实验指导书（一）

一、实验环境搭建

- 方式1（推荐）：使用`virtual box`镜像搭建
  - 虚拟机软件`virtual box`，可在官网自行下载:https://download.virtualbox.org/virtualbox/6.1.32/VirtualBox-6.1.32-149290-Win.exe
  - 本实验提供`virtual box`虚拟机镜像文件（`sdn_exp_2023.ova`），已配置`Mininet`和`Ryu`。
  - 环境搭建步骤如下：
    - 安装`virtual box`
    - 导入镜像文件`sdn_exp_2023.ova`（`root`账户密码并未设置，需要的同学可以参考`sudo passwd root`指令）。
- 方式2：使用`VMWare`镜像搭建
  - 虚拟机软件`VMWare`，可在官网自行下载：https://customerconnect.vmware.com/en/downloads/info/slug/desktop_end_user_computing/vmware_workstation_player/17_0
  - 本实验也提供`VMWare`虚拟机镜像文件（`sdn_exp_2023_vmware`），已配置`Mininet`和`Ryu`。
  - 环境搭建步骤如下：
    - 安装`VMWare`
    - 导入镜像文件`sdn_exp_2023_vmware`（密码`sdn`）。
- 方式3：源码安装

```
# 参考视频
# `Workstaion`和`Ubuntu`的安装：https://www.bilibili.com/video/BV1ng4y1z77g
# SDN环境搭建（`Mininet`）：https://www.bilibili.com/video/BV1nC4y1x7Z8

# 安装mininet
git clone https://github.com/mininet/mininet.git
cd mininet/util
sudo ./install.sh -n3v

# 安装wireshark
sudo add-apt-repository ppa:wireshark-dev/stable
sudo apt update
sudo apt install wireshark
```

二、实验工具介绍

1.整体框架

实验需要用到`Mininet`、`Open vSwitch`、`WireShark`等工具。

- `Mininet`：用来在单台计算机上创建一个包含多台网络设备的虚拟网络。
- `Open vSwitch`：`Mininet`中使用的虚拟交换机。
- `WireShark`：抓包工具。

2.Mininet

2.1Mininet命令行

- 启动`mininet`

```
# shell prompt
mn -h # 查看mininet命令中的各个选项
sudo mn -c # 不正确退出时清理mininet
sudo mn #创建默认拓扑，两个主机h1、h2连接到同一交换机s1
```

```
sdn@ubuntu:~/Desktop$ sudo mn
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2
*** Adding switches:
s1
*** Adding links:
(h1, s1) (h2, s1)
*** Configuring hosts
h1 h2
*** Starting controller
c0
*** Starting 1 switches
s1 ...
*** Starting CLI:
mininet>
```

- 常用命令

```
# mininet CLI 中输入
nodes # 查看网络节点
links # 查看网络连接的情况
net # 显示当前网络拓扑
dump # 显示当前网络拓扑的详细信息
xterm h1 # 给节点h1打开一个终端模拟器
sh [COMMAND] # 在mininet命令行中执行COMMAND命令
h1 ping -c3 h2 # 即h1 ping h2 3次
pingall # 即ping all
h1 ifconfig # 查看h1的网络端口及配置
h1 arp # 查看h1的arp表
link s1 h1 down/up # 断开/连接s1和h1的链路
exit # 退出mininet CLI
```

```
mininet> nodes
available nodes are:
c0 h1 h2 s1
mininet> links
h1-eth0<->s1-eth1 (OK OK)
h2-eth0<->s1-eth2 (OK OK)
mininet> net
h1 h1-eth0:s1-eth1
h2 h2-eth0:s1-eth2
s1 lo:  s1-eth1:h1-eth0 s1-eth2:h2-eth0
c0
mininet>
```

2.2创建拓扑

- 命令行拓扑

```
sudo mn --mac --topo=tree,m,n
```

--mac 指定mac地址从1开始递增，而不是无序的mac，方便观察。

--topo 指定拓扑参数，可选用single和linear等参数。

- 自定义拓扑（方式1）：使用`Mininet Python API`创建自定义拓扑，并通过命令行运行

示例位于`mininet/custom/topo-2sw-2host.py`：

```
from mininet.topo import Topo

class MyTopo( Topo ):
    "Simple topology example."

    def build( self ):
        "Create custom topo."

        # Add hosts and switches
        leftHost = self.addHost( 'h1' )
        rightHost = self.addHost( 'h2' )
        leftSwitch = self.addSwitch( 's3' )
        rightSwitch = self.addSwitch( 's4' )

        # Add links
        self.addLink( leftHost, leftSwitch )
        self.addLink( leftSwitch, rightSwitch )
        self.addLink( rightSwitch, rightHost )
```

```
topos = { 'mytopo': ( lambda: MyTopo() ) }
```

命令行运行：

```
cd ~/sdn/mininet/custom
sudo mn --custom topo-2sw-2host.py --topo mytopo --controller=none
```

- 自定义拓扑（方式2）：使用`Mininet Python API`直接创建并运行网络

```
# sudo python topo_recommend.py
from mininet.topo import Topo
from mininet.net import Mininet
from mininet.cli import CLI
from mininet.log import setLogLevel

class S1H2(Topo):
    def build(self):
        s1 = self.addSwitch('s1')
        h1 = self.addHost('h1')
        h2 = self.addHost('h2')

        self.addLink(s1, h1)
        self.addLink(s1, h2)


def run():
    topo = S1H2()
    net = Mininet(topo)

    net.start()
    CLI(net)
    net.stop()


if __name__ == '__main__':
    setLogLevel('info') # output, info, debug
    run()
```

这种方式写法较为复杂，但简化了运行命令：

```
sudo python topo_recommend.py
```

2.3参考文档

进一步学习可以参考`Mininet`官网：http://mininet.org/

3.OVS

3.1实验中常用的几条指令

- 查看交换机的基本信息

以默认拓扑启动`mininet`，打开新终端，输入`sudo ovs-vsctl show`

```
sdn@ubuntu:~/Desktop$ sudo ovs-vsctl show
02e3be6e-16d8-44c3-9f04-998d777c591f
    Bridge s1
        Controller "ptcp:6654"
        Controller "tcp:127.0.0.1:6653"
        fail_mode: secure
        Port s1-eth2
            Interface s1-eth2
        Port s1-eth1
            Interface s1-eth1
        Port s1
            Interface s1
                type: internal
    ovs_version: "2.13.8"
```

```
root@ubuntu:/home/sdn/Desktop# wireshark
 ** (wireshark:4008) 12:16:58.266010 [GUI WARNING] -- QStandardPaths: XDG_RUNTIM
me-root'
```

- 生成树协议

```
sudo ovs-vsctl set bridge s1 stp_enable=true #开启STP，s1为设备名
sudo ovs-vsctl get bridge s1 stp_enable
sudo ovs-vsctl list bridge
```

- 查看mac表
  - 启动`mininet`，注意禁用控制器，否则mac表可能学习不到内容

```
sdn@ubuntu:~/Desktop$ sudo mn --mac --topo=tree,2,2 --controller=none
*** Creating network
*** Adding controller
*** Adding hosts:
h1 h2 h3 h4
*** Adding switches:
s1 s2 s3
*** Adding links:
(s1, s2) (s1, s3) (s2, h1) (s2, h2) (s3, h3) (s3, h4)
*** Configuring hosts
h1 h2 h3 h4
*** Starting controller

*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet> nodes
available nodes are:
h1 h2 h3 h4 s1 s2 s3
```

  - 对每个交换机执行`sudo ovs-vsctl del-fail-mode xx`，否则mac表将仍然学习不到东西

```
sdn@ubuntu:~/Desktop$ sudo ovs-vsctl del-fail-mode s1
sdn@ubuntu:~/Desktop$ sudo ovs-vsctl del-fail-mode s2
sdn@ubuntu:~/Desktop$ sudo ovs-vsctl del-fail-mode s3
```

  - `pingall`令所有主机发送数据包，防止沉默主机现象

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4
h2 -> h1 h3 h4
h3 -> h1 h2 h4
h4 -> h1 h2 h3
*** Results: 0% dropped (12/12 received)
```

- `sudo ovs-appctl fdb/show xx` 查看mac表

```
sdn@ubuntu:~/Desktop$ sudo ovs-appctl fdb/show s1
 port  VLAN  MAC                Age
    1     0  5a:14:cc:b4:16:a9   27
    1     0  00:00:00:00:00:02   26
    1     0  00:00:00:00:00:01   26
    2     0  00:00:00:00:00:03   25
    2     0  56:45:65:64:3b:83   24
    2     0  00:00:00:00:00:04   24
sdn@ubuntu:~/Desktop$ sudo ovs-appctl fdb/show s2
 port  VLAN  MAC                Age
    3     0  46:82:18:f7:23:1c   29
    1     0  00:00:00:00:00:01   28
    2     0  00:00:00:00:00:02   27
    3     0  00:00:00:00:00:03   27
    3     0  56:45:65:64:3b:83   26
    3     0  00:00:00:00:00:04   26
sdn@ubuntu:~/Desktop$ sudo ovs-appctl fdb/show s3
 port  VLAN  MAC                Age
    3     0  46:02:11:b0:73:f1   30
    3     0  5a:14:cc:b4:16:a9   30
    3     0  00:00:00:00:00:02   29
    3     0  00:00:00:00:00:01   29
    1     0  00:00:00:00:00:03   28
    2     0  00:00:00:00:00:04   27
```

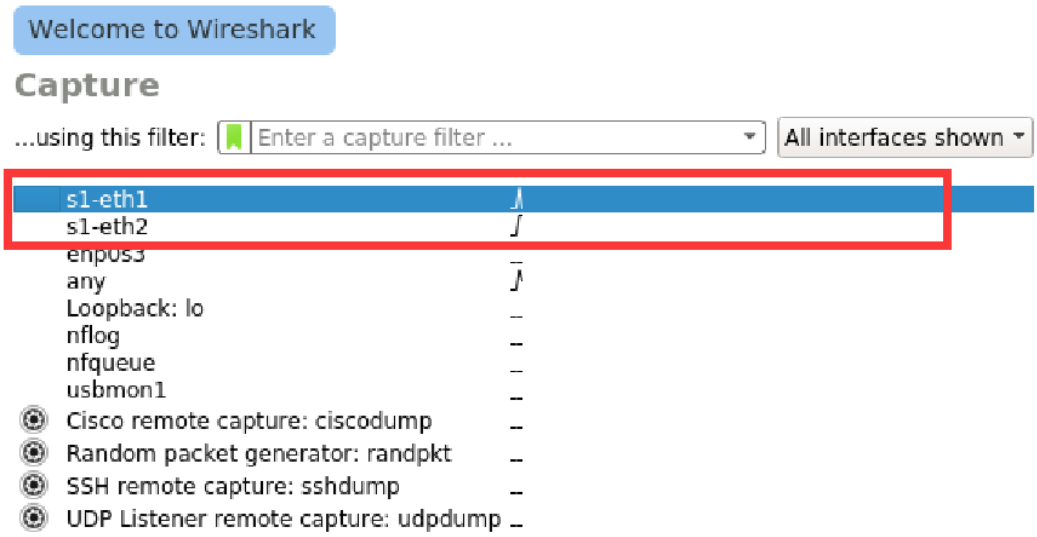
3.2参考文档

`ovs` 的详细学习可参考官方网站http://www.openvswitch.org/

4.WireShark

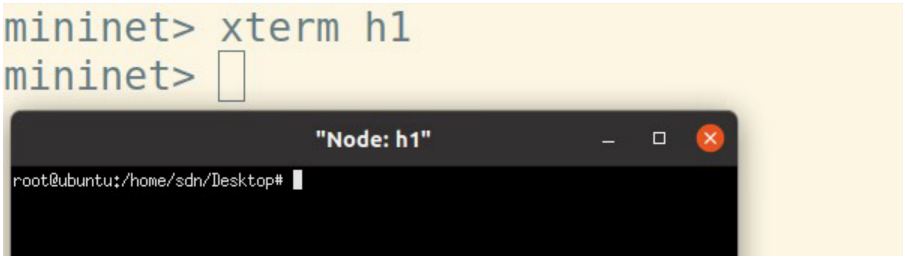
- 抓交换机的包

`sudo wireshark`，选择相应端口：

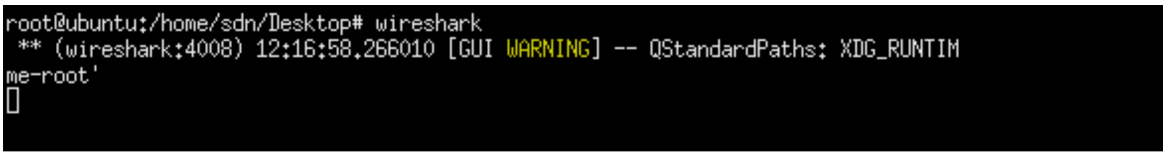


- 抓主机的包

在`mininet CLI`中执行`xterm h1`，打开h1的终端:



在h1终端中运行`wireshark`：



三、实验任务

- 使用`Mininet`的Python API搭建`k=4`的`fat tree`拓扑；
- 使用`pingall`查看各主机之间的连通情况;
- 若主机之间未连通，分析原因并解决（使用`wireshark`抓包分析）
- 若主机连通，分析数据包的路径（提示：`ovs-appctl fdb/show`查看MAC表）
- 完成实验报告并提交到思源学堂
- 要求不能使用控制器