Computer Network lab2 实验报告

姓名 曾闻天

学号 231880147

邮箱 2249428377@qq.com

完成日期 2025.6.3

1 实验名称

Lab 2: Learning Switch

2 实验目的

1. 理解并实现以太网交换机的基本功能。
2. 学习如何通过交换机的端口接收和转发以太网帧。
3. 掌握交换机的 MAC 地址学习机制和表项管理策略。
4. 熟悉 Switchyard 框架的使用方法。

3 实验进行

3.1 Preparation

下载实验模板代码，初始化项目。
使用 Switchyard 框架搭建基本的路由器框架。

3.2 Basic Switch

实现基本交换机功能，接收时学习 MAC 地址，发送时根据 MAC 地址表进行转发或泛洪。

```
import switchyard
from switchyard.lib.userlib import *


def main(net: switchyard.llnetbase.LLNetBase):
    my_interfaces = net.interfaces()
    mymacs = [intf.ethaddr for intf in my_interfaces]
    eth_dict = {}

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

        log_debug (f"In {net.name} received packet {packet} on {fromIface}")
        eth = packet.get_header(Ethernet)
        if eth is None:
            log_info("Received a non-Ethernet packet?!")
            return
        if eth.dst in mymacs:
            eth_dict [eth.src] = fromIface
            log_info("Received a packet intended for me")
        else:
            eth_dict [eth.src] = fromIface
            if eth.dst not in eth_dict:
                for intf in my_interfaces:
                    if fromIface!= intf.name:
                        log_info (f"Flooding packet {packet} to {intf.name}")
                        net.send_packet(intf, packet)
            else:
                log_info (f"Sending packet {packet} to {eth_dict[eth.dst]}")
                net.send_packet(eth_dict[eth.dst], packet)

    net.shutdown()
```

3.3 Timeouts

在基本交换机的基础上，为 MAC 地址表项增加超时机制，使表项在一定时间后自动失效。

```
import switchyard
```

```
import time
from switchyard.lib.userlib import *


def main(net: switchyard.llnetbase.LLNetBase):
    my_interfaces = net.interfaces()
    mymacs = [intf.ethaddr for intf in my_interfaces]
    eth_dict = {}

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

        log_debug (f"In {net.name} received packet {packet} on {fromIface}")
        eth = packet.get_header(Ethernet)
        if eth is None:
            log_info("Received a non-Ethernet packet?!")
            return
        if eth.dst in mymacs:
            eth_dict [eth.src] = (fromIface, time.time())
            log_info("Received a packet intended for me")
        else:
            eth_dict [eth.src] = (fromIface, time.time())
            if eth.dst not in eth_dict:
                for intf in my_interfaces:
                    if fromIface!= intf.name:
                        log_info (f"Flooding packet {packet} to {intf.name}")
                        net.send_packet(intf, packet)
            else:
                if time.time() - eth_dict[eth.dst][1] < 10:
                    log_info (f"Sending packet {packet} to
{eth_dict[eth.dst][0]}")
                    net.send_packet(eth_dict[eth.dst][0], packet)
                else:
                    log_info (f"Removing forwarding table entry
{eth_dict[eth.dst][0]}")
                    del eth_dict[eth.dst]
                    for intf in my_interfaces:
                        if fromIface!= intf.name:
```

```
                            log_info (f"Flooding packet {packet} to
{intf.name}")
                            net.send_packet(intf, packet)

    net.shutdown()
```

3.4 Least Recently Used

当 MAC 地址表满时，根据最近最少使用（LRU）规则删除表项。

```
import switchyard
import collections
from switchyard.lib.userlib import *


def main(net: switchyard.llnetbase.LLNetBase):
    my_interfaces = net.interfaces()
    mymacs = [intf.ethaddr for intf in my_interfaces]
    eth_dict = collections.OrderedDict()

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

        log_debug (f"In {net.name} received packet {packet} on {fromIface}")
        eth = packet.get_header(Ethernet)
        if eth is None:
            log_info("Received a non-Ethernet packet?!")
            return
        if eth.dst in mymacs:
            if len(eth_dict) == 5 and eth.src not in eth_dict:
                eth_dict.popitem(last=False)
            if eth.src not in eth_dict
                eth_dict [eth.src] = fromIface
            log_info("Received a packet intended for me")
        else:
            if len(eth_dict) == 5 and eth.src not in eth_dict:
                eth_dict.popitem(last=False)
            if eth.src not in eth_dict
                eth_dict [eth.src] = fromIface
            if eth.dst not in eth_dict:
```

```
                for intf in my_interfaces:
                    if fromIface!= intf.name:
                        log_info (f"Flooding packet {packet} to {intf.name}")
                        net.send_packet(intf, packet)
            else:
                log_info (f"Sending packet {packet} to {eth_dict[eth.dst]}")
                net.send_packet(eth_dict[eth.dst], packet)

    net.shutdown()
```

3.5 Least Traffic Volume

当 MAC 地址表满时，根据流量最少的规则删除表项。

```
import switchyard
import random
from switchyard.lib.userlib import *


def main(net: switchyard.llnetbase.LLNetBase):
    my_interfaces = net.interfaces()
    mymacs = [intf.ethaddr for intf in my_interfaces]
    eth_dict = {}

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

        log_debug (f"In {net.name} received packet {packet} on {fromIface}")
        eth = packet.get_header(Ethernet)
        if eth is None:
            log_info("Received a non-Ethernet packet?!")
            return
        if eth.dst in mymacs:
            if eth.src not in eth_dict:
                if len(eth_dict) == 5:
                    tmpk = random.choice(list(eth_dict.keys()))
                    for k, v in eth_dict.items():
                        if v[1] < eth_dict[tmpk][1]:
                            tmpk = k
                    del eth_dict[tmpk]
```

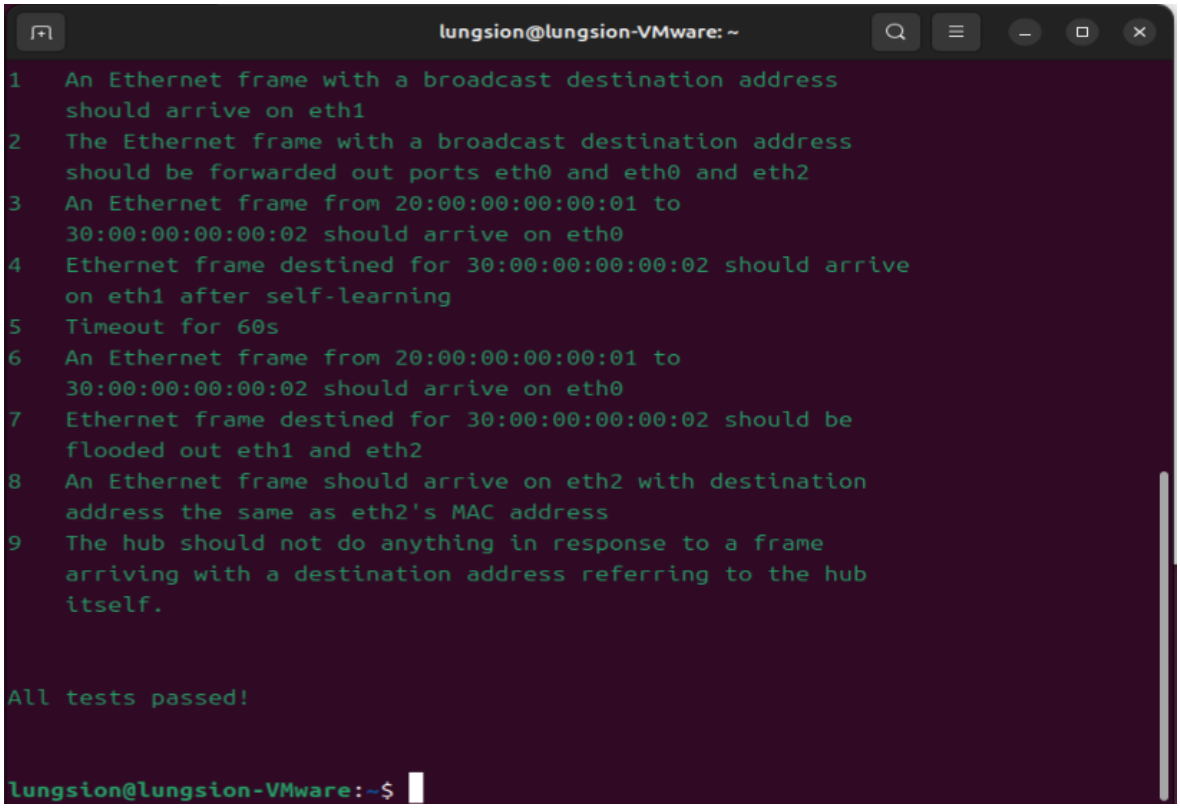
```
                eth_dict[eth.src] = [fromIface, 0]
            log_info("Received a packet intended for me")
        else:
            if eth.src not in eth_dict:
                if len(eth_dict) == 5:
                    tmpk = random.choice(list(eth_dict.keys()))
                    for k, v in eth_dict.items():
                        if v[1] < eth_dict[tmpk][1]:
                            tmpk = k
                    del eth_dict[tmpk]
                eth_dict[eth.src] = [fromIface, 0]
            if eth.dst not in eth_dict:
                for intf in my_interfaces:
                    if fromIface!= intf.name:
                        log_info (f"Flooding packet {packet} to {intf.name}")
                        net.send_packet(intf, packet)
            else:
                log_info (f"Sending packet {packet} to
{eth_dict[eth.dst][0]}")
                eth_dict[eth.dst][1] += 1
                net.send_packet(eth_dict[eth.dst][0], packet)

    net.shutdown()
```

3.6 测试与部署

使用 Switchyard 提供的测试用例进行功能测试。

```
swyard -t testcases/myswitch_to_testscenario.srpy myswitch_to.py
```



在 Mininet 环境中部署交换机，验证其在实际网络环境中的运行情况。

sudo python start_mininet.py

mininet> xterm switch

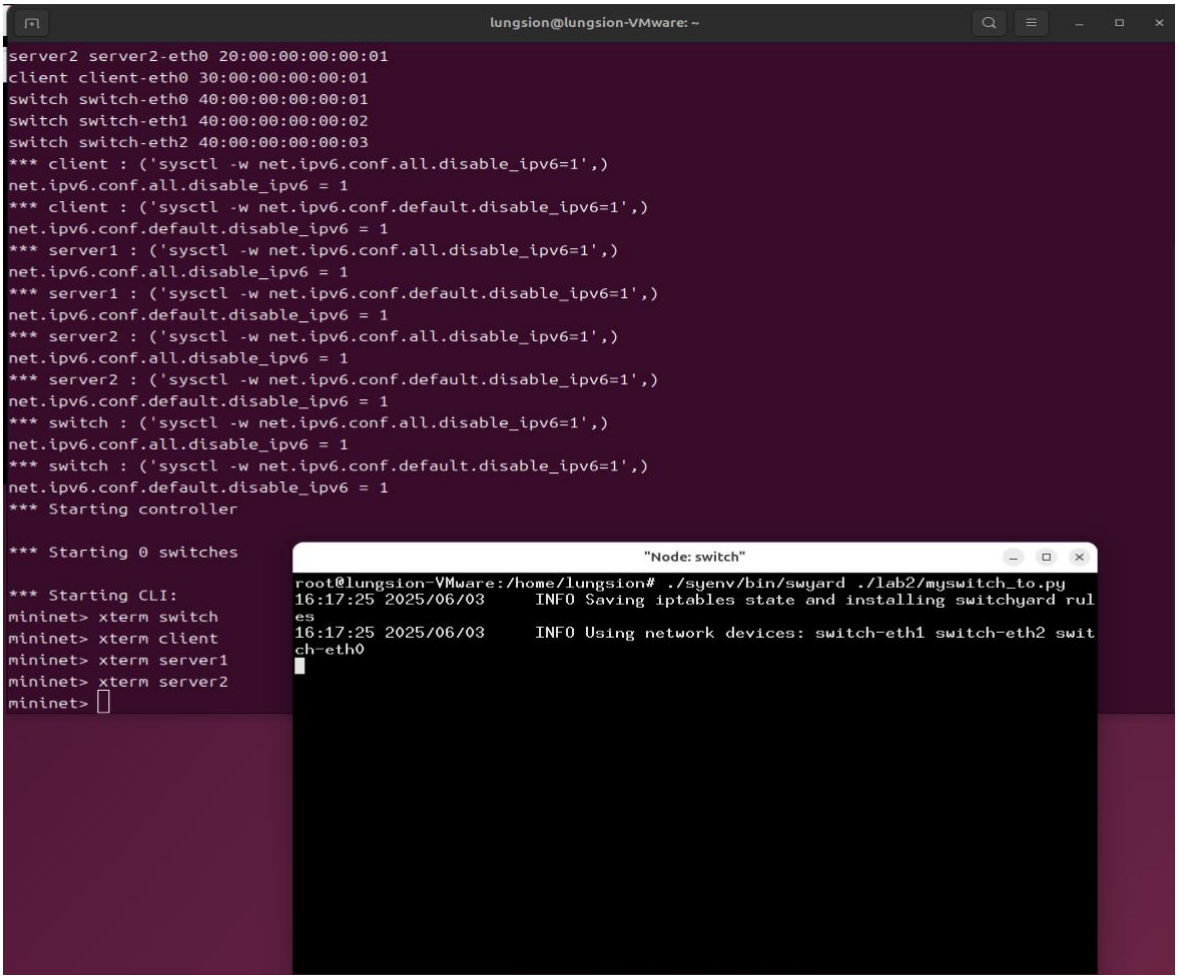
/home/syenv/bin/swyard myswitch_to.py

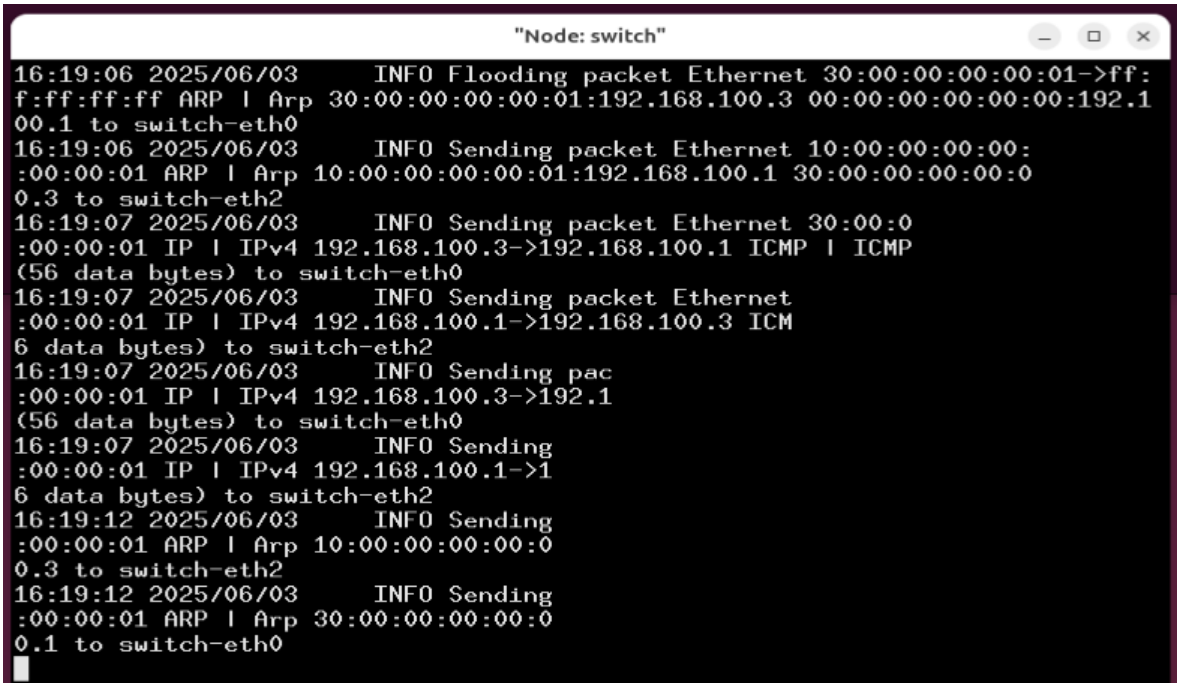
mininet> xterm client

mininet> xterm server1

mininet> xterm server2

client> ping -c 2 192.168.100.1


lungsion@lungsion-VMware: ~
server2 server2-eth0 20:00:00:00:00:01
client client-eth0 30:00:00:00:00:01
switch switch-eth0 40:00:00:00:00:01
switch switch-eth1 40:00:00:00:00:02
switch switch-eth2 40:00:00:00:00:03
*** 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
*** 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
*** server2 : ('sysctl -w net.ipv6.conf.all.disable_ipv6=1',)
net.ipv6.conf.all.disable_ipv6 = 1
*** server2 : ('sysctl -w net.ipv6.conf.default.disable_ipv6=1',)
net.ipv6.conf.default.disable_ipv6 = 1
*** switch : ('sysctl -w net.ipv6.conf.all.disable_ipv6=1',)
net.ipv6.conf.all.disable_ipv6 = 1
*** switch : ('sysctl -w net.ipv6.conf.default.disable_ipv6=1',)
net.ipv6.conf.default.disable_ipv6 = 1
*** Starting controller
*** Starting 0 switches
*** Starting CLI:
mininet> xterm switch
mininet> xterm client
mininet> xterm server1
mininet> xterm server2
mininet>
"Node: switch"
root@lungsion-VMware:/home/lungsion# ./syenv/bin/swyard ./lab2/myswitch_to.py
16:17:25 2025/06/03 INFO Saving iptables state and installing switchyard rul
es
16:17:25 2025/06/03 INFO Using network devices: switch-eth1 switch-eth2 swit
ch-eth0



4 实验总结

通过本次实验，我深入理解了以太网交换机的工作原理，掌握了如何通过交换机的端口接收和转发以太网帧，以及如何实现 MAC 地址的学习和表项管理。实验过程中，我学习了以下内容：

以太网交换机的基本功能：了解了交换机如何通过 MAC 地址表进行帧的转发和泛洪。

表项管理策略：学会了如何实现超时机制、最近最少使用（LRU）规则和流量最少的删除规则。

Switchyard 框架的使用：熟悉了 Switchyard 框架的编程方法，包括如何接收和发送数据包。

Mininet 与网络测试：掌握了 Mininet 网络仿真环境的使用方法，能够通过 Wireshark 捕获并分析网络数据包。

本次实验为后续学习更复杂的网络设备和协议奠定了基础，使我更加深入地理解了计算机网络中交换机的工作机制。

致谢 在此,我向对本报告的工作给予支持和建议的老师和助教,南京大学南京大学智能软件与工程学院殷亚凤教授及其领导的助教们表示感谢.