



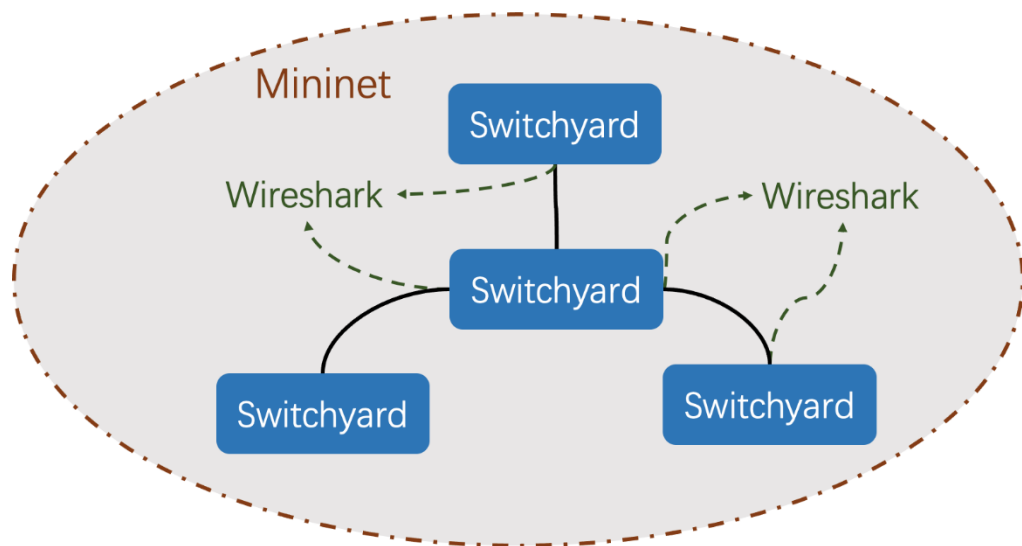
南京大學

LAB 1: Switchyard & Mininet

课程名称:	计算机网络
姓名:	孙文博
学号:	201830210
学院:	计算机科学与技术系
Email:	201830210@smail.nju.edu.cn
任课教师:	李文中
实验时间:	2022. 3. 1 – 2022. 3. 10

一、实验目的

本次实验中，我们需要根据实验手册要求配置好实验环境，包括安装一台用于模拟实验的 VM 虚拟机，配置好运行网络设备的平台 Switchyard，构建网络的 Mininet，以及用于捕获数据包的 Wireshark，之后熟悉一个 hub 实例，了解设备之间如何通过 hub 互相通信，在此基础上修改样例代码，构建一个新的网络拓扑并加以测试分析，最后将相关文件提交到 GitHub classroom 中，任务完成！上述软件之间的关系如下图所示：



图表 1：Mininet，Switchyard 与 Wireshark

二、实验内容

1. 修改 Mininet 拓扑

在 Task3 的拓扑结构中，有三个 host 与 hub 相连，如下图所示：

```
# Host and link configuration
#
#|
# server1
#      \
#      hub----client
#      /
# server2
#
```

图表 2：例子中的拓扑结构

按照手册要求删去 server2，构成新的拓扑如下图所示：

```
# server1 -- hub ---client
```

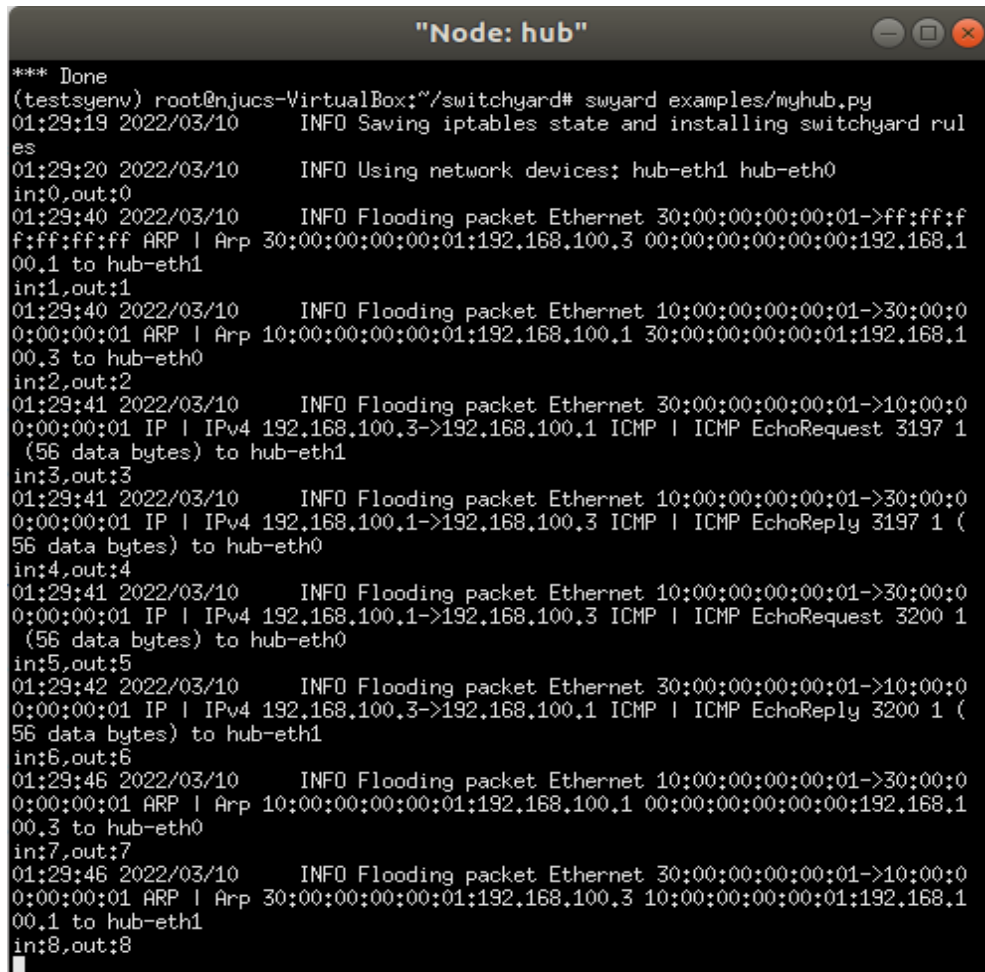
图表 3：修改后的拓扑结构

这个拓扑由一个集线器和两个主机相连，类似于一个杠铃型拓扑。

2. 修改设备逻辑

接下来我们需要统计有多少数据包通过集线器进出，结合课内知识可知，每收到一个不是发给 hub 本身的数据包，集线器都会将这个数据包转发给剩余 $n-1$ 个端口（除了传入端口），其中 n 为端口数。在上面的拓扑结构中，hub 与两个 host 相连，故每次收到不是发给自己的数据包时，都会将数据包转发 $2-1=1$ 次。事实上，在我们执行 pingall 指令时不会有数据包发给 hub 的 mac 地址，因此有 $in=out$ 。

在 Mininet 中运行集线器的工作日志如下：



```

**** Done
(testsenv) root@njucs-VirtualBox:~/switchyard# swyard examples/myhub.py
01:29:19 2022/03/10 INFO Saving iptables state and installing switchyard rules
01:29:20 2022/03/10 INFO Using network devices: hub-eth1 hub-eth0
in:0,out:0
01:29:40 2022/03/10 INFO Flooding packet Ethernet 30:00:00:00:00:01->ff:ff:ff:ff:ff:ff ARP | Arp 30:00:00:00:00:01:192.168.100.3 00:00:00:00:00:00:192.168.100.1 to hub-eth1
in:1,out:1
01:29:40 2022/03/10 INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 30:00:00:00:00:01:192.168.100.3 to hub-eth0
in:2,out:2
01:29:41 2022/03/10 INFO Flooding packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoRequest 3197 1 (56 data bytes) to hub-eth1
in:3,out:3
01:29:41 2022/03/10 INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoReply 3197 1 (56 data bytes) to hub-eth0
in:4,out:4
01:29:41 2022/03/10 INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 IP | IPv4 192.168.100.1->192.168.100.3 ICMP | ICMP EchoRequest 3200 1 (56 data bytes) to hub-eth0
in:5,out:5
01:29:42 2022/03/10 INFO Flooding packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 IP | IPv4 192.168.100.3->192.168.100.1 ICMP | ICMP EchoReply 3200 1 (56 data bytes) to hub-eth1
in:6,out:6
01:29:46 2022/03/10 INFO Flooding packet Ethernet 10:00:00:00:00:01->30:00:00:00:00:01 ARP | Arp 10:00:00:00:00:01:192.168.100.1 00:00:00:00:00:00:192.168.100.3 to hub-eth0
in:7,out:7
01:29:46 2022/03/10 INFO Flooding packet Ethernet 30:00:00:00:00:01->10:00:00:00:00:01 ARP | Arp 30:00:00:00:00:01:192.168.100.3 10:00:00:00:00:01:192.168.100.1 to hub-eth1
in:8,out:8

```

图表 4: hub 的运行日志

分析运行日志发现，in_packet 的数量与 out_packet 的数量相同，与我们的预期结果一致。但是在后面的测试样例中，我们会故意添加向 hub 的 mac 地址发送的数据包，从而会被 hub 吞掉，此时的 out_packet 增加 0（但是通常这种情况不会发生）。

3. 修改设备测试场景

现在我们来修改 myhub_testscenario.py 文件中的测试样例，在现有的一些测试样例中加入新的测试样例，以检测该拓扑结构的鲁棒性。使用 Switchyard 测试得到的结果如下：

LAB 1 : Switchyard & Mininet

```
(testsyenv) njucs@njucs-VirtualBox:~/switchyard$ swyard -t examples/myhub_testscenario.py examples/myhub.py
It works!
01:56:01 2022/03/10      INFO Starting test scenario examples/myhub_testscenario.py
in:0,out:0
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 17
2.16.42.2->255.255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth0
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:00:00:00:00:02->ff:ff:ff:ff:ff:ff IP | IPv4 17
2.16.42.2->255.255.255.255 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
in:1,out:2
01:56:01 2022/03/10      INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 19
2.168.1.100->172.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth1
01:56:01 2022/03/10      INFO Flooding packet Ethernet 20:00:00:00:00:01->30:00:00:00:00:02 IP | IPv4 19
2.168.1.100->172.16.42.2 ICMP | ICMP EchoRequest 0 0 (0 data bytes) to eth2
in:2,out:4
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:00:00:00:00:02->20:00:00:00:00:01 IP | IPv4 17
2.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth0
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:00:00:00:00:02->20:00:00:00:00:01 IP | IPv4 17
2.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth2
in:3,out:6
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:11:11:11:11:01->20:11:11:11:11:02 IP | IPv4 17
2.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth0
01:56:01 2022/03/10      INFO Flooding packet Ethernet 30:11:11:11:11:01->20:11:11:11:11:02 IP | IPv4 17
2.16.42.2->192.168.1.100 ICMP | ICMP EchoReply 0 0 (0 data bytes) to eth1
in:4,out:8
01:56:01 2022/03/10      INFO Received a packet intended for me
in:5,out:8
01:56:01 2022/03/10      INFO Received a packet intended for me
in:6,out:8
in:6,out:8
Results for test scenario hub tests: 11 passed, 0 failed, 0 pending
```

图表 5: Switchyard 测试结果

这里的测试文件中, hub 有三个端口 eth0,1,2, 所以每次收到一个数据包会转发给另外两个端口, 但最后两个数据包是发给 hub 的 mac 地址, 因此 out 不增加, 且返回 “Received a packet intended for me ”。测试的提示输出如下:

```
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 outeth1 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 from 30:11:11:11:11:01 to
   20:11:11:11:11:02 should arrive on eth2
8  Ethernet frame destined to 20:11:11:11:11:02 should be
   flooded outeth0 and eth1 My test passed!
9  An Ethernet frame should arrive on eth1 with destination
   address the same as eth1's MAC address
10 An Ethernet frame should arrive on eth2 with destination
   address the same as eth2's MAC address
11 The hub should not do anything in response to a frame
   arriving with a destination address referring to the hub
   itself.

All tests passed!
```

图表 6: 测试样例结果

4. 在 Mininet 中运行新设备

接下来我们在 Mininet 中运行我们的 Switchyard 程序，使用 python 命令

打开 start_mininet.py 文件，可以看到目前的网络拓扑：

```
(testsyenv) njucs@njucs-VirtualBox:~/switchyard$ sudo python examples/start_mininet.py
[sudo] password for njucs:
*** Creating network
*** Adding hosts:
client hub server1
*** Adding switches:

*** Adding links:
(10.00Mbit 100ms delay) (10.00Mbit 100ms delay) (client, hub) (10.00Mbit 100ms delay) (10.00Mbit 100ms
delay) (server1, hub)
*** Configuring hosts
client hub server1
('client', <TCIntf client-eth0>, '30:00:00:00:00:01')
('server1', <TCIntf server1-eth0>, '10:00:00:00:00:01')
('hub', <TCIntf hub-eth0>, '40:00:00:00:00:01')
('hub', <TCIntf hub-eth1>, '40:00:00:00:00:02')
*** client : ('sysctl -w net.ipv6.conf.all.disable_ipv6=1',)
net.ipv6.conf.all.disable_ipv6 = 1
*** client : ('sysctl -w net.ipv6.conf.default.disable_ipv6=1',)
net.ipv6.conf.default.disable_ipv6 = 1
*** hub : ('sysctl -w net.ipv6.conf.all.disable_ipv6=1',)
net.ipv6.conf.all.disable_ipv6 = 1
*** hub : ('sysctl -w net.ipv6.conf.default.disable_ipv6=1',)
net.ipv6.conf.default.disable_ipv6 = 1
*** server1 : ('sysctl -w net.ipv6.conf.all.disable_ipv6=1',)
net.ipv6.conf.all.disable_ipv6 = 1
*** server1 : ('sysctl -w net.ipv6.conf.default.disable_ipv6=1',)
net.ipv6.conf.default.disable_ipv6 = 1
*** Starting controller
*** Starting 0 switches
```

图表 7：Mininet 运行程序

在 mininet 中使用 xterm 命令打开集线器，这一步是将我们模拟的一个节点作为集线器工作，hub 的工作日志如图 4 所示。

使用 pingall 指令可以查看模拟情况：

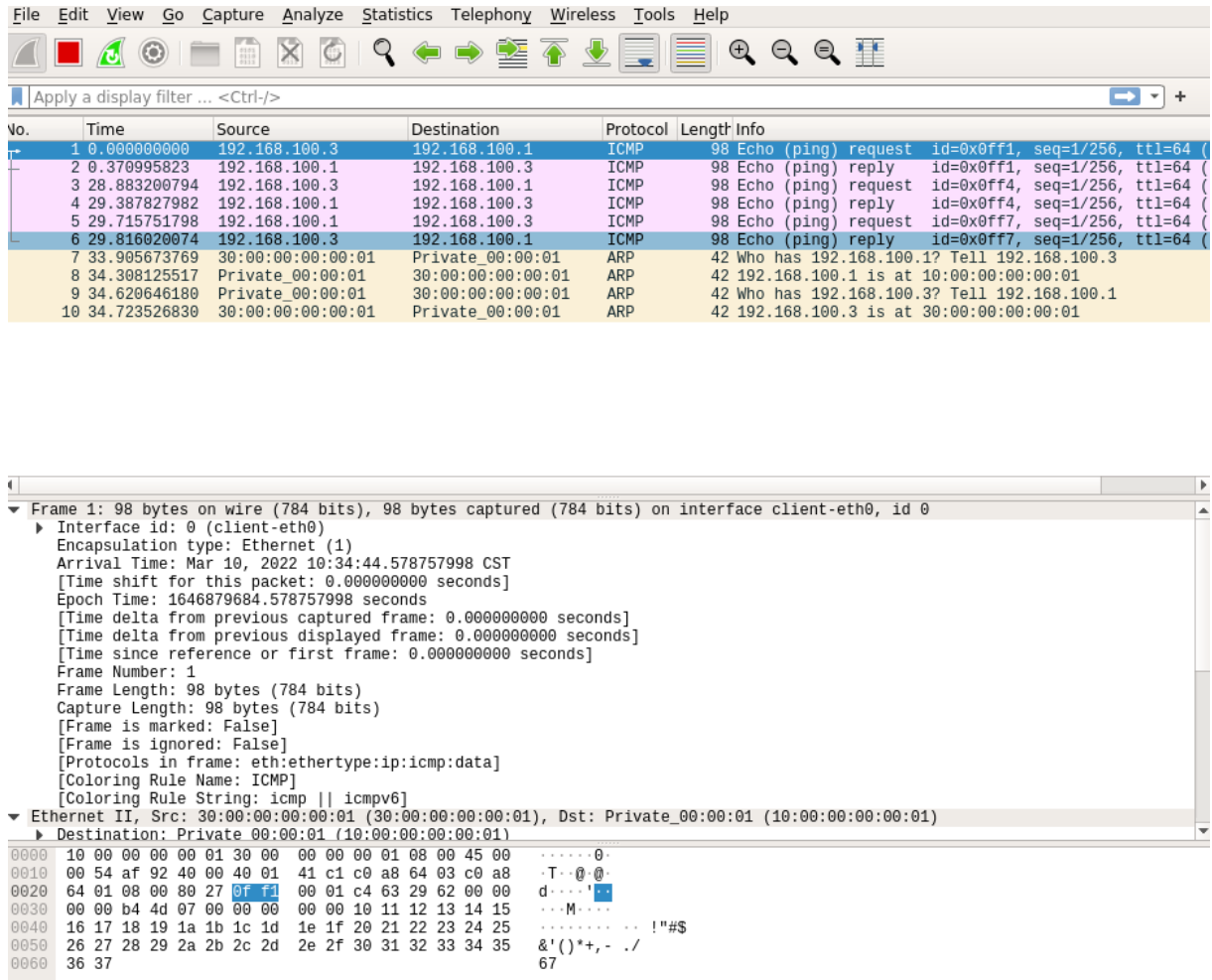
```
*** Starting CLI:
mininet> xterm hub
mininet> pingall
*** Ping: testing ping reachability
client -> X server1
hub -> X X
server1 -> client X
*** Results: 66% dropped (2/6 received)
mininet>
```

图表 8：Mininet 运行情况

这里的丢包率是由于发给 hub 的数据包将会被它吞掉造成的。

5. 使用 Wireshark 进行捕获

在 mininet 中通过 client wireshark &指令使用 Wireshark 捕获 client 主机上的数据包，如图所示：



图表 9：Wireshark 捕获结果

上面是我们的捕获结果，其中 Time 记录收发各数据包的时间戳，Source 是源地址，Destination 是目标地址；192.168.100.3 是 client 的 IP 地址，192.168.100.1 是 server 的 IP 地址；ARP 的四条语句是在 client ping server 的过程中，由于不知道 server 的 mac 地址，因此需要通过一问一答的形式询问 server 实现。

三、核心代码

本次实验需要修改的三个文件代码截图如下,具体代码已提交到classroom中。

```
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}",
    }
}
```

图表 10: Start_mininet.py 文件

```
def main(net: switchyard.llnetbase.LLNetBase):
    my_interfaces = net.interfaces()
    mymacs = [intf.ethaddr for intf in my_interfaces]
    in_count=0;
    out_count=0;

    while True:
        print(f"in:{in_count},out:{out_count}")
        try:
            _, fromIface, packet = net.recv_packet()
        except NoPackets:
            continue
        except Shutdown:
            break

        log_debug(f"In {net.name} received packet {packet} on {fromIface}")
        in_count=in_count+1
        eth = packet.get_header(Ethernet)
        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:
                    out_count=out_count+1
                    log_info(f"Flooding packet {packet} to {intf.name}")
                    net.send_packet(intf, packet)

    net.shutdown()
```

图表 11: My_hub.py 文件


```
# my test
respkt = new_packet(
    "30:11:11:11:11:01",
    "20:11:11:11:11:02",
    '172.16.42.2',
    '192.168.1.100',
    reply=True
)
s.expect(
    PacketInputEvent("eth2", respkt, display=Ethernet),
    ("An Ethernet frame from 30:11:11:11:11:01 to 20:11:11:11:11:02 "
     "should arrive on eth2")
)
s.expect(
    PacketOutputEvent("eth0", respkt, "eth1", respkt, display=Ethernet),
    ("Ethernet frame destined to 20:11:11:11:11:02 should be flooded out"
     "eth0 and eth1"
     " My test passed!")
)
```

图表 12: Myhub_testscenario.py 文件

四、总结与反思

本次实验是我们第一次验收实验，目的是配置好实验的模拟环境并进一步巩固课内知识点，配置环境的过程中遇到了很多困难，比如安装虚拟机，配置python 虚拟环境等等，但在同学和助教的帮助下逐个解决了，再次感谢助教和同学们的帮助，希望接下来的实验可以再接再厉！💪