**Problem statements focusing on various aspects of Mininet and Mininet-WiFi**

**1. Custom Topology Creation in Mininet**

**- Design and implement a custom network topology in Mininet that includes at least three switches and six hosts. Ensure that the topology allows for redundant paths between switches and demonstrate how to test the connectivity between all hosts.**

import sys

from mininet.topo import Topo

from mininet.net import Mininet

from mininet.node import OVSSwitch, Controller

from mininet.cli import CLI

from mininet.log import setLogLevel, info

class CustomTopo(Topo):

def build(self):

# Add switches

s1 = self.addSwitch('s1', stp=True)

s2 = self.addSwitch('s2', stp=True)

s3 = self.addSwitch('s3', stp=True)

# Add hosts

h1 = self.addHost('h1', ip='10.0.0.1/24')

h2 = self.addHost('h2', ip='10.0.0.2/24')

h3 = self.addHost('h3', ip='10.0.0.3/24')

h4 = self.addHost('h4', ip='10.0.0.4/24')

h5 = self.addHost('h5', ip='10.0.0.5/24')

h6 = self.addHost('h6', ip='10.0.0.6/24')

# Add links between hosts and switches

self.addLink(h1, s1)

self.addLink(h2, s1)

self.addLink(h3, s2)

self.addLink(h4, s2)

self.addLink(h5, s3)

self.addLink(h6, s3)

# Add redundant links between switches

self.addLink(s1, s2)

self.addLink(s2, s3)

self.addLink(s3, s1)

def run():

# Set up the topology and network

topo = CustomTopo()

net = Mininet(topo=topo, switch=OVSSwitch, controller=Controller)

net.start()

# Add a simple OpenFlow rule to each switch to forward packets

for switch in net.switches:

switch.cmd('ovs-ofctl add-flow {} "priority=1,actions=flood"'.format(switch.name))

# Test connectivity

info("Testing network connectivity\n")

net.pingAll()

# Start CLI

CLI(net)

# Stop the network

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

run()

**2. Custom Topology Creation in Mininet-WiFi**

**- Create a custom wireless network topology in Mininet-WiFi with multiple access points and mobile stations. Ensure the topology reflects a realistic deployment scenario, and validate the q**

#!/usr/bin/python

import sys

from mininet.log import setLogLevel, info

from mn\_wifi.cli import CLI

from mn\_wifi.net import Mininet\_wifi

from mininet.node import Controller

def topology():

"Create a network."

net = Mininet\_wifi(controller=Controller)

info("\*\*\* Creating nodes\n")

ap1 = net.addAccessPoint('ap1', ssid='ssid-ap1', mode='g', channel='1', position='10,30,0')

ap2 = net.addAccessPoint('ap2', ssid='ssid-ap2', mode='g', channel='6', position='50,30,0')

ap3 = net.addAccessPoint('ap3', ssid='ssid-ap3', mode='g', channel='11', position='90,30,0')

sta1 = net.addStation('sta1', ip='10.0.0.1', position='15,30,0')

sta2 = net.addStation('sta2', ip='10.0.0.2', position='20,40,0')

sta3 = net.addStation('sta3', ip='10.0.0.3', position='25,50,0')

sta4 = net.addStation('sta4', ip='10.0.0.4', position='30,60,0')

sta5 = net.addStation('sta5', ip='10.0.0.5', position='35,70,0')

sta6 = net.addStation('sta6', ip='10.0.0.6', position='40,80,0')

c0 = net.addController('c0')

info("\*\*\* Configuring wifi nodes\n")

net.configureWifiNodes()

info("\*\*\* Creating links\n")

net.addLink(ap1, ap2)

net.addLink(ap2, ap3)

info("\*\*\* Starting network\n")

net.build()

c0.start()

ap1.start([c0])

ap2.start([c0])

ap3.start([c0])

info("\*\*\* Running CLI\n")

CLI(net)

info("\*\*\* Stopping network\n")

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

topology()

**3. Running a Simple Web Server and Client:**

**- Set up a simple network in Mininet with one host acting as a web server and another as a client. Configure the web server to host a basic HTML page and write a script to retrieve this page using the client, demonstrating successful communication.**

#!/usr/bin/python

import os

from mininet.net import Mininet

from mininet.node import Controller

from mininet.cli import CLI

from mininet.log import setLogLevel, info

from mininet.link import TCLink

def setup\_network():

"Create a network."

net = Mininet(controller=Controller, link=TCLink)

info("\* Adding controller\n")

net.addController('c0')

info("\* Adding hosts\n")

h1 = net.addHost('h1', ip='10.0.0.1')

h2 = net.addHost('h2', ip='10.0.0.2')

info("\* Adding switch\n")

s1 = net.addSwitch('s1')

info("\* Creating links\n")

net.addLink(h1, s1)

net.addLink(h2, s1)

info("\* Starting network\n")

net.start()

info("\* Configuring web server on h1\n")

h1.cmd('mkdir -p /tmp/www')

os.system('cp index.html /tmp/www')

h1.cmd('cd /tmp/www && python3 -m http.server 80 &')

info("\* Running CLI\n")

CLI(net)

info("\* Stopping network\n")

net.stop()

if \_name\_ == '\_main\_':

setLogLevel('info')

setup\_network()

index.html:

<!DOCTYPE html>

<html>

<head>

<title>Mininet Web Server</title>

</head>

<body>

<h1>Hello from Mininet!</h1>

</body>

</html>

Mininet> h2 wget<http://10.0.0.1>

Mininet> h2 cat index.html

**4. 802.11 Wireless LAN Emulation**

**- Implement an 802.11 wireless LAN using Mininet-WiFi with multiple access points and mobile stations. Emulate a realistic environment where stations move between different access points and demonstrate how handoff occurs.**

#!/usr/bin/env python

'Example for Handover'

import sys

from mininet.log import setLogLevel, info

from mn\_wifi.cli import CLI

from mn\_wifi.net import Mininet\_wifi

def topology(args):

"Create a network."

net = Mininet\_wifi()

info("\* Creating nodes\n")

sta1\_args, sta2\_args = {}, {}

if '-s' in args:

sta1\_args['position'], sta2\_args['position'] = '20,30,0', '60,30,0'

sta1 = net.addStation('sta1', mac='00:00:00:00:00:01', \*\*sta1\_args)

sta2 = net.addStation('sta2', mac='00:00:00:00:00:02', \*\*sta2\_args)

ap1 = net.addAccessPoint('ap1', ssid='ssid-ap1', channel='1', position='15,30,0')

ap2 = net.addAccessPoint('ap2', ssid='ssid-ap2', channel='6', position='55,30,0')

c1 = net.addController('c1')

net.setPropagationModel(model="logDistance", exp=5)

info("\* Configuring nodes\n")

net.configureNodes()

info("\* Creating links\n")

net.addLink(ap1, ap2)

if '-p' not in args:

net.plotGraph(max\_x=100, max\_y=100)

if '-s' not in args:

net.startMobility(time=0)

net.mobility(sta1, 'start', time=1, position='10,30,0')

net.mobility(sta2, 'start', time=2, position='10,40,0')

net.mobility(sta1, 'stop', time=40, position='60,30,0')

net.mobility(sta2, 'stop', time=40, position='25,40,0')

net.stopMobility(time=41)

info("\* Starting network\n")

net.build()

c1.start()

ap1.start([c1])

ap2.start([c1])

info("\* Running CLI\n")

CLI(net)

info("\* Stopping network\n")

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

topology(sys.argv)

sudo python question4.py

**5. Capturing Wireless Control Traffic in Mininet-WiFi**

**- Create a Mininet-WiFi topology with two access points and three mobile stations. Set up a packet capture mechanism to monitor and record wireless control traffic such as beacons, probe requests, and responses. Analyse the captured traffic to understand the wireless communication process.**

import sys

from mininet.node import Controller, OVSKernelSwitch

from mininet.log import setLogLevel, info

from mn\_wifi.net import Mininet\_wifi

from mn\_wifi.cli import CLI

from mn\_wifi.link import wmediumd

from mn\_wifi.wmediumdConnector import interference

def topology():

"Create a network."

net = Mininet\_wifi()

info("\*\*\* Creating nodes\n")

ap1 = net.addAccessPoint('ap1', ssid='ssid-ap1', position='10,30,0')

ap2 = net.addAccessPoint('ap2', ssid='ssid-ap2', position='50,30,0')

sta1 = net.addStation('sta1', ip='10.0.0.1/24', position='15,30,0')

sta2 = net.addStation('sta2', ip='10.0.0.2/24', position='20,30,0')

sta3 = net.addStation('sta3', ip='10.0.0.3/24', position='25,30,0')

c1 = net.addController('c1')

info("\*\*\* Configuring wifi nodes\n")

net.configureWifiNodes()

info("\*\*\* Associating and creating links\n")

net.addLink(ap1, ap2)

net.addLink(sta1,ap1)

net.addLink(sta2,ap2)

net.addLink(sta3,ap1)

net.plotGraph(max\_x=100, max\_y=100)

net.startMobility(time=0)

net.mobility(sta1, 'start', time=1, position='10,30,0')

net.mobility(sta2, 'start', time=2, position='10,30,0')

net.mobility(sta3, 'start', time=3, position='10,30,0')

net.mobility(sta1, 'stop', time=10, position='50,30,0')

net.mobility(sta2, 'stop', time=11, position='55,30,0')

net.mobility(sta3, 'stop', time=12, position='60,30,0')

net.stopMobility(time=13)

info("\*\*\* Starting network\n")

net.build()

c1.start()

ap1.start([c1])

ap2.start([c1])

# Start packet capture on wireless interfaces

info("\*\*\* Starting packet capture\n")

ap1.cmd('tcpdump -i ap1-wlan1 -w /tmp/ap1-wlan1.pcap &')

ap2.cmd('tcpdump -i ap2-wlan1 -w /tmp/ap2-wlan1.pcap &')

info("\*\*\* Running CLI\n")

CLI(net)

info("\*\*\* Stopping network\n")

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

topology()

**6. Handling Multiple Access Points**

**- Design a network topology in Mininet-WiFi with three access points and six mobile stations. Ensure that the stations can roam between access points without losing connectivity. Implement mechanisms to monitor and demonstrate seamless handover between access points.**

#!/usr/bin/python

from mininet.net import Mininet

from mininet.node import Controller

from mininet.log import setLogLevel, info

from mn\_wifi.node import OVSKernelAP

from mn\_wifi.cli import CLI

from mn\_wifi.net import Mininet\_wifi

from mn\_wifi.wmediumdConnector import interference

from mn\_wifi.link import wmediumd, mesh

def topology():

net = Mininet\_wifi(controller=Controller, link=wmediumd, wmediumd\_mode=interference)

info("\*\*\* Creating nodes\n")

ap1 = net.addAccessPoint('ap1', ssid='ssid1', mode='g', channel='1', position='10,30,0', range=30)

ap2 = net.addAccessPoint('ap2', ssid='ssid2', mode='g', channel='6', position='50,30,0', range=50)

ap3 = net.addAccessPoint('ap3', ssid='ssid3', mode='g', channel='11', position='90,30,0', range=30)

sta1 = net.addStation('sta1', position='10,20,0')

sta2 = net.addStation('sta2', position='20,20,0')

sta3 = net.addStation('sta3', position='30,20,0')

sta4 = net.addStation('sta4', position='40,20,0')

sta5 = net.addStation('sta5', position='50,20,0')

sta6 = net.addStation('sta6', position='60,20,0')

c0 = net.addController('c0')

info("\*\*\* Configuring WiFi nodes\n")

net.configureWifiNodes()

info("\*\*\* Associating stations with access points\n")

net.addLink(sta1, ap1)

net.addLink(sta2, ap1)

net.addLink(sta3, ap2)

net.addLink(sta4, ap2)

net.addLink(sta5, ap3)

net.addLink(sta6, ap3)

net.plotGraph(max\_x=100, max\_y=100)

info("\*\*\* Setting Mobility\n")

net.startMobility(time=0)

net.mobility(sta1, 'start', time=1, position='10,20,0')

net.mobility(sta1, 'stop', time=20, position='50,30,0')

net.mobility(sta2, 'start', time=1, position='20,20,0')

net.mobility(sta2, 'stop', time=20, position='90,30,0')

net.stopMobility(time=30)

info("\*\*\* Starting network\n")

net.build()

c0.start()

ap1.start([c0])

ap2.start([c0])

ap3.start([c0])

info("\*\*\* Running CLI\n")

CLI(net)

info("\*\*\* Stopping network\n")

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

topology()

**7. Implementing a Simple Mobility Scenario**

**- Create a scenario in Mininet-WiFi where mobile stations move within the coverage area of multiple access points. Develop a script to simulate the movement of stations and handle handover mechanisms, demonstrating how stations maintain connectivity while moving.**

#!/usr/bin/env python

'Setting the position of nodes and providing mobility'

import sys

from mininet.log import setLogLevel, info

from mn\_wifi.cli import CLI

from mn\_wifi.net import Mininet\_wifi

def topology(args):

"Create a network."

net = Mininet\_wifi()

info("\* Creating nodes\n")

h1 = net.addHost('h1', mac='00:00:00:00:00:01', ip='10.0.0.1/8')

sta1 = net.addStation('sta1', mac='00:00:00:00:00:02', ip='10.0.0.2/8')

sta2 = net.addStation('sta2', mac='00:00:00:00:00:03', ip='10.0.0.3/8')

ap1 = net.addAccessPoint('ap1', ssid='new-ssid', mode='g', channel='1',

position='45,40,0')

c1 = net.addController('c1')

info("\* Configuring propagation model\n")

net.setPropagationModel(model="logDistance", exp=4.5)

info("\* Configuring nodes\n")

net.configureNodes()

info("\* Associating and Creating links\n")

net.addLink(ap1, h1)

if '-p' not in args:

net.plotGraph(max\_x=200, max\_y=200)

if '-c' in args:

sta1.coord = ['40.0,30.0,0.0', '31.0,10.0,0.0', '31.0,30.0,0.0']

sta2.coord = ['40.0,40.0,0.0', '55.0,31.0,0.0', '55.0,81.0,0.0']

net.startMobility(time=0, mob\_rep=1, reverse=False)

p1, p2, p3, p4 = {}, {}, {}, {}

if '-c' not in args:

p1 = {'position': '40.0,30.0,0.0'}

p2 = {'position': '40.0,40.0,0.0'}

p3 = {'position': '31.0,10.0,0.0'}

p4 = {'position': '55.0,31.0,0.0'}

net.mobility(sta1, 'start', time=1, \*\*p1)

net.mobility(sta2, 'start', time=2, \*\*p2)

net.mobility(sta1, 'stop', time=12, \*\*p3)

net.mobility(sta2, 'stop', time=22, \*\*p4)

net.stopMobility(time=23)

info("\* Starting network\n")

net.build()

c1.start()

ap1.start([c1])

info("\* Running CLI\n")

CLI(net)

info("\* Stopping network\n")

net.stop()

if \_name\_ == '\_main\_':

setLogLevel('info')

topology(sys.argv)

**8. Performance Evaluation of a Web Server in a Wireless Network**

**- Create a wireless network topology in Mininet-WiFi with one host acting as a web server and multiple mobile clients accessing the server. Evaluate the performance of the web server under different conditions, such as varying client mobility patterns and network load, and report the results.**

#!/usr/bin/python

from mininet.net import Mininet

from mininet.node import Controller

from mininet.log import setLogLevel, info

from mn\_wifi.node import OVSKernelAP

from mn\_wifi.cli import CLI

from mn\_wifi.net import Mininet\_wifi

from mn\_wifi.wmediumdConnector import interference

from mn\_wifi.link import wmediumd

def topology():

net = Mininet\_wifi(controller=Controller, link=wmediumd, wmediumd\_mode=interference)

info("\*\*\* Creating nodes\n")

ap1 = net.addAccessPoint('ap1', ssid='ssid1', mode='g', channel='1', position='50,50,0', range=50)

server = net.addHost('server', ip='10.0.0.1')

sta1 = net.addStation('sta1', ip='10.0.0.2', position='10,20,0')

sta2 = net.addStation('sta2', ip='10.0.0.3', position='20,20,0')

sta3 = net.addStation('sta3', ip='10.0.0.4', position='30,20,0')

sta4 = net.addStation('sta4', ip='10.0.0.5', position='40,20,0')

sta5 = net.addStation('sta5', ip='10.0.0.6', position='50,20,0')

sta6 = net.addStation('sta6', ip='10.0.0.7', position='60,20,0')

c0 = net.addController('c0')

info("\*\*\* Configuring WiFi nodes\n")

net.configureWifiNodes()

info("\*\*\* Creating links\n")

net.addLink(ap1, server)

net.addLink(ap1, sta1)

net.addLink(ap1, sta2)

net.addLink(ap1, sta3)

net.addLink(ap1, sta4)

net.addLink(ap1, sta5)

net.addLink(ap1, sta6)

net.plotGraph(max\_x=100, max\_y=100)

info("\*\*\* Setting Mobility Model\n")

net.startMobility(time=0)

net.mobility(sta1, 'start', time=1, position='10,20,0')

net.mobility(sta1, 'stop', time=20, position='70,20,0')

net.mobility(sta2, 'start', time=1, position='20,20,0')

net.mobility(sta2, 'stop', time=20, position='80,20,0')

net.mobility(sta3, 'start', time=1, position='30,20,0')

net.mobility(sta3, 'stop', time=20, position='90,20,0')

net.mobility(sta4, 'start', time=1, position='40,20,0')

net.mobility(sta4, 'stop', time=20, position='100,20,0')

net.mobility(sta5, 'start', time=1, position='50,20,0')

net.mobility(sta5, 'stop', time=20, position='110,20,0')

net.mobility(sta6, 'start', time=1, position='60,20,0')

net.mobility(sta6, 'stop', time=20, position='120,20,0')

net.stopMobility(time=30)

info("\*\*\* Setting up web server\n")

server.cmd('python3 -m http.server 80 &')

info("\*\*\* Starting network\n")

net.build()

c0.start()

ap1.start([c0])

info("\*\*\* Running CLI\n")

CLI(net)

info("\*\*\* Stopping network\n")

server.cmd('kill %python3')

net.stop()

if \_\_name\_\_ == '\_\_main\_\_':

setLogLevel('info')

topology()