Universidade Federal de Pernambuco - Centro de Informática

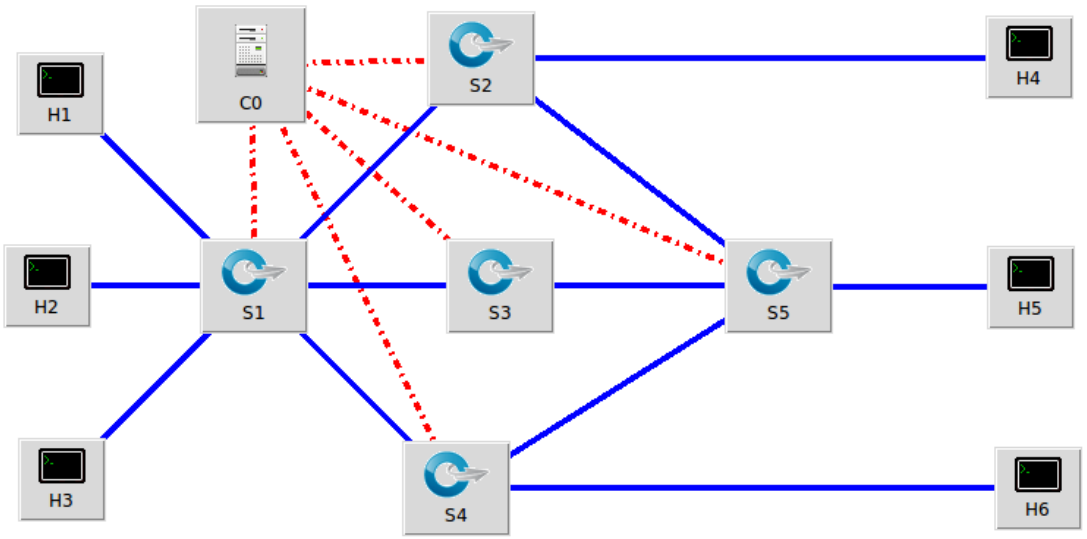
Disciplina: Redes de Computadores, Prof. Djamel Sadok

Estudante: Matheus Johann Araújo

Data de entrega: 11/01/2022

Projeto Final de SDN – Pox Controller e Mininet

Topologia da Rede (MiniEdit) – Arquivo: “projeto_rede.mn”



Código da Rede (Mininet) – Arquivo: “projeto_rede.py”

```
#!/usr/bin/python
# Projeto SDN - Controlador POX e Rede Mininet
# Estudante: Matheus Johann Araujo

from mininet.net import Mininet
from mininet.node import Controller, RemoteController, OVSController
from mininet.node import CPULimitedHost, Host, Node
from mininet.node import OVSKernelSwitch, UserSwitch
from mininet.node import IVSSwitch
from mininet.cli import CLI
```

```
from mininet.log import setLogLevel, info
from mininet.link import TCLink, Intf
from subprocess import call

def myNetwork():

    net = Mininet( topo=None,
                   build=False,
                   ipBase='16.32.64.0/8')

    info( '*** Adding controller\n' )
    C0=net.addController(name='C0',
                         controller=RemoteController,
                         ip='127.0.0.1',
                         protocol='tcp',
                         port=6633)

    info( '*** Add switches\n')
    S1 = net.addSwitch('S1', cls=OVSKernelSwitch, dpid='1')
    S2 = net.addSwitch('S2', cls=OVSKernelSwitch, dpid='2')
    S3 = net.addSwitch('S3', cls=OVSKernelSwitch, dpid='3')
    S4 = net.addSwitch('S4', cls=OVSKernelSwitch, dpid='4')
    S5 = net.addSwitch('S5', cls=OVSKernelSwitch, dpid='5')

    info( '*** Add hosts\n')
    H1 = net.addHost('H1', cls=Host, ip='16.32.64.1', mac='a0:00:00:00:00:01',
defaultRoute=None)
    H2 = net.addHost('H2', cls=Host, ip='16.32.64.2', mac='a0:00:00:00:00:02',
defaultRoute=None)
    H3 = net.addHost('H3', cls=Host, ip='16.32.64.3', mac='a0:00:00:00:00:03',
defaultRoute=None)
    H4 = net.addHost('H4', cls=Host, ip='16.32.64.4', mac='b0:00:00:00:00:04',
defaultRoute=None)
    H5 = net.addHost('H5', cls=Host, ip='16.32.64.5', mac='c0:00:00:00:00:05',
defaultRoute=None)
    H6 = net.addHost('H6', cls=Host, ip='16.32.64.6', mac='d0:00:00:00:00:06',
defaultRoute=None)

    info( '*** Add links\n')

    #S1H1 = S1H2 = S1H3 = S2H4 = S5H5 = S4H6 = {'bw':100,'max_queue_size':100}
    #S1S2 = S1S3 = S1S4 = S2S5 = S3S5 = S4S5 = {'bw':100,'max_queue_size':100}

    S1H1 = S1H2 = S1H3 = S2H4 = S5H5 = S4H6 =
{'bw':100,'max_queue_size':100,'loss':1}
    S1S2 = S1S3 = S1S4 = S2S5 = S3S5 = S4S5 =
{'bw':100,'max_queue_size':100,'loss':2,'delay':'1ms'}

    net.addLink(S1, H1, cls=TCLink , **S1H1)
```

```
    net.addLink(S1, H2, cls=TCLink , **S1H2)
    net.addLink(S1, H3, cls=TCLink , **S1H3)
    net.addLink(S1, S2, cls=TCLink , **S1S2)
    net.addLink(S1, S3, cls=TCLink , **S1S3)
    net.addLink(S1, S4, cls=TCLink , **S1S4)
    net.addLink(S2, S5, cls=TCLink , **S2S5)
    net.addLink(S2, H4, cls=TCLink , **S2H4)
    net.addLink(S3, S5, cls=TCLink , **S3S5)
    net.addLink(S4, S5, cls=TCLink , **S4S5)
    net.addLink(S5, H5, cls=TCLink , **S5H5)
    net.addLink(S4, H6, cls=TCLink , **S4H6)

    info( '*** Starting network\n')
    net.build()
    info( '*** Starting controllers\n')
    for controller in net.controllers:
        controller.start()

    info( '*** Starting switches\n')

    net.get('S1').start([C0])
    net.get('S2').start([C0])
    net.get('S3').start([C0])
    net.get('S5').start([C0])
    net.get('S4').start([C0])

    info( '*** Post configure switches and hosts\n')

    net.start()

    info( '*** Testing network\n')

    cmd_ping = "ping -c 1"

    print("")
    print("H1 {} H2".format(cmd_ping))
    print H1.cmd(cmd_ping, H2.IP())

    print("")
    print("H2 {} H3".format(cmd_ping))
    print H2.cmd(cmd_ping, H3.IP())

    print("")
    print("H3 {} H4".format(cmd_ping))
    print H3.cmd(cmd_ping, H4.IP())

    print("")
    print("H1 {} H5".format(cmd_ping))
    print H1.cmd(cmd_ping, H5.IP())
```

```
    print("")
    print("H3 {} H6".format(cmd_ping))
    print H3.cmd(cmd_ping, H6.IP())

    print("")
    print("pingall")
    net.pingAll()

    CLI(net)
    net.stop()

if __name__ == '__main__':
    setLogLevel( 'info' )
    myNetwork()
```

Código do Controlador (POX) – Arquivo: “projeto_controlador.py”

```
# ext/projeto_controlador.py
# Projeto SDN - Controlador POX e Rede Mininet
# Estudante: Matheus Johann Araujo

from pox.core import core
import pox.openflow.libopenflow_01 as of
from pox.lib.revent import *
from pox.lib.addresses import EthAddr, IPAddr
from pox.openflow.discovery import Discovery
from pox.host_tracker import host_tracker
from host_tracker import launch as launch_host_tracker

class projeto_controlador(EventMixin):

    def __init__(self):
        launch_host_tracker()
        def startup():
            core.openflow.addListeners(self, priority = 0)
            core.openflow_discovery.addListeners(self)
            core.host_tracker.addListeners(self)
        core.call_when_ready(startup, ('openflow','openflow_discovery',
'host_tracker'))
        self.listenTo(core.openflow)

    def _handle_LinkEvent(self, event):
        print("-" * 160)
        link = event.link
        if event.added:
```

```
            print("Link Switches Added")
        elif event.removed:
            print("Link Switches Removed")
        print("S{}-eth{}:S{}-eth{}".format(link.dpid1, link.port1, link.dpid2,
link.port2))

    def _handle_HostEvent (self, event):
        print("-" * 160)
        print("Host: {} | Switch dpid: {} | Port:
{}".format(EthAddr(event.entry.macaddr), event.entry.dpid, event.entry.port))

    # Variavel que armazena a conexao com os Switches
    switches = {}

    # DPIDs Switches
    dpid_S1 = None
    dpid_S2 = None
    dpid_S3 = None
    dpid_S4 = None
    dpid_S5 = None

    def _handle_ConnectionUp(self, event) :
        print("-" * 160)
        print("Connection UP from Switch dpid: {}".format(event.dpid)) # Numero ID da
conexao com o Switch
        self.switches[event.dpid] = event.connection # Conexao com o
Switch
        for m in event.connection.features.ports:
            if m.port_no == 65534:
                continue
            print("Switch Eth: {} | Mac Port: {}".format(m.name, EthAddr(m.hw_addr)))
            if m.name == "S1-eth1":
                self.dpid_S1 = event.connection.dpid
            elif m.name == "S2-eth1":
                self.dpid_S2 = event.connection.dpid
            elif m.name == "S3-eth1":
                self.dpid_S3 = event.connection.dpid
            elif m.name == "S4-eth1":
                self.dpid_S4 = event.connection.dpid
            elif m.name == "S5-eth1":
                self.dpid_S5 = event.connection.dpid
        print("-" * 160)
        print("S1 dpid: {} | S2 dpid: {} | S3 dpid: {} | S4 dpid: {} | S5 dpid:
{}".format(self.dpid_S1, self.dpid_S2, self.dpid_S3, self.dpid_S4, self.dpid_S5))

    def packet_type(self, packet):
        if packet.find('icmp'): return 'icmp'
        if packet.find('arp'):  return 'arp'
        if packet.find('dhcp'): return 'dhcp'
```

```
        if packet.find('tcp'):  return 'tcp'
        if packet.find('udp'):  return 'udp'
        if parsed.find('ipv4'): return "ipv4"
        return 'unknown'

    def flow_add(self, dpid, out_port = None, dl_type = None, priority = None,
idle_timeout = None, hard_timeout = None, nw_src = None, nw_dst = None, in_port =
None):
        msg = of.ofp_flow_mod()
        # PORT
        if out_port == None:
            out_port = [of.OFPP_ALL]
        elif isinstance(out_port, int):
            out_port = [out_port]
        if not isinstance(out_port, list):
            return
        # IPv4
        if isinstance(nw_src, str):
            nw_src = IPAddr(nw_src)
        if isinstance(nw_dst, str):
            nw_dst = IPAddr(nw_dst)
        print("-" * 160)
        print("SWITCH DPID: {} | IN_PORT: {} | OUT_PORT: {} | DL_TYPE: {} | PRIORITY:
{} | IDLE_TIMEOUT: {} | HARD_TIMEOUT: {} | NW_SRC: {} | NW_DST: {}".format(dpid,
in_port, out_port, dl_type, priority, idle_timeout, hard_timeout, nw_src, nw_dst))
        # priority
        if priority != None:
            msg.priority = priority
        # idle_timeout
        if idle_timeout != None:
            msg.idle_timeout = idle_timeout
        # hard_timeout
        if hard_timeout != None:
            msg.hard_timeout = hard_timeout
        # dl_type
        if dl_type != None:
            msg.match.dl_type = dl_type
        # nw_src
        if nw_src != None:
            msg.match.nw_src = nw_src
        # nw_dst
        if nw_dst != None:
            msg.match.nw_dst = nw_dst
        # in_port
        if in_port:
            msg.match.in_port = in_port
        for num in out_port:
            msg.actions.append(of.ofp_action_output(port = num))
        self.switches[dpid].send(msg)
```

```
    def flow_add_arp(self, dpid, out_port = None):
        self.flow_add(dpid, out_port=out_port, dl_type=0x0806, priority=1,
idle_timeout=10, hard_timeout=10)

    def flow_add_ip(self, dpid, out_port = None, nw_src = None, nw_dst = None,
in_port = None):
        self.flow_add(dpid, out_port=out_port, dl_type=0x0800, priority=10,
idle_timeout=0, hard_timeout=0, nw_src=nw_src, nw_dst=nw_dst, in_port=in_port)

    def flow_add_in_port(self, dpid, out_port = None, nw_src = None, nw_dst = None,
in_port = None):
        self.flow_add(dpid, out_port=out_port, priority=10, idle_timeout=0,
hard_timeout=0, nw_src=nw_src, nw_dst=nw_dst, in_port=in_port)

    def set_flow_S1(self, dpid):
        print("Set Flow S1")
        # S1-eth1:H1-eth0 S1-eth2:H2-eth0 S1-eth3:H3-eth0 S1-eth4:S2-eth1 S1-eth5:S3-
eth1 S1-eth6:S4-eth1
        self.flow_add_arp(dpid)
        # S1-eth1:H1-eth0
        self.flow_add_ip(dpid, out_port = 1, nw_dst = "16.32.64.1")
        # S1-eth2:H2-eth0
        self.flow_add_ip(dpid, out_port = 2, nw_dst = "16.32.64.2")
        # S1-eth3:H3-eth0
        self.flow_add_ip(dpid, out_port = 3, nw_dst = "16.32.64.3")
        # S1-eth4:S2-eth1
        self.flow_add_ip(dpid, out_port = [4, 5], nw_dst = "16.32.64.4")
        # S1-eth5:S3-eth1
        self.flow_add_ip(dpid, out_port = 5, nw_dst = "16.32.64.5")
        # S1-eth6:S4-eth1
        self.flow_add_ip(dpid, out_port = [5, 6], nw_src = "16.32.64.3", nw_dst =
"16.32.64.6")

    def set_flow_S2(self, dpid):
        print("Set Flow S2")
        # S2-eth1:S1-eth4 S2-eth3:H4-eth0
        self.flow_add_arp(dpid, out_port = [1, 3])
        # S2-eth3:H4-eth0
        self.flow_add_ip(dpid, out_port = 3, nw_dst = "16.32.64.4")
        # S2-eth1:S1-eth4
        self.flow_add_in_port(dpid, out_port = 1, in_port = 3)
        # S2-eth2:S5-eth1
        self.flow_add_in_port(dpid, out_port = 3, in_port = 2)

    def set_flow_S3(self, dpid):
        print("Set Flow S3")
        # S3-eth1:S1-eth5 S3-eth2:S5-eth2
        self.flow_add_arp(dpid)
```

```
        # S3-eth2:S5-eth2
        self.flow_add_in_port(dpid, out_port = 2, in_port = 1)
        # S3-eth1:S1-eth5
        self.flow_add_in_port(dpid, out_port = 1, in_port = 2)

    def set_flow_S4(self, dpid):
        print("Set Flow S4")
        # S4-eth1:S1-eth6 S4-eth3:H6-eth0
        self.flow_add_arp(dpid, out_port = [1, 3])
        # S4-eth3:H6-eth0
        self.flow_add_ip(dpid, out_port = 3, nw_src = "16.32.64.3", nw_dst =
"16.32.64.6")
        # S4-eth1:S1-eth6
        self.flow_add_in_port(dpid, out_port = 1, in_port = 3)

    def set_flow_S5(self, dpid):
        print("Set Flow S5")
        # S5-eth2:S3-eth2 S5-eth4:H5-eth0
        self.flow_add_arp(dpid, out_port = [2, 4])
        # S5-eth4:H5-eth0
        self.flow_add_ip(dpid, out_port = 4, nw_dst = "16.32.64.5", in_port = 2)
        # S5-eth2:S3-eth2
        self.flow_add_in_port(dpid, out_port = 2, in_port = 4)
        # S5-eth1:S2-eth2 S5-eth2:S3-eth2
        self.flow_add_ip(dpid, out_port = 1, nw_dst = "16.32.64.4", in_port = 2)
        # S5-eth3:S4-eth2 S5-eth2:S3-eth2
        self.flow_add_ip(dpid, out_port = 3, nw_dst = "16.32.64.6", in_port = 2)

    def set_flow(self, dpid):
        print("-" * 160)
        if dpid == self.dpid_S1:
            self.set_flow_S1(dpid)
        elif dpid == self.dpid_S2:
            self.set_flow_S2(dpid)
        elif dpid == self.dpid_S3:
            self.set_flow_S3(dpid)
        elif dpid == self.dpid_S4:
            self.set_flow_S4(dpid)
        elif dpid == self.dpid_S5:
            self.set_flow_S5(dpid)

    def _handle_PacketIn (self, event):
        packet = event.parsed
        packet_type = self.packet_type(packet)
        print("-" * 160)
        print("PACKET TYPE: {} | SRC: {} | DST: {} | MULTICAST DST:
{}".format(packet_type, EthAddr(packet.src), EthAddr(packet.dst),
packet.dst.is_multicast))
        if packet_type == "unknown":
```

```
            return
        dpid = event.connection.dpid
        self.set_flow(dpid)

def launch():
    core.openflow.miss_send_len = 1024
    core.registerNew(projeto_controlador)
```

Executando o Projeto de SDN

Para fazer com que a rede funcione corretamente é necessário executar os códigos (*script* do controlador e o *script* da rede) dentro de terminais Linux. O computador deve possuir o POX *Controller* e Mininet/MiniEdit previamente instalados e configurados para que os *scripts* sejam executados sem problemas.

Comando usado somente para desenvolver a topologia da rede no MiniEdit:

sudo ~/mininet/examples/miniedit.py

Comando para iniciar o controlador POX:

sudo pkill -9 python; clear; cd ~/pox; python pox.py –verbose projeto_controlador openflow.discovery host_tracker py log --no-default --file=/tmp/mylog.log

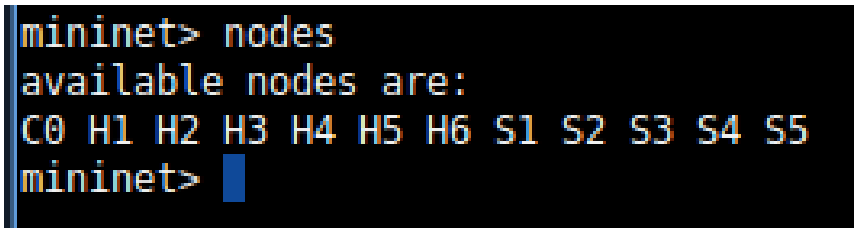
Comando para iniciar a rede Mininet:

sudo mn -c; clear; sudo python projeto_rede.py

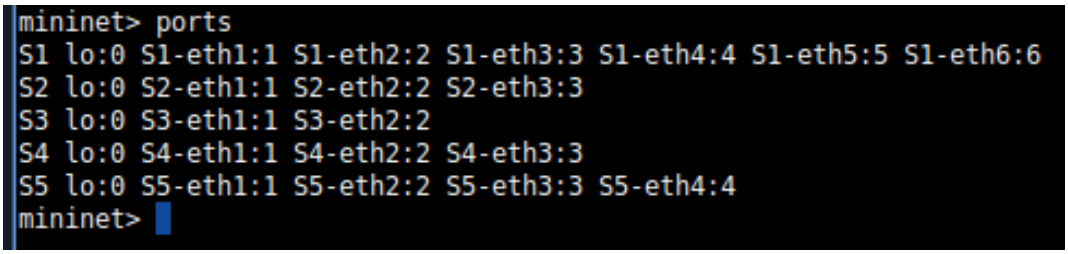
Executando comandos dentro do CLI do Mininet

Para visualizar toda a configuração da topologia da rede no CLI do Mininet, foram usados os seguintes comandos:

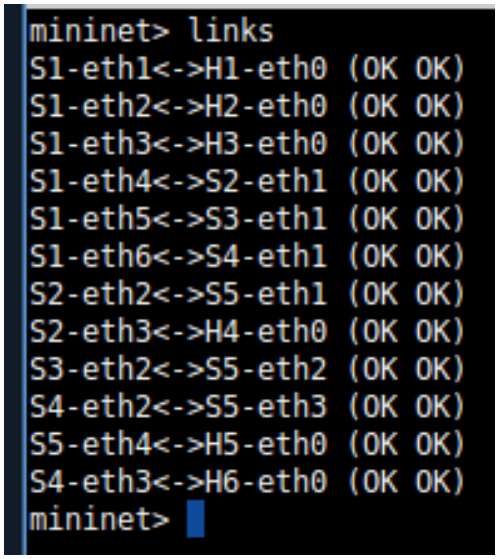
Comando “nodes”:



Comando “ports”:



Comando “links”:



Comando “net”:

```
mininet> net
H1 H1-eth0:S1-eth1
H2 H2-eth0:S1-eth2
H3 H3-eth0:S1-eth3
H4 H4-eth0:S2-eth3
H5 H5-eth0:S5-eth4
H6 H6-eth0:S4-eth3
S1 lo:  S1-eth1:H1-eth0 S1-eth2:H2-eth0 S1-eth3:H3-eth0 S1-eth4:S2-eth1 S1-eth5:S3-eth1 S1-eth6:S4-eth1
S2 lo:  S2-eth1:S1-eth4 S2-eth2:S5-eth1 S2-eth3:H4-eth0
S3 lo:  S3-eth1:S1-eth5 S3-eth2:S5-eth2
S4 lo:  S4-eth1:S1-eth6 S4-eth2:S5-eth3 S4-eth3:H6-eth0
S5 lo:  S5-eth1:S2-eth2 S5-eth2:S3-eth2 S5-eth3:S4-eth2 S5-eth4:H5-eth0
C0
mininet>
```

Comando “dump”:

```
mininet> dump
<Host H1: H1-eth0:16.32.64.1 pid=13391>
<Host H2: H2-eth0:16.32.64.2 pid=13396>
<Host H3: H3-eth0:16.32.64.3 pid=13401>
<Host H4: H4-eth0:16.32.64.4 pid=13406>
<Host H5: H5-eth0:16.32.64.5 pid=13411>
<Host H6: H6-eth0:16.32.64.6 pid=13416>
<OVSSwitch S1: lo:127.0.0.1,S1-eth1:None,S1-eth2:None,S1-eth3:None,S1-eth4:None,S1-eth5:None,S1-eth6:None pid=13374>
<OVSSwitch S2: lo:127.0.0.1,S2-eth1:None,S2-eth2:None,S2-eth3:None pid=13377>
<OVSSwitch S3: lo:127.0.0.1,S3-eth1:None,S3-eth2:None pid=13380>
<OVSSwitch S4: lo:127.0.0.1,S4-eth1:None,S4-eth2:None,S4-eth3:None pid=13383>
<OVSSwitch S5: lo:127.0.0.1,S5-eth1:None,S5-eth2:None,S5-eth3:None,S5-eth4:None pid=13386>
<RemoteController C0: 127.0.0.1:6633 pid=13367>
mininet>
```

Para visualizar as regras de encaminhamento definidas nos Switches pelo controlador POX foi utilizado o comando “dpctl dump-flows”, resultado da execução abaixo:

```
mininet> dpctl dump-flows
*** S1 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=376.661s, table=0, n_packets=198, n_bytes=8118, idle_age=2,
priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=376.622s, table=0, n_packets=12, n_bytes=504, idle_age=75,
priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLLER:65535
 cookie=0x0, duration=305.327s, table=0, n_packets=0, n_bytes=0, idle_age=336,
priority=10,ip,nw_dst=16.32.64.1 actions=output:1
 cookie=0x0, duration=305.327s, table=0, n_packets=0, n_bytes=0, idle_age=326,
priority=10,ip,nw_dst=16.32.64.2 actions=output:2
 cookie=0x0, duration=305.327s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,nw_dst=16.32.64.3 actions=output:3
 cookie=0x0, duration=305.327s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,nw_dst=16.32.64.4 actions=output:4,output:5
```

```
 cookie=0x0, duration=305.327s, table=0, n_packets=0, n_bytes=0, idle_age=306,
priority=10,ip,nw_dst=16.32.64.5 actions=output:5
 cookie=0x0, duration=305.326s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,nw_src=16.32.64.3,nw_dst=16.32.64.6 actions=output:5,output:6
*** S2 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=376.619s, table=0, n_packets=131, n_bytes=5371, idle_age=1,
priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=376.583s, table=0, n_packets=0, n_bytes=0, idle_age=376,
priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLLER:65535
 cookie=0x0, duration=306.344s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,nw_dst=16.32.64.4 actions=output:3
 cookie=0x0, duration=306.344s, table=0, n_packets=0, n_bytes=0, idle_age=311,
priority=10,in_port=3 actions=output:1
 cookie=0x0, duration=306.343s, table=0, n_packets=0, n_bytes=0, idle_age=367,
priority=10,in_port=2 actions=output:3
*** S3 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=376.547s, table=0, n_packets=131, n_bytes=5371, idle_age=1,
priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=376.511s, table=0, n_packets=0, n_bytes=0, idle_age=376,
priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLLER:65535
 cookie=0x0, duration=374.730s, table=0, n_packets=75, n_bytes=4326, idle_age=304,
priority=10,in_port=1 actions=output:2
 cookie=0x0, duration=374.730s, table=0, n_packets=27, n_bytes=1750, idle_age=304,
priority=10,in_port=2 actions=output:1
*** S4 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=376.401s, table=0, n_packets=132, n_bytes=5412, idle_age=0,
priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=376.387s, table=0, n_packets=0, n_bytes=0, idle_age=376,
priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLLER:65535
 cookie=0x0, duration=306.348s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,nw_src=16.32.64.3,nw_dst=16.32.64.6 actions=output:3
 cookie=0x0, duration=306.348s, table=0, n_packets=1, n_bytes=42, idle_age=304,
priority=10,in_port=3 actions=output:1
*** S5 ------------------------------------------------------------------------
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=376.521s, table=0, n_packets=196, n_bytes=8036, idle_age=0,
priority=65000,dl_dst=01:23:20:00:00:01,dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=376.511s, table=0, n_packets=0, n_bytes=0, idle_age=376,
priority=32769,arp,dl_dst=02:00:00:00:be:ef actions=CONTROLLER:65535
 cookie=0x0, duration=304.339s, table=0, n_packets=0, n_bytes=0, idle_age=306,
priority=10,ip,in_port=2,nw_dst=16.32.64.5 actions=output:4
 cookie=0x0, duration=304.339s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,in_port=2,nw_dst=16.32.64.4 actions=output:1
 cookie=0x0, duration=304.339s, table=0, n_packets=0, n_bytes=0, idle_age=316,
priority=10,ip,in_port=2,nw_dst=16.32.64.6 actions=output:3
```

```
 cookie=0x0, duration=304.339s, table=0, n_packets=0, n_bytes=0, idle_age=304,
priority=10,in_port=4 actions=output:2
mininet>
```

Para visualizar o mapeamento IP-MAC pode ser utilizado o comando “arp” nos *hosts* da rede, abaixo é demostrado o resultado no terceiro *host* (H3).

```
mininet> H3 arp
Address                  HWtype  HWaddress           Flags Mask            Iface
16.32.64.6               ether   d0:00:00:00:00:06   C                     H3-eth0
16.32.64.4               ether   b0:00:00:00:00:04   C                     H3-eth0
16.32.64.5               ether   c0:00:00:00:00:05   C                     H3-eth0
16.32.64.2               ether   a0:00:00:00:00:02   C                     H3-eth0
16.32.64.1               ether   a0:00:00:00:00:01   C                     H3-eth0
mininet>
```

Dentro da CLI (*Command Line Interface*) do Mininet foi utilizado o comando “ping” (utilitário que usa o protocolo ICMP) para testar a conectividade entre os *hosts* (H1, H2, H3, H4, H5 e H6). No Mininet tem-se o comando “pingall” que avalia a conectividade de comunicação de todos *hosts* existentes na topologia da rede.

Executei o “pingall” duas vezes para demostrar os efeitos da configuração dos links *(Host:Switch* e *Switch:Switch*) no que diz respeito ao *loss* e *delay*.

Primeiro resultado do comando “pingall”, script Mininet sem as configurações de *loss* e *delay*.

```
S1H1 = S1H2 = S1H3 = S2H4 = S5H5 = S4H6 = {'bw':100,'max_queue_size':100}
S1S2 = S1S3 = S1S4 = S2S5 = S3S5 = S4S5 = {'bw':100,'max_queue_size':100}

net.addLink(S1, H1, cls=TCLink , **S1H1)
net.addLink(S1, H2, cls=TCLink , **S1H2)
net.addLink(S1, H3, cls=TCLink , **S1H3)
net.addLink(S1, S2, cls=TCLink , **S1S2)
net.addLink(S1, S3, cls=TCLink , **S1S3)
net.addLink(S1, S4, cls=TCLink , **S1S4)
net.addLink(S2, S5, cls=TCLink , **S2S5)
net.addLink(S2, H4, cls=TCLink , **S2H4)
net.addLink(S3, S5, cls=TCLink , **S3S5)
net.addLink(S4, S5, cls=TCLink , **S4S5)
net.addLink(S5, H5, cls=TCLink , **S5H5)
net.addLink(S4, H6, cls=TCLink , **S4H6)
```

```
*** Ping: testing ping reachability
H1 -> H2 H3 H4 H5 X
H2 -> H1 H3 H4 H5 X
H3 -> H1 H2 H4 H5 H6
H4 -> H1 H2 H3 H5 X
H5 -> H1 H2 H3 H4 X
H6 -> X X H3 X X
*** Results: 26% dropped (22/30 received)
mininet>
```

Segundo resultado do comando "pingall", script Mininet com as configurações de *loss* e *delay*.

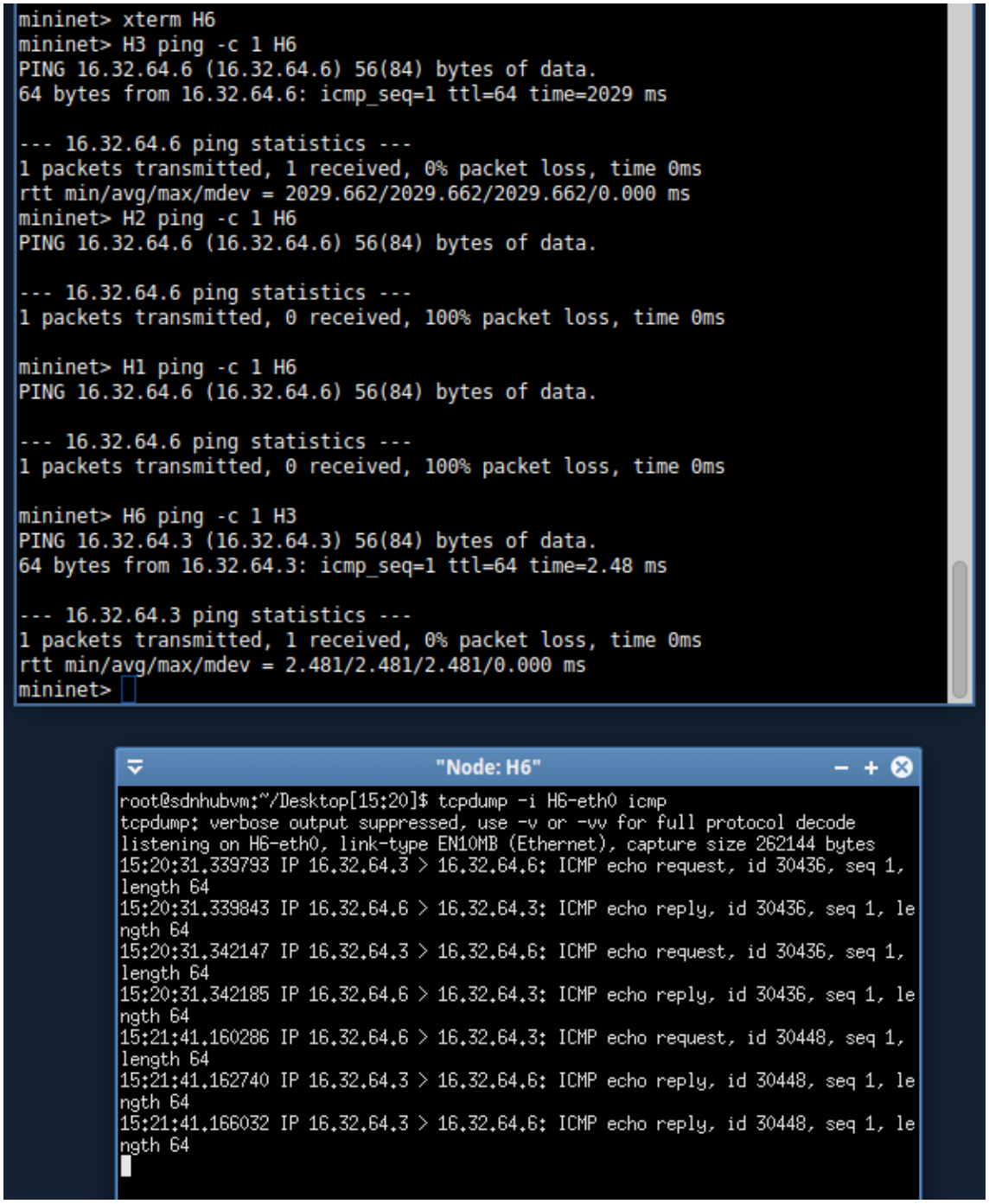
```
S1H1 = S1H2 = S1H3 = S2H4 = S5H5 = S4H6 = {'bw':100,'max_queue_size':100,'loss':1}
S1S2 = S1S3 = S1S4 = S2S5 = S3S5 = S4S5 = {'bw':100,'max_queue_size':100,'loss':2,'delay':'1ms'}

net.addLink(S1, H1, cls=TCLink , **S1H1)
net.addLink(S1, H2, cls=TCLink , **S1H2)
net.addLink(S1, H3, cls=TCLink , **S1H3)
net.addLink(S1, S2, cls=TCLink , **S1S2)
net.addLink(S1, S3, cls=TCLink , **S1S3)
net.addLink(S1, S4, cls=TCLink , **S1S4)
net.addLink(S2, S5, cls=TCLink , **S2S5)
net.addLink(S2, H4, cls=TCLink , **S2H4)
net.addLink(S3, S5, cls=TCLink , **S3S5)
net.addLink(S4, S5, cls=TCLink , **S4S5)
net.addLink(S5, H5, cls=TCLink , **S5H5)
net.addLink(S4, H6, cls=TCLink , **S4H6)
```

```
mininet> pingall
*** Ping: testing ping reachability
H1 -> X H3 H4 H5 X
H2 -> H1 H3 H4 H5 X
H3 -> X H2 H4 H5 H6
H4 -> H1 H2 H3 H5 X
H5 -> H1 H2 H3 H4 X
H6 -> X X H3 X X
*** Results: 33% dropped (20/30 received)
mininet>
```

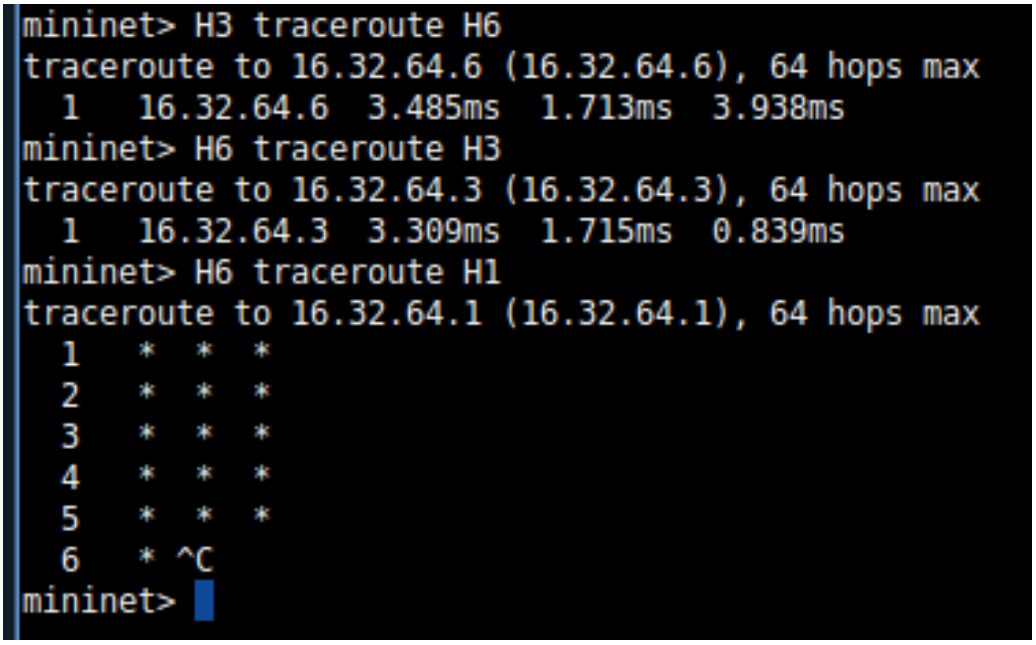
Capturando o tráfego de dados no sexto *host* (H6)

Os comandos utilizados foram “xterm H6”, no terminal do H6 é usado o comando “tcpdump -i H6-eth0 icmp” para monitorar e mostrar o tráfego de dados ICMP recebido e enviado. No CLI do Mininet é realizado “pings” dos *hosts* H1, H2, H3 para o H6 e por último é feito um “ping” do H6 para o H3.



Usando o comando “traceroute”

Na figura abaixo é mostrado a utilização do comando “traceroute” nos *hosts* H1, H3 e H6 da rede Mininet.



Conforme apresentado na figura do comando “pingall”, na imagem da captura do tráfego de dados no sexto *host* e na ilustração do comando “traceroute”; é possível observar que o sexto *host* (H6) se comunica apenas com o terceiro *host* (H3).

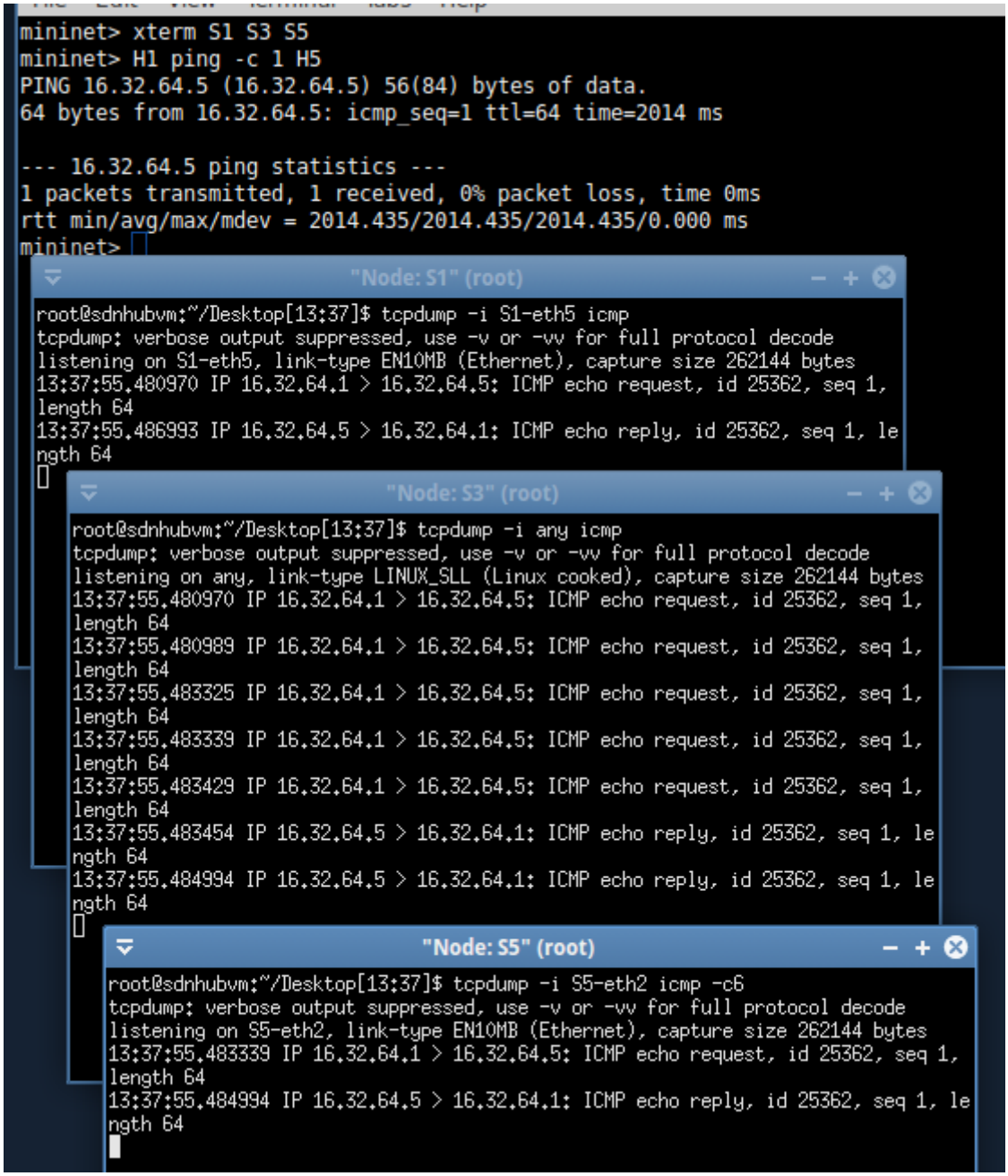
No caso do H3, ele pode se comunicar com todos os demais *hosts* da rede, sendo a restrição de comunicação aplicada somente ao H6.

Usando o comando “tcpdump” em Switches

Para mostrar que todo o tráfego de rede com destino ao quinto *host* (H5) passa pelos *switches* (S1, S3 e S5) utilizei o comando “tcpdump” em cada *switch.*

No S1 foi especificada a porta que o liga ao S3 (S1-eth5:S3-eth1), já no S5 a porta que o interliga o S3 (S5-eth2:S3-eth2).

Para que possa ser monitorado o *Ping Pong* do ICMP em ambas interfaces de interconexão do S3 (S3-eth1:S1-eth5 S3-eth2:S5-eth2) usei o “any” no argumento das interfaces do comando “tcpdump -i”.



Saída (*dump*) da execução do script do controlador POX

```
> sudo pkill -9 python; clear; cd ~/pox; python pox.py --verbose projeto_controlador
openflow.discovery host_tracker py log --no-default --file=/tmp/mylog.log

INFO:host_tracker:host_tracker ready
WARNING:core:Warning: Registered 'host_tracker' multipled times
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Connection UP from Switch dpid: 1
Switch Eth: S1-eth1 | Mac Port: 92:de:1c:fe:c8:1f
Switch Eth: S1-eth6 | Mac Port: c2:cc:f2:8f:b1:c7
Switch Eth: S1-eth4 | Mac Port: fe:80:69:4a:78:b3
Switch Eth: S1-eth5 | Mac Port: aa:55:a5:19:ab:1a
Switch Eth: S1-eth2 | Mac Port: 3e:10:de:82:94:85
Switch Eth: S1-eth3 | Mac Port: da:a3:eb:76:d3:c0
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Connection UP from Switch dpid: 2
Switch Eth: S2-eth1 | Mac Port: da:f2:c4:47:dd:e9
Switch Eth: S2-eth2 | Mac Port: 92:60:4c:53:1e:9a
Switch Eth: S2-eth3 | Mac Port: 5e:16:b8:91:ec:83
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Connection UP from Switch dpid: 4
Switch Eth: S4-eth1 | Mac Port: ba:49:e0:8b:72:32
Switch Eth: S4-eth2 | Mac Port: ce:eb:75:9b:0c:8e
Switch Eth: S4-eth3 | Mac Port: f6:db:4b:5b:56:f2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Connection UP from Switch dpid: 3
Switch Eth: S3-eth1 | Mac Port: 16:1b:82:1c:ce:1a
Switch Eth: S3-eth2 | Mac Port: d6:a0:e1:e2:35:38
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Connection UP from Switch dpid: 5
Switch Eth: S5-eth1 | Mac Port: 1e:ac:f3:99:7a:9c
Switch Eth: S5-eth4 | Mac Port: 0a:47:ed:54:74:f0
Switch Eth: S5-eth2 | Mac Port: c2:24:a7:5d:7b:1f
Switch Eth: S5-eth3 | Mac Port: 9a:32:7a:8d:dd:18
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
S1 dpid: 1 | S2 dpid: 2 | S3 dpid: 3 | S4 dpid: 4 | S5 dpid: 5
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: a0:00:00:00:00:01 | Switch dpid: 1 | Port: 1
```

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: ba:49:e0:8b:72:32 | Switch dpid: 1 | Port: 6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: d0:00:00:00:00:06 | Switch dpid: 4 | Port: 3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: 1e:ac:f3:99:7a:9c | Switch dpid: 2 | Port: 2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: c0:00:00:00:00:05 | Switch dpid: 5 | Port: 4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: 9a:32:7a:8d:dd:18 | Switch dpid: 4 | Port: 2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: c2:24:a7:5d:7b:1f | Switch dpid: 3 | Port: 2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: 92:60:4c:53:1e:9a | Switch dpid: 5 | Port: 1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: c2:cc:f2:8f:b1:c7 | Switch dpid: 4 | Port: 1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: a0:00:00:00:00:03 | Switch dpid: 1 | Port: 3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: d6:a0:e1:e2:35:38 | Switch dpid: 5 | Port: 2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: b0:00:00:00:00:04 | Switch dpid: 2 | Port: 3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Host: ce:eb:75:9b:0c:8e | Switch dpid: 5 | Port: 3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Link Switches Added
S1-eth4:S2-eth1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Link Switches Added
S1-eth5:S3-eth1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Link Switches Added
S1-eth6:S4-eth1
```

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S2-eth1:S1-eth4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S2-eth2:S5-eth1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S3-eth1:S1-eth5
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S3-eth2:S5-eth2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S5-eth1:S2-eth2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S5-eth2:S3-eth2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S5-eth3:S4-eth2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S4-eth1:S1-eth6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Link Switches Added
S4-eth2:S5-eth3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:01 | DST: ff:ff:ff:ff:ff:ff | MULTICAST DST:
True
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Set Flow S1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [65532] | DL_TYPE: 2054 | PRIORITY: 1 |
IDLE_TIMEOUT: 10 | HARD_TIMEOUT: 10 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
```

```
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [1] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.1
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [2] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [3] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [4, 5] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [5] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.5
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 1 | IN_PORT: None | OUT_PORT: [5, 6] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: 16.32.64.3 | NW_DST: 16.32.64.6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:01 | DST: ff:ff:ff:ff:ff:ff | MULTICAST DST:
True
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
Set Flow S2
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 2 | IN_PORT: None | OUT_PORT: [1, 3] | DL_TYPE: 2054 | PRIORITY: 1 |
IDLE_TIMEOUT: 10 | HARD_TIMEOUT: 10 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 2 | IN_PORT: None | OUT_PORT: [3] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 2 | IN_PORT: 3 | OUT_PORT: [1] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
SWITCH DPID: 2 | IN_PORT: 2 | OUT_PORT: [3] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:02 | DST: a0:00:00:00:00:01 | MULTICAST DST:
False
```

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Set Flow S3
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 3 | IN_PORT: None | OUT_PORT: [65532] | DL_TYPE: 2054 | PRIORITY: 1 |
IDLE_TIMEOUT: 10 | HARD_TIMEOUT: 10 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 3 | IN_PORT: 1 | OUT_PORT: [2] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 3 | IN_PORT: 2 | OUT_PORT: [1] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:03 | DST: a0:00:00:00:00:02 | MULTICAST DST:
False
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Set Flow S4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 4 | IN_PORT: None | OUT_PORT: [1, 3] | DL_TYPE: 2054 | PRIORITY: 1 |
IDLE_TIMEOUT: 10 | HARD_TIMEOUT: 10 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 4 | IN_PORT: None | OUT_PORT: [3] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: 16.32.64.3 | NW_DST: 16.32.64.6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 4 | IN_PORT: 3 | OUT_PORT: [1] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
PACKET TYPE: icmp | SRC: a0:00:00:00:00:03 | DST: b0:00:00:00:00:04 | MULTICAST DST:
False
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:02 | DST: a0:00:00:00:00:01 | MULTICAST DST:
False
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
Set Flow S5
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 5 | IN_PORT: None | OUT_PORT: [2, 4] | DL_TYPE: 2054 | PRIORITY: 1 |
IDLE_TIMEOUT: 10 | HARD_TIMEOUT: 10 | NW_SRC: None | NW_DST: None
```

```
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 5 | IN_PORT: 2 | OUT_PORT: [4] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.5
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 5 | IN_PORT: 4 | OUT_PORT: [2] | DL_TYPE: None | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: None
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 5 | IN_PORT: 2 | OUT_PORT: [1] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.4
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
SWITCH DPID: 5 | IN_PORT: 2 | OUT_PORT: [3] | DL_TYPE: 2048 | PRIORITY: 10 |
IDLE_TIMEOUT: 0 | HARD_TIMEOUT: 0 | NW_SRC: None | NW_DST: 16.32.64.6
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
PACKET TYPE: arp | SRC: a0:00:00:00:00:02 | DST: a0:00:00:00:00:01 | MULTICAST DST:
False
--------------------------------------------------------------------------------
--------------------------------------------------------------------------
```

Referências:

http://mininet.org/walkthrough/

https://noxrepo.github.io/pox-doc/html/

http://csie.nqu.edu.tw/smallko/sdn/mySDN.pdf

https://www.grotto-networking.com/SDNfun.html

http://xuyansen.work/discovery-topology-in-mininet/

http://csie.nqu.edu.tw/smallko/sdn/measure_traffic.pdf

https://osrg.github.io/ryu-book/en/html/spanning_tree.html

https://www.sciencedirect.com/science/article/pii/S0140366415003527

http://www.baburd.com.np/material/NG/Paper2-SDN-POX-Controller.pdf

https://www.comp.nus.edu.sg/~tbma/teaching/cs4226y15_past/tut-pox-pdf.pdf

https://programtalk.com/python-examples/pox.openflow.libopenflow_01.ofp_flow_mod/

https://mailman.stanford.edu/pipermail/mininet-discuss/2014-September/005044.html

https://stackoverflow.com/questions/31075492/setting-icmp-match-with-pox-controller

https://homepages.dcc.ufmg.br/~mmvieira/cc/OpenFlow%20Tutorial%20-%20OpenFlow%20Wiki.htm

https://python.hotexamples.com/pt/examples/pox.openflow.libopenflow_01/-/ofp_flow_mod/python-ofp_flow_mod-function-examples.html