Mawlana Bhashani Science and Technology University

Lab-Report

Report No: 06

Course code: ICT-4202

Course title: Wireless and Mobile Communication Lab

Date of Performance: 25.09.2020

Date of Submission: 30.09.2020

Submitted by

Name: Md. Afzalur Rahman Tanzin

ID:IT-16026

4th year 2nd semester

Session: 2015-2016

Dept. of ICT

MBSTU.

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

Experiment No: 06

Experiment Name: Switching an interface to move a host around a network using mininet.

Objective:

- Downloading and running mininet on virtual machine(VM)
- Switching an interface to move a host around the network
- Using mobility.py

Code:

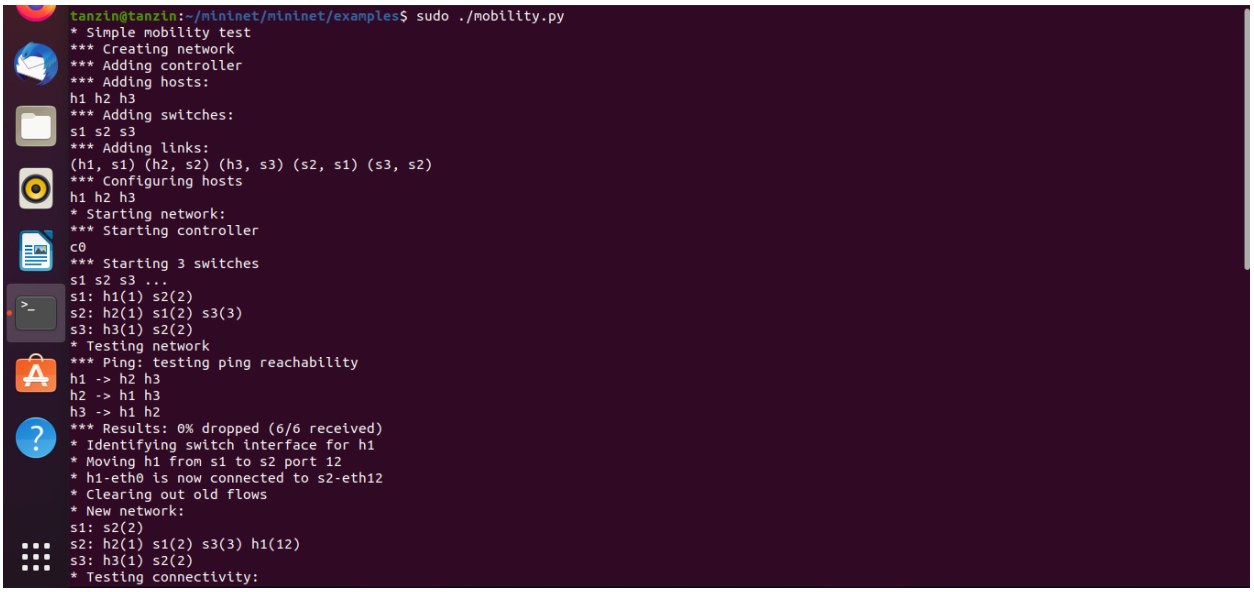
```
from mininet.node import OVSSwitch
from mininet.net import Mininet
from mininet.topo import LinearTopo
from mininet.log import info, output, warn, setLogLevel


from random import randint



class MobilitySwitch( OVSSwitch ):
    "Switch that can reattach and rename interfaces"


    def delIntf( self, intf ):
        "Remove (and detach) an interface"
        port = self.ports[ intf ]
        del self.ports[ intf ]
        del self.intfs[ port ]
        del self.nameToIntf[ intf.name ]


    def addIntf( self, intf, rename=False, **kwargs ):
        "Add (and reparent) an interface"
        OVSSwitch.addIntf( self, intf, **kwargs )
        intf.node = self
        if rename:
            self.renameIntf( intf )
```

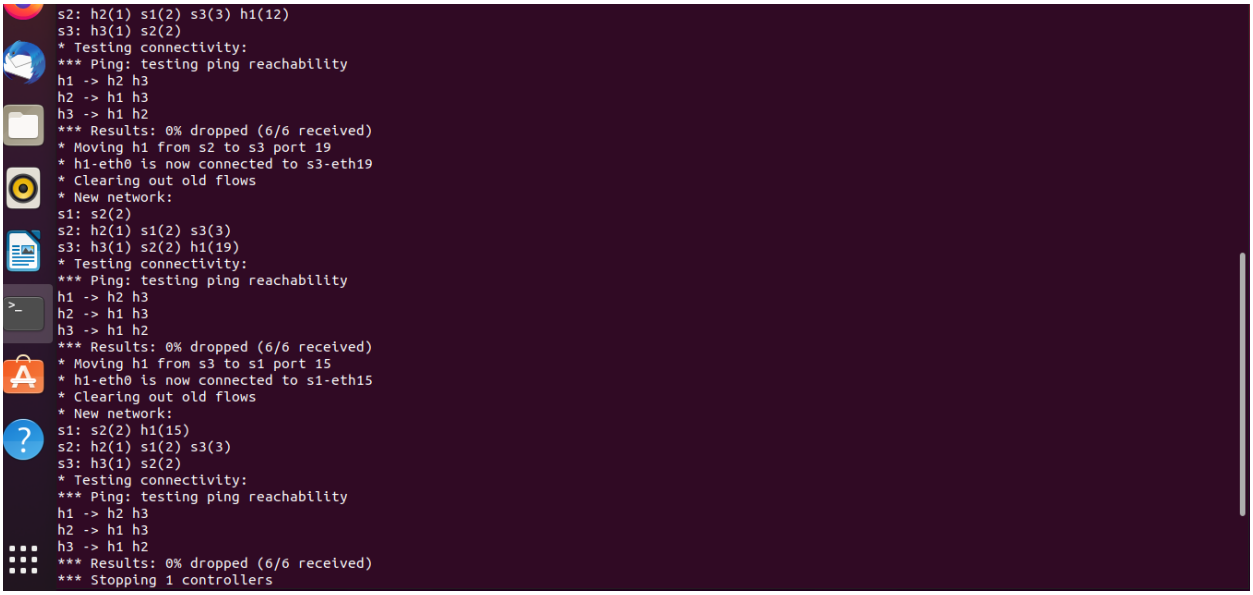
```
def attach( self, intf ):
    "Attach an interface and set its port"
    port = self.ports[ intf ]
    if port:
        if self.isOldOVS():
            self.cmd( 'ovs-vsctl add-port', self, intf )
        else:
            self.cmd( 'ovs-vsctl add-port', self, intf,
                      '-- set Interface', intf,
                      'ofport_request=%s' % port )
        self.validatePort( intf )


def validatePort( self, intf ):
    "Validate intf's OF port number"
    ofport = int( self.cmd( 'ovs-vsctl get Interface', intf,
                            'ofport' ) )
    if ofport != self.ports[ intf ]:
        warn( 'WARNING: ofport for', intf, 'is actually', ofport,
              '\n' )


def renameIntf( self, intf, newname='' ):
    "Rename an interface (to its canonical name)"
    intf.ifconfig( 'down' )
    if not newname:
        newname = '%s-eth%d' % ( self.name, self.ports[ intf ] )
    intf.cmd( 'ip link set', intf, 'name', newname )
    del self.nameToIntf[ intf.name ]
    intf.name = newname
    self.nameToIntf[ intf.name ] = intf
    intf.ifconfig( 'up' )


def moveIntf( self, intf, switch, port=None, rename=True ):
    "Move one of our interfaces to another switch"
    self.detach( intf )
    self.delIntf( intf )
    switch.addIntf( intf, port=port, rename=rename )
    switch.attach( intf )
```

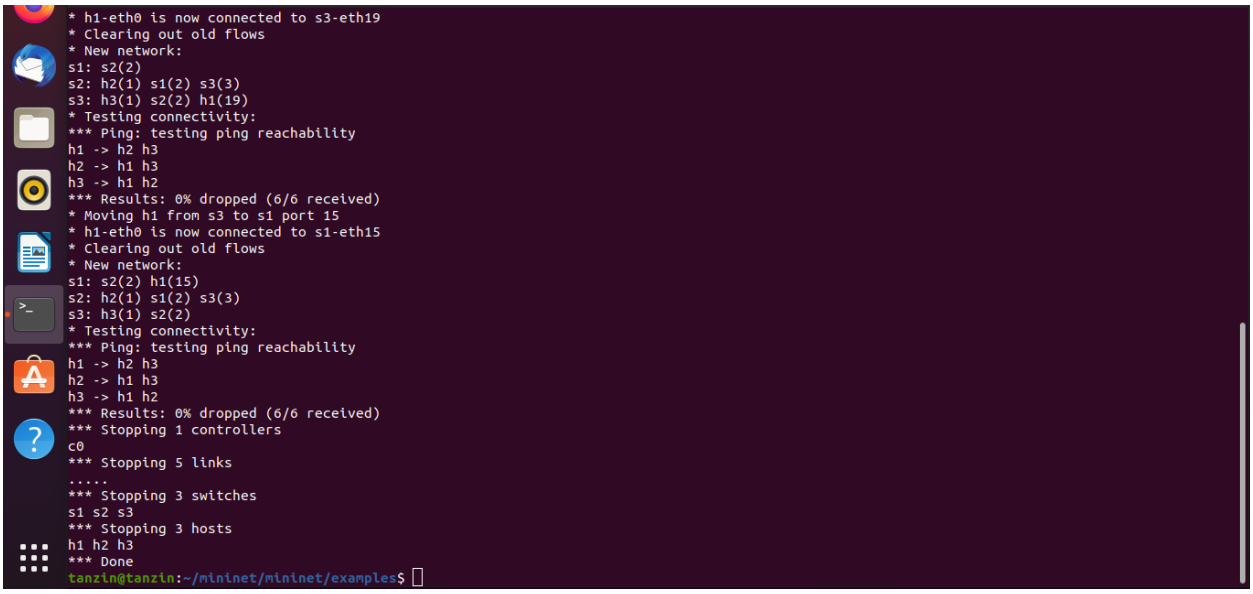
```
def printConnections( switches ):
    "Compactly print connected nodes to each switch"
    for sw in switches:
        output( '%s: ' % sw )
        for intf in sw.intfList():
            link = intf.link
            if link:
                intf1, intf2 = link.intf1, link.intf2
                remote = intf1 if intf1.node != sw else intf2
                output( '%s(%s) ' % ( remote.node, sw.ports[ intf ] ) )
        output( '\n' )




def moveHost( host, oldSwitch, newSwitch, newPort=None ):
    "Move a host from old switch to new switch"
    hintf, sintf = host.connectionsTo( oldSwitch )[ 0 ]
    oldSwitch.moveIntf( sintf, newSwitch, port=newPort )
    return hintf, sintf




def mobilityTest():
    "A simple test of mobility"
    info( '* Simple mobility test\n' )
    net = Mininet( topo=LinearTopo( 3 ), switch=MobilitySwitch )
    info( '* Starting network:\n' )
    net.start()
    printConnections( net.switches )
    info( '* Testing network\n' )
    net.pingAll()
    info( '* Identifying switch interface for h1\n' )
    h1, old = net.get( 'h1', 's1' )
    for s in 2, 3, 1:
        new = net[ 's%d' % s ]
        port = randint( 10, 20 )
        info( '* Moving', h1, 'from', old, 'to', new, 'port', port, '\n' )
        hintf, sintf = moveHost( h1, old, new, newPort=port )
        info( '*', hintf, 'is now connected to', sintf, '\n' )
```

```
        info( '* Clearing out old flows\n' )
        for sw in net.switches:
            sw.dpctl( 'del-flows' )
        info( '* New network:\n' )
        printConnections( net.switches )
        info( '* Testing connectivity:\n' )
        net.pingAll()
        old = new
    net.stop()


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

Output:







Conclusion:

In this experiment we switches an interface to move a host around a network using mininet. Mininet is a very convenient tool for this task. It clearly shows the wifi mobility test using mobility.py.