算机科学与技术系**	专业（方向）	计算机科学与技术系
学号	211220027	姓名	王秋博
Email	211220027@nju.edu.cn	开始/完成日期	9.30/10.13

1. 实验名称

Lab1: Switchyard & Mininet

2. 实验目的

熟悉实验流程、环境。

3. 实验内容

Step 1: Modify the Mininet topology

实现选项一：Delete server2 in the topology

只需注释 start_mininet.py 中 nodes 中 server2 的节点信息即可，如下图

```
25  nodes = {
26      "server1": {
27          "mac": "10:00:00:00:00:{:02x}",
28          "ip": "192.168.100.1/24"
29      },
30      #"server2": {
31      #   "mac": "20:00:00:00:00:{:02x}",
32      #   "ip": "192.168.100.2/24"
33      #},
34      "client": {
35          "mac": "30:00:00:00:00:{:02x}",
36          "ip": "192.168.100.3/24"
37      },
38      "hub": {
```

启动 mininet 后使用 nodes 指令查看拓扑结构中所有节点，可发现 server2 被成功删除

```
mininet> nodes
available nodes are:
client hub server1
mininet>
```

Step 2: Modify the logic of a device

要求记录每次收到一个数据包时的统计结果，首先定义两个局部变量 inpacket 和 outpacket 来记录实时结果

```
12      mymacs = [intf.ethaddr for intf in my_interfaces]
13      inpacket=outpacket=0
14      while True:
```

在 while 循环体内确认收到后自增 inpacket，在广播转发时自增 outpacket 并在最后输出 log；如图

```
    log_debug (f"In {net.name} received packet {packet} on {fromIface}")
    eth = packet.get_header(Ethernet)
    inpacket+=1
    if eth is None:
        log_info("Received a non-Ethernet packet?!")
        return
    if eth.dst in mymacs:
        log_info("Received a packet intended for me")
    else:
        for intf in my_interfaces:
            if fromIface!= intf.name:
                outpacket+=1
                log_info (f"Flooding packet {packet} to {intf.name}")
                net.send_packet(intf, packet)
    log_info(f"in:{inpacket} out:{outpacket}")
```

运行修改后的 mininet 网络拓扑可得

```
njucs@njucs-VirtualBox: ~/workspace/lab-1-Dexter2008-1
File  Edit  View  Search  Terminal  Help
(syenv) njucs@njucs-VirtualBox:~/workspace/lab-1-Dexter2008-1$ swyard -t testcases/myhub_testscenari
o.py myhub.py
01:34:52 2023/10/13     INFO Starting test scenario testcases/myhub_testscenario.py
01:34:52 2023/10/13     INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4
 172.16.42.2->255.255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
01:34:52 2023/10/13     INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4
 172.16.42.2->255.255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
01:34:52 2023/10/13     INFO in:1 out:2
01:34:52 2023/10/13     INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4
 192.168.1.100->172.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
01:34:52 2023/10/13     INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4
 192.168.1.100->172.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
01:34:52 2023/10/13     INFO in:2 out:4
01:34:52 2023/10/13     INFO Flooding packet Ethernet 30:00:00:00:00:02->20:00:00:00:00:01 IP | IPv4
 172.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth0
01:34:52 2023/10/13     INFO Flooding packet Ethernet 30:00:00:00:00:02->20:00:00:00:00:01 IP | IPv4
 172.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth2
01:34:52 2023/10/13     INFO in:3 out:6
01:34:52 2023/10/13     INFO Received a packet intended for me
01:34:52 2023/10/13     INFO in:4 out:6

Results for test scenario hub tests: 8 passed, 0 failed, 0 pending
```

Step 3: Modify the test scenario of a device

使用具有不同参数的给定函数 new_packet 创建一个测试用例

设计了一个由 20:00:00:00:00:01 发给广播地址的样例，应有 hub 的 eth0 接口接收到数据包，并广播到 eth1 接口和 eth2 接口

```
#test case 4   My_Own_testcase
mytestpkt = new_packet(
    "20:00:00:00:00:01",
    "ff:ff:ff:ff:ff:ff",
    "192.168.1.100",
    "255.255.255.255"
)
s.expect(
    PacketInputEvent("eth0", mytestpkt, display=Ethernet),
    ("An Ethernet frame with a broadcast destination address "
     "should arrive on eth0")
)
s.expect(
    PacketOutputEvent("eth1", mytestpkt, "eth2",mytestpkt, display=Ethernet),
    ("The Ethernet frame with a broadcast destination address should be "
     "forwarded out ports eth1 and eth2")
)
```

Step 4: Run your device in Mininet

根据手册，在 mininet 上运行 switchyard 程序

首先进入虚拟环境并启动 mininet

```
njucs@njucs-VirtualBox:~/workspace$ source ./syenv/bin/activate
(syenv) njucs@njucs-VirtualBox:~/workspace$ cd lab-1-Dexter2008-1
(syenv) njucs@njucs-VirtualBox:~/workspace/lab-1-Dexter2008-1$ sudo python start
_mininet.py
[sudo] password for njucs:
```

在 xterm 中启动网络拓扑：

```
root@njucs-VirtualBox:~/workspace# source ./syenv/bin/activate
(syenv) root@njucs-VirtualBox:~/workspace# cd lab-1-Dexter2008-1
(syenv) root@njucs-VirtualBox:~/workspace/lab-1-Dexter2008-1# swyard myhub.py
02:58:06 2023/10/13     INFO Saving iptables state and installing switchyard rul
es
02:58:06 2023/10/13     INFO Using network devices: hub-eth1 hub-eth0
```

在终端 pingall：

```
mininet> xterm hub
mininet> pingall
*** Ping: testing ping reachability
client -> X server1
hub -> X X
server1 -> client X
*** Results: 66% dropped (2/6 received)
```

在 xterm 窗口可看到数据包收发的情况

```
0:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 7728 1
 (56 data bytes) to hub-eth1
02:58:39 2023/10/13     INFO in:3 out:3
02:58:39 2023/10/13     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 7728 1 (
56 data bytes) to hub-eth0
02:58:39 2023/10/13     INFO in:4 out:4
02:58:39 2023/10/13     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 7731 1
 (56 data bytes) to hub-eth0
02:58:39 2023/10/13     INFO in:5 out:5
02:58:40 2023/10/13     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 7731 1 (
56 data bytes) to hub-eth1
02:58:40 2023/10/13     INFO in:6 out:6
02:58:44 2023/10/13     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-eth0
02:58:44 2023/10/13     INFO in:7 out:7
02:58:45 2023/10/13     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-eth1
02:58:45 2023/10/13     INFO in:8 out:8
```

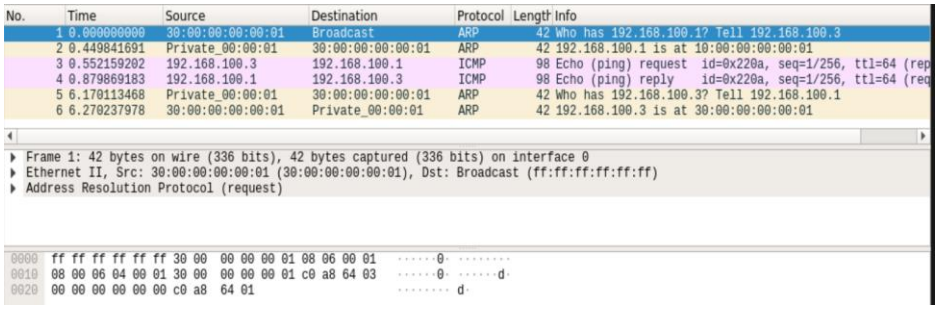
Step 5: Capture using Wireshark

根据手册，重启 xterm 的 myhub 程序，并用 wireshark 程序监听 client 的 eth0 接口，再 ping

```
mininet> xterm hub
mininet> client wireshark &
mininet> client ping -c1 server1
QStandardPaths: XDG_RUNTIME_DIR not set, defaulting to '/tmp/runtime-root'
PING 192.168.100.1 (192.168.100.1) 56(84) bytes of data.
64 bytes from 192.168.100.1: icmp_seq=1 ttl=64 time=980 ms

--- 192.168.100.1 ping statistics ---
1 packets transmitted, 1 received, 0% packet loss, time 0ms
rtt min/avg/max/mdev = 980.039/980.039/980.039/0.000 ms
```

Wireshark 捕获的数据包如下图所示



可以看到有六个数据包

一个包由源 192.168.100.1 向外广播发出，使用 ARP 协议，长度为 42。

一个包由 server1 发给 30:00:00:00:01，使用 ARP 协议，长度为 42。

两个包先由 192.168.100.3 发给 192.168.100.1，使用 ICMP 协议，长度 98.

然后再由 192.168.100.1 发给 192.168.100.3，也使用 ICMP 协议，长度为 98。

第 5 个包与第 2 个包基本类似

最后一个包由 30:00:00:00:01 发给 server1，使用 ARP 协议，长度为 42。

4. 实验结果

本节实验结果基本于实验过程中阐述，不再赘述

5. 核心代码

同实验结果

6. 总结与感想

本次实验主要进行了环境配置和对 Mininet、switchyard、wireshark 等工具的熟悉。通过本次实验，掌握了 python 语言的一些编程知识，获得了对网络拓扑的一部分理解。为未来实验奠定了基础，也希望以后实验顺利。