

Table of Contents

Introduction:	3
TASK 1	5
Screenshot from the Mininet Wi-Fi GUI	5
Prior to Mobility:.....	5
At the completion of mobility:.....	5
Ping results:	6
STA1 <- - -> STA2.....	6
STA2 <- - -> STA3.....	6
STA1 <- - -> STA3.....	6
STA1 <- - -> STA4.....	7
A TCP flow of 1GB using the socket assigned:	7
The server and the client statistics	7
Wireshark, while the transfer is in progress	7
Wireshark, throughput when the transfer is complete	8
I/O, when the transfer is complete.....	8
TASK 2	9
ICMP stream	9
sta4ad <- - -> sta5ad.....	9
sta5ad <- - -> sta6ad.....	9
sta4ad <- - -> sta6ad.....	9
sta7M <- - -> sta8M	10
Sta8M <- - -> sta9M.....	10
sta7M <- - -> sta9M	10
Analysis	11
TASK 3	12
ONOS GUI.....	12
ICMP stream	12
Link configurations	13
Links status:	13
TASK 4	14
UDP flow to total of 2GB traffic using the port assigned	14
I/O Graph	14
ONOS GUI Server and Client	15
UDP Wireshark.....	15
Video stream	16

Wireshark while streaming.....	16
Packets route while streaming	17
Capture File Properties	17
Analysis	18
References:.....	18

Introduction:

Simulation of wireless networks is important at all stages of their life including the design, operational, and testing stages. Simulation is used to predict the performance of a wireless network's architecture, protocol, device, topology, etc. It is a cost-effective and flexible technique to performance evaluation of wireless systems. This chapter aims at reviewing the main aspects of wireless systems including wireless node object model, radio propagation, physical and media access control layers, and wireless network architectures. Then we review the simulation tools and packages that are optimized for this task. Case studies on simulation of wireless network systems are presented to demonstrate the main concepts.

We used these network simulation tools during this period :

Mininet:

Mininet is a network emulator which creates a network of virtual hosts, switches, controllers, and links. Mininet hosts run standard Linux network software, and its switches support OpenFlow for highly flexible custom routing and Software-Defined Networking.

Mininet supports research, development, learning, prototyping, testing, debugging, and any other tasks that could benefit from having a complete experimental network on a laptop or other PC.

ONOS:

ONOS stands for Open Network Operating System. ONOS provides the control plane for a software-defined network (SDN), managing network components, such as switches and links, and running software programs or modules to provide communication services to end hosts and neighboring networks.

The most important benefit of an operating system is that it provides a useful and usable platform for software programs designed for a particular application or use case. ONOS applications and use cases often consist of customized communication routing, management, or monitoring services for software-defined networks. Some examples of things which you can do with ONOS, and software written to run on ONOS, may be found in Apps and Use Cases.

Wireshark:

Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible.

You could think of a network packet analyzer as a measuring device for examining what is happening inside a network cable, just like an electrician uses a voltmeter for examining what's happening inside an electric cable (but at a higher level, of course). In the past, such tools were either very expensive, proprietary, or both. However, with the advent of Wireshark, that has changed. Wireshark is available for free, is open source, and is one of the best packet analyzers available today.

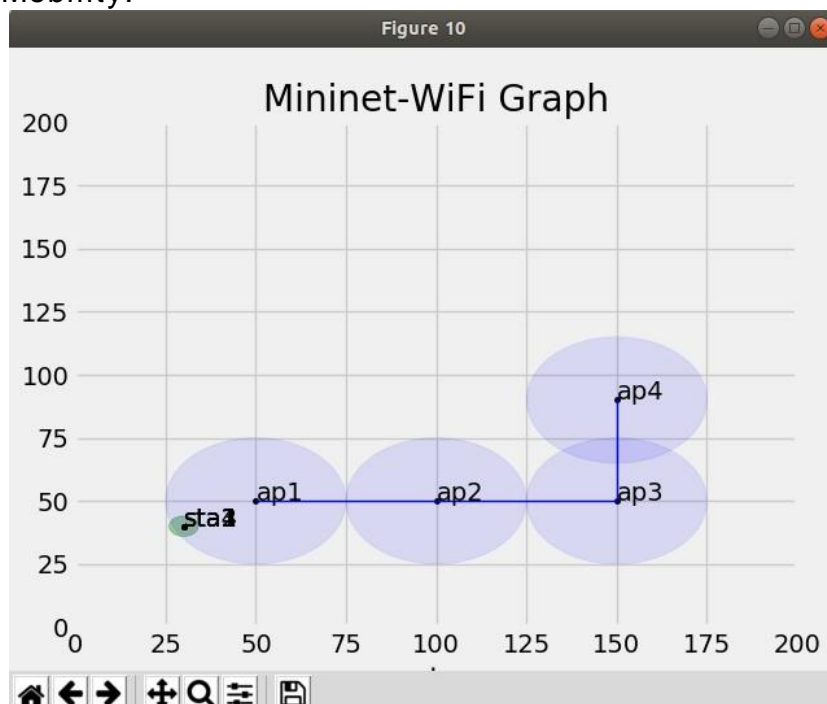
Iperf:

Iperf is a tool for network performance measurement and tuning. It is a cross-platform tool that can produce standardized performance measurements for any network. Iperf has client and server functionality and can create data streams to measure the throughput between the two ends in one or both directions. Typical Iperf output contains a time-stamped report of the amount of data transferred and the throughput measured.

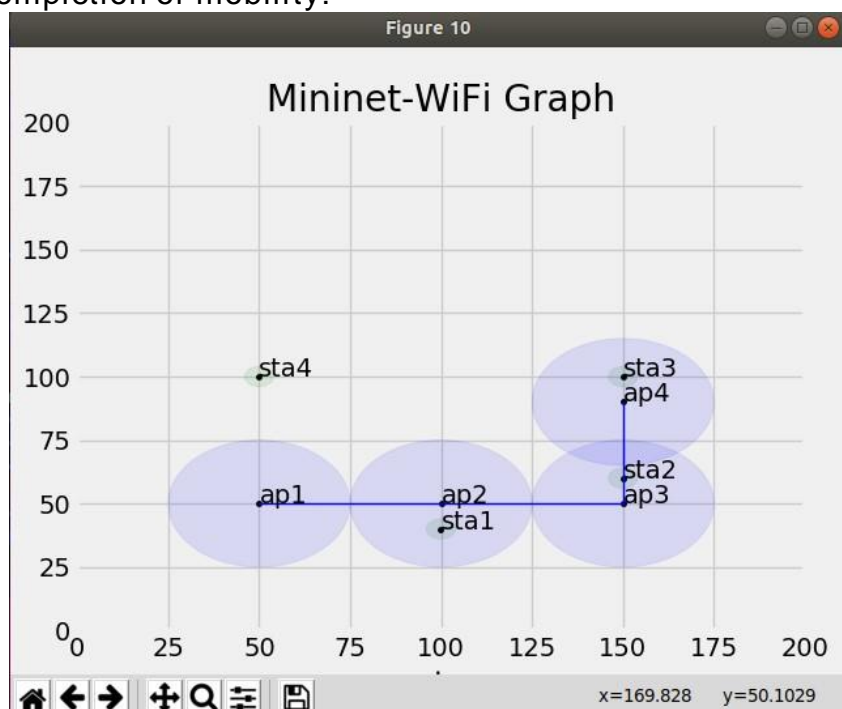
TASK 1

Screenshot from the Mininet Wi-Fi GUI

Prior to Mobility:



At the completion of mobility:



Ping results:

STA1 <- -> STA2

```
mininet-wifi> sta1 ping sta2
PING 10.0.0.3 (10.0.0.3) 56(84) bytes of data.
64 bytes from 10.0.0.3: icmp_seq=1 ttl=64 time=51.5 ms
64 bytes from 10.0.0.3: icmp_seq=2 ttl=64 time=0.856 ms
64 bytes from 10.0.0.3: icmp_seq=3 ttl=64 time=0.165 ms
64 bytes from 10.0.0.3: icmp_seq=4 ttl=64 time=0.261 ms
64 bytes from 10.0.0.3: icmp_seq=5 ttl=64 time=0.328 ms
64 bytes from 10.0.0.3: icmp_seq=6 ttl=64 time=0.200 ms
64 bytes from 10.0.0.3: icmp_seq=7 ttl=64 time=0.225 ms
64 bytes from 10.0.0.3: icmp_seq=8 ttl=64 time=0.251 ms
64 bytes from 10.0.0.3: icmp_seq=9 ttl=64 time=0.343 ms
64 bytes from 10.0.0.3: icmp_seq=10 ttl=64 time=0.177 ms
64 bytes from 10.0.0.3: icmp_seq=11 ttl=64 time=0.281 ms
64 bytes from 10.0.0.3: icmp_seq=12 ttl=64 time=0.294 ms
^C
--- 10.0.0.3 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11196ms
rtt min/avg/max/mdev = 0.165/4.579/51.572/14.170 ms
mininet-wifi>
```

STA2 <- -> STA3

```
rtt min/avg/max/mdev = 0.165/4.579/51.572/14.170 ms
mininet-wifi> sta2 ping sta3
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=57.0 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=0.638 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.195 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.239 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.144 ms
64 bytes from 10.0.0.4: icmp_seq=6 ttl=64 time=0.446 ms
64 bytes from 10.0.0.4: icmp_seq=7 ttl=64 time=0.152 ms
64 bytes from 10.0.0.4: icmp_seq=8 ttl=64 time=0.426 ms
64 bytes from 10.0.0.4: icmp_seq=9 ttl=64 time=0.259 ms
^C
--- 10.0.0.4 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8164ms
rtt min/avg/max/mdev = 0.144/6.615/57.038/17.827 ms
mininet-wifi>
```

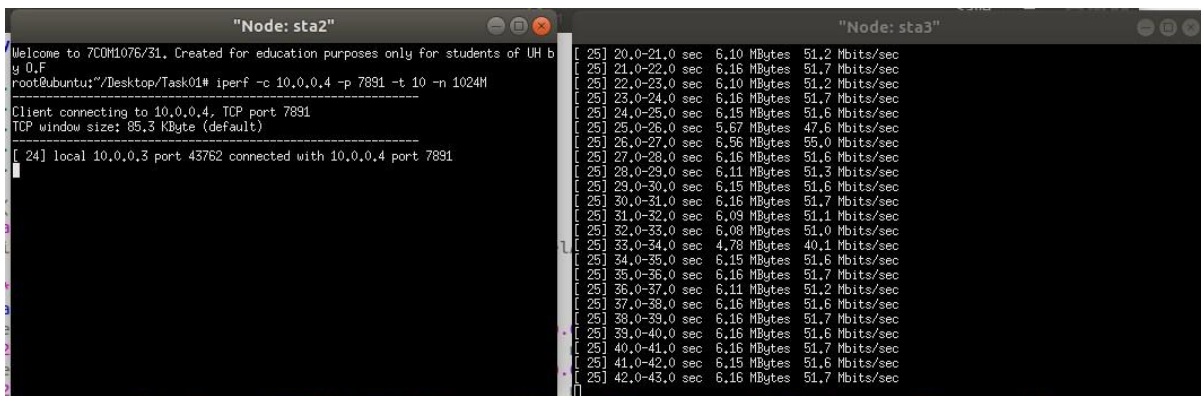
STA1 <- -> STA3

```
rtt min/avg/max/mdev = 0.144/6.615/57.038/17.827 ms
mininet-wifi> sta1 ping sta3
PING 10.0.0.4 (10.0.0.4) 56(84) bytes of data.
64 bytes from 10.0.0.4: icmp_seq=1 ttl=64 time=66.0 ms
64 bytes from 10.0.0.4: icmp_seq=2 ttl=64 time=1.11 ms
64 bytes from 10.0.0.4: icmp_seq=3 ttl=64 time=0.238 ms
64 bytes from 10.0.0.4: icmp_seq=4 ttl=64 time=0.452 ms
64 bytes from 10.0.0.4: icmp_seq=5 ttl=64 time=0.186 ms
64 bytes from 10.0.0.4: icmp_seq=6 ttl=64 time=0.393 ms
64 bytes from 10.0.0.4: icmp_seq=7 ttl=64 time=0.252 ms
64 bytes from 10.0.0.4: icmp_seq=8 ttl=64 time=0.157 ms
^C
--- 10.0.0.4 ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7106ms
rtt min/avg/max/mdev = 0.157/8.605/66.052/21.714 ms
```


STA1 <- -> STA2

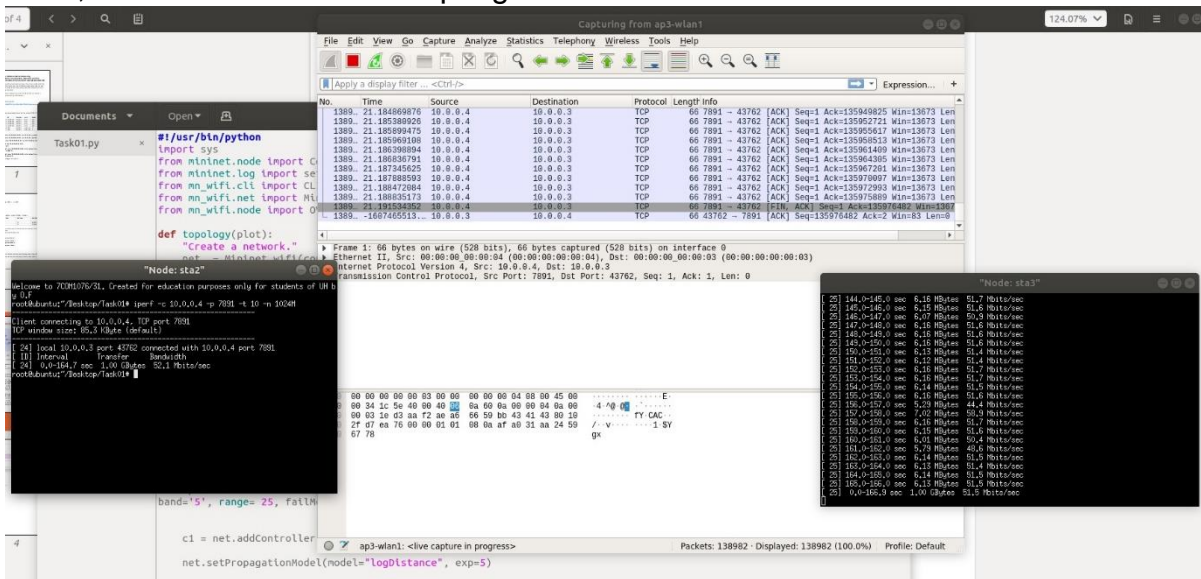
```
rtt min/avg/max/mdev = 0.157/8.605/66.052/21.714 ms
mininet-wifi> sta1 ping sta4
PING 10.0.0.5 (10.0.0.5) 56(84) bytes of data.
From 10.0.0.2 icmp_seq=1 Destination Host Unreachable
From 10.0.0.2 icmp_seq=2 Destination Host Unreachable
From 10.0.0.2 icmp_seq=3 Destination Host Unreachable
From 10.0.0.2 icmp_seq=4 Destination Host Unreachable
From 10.0.0.2 icmp_seq=5 Destination Host Unreachable
From 10.0.0.2 icmp_seq=6 Destination Host Unreachable
From 10.0.0.2 icmp_seq=7 Destination Host Unreachable
From 10.0.0.2 icmp_seq=8 Destination Host Unreachable
From 10.0.0.2 icmp_seq=9 Destination Host Unreachable
^C
--- 10.0.0.5 ping statistics ---
10 packets transmitted, 0 received, +9 errors, 100% packet loss, time 922ms
pipe 4
mininet-wifi>
```

A TCP flow of 1GB using the socket assigned:
The server and the client statistics



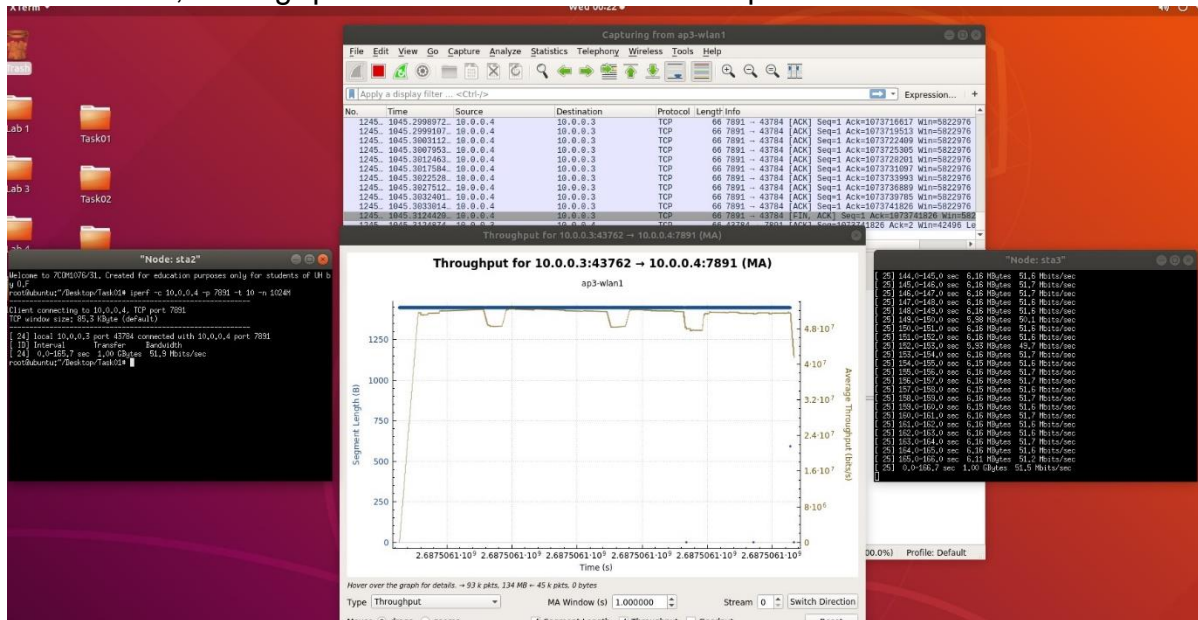
The image shows two terminal windows. The left window, titled "Node: sta2", displays the output of the 'iperf -c 10.0.0.4 -p 7891 -t 10 -n 1024M' command. It shows a client connecting to 10.0.0.4 port 7891 and then a list of statistics for 24 local connections. The right window, titled "Node: sta3", shows a list of statistics for 25 connections, each with a time range, size, and rate.

Wireshark, while the transfer is in progress

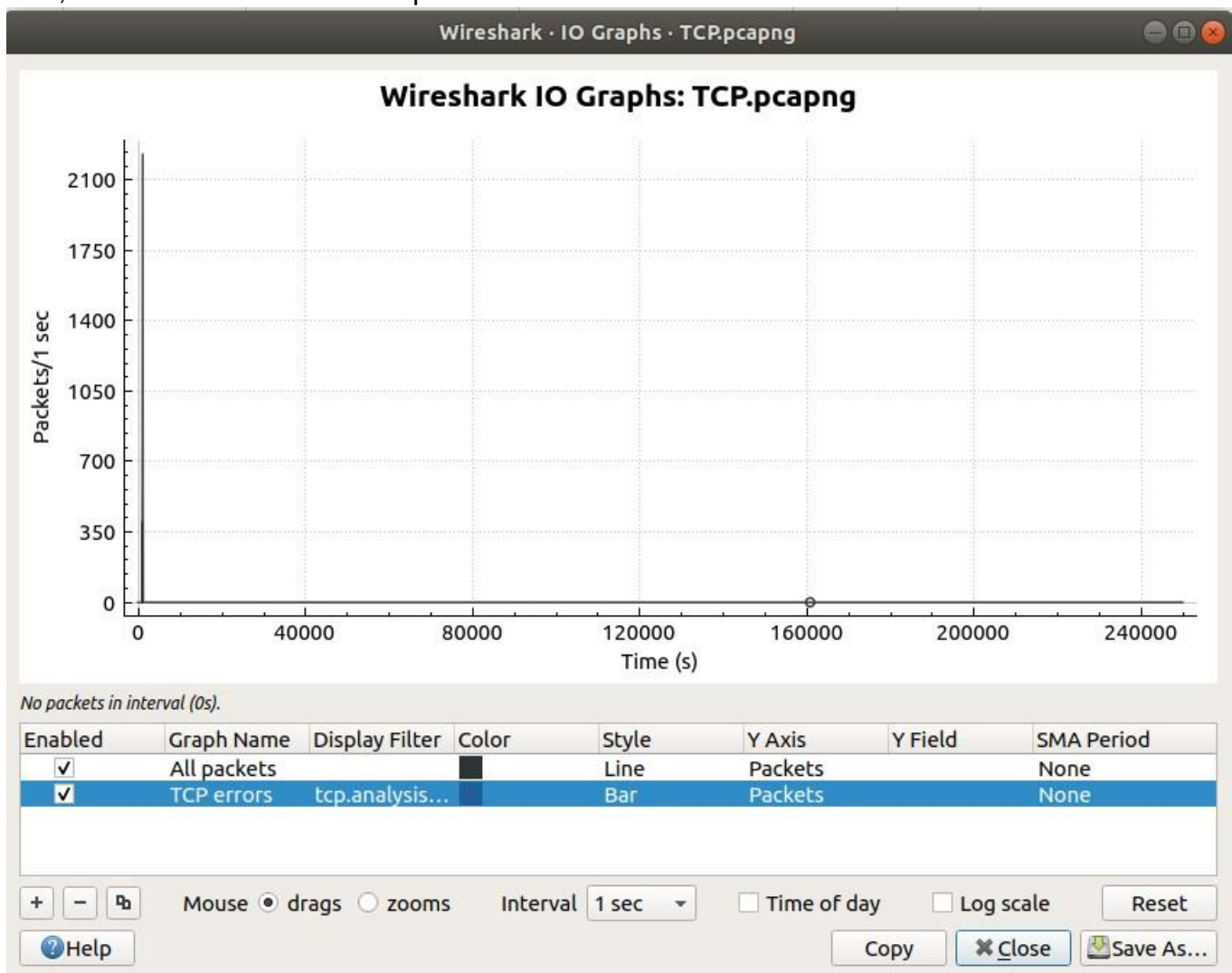


The image is a screenshot of the Wireshark network protocol analyzer. It shows a capture of traffic on the 'ap3-wlan1' interface. The packet list on the left shows a series of TCP segments. The packet details pane in the center shows the structure of a selected TCP segment, including the Ethernet II header, Internet Protocol Version 4 header, and Transmission Control Protocol header. The packet bytes pane at the bottom shows the raw data of the selected packet. The status bar at the bottom indicates that 138982 packets have been displayed.

Wireshark, throughput when the transfer is complete



I/O, when the transfer is complete



TASK 2

ICMP stream

sta4ad < - - -> sta5ad

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
64 bytes from 10.0.0.6: icmp_seq=11 ttl=64 time=0.078 ms
64 bytes from 10.0.0.6: icmp_seq=12 ttl=64 time=0.033 ms
^C
--- 10.0.0.6 ping statistics ---
12 packets transmitted, 12 received, 0% packet loss, time 11273ms
rtt min/avg/max/mdev = 0.033/0.060/0.123/0.026 ms
mininet-wifi> sta4ad ping sta5ad
PING 10.0.0.6 (10.0.0.6) 56(84) bytes of data.
64 bytes from 10.0.0.6: icmp_seq=1 ttl=64 time=23.2 ms
64 bytes from 10.0.0.6: icmp_seq=2 ttl=64 time=2.88 ms
64 bytes from 10.0.0.6: icmp_seq=3 ttl=64 time=3.02 ms
64 bytes from 10.0.0.6: icmp_seq=4 ttl=64 time=2.96 ms
64 bytes from 10.0.0.6: icmp_seq=5 ttl=64 time=2.86 ms
64 bytes from 10.0.0.6: icmp_seq=6 ttl=64 time=3.78 ms
64 bytes from 10.0.0.6: icmp_seq=7 ttl=64 time=3.99 ms
64 bytes from 10.0.0.6: icmp_seq=8 ttl=64 time=10.0 ms
64 bytes from 10.0.0.6: icmp_seq=9 ttl=64 time=4.08 ms
64 bytes from 10.0.0.6: icmp_seq=10 ttl=64 time=3.91 ms
64 bytes from 10.0.0.6: icmp_seq=11 ttl=64 time=4.17 ms
^C
--- 10.0.0.6 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10015ms
rtt min/avg/max/mdev = 2.864/5.912/23.290/5.825 ms
mininet-wifi>
```

sta5ad < - - -> sta6ad

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
*** Removing fakelb module and Configurations
s7com1030@ubuntu:~/Desktop/Task02$ sudo ./Task02.py
*** Creating nodes
*** Configuring wifi nodes
*** Connecting to wmediumd server /var/run/wmediumd.sock
*** Creating links
*** Starting network
*** Configuring nodes
*** Running CLI
*** Starting CLI:
mininet-wifi> sta5ad ping sta6ad
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
64 bytes from 10.0.0.7: icmp_seq=1 ttl=64 time=44.7 ms
64 bytes from 10.0.0.7: icmp_seq=2 ttl=64 time=3.20 ms
64 bytes from 10.0.0.7: icmp_seq=3 ttl=64 time=3.05 ms
64 bytes from 10.0.0.7: icmp_seq=4 ttl=64 time=3.16 ms
64 bytes from 10.0.0.7: icmp_seq=5 ttl=64 time=3.83 ms
64 bytes from 10.0.0.7: icmp_seq=6 ttl=64 time=3.38 ms
64 bytes from 10.0.0.7: icmp_seq=7 ttl=64 time=3.30 ms
64 bytes from 10.0.0.7: icmp_seq=8 ttl=64 time=3.82 ms
64 bytes from 10.0.0.7: icmp_seq=9 ttl=64 time=4.13 ms
64 bytes from 10.0.0.7: icmp_seq=10 ttl=64 time=3.93 ms
^C
--- 10.0.0.7 ping statistics ---
```

sta4ad < - - -> sta6ad

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
64 bytes from 10.0.0.7: icmp_seq=10 ttl=64 time=3.93 ms
^C
--- 10.0.0.7 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9013ms
rtt min/avg/max/mdev = 3.059/7.659/44.749/12.368 ms
mininet-wifi> sta4ad ping sta6ad
PING 10.0.0.7 (10.0.0.7) 56(84) bytes of data.
From 10.0.0.5 icmp_seq=1 Destination Host Unreachable
From 10.0.0.5 icmp_seq=2 Destination Host Unreachable
From 10.0.0.5 icmp_seq=3 Destination Host Unreachable
From 10.0.0.5 icmp_seq=4 Destination Host Unreachable
From 10.0.0.5 icmp_seq=5 Destination Host Unreachable
From 10.0.0.5 icmp_seq=6 Destination Host Unreachable
From 10.0.0.5 icmp_seq=7 Destination Host Unreachable
From 10.0.0.5 icmp_seq=8 Destination Host Unreachable
From 10.0.0.5 icmp_seq=9 Destination Host Unreachable
From 10.0.0.5 icmp_seq=10 Destination Host Unreachable
From 10.0.0.5 icmp_seq=11 Destination Host Unreachable
From 10.0.0.5 icmp_seq=12 Destination Host Unreachable
^C
--- 10.0.0.7 ping statistics ---
13 packets transmitted, 0 received, +12 errors, 100% packet loss, time 12277ms
pipe 4
mininet-wifi>
```

sta7M < - - -> sta8M

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
64 bytes from 10.0.0.6: icmp_seq=10 ttl=64 time=3.91 ms
64 bytes from 10.0.0.6: icmp_seq=11 ttl=64 time=4.17 ms
^C
--- 10.0.0.6 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10015ms
rtt min/avg/max/mdev = 2.864/5.912/23.290/5.825 ms
mininet-wifi> sta7M ping sta8M
PING 10.0.0.9 (10.0.0.9) 56(84) bytes of data.
64 bytes from 10.0.0.9: icmp_seq=1 ttl=64 time=39.4 ms
64 bytes from 10.0.0.9: icmp_seq=2 ttl=64 time=2.95 ms
64 bytes from 10.0.0.9: icmp_seq=3 ttl=64 time=2.99 ms
64 bytes from 10.0.0.9: icmp_seq=4 ttl=64 time=3.02 ms
64 bytes from 10.0.0.9: icmp_seq=5 ttl=64 time=3.88 ms
64 bytes from 10.0.0.9: icmp_seq=6 ttl=64 time=4.04 ms
64 bytes from 10.0.0.9: icmp_seq=7 ttl=64 time=4.08 ms
64 bytes from 10.0.0.9: icmp_seq=8 ttl=64 time=9.02 ms
64 bytes from 10.0.0.9: icmp_seq=9 ttl=64 time=3.43 ms
64 bytes from 10.0.0.9: icmp_seq=10 ttl=64 time=4.19 ms
64 bytes from 10.0.0.9: icmp_seq=11 ttl=64 time=4.05 ms
^C
--- 10.0.0.9 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10013ms
rtt min/avg/max/mdev = 2.954/7.374/39.411/10.257 ms
mininet-wifi>
```

Sta8M < - - -> sta9M

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
*** Configuring wifi nodes
*** Connecting to wmediumd server /var/run/wmediumd.sock
*** Creating links
*** Starting network
*** Configuring nodes
*** Running CLI
*** Starting CLI:
mininet-wifi> sta8M ping sta9M
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=37.5 ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=64 time=3.02 ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=64 time=3.07 ms
64 bytes from 10.0.0.10: icmp_seq=4 ttl=64 time=2.29 ms
64 bytes from 10.0.0.10: icmp_seq=5 ttl=64 time=4.00 ms
64 bytes from 10.0.0.10: icmp_seq=6 ttl=64 time=4.25 ms
64 bytes from 10.0.0.10: icmp_seq=7 ttl=64 time=3.76 ms
64 bytes from 10.0.0.10: icmp_seq=8 ttl=64 time=3.73 ms
64 bytes from 10.0.0.10: icmp_seq=9 ttl=64 time=3.98 ms
64 bytes from 10.0.0.10: icmp_seq=10 ttl=64 time=10.0 ms
^C
--- 10.0.0.10 ping statistics ---
10 packets transmitted, 10 received, 0% packet loss, time 9014ms
rtt min/avg/max/mdev = 2.295/7.572/37.576/10.202 ms
mininet-wifi>
```

sta7M < - - -> sta9M

```
s7com1030@ubuntu: ~/Desktop/Task02
File Edit View Search Terminal Help
64 bytes from 10.0.0.9: icmp_seq=10 ttl=64 time=4.19 ms
64 bytes from 10.0.0.9: icmp_seq=11 ttl=64 time=4.05 ms
^C
--- 10.0.0.9 ping statistics ---
11 packets transmitted, 11 received, 0% packet loss, time 10013ms
rtt min/avg/max/mdev = 2.954/7.374/39.411/10.257 ms
mininet-wifi> sta7M ping sta9M
PING 10.0.0.10 (10.0.0.10) 56(84) bytes of data.
64 bytes from 10.0.0.10: icmp_seq=1 ttl=64 time=42.8 ms
64 bytes from 10.0.0.10: icmp_seq=2 ttl=64 time=7.42 ms
64 bytes from 10.0.0.10: icmp_seq=3 ttl=64 time=7.12 ms
64 bytes from 10.0.0.10: icmp_seq=4 ttl=64 time=10.3 ms
64 bytes from 10.0.0.10: icmp_seq=5 ttl=64 time=7.98 ms
64 bytes from 10.0.0.10: icmp_seq=7 ttl=64 time=8.07 ms
64 bytes from 10.0.0.10: icmp_seq=8 ttl=64 time=8.08 ms
64 bytes from 10.0.0.10: icmp_seq=9 ttl=64 time=8.04 ms
64 bytes from 10.0.0.10: icmp_seq=10 ttl=64 time=14.1 ms
64 bytes from 10.0.0.10: icmp_seq=11 ttl=64 time=7.96 ms
64 bytes from 10.0.0.10: icmp_seq=12 ttl=64 time=7.97 ms
^C
--- 10.0.0.10 ping statistics ---
12 packets transmitted, 11 received, 8% packet loss, time 11047ms
rtt min/avg/max/mdev = 7.120/11.819/42.837/9.988 ms
mininet-wifi>
```

Analysis

- o Calculate TCP Success rate, this can be done by the statistics collected

Results:

Link bandwidth (Mbit/s):	52.1
Max achievable TCP throughput limited by TCP overhead (Mbit/s):	49.4577
Bandwidth-Delay Product (BDP) (bit):	837455400
Minimum required TCP RWND (Byte):	104681925
Max TCP throughput limited by packet loss (Mathis et.al. formula) (Mbit/s):	0.726639
Max TCP throughput limited by TCP RWND (Mbit/s):	0.042454
Expected maximum TCP throughput (Mbit/s):	0.042454
Minimum transfer time for a 1000 Megabytes file (D:H:M:S):	2:04:20:39

- o Critically evaluate the reason for success or failure of the ICMP streams between sta4ad < - - -> sta6ad and sta7M < - - -> sta9M. Conduct a discussion of the results with evidence (screenshots) and reference:

sta4ad cannot ping sta6ad but sta7M can ping sta9M this is because they are in a mesh network and sta8M transfers the data and we can see that the ping time for sta7M to sta9M equals to the ping time from sta7M to sta8M + sta8M to sta9M.

- o If the nodes are in mobility during the transmission of the TCP stream, will the performance deviate from the collected in any way? Conduct a discussion based on this experiment. If needed add reference from background research to further support your claims.

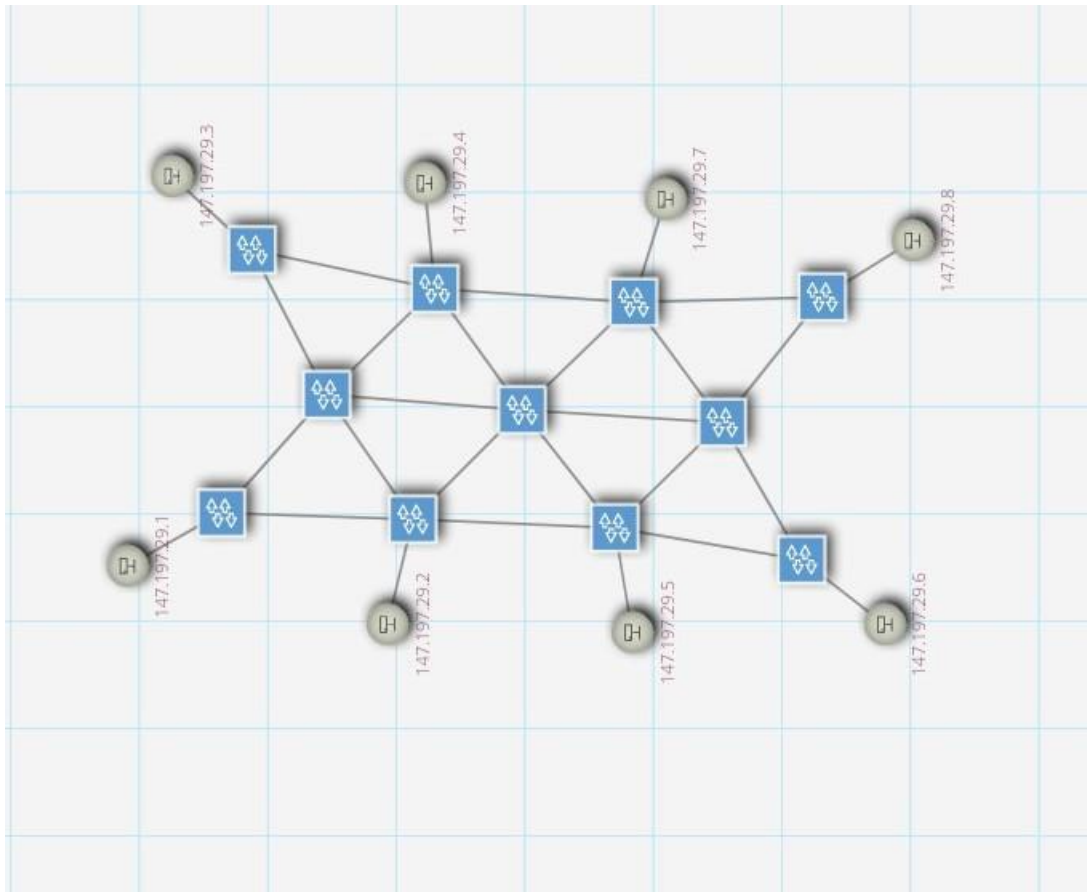
Yes, because a Handoff between the new access point and the old access point is taking place when the stations are moving between the access points.

- o Critically evaluate why STA1 < - - > STA4 ping fail in Task 1? How can a successful ping be achieved?

We cannot ping from Sta1 to Sta4 because sta4 is not in range of any of the access points

TASK 3

ONOS GUI



ICMP stream

```
mininet-wifi> pingall
*** Ping: testing ping reachability
h1 -> X h3 X X X h7 X
h2 -> X X h4 X h6 X h8
h3 -> h1 X X h5 X h7 X
h4 -> X h2 X X h6 X h8
h5 -> h1 X h3 X X h7 X
h6 -> X h2 X h4 X X X
h7 -> h1 X h3 X h5 X X
h8 -> X h2 X h4 X h6 X
*** Results: 60% dropped (22/56 received)
mininet-wifi>
```

Link configurations

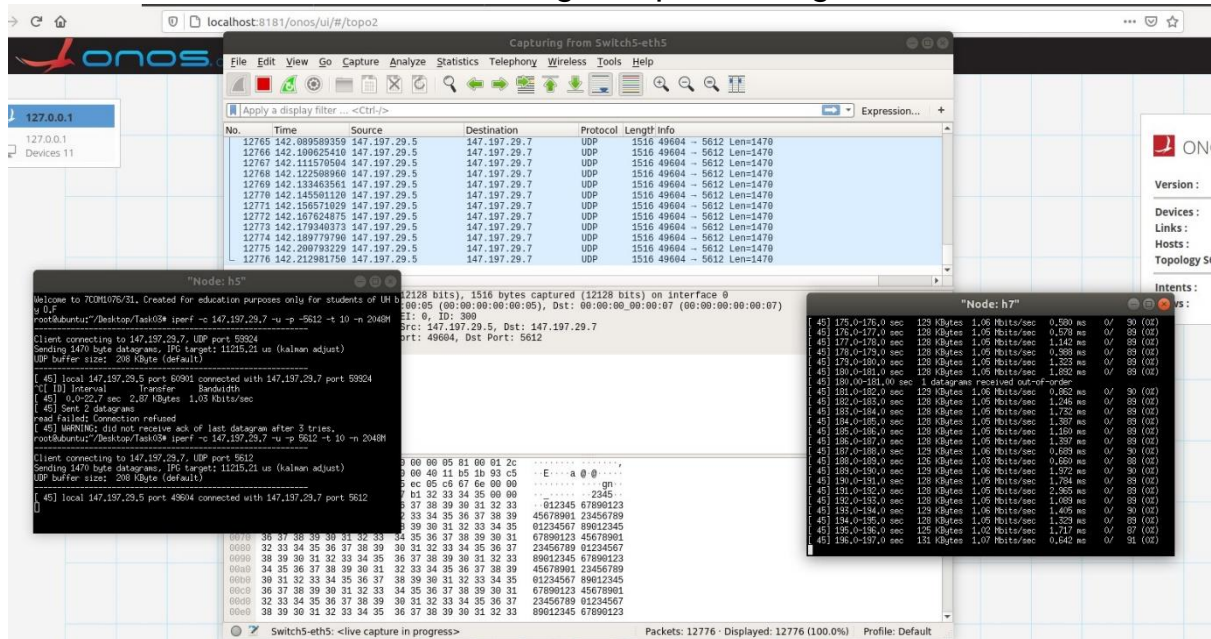
[illegible]

Links status:

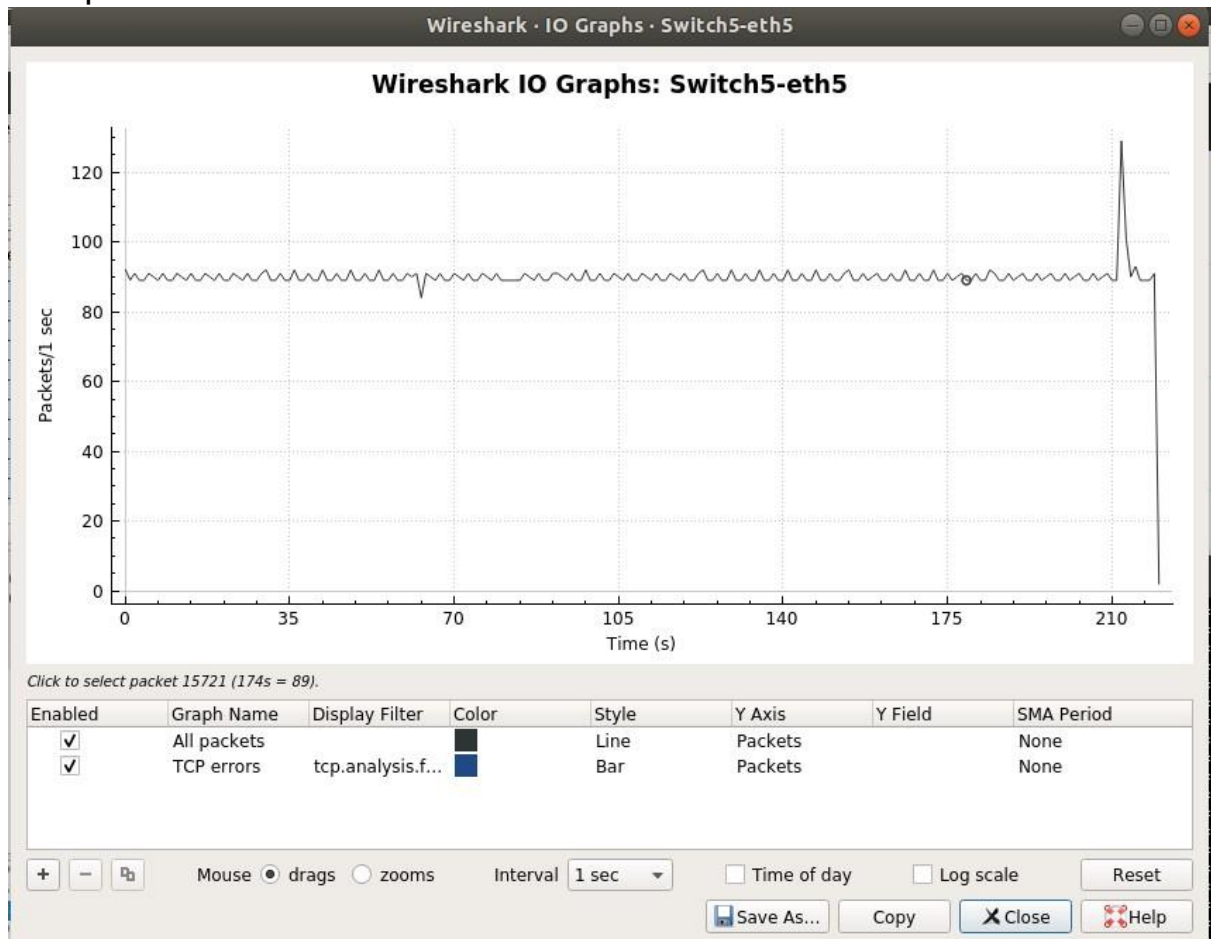
```
c0
mininet-wifi> links
Switch1-eth1<->Switch2-eth1 (OK OK)
Switch1-eth2<->Switch9-eth1 (OK OK)
Switch2-eth4<->Switch5-eth1 (OK OK)
Switch2-eth2<->Switch9-eth2 (OK OK)
Switch2-eth3<->Switch10-eth1 (OK OK)
Switch4-eth3<->Switch3-eth2 (OK OK)
Switch5-eth4<->Switch6-eth1 (OK OK)
Switch5-eth2<->Switch10-eth2 (OK OK)
Switch5-eth3<->Switch11-eth1 (OK OK)
Switch6-eth2<->Switch11-eth2 (OK OK)
Switch7-eth3<->Switch4-eth4 (OK OK)
Switch8-eth2<->Switch7-eth4 (OK OK)
Switch9-eth3<->Switch3-eth1 (OK OK)
Switch9-eth4<->Switch4-eth1 (OK OK)
Switch9-eth5<->Switch10-eth3 (OK OK)
Switch10-eth4<->Switch4-eth2 (OK OK)
Switch10-eth5<->Switch7-eth1 (OK OK)
Switch10-eth6<->Switch11-eth3 (OK OK)
Switch11-eth4<->Switch7-eth2 (OK OK)
Switch11-eth5<->Switch8-eth1 (OK OK)
h1-eth0.300<->Switch1-eth3 (OK OK)
h2-eth0.400<->Switch2-eth5 (OK OK)
h3-eth0.300<->Switch3-eth3 (OK OK)
h4-eth0.400<->Switch4-eth5 (OK OK)
h5-eth0.300<->Switch5-eth5 (OK OK)
h6-eth0.400<->Switch6-eth3 (OK OK)
h7-eth0.300<->Switch7-eth5 (OK OK)
h8-eth0.400<->Switch8-eth3 (OK OK)
mininet-wifi>
```


TASK 4

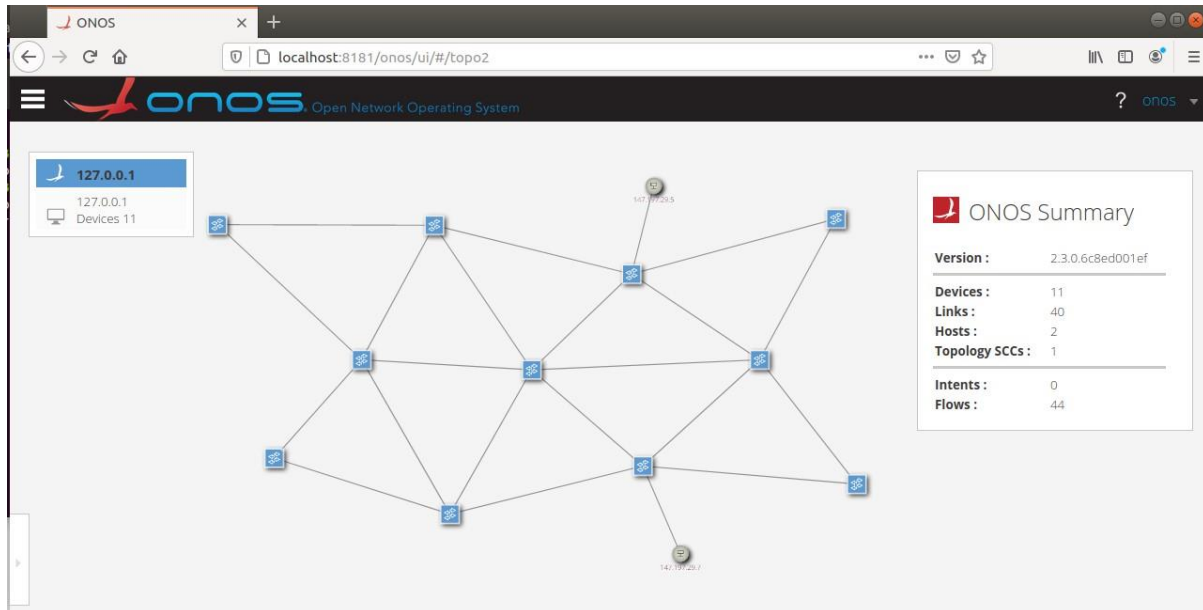
UDP flow to total of 2GB traffic using the port assigned



I/O Graph



ONOS GUI Server and Client



UDP Wireshark

Capturing from Switch5-eth5

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

Apply a display filter ... <Ctrl-/> Expression...

No.	Time	Source	Destination	Protocol	Length	Info
7277	81.010778235	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7278	81.022197674	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7279	81.033450174	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7280	81.044109596	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7281	81.055516147	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7282	81.066688431	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7283	81.077979278	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470
7284	81.089172394	147.197.29.5	147.197.29.7	UDP	1516	40886 → 5612 Len=1470

Frame 4741: 1516 bytes on wire (12128 bits), 1516 bytes captured (12128 bits) on interface 0

Ethernet II, Src: 00:00:00:00:00:05 (00:00:00:00:00:05), Dst: 00:00:00:00:00:07 (00:00:00:00:00:07)

802.1Q Virtual LAN, PRI: 0, DEI: 0, ID: 300

Internet Protocol Version 4, Src: 147.197.29.5, Dst: 147.197.29.7

0100 = Version: 4

.... 0101 = Header Length: 20 bytes (5)

Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)

Total Length: 1498

Identification: 0x3402 (13314)

Flags: 0x4000, Don't fragment

Time to live: 64

Protocol: UDP (17)

Header checksum: 0x9f7a [validation disabled]

[Header checksum status: Unverified]

Source: 147.197.29.5

Destination: 147.197.29.7

User Datagram Protocol, Src Port: 40886, Dst Port: 5612

Source Port: 40886

Destination Port: 5612

Length: 1478

Checksum: 0x676e [unverified]

[Checksum Status: Unverified]

[Stream index: 0]

Data (1470 bytes)

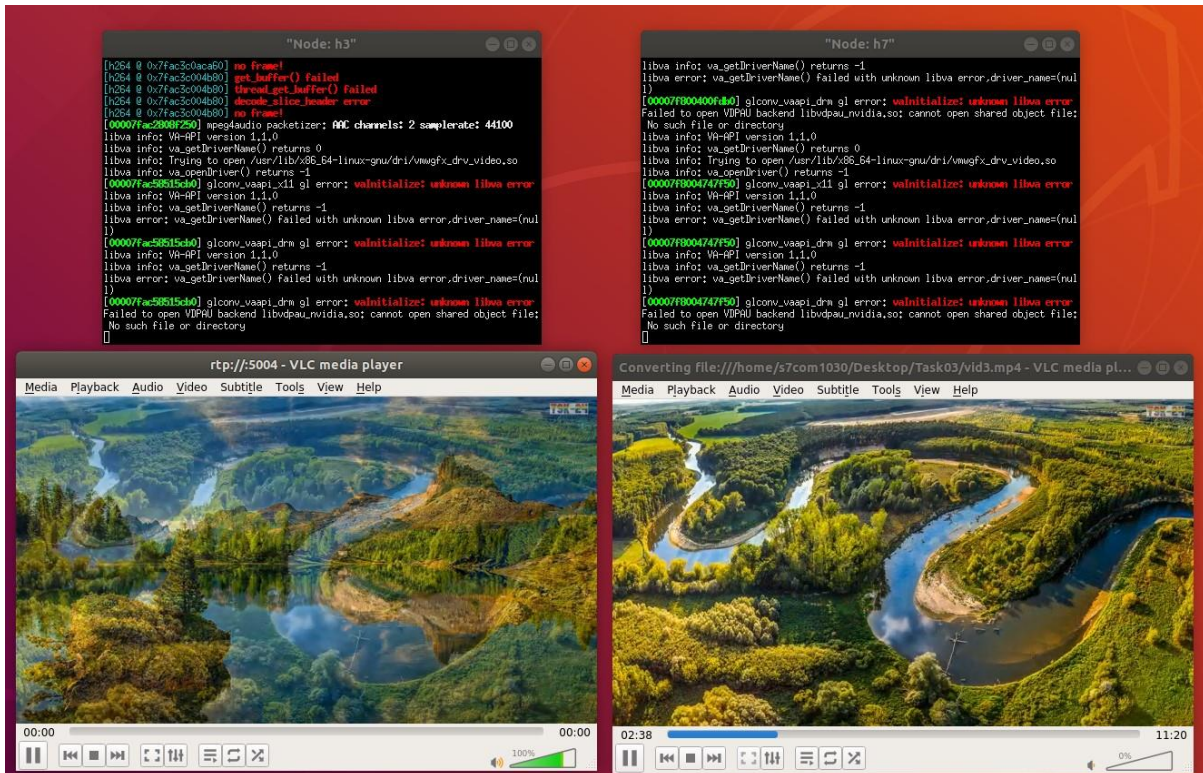
Data: 000096915ff32ae5000c4b3c323334350000000030313233...

[Length: 1470]

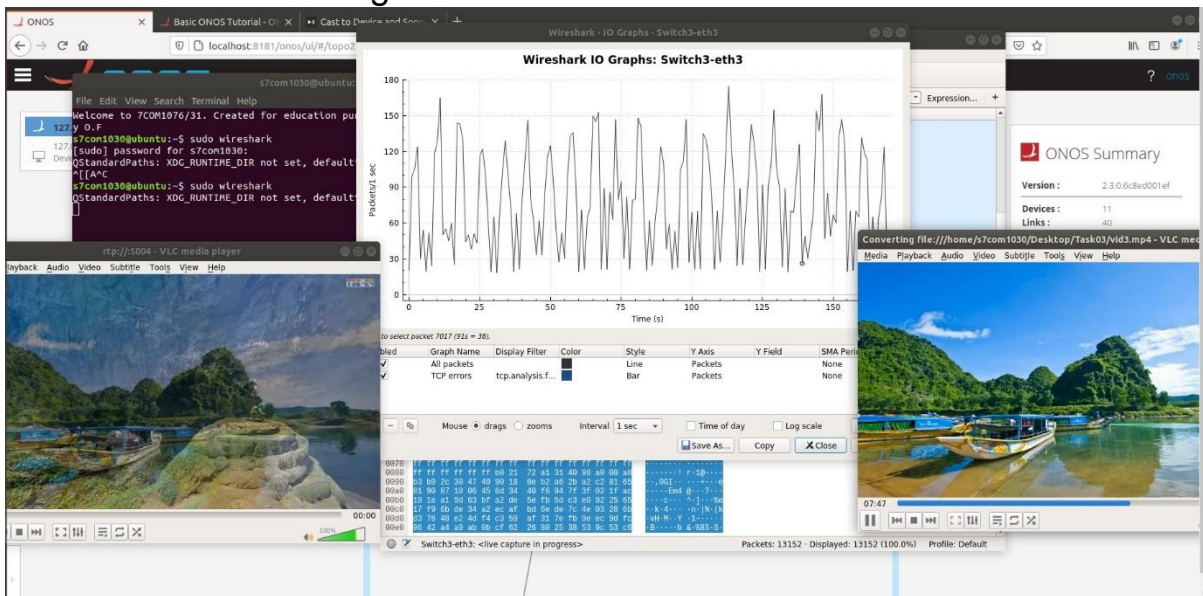
0000 00 00 00 00 07 00 00 00 00 05 81 00 01 2c
 0010 08 00 45 00 05 da 34 02 40 00 40 11 9f 7a 93 c5 ..E...4. @ @ .z..
 0020 1d 05 93 c5 1d 07 9f b6 15 ec 05 c6 67 6e 00 00gn..
 0030 96 91 5f f3 2a e5 00 0c 4b 3c 32 33 34 35 00 00K<2345..
 0040 00 00 30 31 32 33 34 35 36 37 38 39 30 31 32 33 ..012345 67890123
 0050 34 35 36 37 38 39 30 31 32 33 34 35 36 37 38 39 45678901 23456789

Switch5-eth5: <live capture in progress> Packets: 7284 · Displayed: 7284 (100.0%) Profile: Default

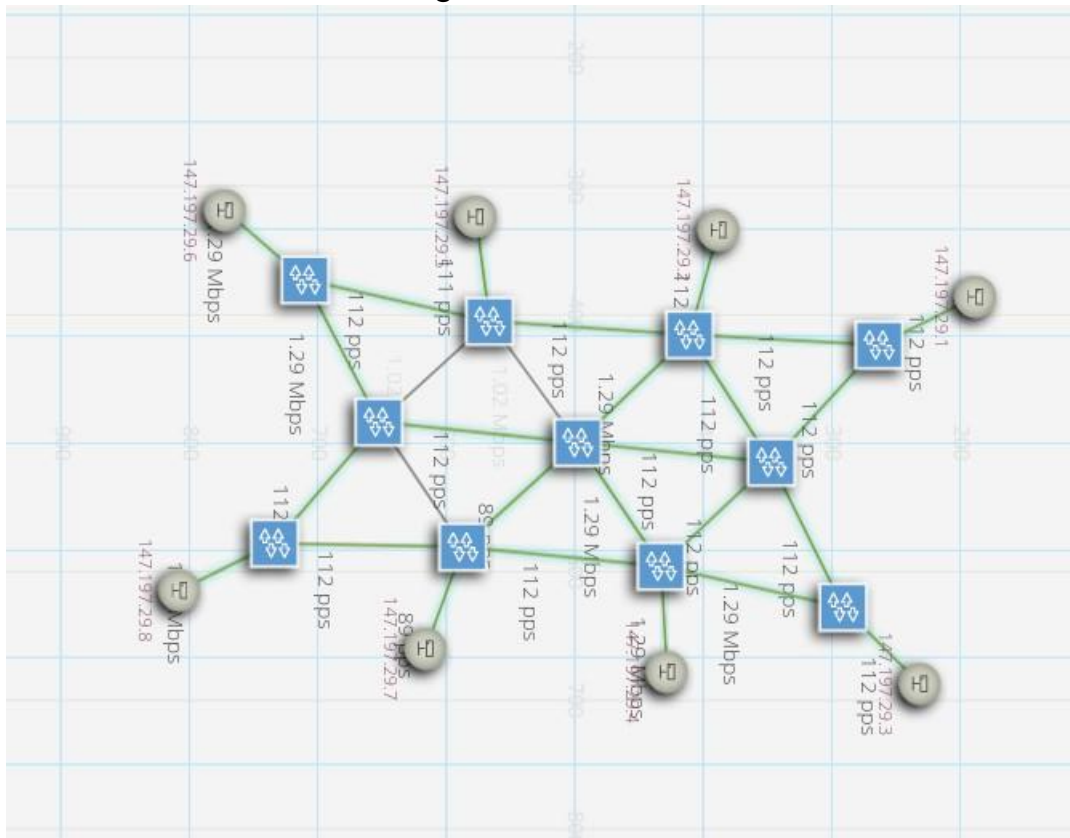
Video stream



Wireshark while streaming



Packets route while streaming



Capture File Properties

Wireshark · Capture File Properties · Switch5-eth5

Details

File

Name: /tmp/wireshark_Switch5-eth5_20210104144817_m2LyRc.pcapng
 Length: 85 MB
 Format: Wireshark/... - pcapng
 Encapsulation: Ethernet

Time

First packet: 2021-01-04 14:48:17
 Last packet: 2021-01-04 14:58:38
 Elapsed: 00:10:21

Capture

Hardware: Intel(R) Core(TM) i7-6820HQ CPU @ 2.70GHz (with SSE4.2)
 OS: Linux 5.4.0-58-generic
 Application: Dumpcap (Wireshark) 2.6.10 (Git v2.6.10 packaged as 2.6.10-1~ubuntu18.04.0)

Interfaces

Interface	Dropped packets	Capture filter	Link type	Packet size limit
Switch5-eth5	Unknown	none	Ethernet	262144 bytes

Statistics

Measurement	Captured	Displayed	Marked
Packets	55895	55895 (100.0%)	—
Time span, s	621.925	621.925	—
Average pps	89.9	89.9	—
Average packet size, B	1505	1505	—
Bytes	84116518	84116518 (100.0%)	0
Average bytes/s	135 k	135 k	—
Average bits/s	1,082 k	1,082 k	—

Capture file comments

Refresh Save Comments Close Copy To Clipboard Help

Analysis

o Should there be more packet loss and delay, should you expect the results to deviate from what you have acquired?

For this task, we are using ONOS and because of our own data connection (internet disconnections, etc.) we are having more packet losses and jitter than expected.

```
[ 45] 7.0- 8.0 sec 543 KBytes 4.45 Mbits/sec 0.248 ms 3/ 381 (0.79%)
[ 45] 7.00-8.00 sec 22 datagrams received out-of-order
[ 45] 8.0- 9.0 sec 0.00 Bytes 0.00 bits/sec 0.000 ms 0/ 0 (0%)
[ 45] 8.00-9.00 sec 514 KBytes 2.98 Mbits/sec 0.571 ms 7/ 226 (3.1%)
[ 45] 9.00-10.00 sec 14 datagrams received out-of-order
```

o Comment on how the variables such as packet loss and delay have contributed towards the overall performance of your network.

Because we are using UDP, there is no retransmissions to ensure reliable messaging. therefore, we are sending the data much faster, but we are also losing some of it.

```
[ 45] local 147.197.29.7 port 5612 connected with 147.197.29.5 port 33244
[ 10] Interval Transfer Bandwidth Jitter Lost/Total Datagrams
[ 45] 0.0- 1.0 sec 838 KBytes 6.87 Mbits/sec 9.780 ms 84/ 668 (13%)
[ 45] 0.00-1.00 sec 90 datagrams received out-of-order
[ 45] 1.0- 2.0 sec 791 KBytes 6.48 Mbits/sec 13.134 ms 20/ 571 (3.5%)
[ 45] 1.00-2.00 sec 70 datagrams received out-of-order
```

References:

Obaidat, M.S. and Green, D.B. (2003). Simulation of Wireless Networks. Applied System Simulation, pp.115–153.

Fontes, R.R., Afzal, S., Brito, S.H.B., Santos, M.A.S. and Rothenberg, C.E. (2015). Mininet-WiFi: Emulating software-defined wireless networks. 2015 11th International Conference on Network and Service Management (CNSM).

GUEANT, V. (2013). iPerf - The TCP, UDP and SCTP network bandwidth measurement tool. [online] lperf.fr. Available at: <https://iperf.fr/>.

Mininet Team (2018). Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet. [online] Mininet.org. Available at: <http://mininet.org/>.

Wireshark.org. (2019). Chapter 1. Introduction. [online] Available at: https://www.wireshark.org/docs/wsug_html_chunked/ChapterIntroduction.html.