DCU School of Electronic Engineering Assignment Submission

Student Name(s):	**Michael Lenehan**
Student Number(s):	**15410402**
Programme:	**B.Eng in Electronic and Computer Engineering**
Module Code:	**EE452**
Lecturer:	**J. McManis**
Project Due Date:	**12/12/2018**

Declaration

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited andacknowledged within the text of my work. I understand thatplagiarism, collusion, and copying is a grave and serious offence in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion, or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged and the source cited are identified in the assignment references.

I have not copied or paraphrased an extract of any length from any source without identifying the source and using quotation marks as appropriate. Any images, audio recordings, video or other materials have likewise been originated and produced by me or are fully acknowledged and identified.

This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study. I have read and understood the referencing guidelines found at http://www.library.dcu.ie/citing&refguide08.pdf and/or recommended in the assignment guidelines.

I understand that I may be required to discuss with the module lecturer/s the contents of this submission.

I/me/my incorporates we/us/our in the case of group work, which is signed by all of us.

Signed: ***Michael Lenehan***

Assignment 2

Michael Lenehan

Introduction

The aim of this assignment is to analyse the 802.11ac Distributed Coordinator Function MAC level implementation, and the throughput of the 802.11ac standard. The throughput will be measured using both the TCP and UDP transfer protocols, each using one source node and one sink node. In order to demonstrate the effects on throughput of transmission to multiple clients, extra sink nodes are added to each example.

Procedure

2.1 Analysis of TCP and UDP

In order to analyse the difference between the throughput of 802.11ac in the TCP and UDP protocols, two systems must be set up, one using the TCP protocol, and one using the UDP protocol. Each system must have an equal number of nodes, an equal number of sources, and an equal number of sinks. Sink nodes must be in the same positions in both the TCP and UDP system.

For this section of the assignment, systems consisting of one source node and one sink node were chosen. This allowed for the analysis to be completed without the need to consider the effects of interference.

Due to an error caused by the RAA's available for use with HT rates in ns-3.29, the FlowMonitor output XML file could not be obtained using ns-3.29. As such, the provided code was modified and executed using ns-3.25.

2.1.1 Setup for UDP Throughput Analysis

With the provided code, in order to correctly analyse the throughput, a FlowMonitor object is created. This object is used to output an XML file, containing all relevant information from the packet transfers between the source and sink nodes.

2.1.2 Setup for TCP Throughput Analysis

As with the UDP system, a FlowMonitor object must be created in order to output an XML used for analysis. Also required in the TCP system is the addition of the "Config::Connect()" method must be called, with the source node as an argument to the method. This is due to the fact that the source also receives packets in TCP, in the form of Acknowledgements.

2.2 Impact of Load on Performance

The impact of load on the performance of the 802.11ac standard may be analysed by adding multiple sink nodes to the TCP and UDP systems. For the purpose of this assignment, it was chosen to use a 5 Sink, 10 Sink and 15 Sink system for the comparison of each protocol.

The provided "ReceivePacket()" and "CalculateThroughput()" methods are used as arguments within "Config::Connect()" and Simulator::Schedule()" methods respectively. However, adding multiple sinks, while calling the same methods for each, will calculate total system throughput. In order to calculate throughput separately for each sink, separate "ReceivePacket()" and "CalculateThroughput()" methods must be implemented.

2.2.1 Modified UDP with multiple Nodes Code Snippets

In order to add multiple sink nodes to the provided "udpWireless.cc" file, the "Create" methods argument must be increased from 2 (one source and one sink) to 6, 11, or 16 for 5, 10 or 15 sink nodes respectively. Position vectors must then be added for each of the new nodes. The distances used are as follows:

Position Vectors for 1 Source and 5 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
```

Position Vectors for 1 Source and 10 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.5, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.5));
```

Position Vectors for 1 Source and 15 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.5, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.5));
positionAlloc -> Add (Vector (1.5, 1.5, 1.5));
positionAlloc -> Add (Vector (1.5, 0.0, 1.5));
positionAlloc -> Add (Vector (1.5, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 1.5));
positionAlloc -> Add (Vector (2.5, 2.5, 2.5));
```

In order to transmit data from the source to the sink node, an additional "OnOffHelper" object must be created for each added sink node. The address of the added sink node is passed as an argument to the "interfaces.GetAddress()" method passed to the objects constructor. This "OnOffHelper" object is then set to transmit from the source node (address 0) using the "Install(nodes.Get())" method on the object.

Source Node Configuration for Additional Nodes

```
OnOffHelper onoff3 ("ns3::TcpSocketFactory",
                    Address (InetSocketAddress (interfaces.GetAddress (3),
                      port)));
```

⋮

```
cpp ApplicationContainer apps3 = onoff3.Install (nodes.Get (0));
```

In order to create the additional sink nodes, "SinkHelper" objects must be initialised, and then installed on the node at the required address.

Sink Node Configuration for Additional Nodes

```
PacketSinkHelper sinkHelper3 ("ns3::TcpSocketFactory",
                              InetSocketAddress (Ipv4Address::GetAny (),
                        port));
ApplicationContainer sinkApp3 = sinkHelper3.Install (nodes.Get(3));
```

Additional "Config::Connect()" methods are required, with the additional sink nodes passed in as arguments.

"Config::Connect()" Method for Added Nodes (5 Sink Example)

```
Config::Connect ("/NodeList/1/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/2/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/3/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect("/NodeList/4/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect("/NodeList/5/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
```

2.2.2 Modified TCP with multiple Nodes Code Snippets

In order to add multiple sink nodes to the provided "tcpWireless.cc" file, the "Create" methods argument must be increased from 2 (one source and one sink) to 6, 11, or 16 for 5, 10 or 15 sink nodes respectively. Position vectors must then be added for each of the new nodes. The distances used are as follows:

Position Vectors for 1 Source and 5 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
```

Position Vectors for 1 Source and 10 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.5, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.5));
```

Position Vectors for 1 Source and 15 Sink Nodes

```
positionAlloc -> Add (Vector (0.0, 0.0, 0.0));
positionAlloc -> Add (Vector (1.0, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.0, 0.0, 1.0));
positionAlloc -> Add (Vector (1.0, 1.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.0, 1.0));
positionAlloc -> Add (Vector (1.5, 0.0, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 0.0, 1.5));
positionAlloc -> Add (Vector (1.5, 1.5, 1.5));
positionAlloc -> Add (Vector (1.5, 0.0, 1.5));
positionAlloc -> Add (Vector (1.5, 1.5, 0.0));
positionAlloc -> Add (Vector (0.0, 1.5, 1.5));
positionAlloc -> Add (Vector (2.5, 2.5, 2.5));
```

As can be seen in the code snippets, the position vectors used for the TCP systems are the same as those used in the UDP systems.

The required modifications for the TCP system code are the same as those made in the UDP system code. One difference exists however, in that a "Config::Connect()" method must be added with the source node (at address 0) as the argument:

"Config::Connect()" Method for Added Nodes (5 Sink Example)

```
Config::Connect ("/NodeList/0/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/1/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/2/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
 MakeCallback(&ReceivePacket) );
Config::Connect("/NodeList/3/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/4/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
Config::Connect ("/NodeList/5/DeviceList/*/$ns3::WifiNetDevice/Mac/MacRx",
  MakeCallback(&ReceivePacket) );
```

Results

3.1 Analysis of TCP and UDP

3.1.1 Listed 802.11ac data rates

The advertised physical layer data rate of 802.11ac is approximately 7Gbps (at 5GHz). Single users may achieve MAC throughput of 500Mbps, with multiple users achieving rates of 1Gbps in total [1]. There are a number of factors within the protocol which influence the increases in throughput from the previous standards. Enhancements have been made to a number of the 802.11n standards specifications, with channel bonding (combining channels to increase throughput) being implemented both statically and dynamically, and the implementation of MU-MIMO, allowing for the increased data rates. Beamforming for MU-MIMO is also a specification of the 802.11ac standard, and the number of spatial streams, i.e. antennas, increased to eight from the previous standard of four [2].

3.1.2 Obtained Throughput

The following figure displays the throughput graph for a system using the TCP protocol, consisting of a single source node, and a single sink node.

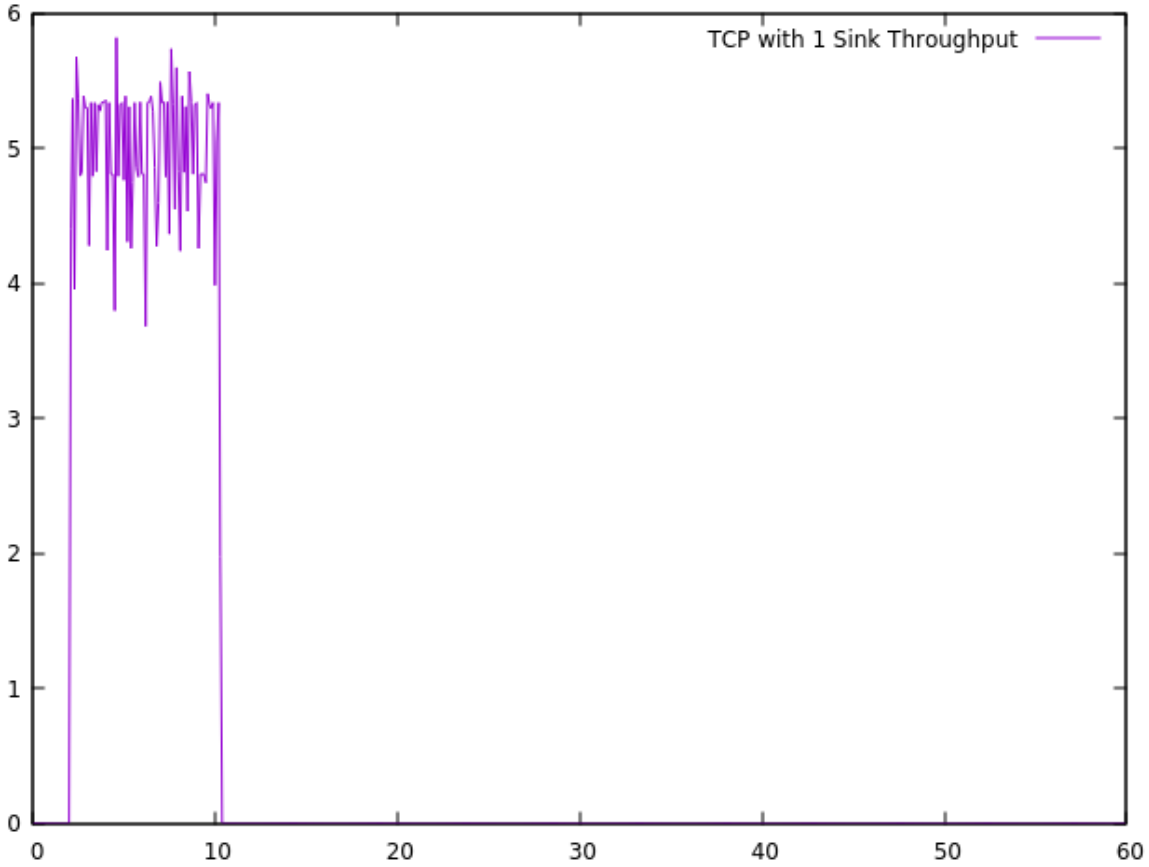

Figure 1: Total Throughput for TCP System with 1 source and 1 Sink Node

The following figure displays the throughput graph for a system using the UDP protocol, consisting of a single source node, and a single sink node.

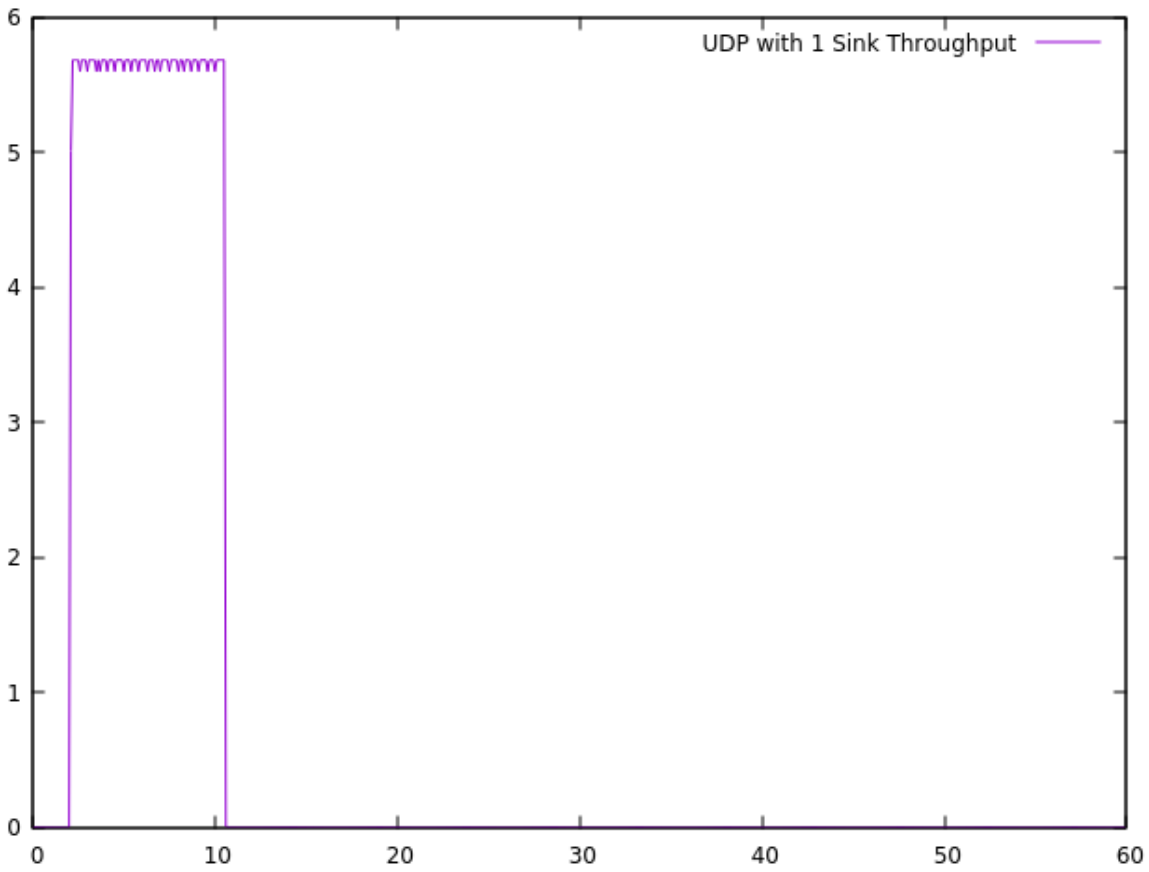

Figure 2: Total Throughput for UDP System with 1 Source and 1 Sink Node

The throughput represented in the above graphs is for the total of the systems, i.e. the average across all nodes at the sampled point in time. As such the separate throughput for each node must be calculated separately, using the output XML from the FlowMonitor.

The obtained throughput for each of the above was calculated using the following formula:

$$Throughput = \frac{rxBytes \times 8}{timeLastRxPacket - timeFirstRxPacket]}$$

The output values obtained for the TCP system were as follows:

Source Node $Throughput = 0.0215 Mbps$

Sink Node $Throughput = 4.68Mbps$

The output values obtained for the UDP system were as follows:

Sink Node $Throughput = 5.62Mbps$

3.1.3 UDP Deviation

Deviations in UDP throughput may be cause by a number of issues. Distance from the source node will cause a change in throughput. 802.11ac is highly susceptible to decreases in throughput due to increasing distances. As the sink node in the code is 5 meters from the source, this may be the main cause for the deviation between the advertised and simulated throughput.

3.1.4 TCP Deviation

Deviations in TCP throughput may be caused by a number of issues. As with UDP, distance from the source node will cause a change in throughput. The sink node is placed at the same distance from the source node as in the UDP system, which, again, may be the main cause of deviation between the advertised and simulated throughput.

3.1.5 Trace

The following timing diagram was developed using Wireshark, and the output trace file from the UDP system described above:

Figure 3: Wireshark Timing Diagram for the transmission of UDP packets

The transmission of a single 1024 byte UDP packet can be seen in the timing diagram below, with the transmission (Tx) and receive (Rx) times listed below the source and sink nodes.

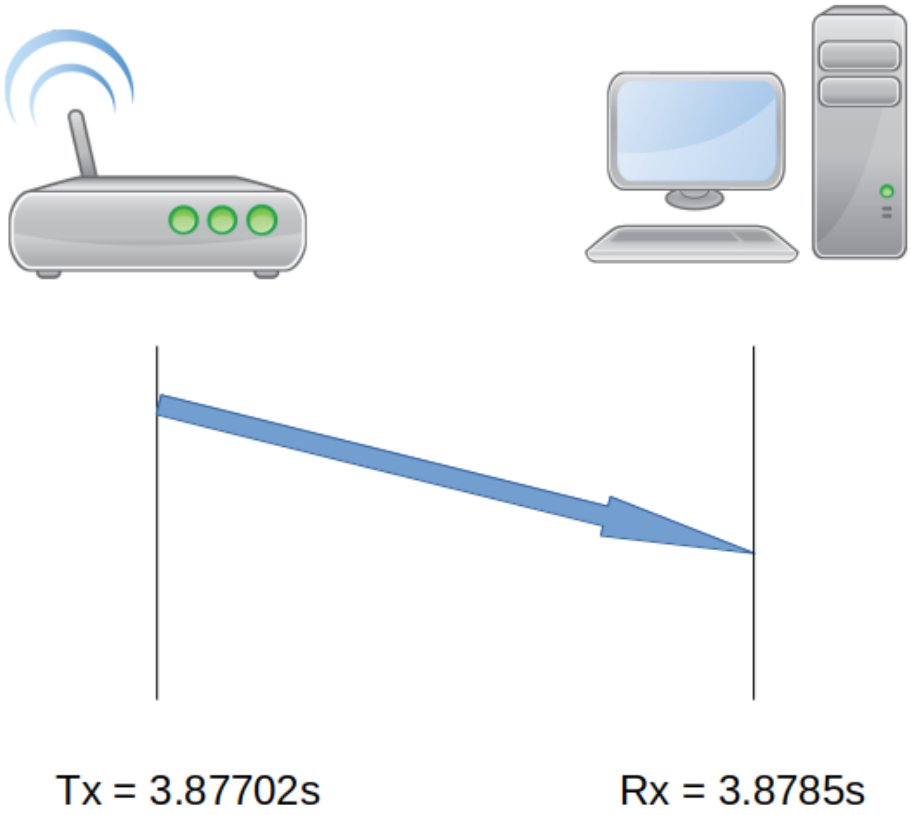

Figure 4: Timing Diagram for the transmission of one 1024 byte UDP packet

Distributed Coordination Function (DCF) is enabled by default within ns-3.25 for 802.11ac UDP packets. DCF allows for collision avoidance, using Request-To-Send and Clear-To-Send (RTS/CTS), which allows each node monitor the channel for transmissions, and, when the channel is idle, transmit data[3].

3.2 Impact on Load on Performance

3.2.1 Measured TCP and UDP Throughput with Multiple Nodes

TCP 5 Sinks The following figure displays the total throughput graph for a system using the TCP protocol, consisting of a single source node, and five sink nodes.

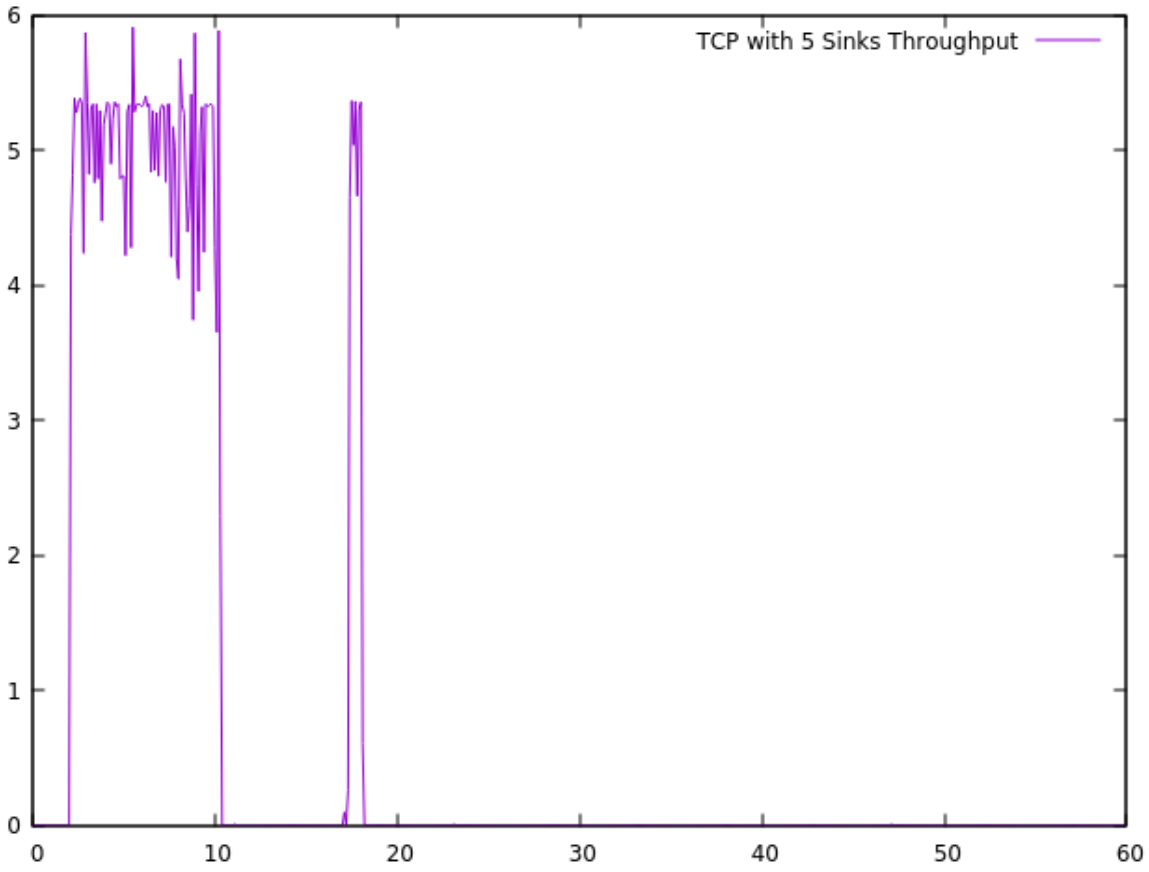

Figure 5: Total Throughput for TCP System with 1 Source and 5 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 0.07832 Mbps$

Sink Node 2 $Throughput = 0.00004 Mbps$

Sink Node 3 $Throughput = 4.724Mbps$

Sink Node 4 $Throughput = 0.73Mbps$

Sink Node 5 $Throughput = 0.0772Mbps$

Source Node $AverageThroughput = 0.0455Mbps$

TCP 10 Sinks The following figure displays the total throughput graph for a system using the TCP protocol, consisting of a single source node, and ten sink node.

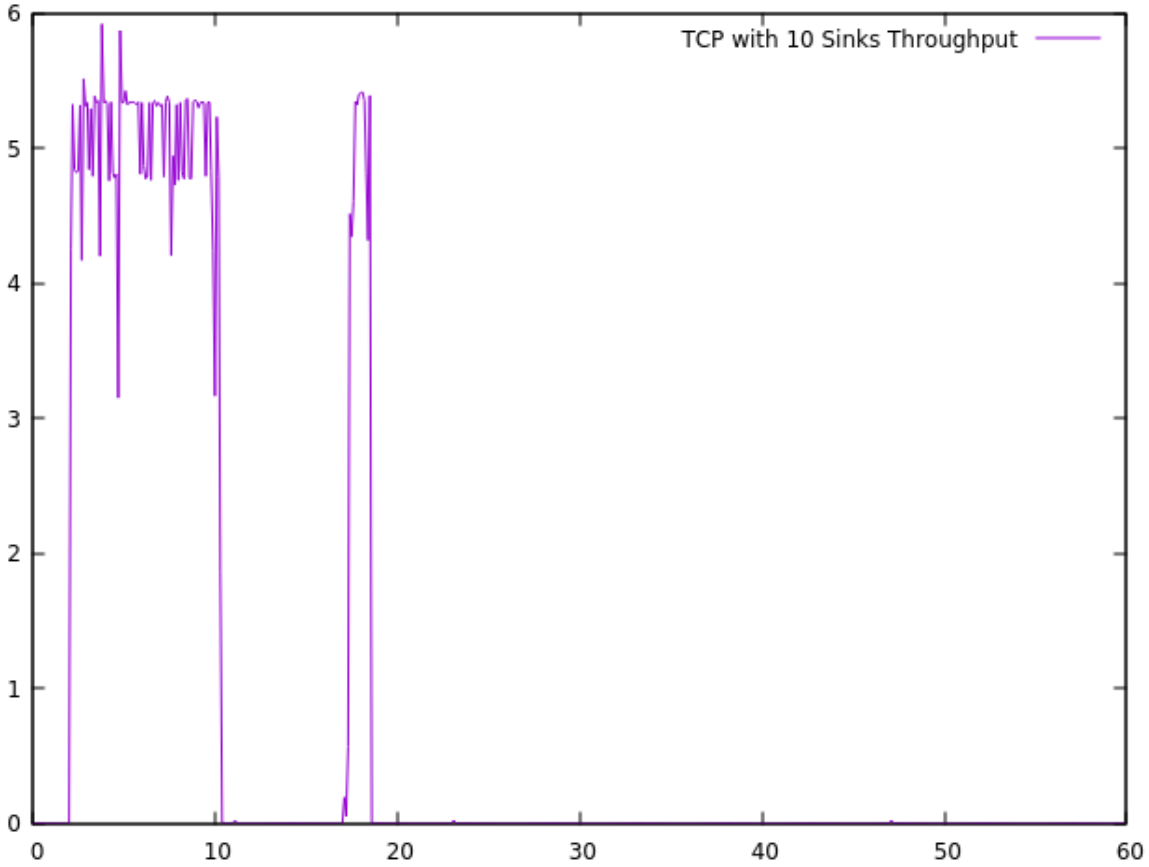

Figure 6: Total Throughput for TCP System with 1 Source and 10 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 0.0748Mbps$

Sink Node 2 $Throughput = 4.7156Mbps$

Sink Node 3 $Throughput = 0.0722Mbps$

Sink Node 4 $Throughput = 0.0722Mbps$

Sink Node 5 $Throughput = 0.0723Mbps$

Sink Node 6 $Throughput = 0.00004Mbps$

Sink Node 7 $Throughput = 0.00004Mbps$

Sink Node 8 $Throughput = 0.00004Mbps$

Sink Node 9 $Throughput = 0.0904Mbps$

Sink Node 10 $Throughput = 0.00004Mbps$

Source Node $AverageThroughput = 0.0234Mbps$

TCP 15 Sinks The following figure displays the total throughput graph for a system using the TCP protocol, consisting of a single source node, and fifteen sink node.

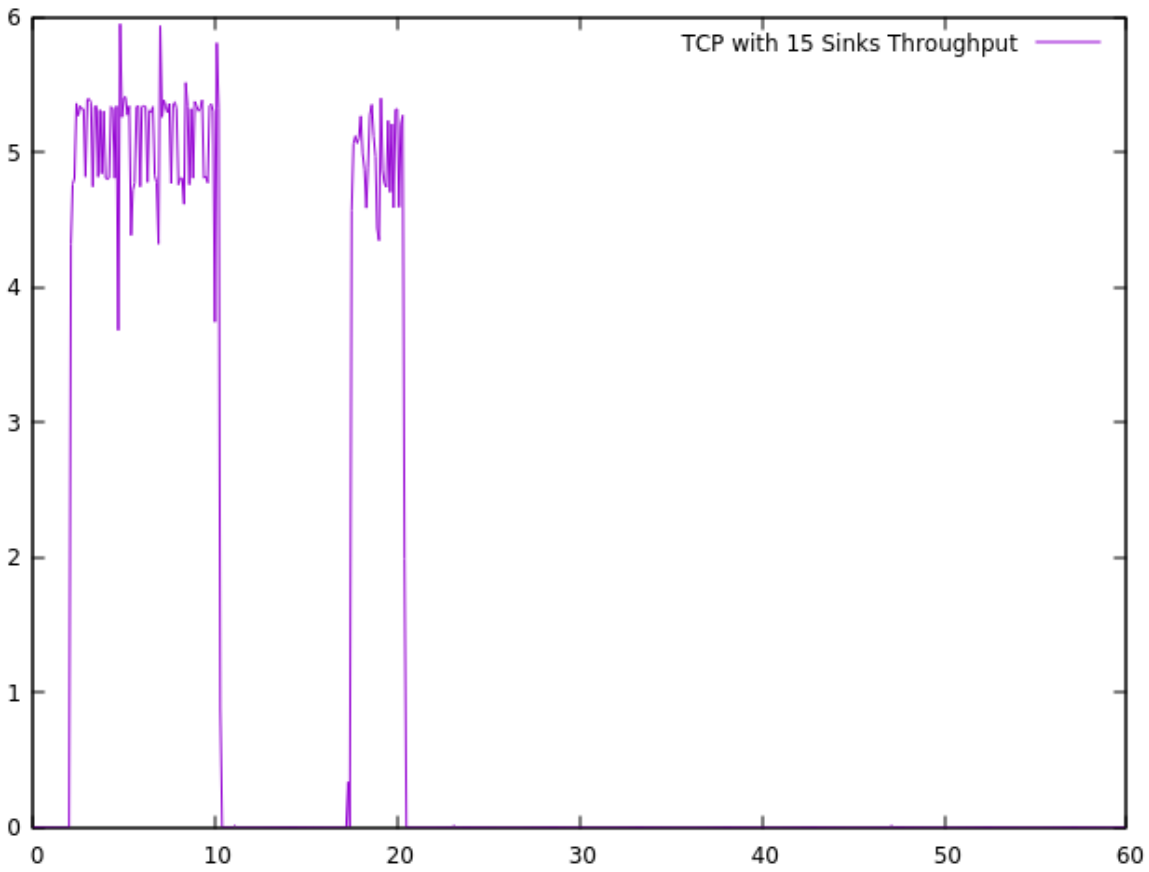

Figure 7: Total Throughput for TCP System with 1 Source and 15 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 0.0722Mbps$

Sink Node 2 $Throughput = 4.653Mbps$

Sink Node 3 $Throughput = 0.0729Mbps$

Sink Node 4 $Throughput = 0.0672Mbps$

Sink Node 5 $Throughput = 0.0653Mbps$

Sink Node 6 $Throughput = 0.07Mbps$

Sink Node 7 $Throughput = 0.066Mbps$

Sink Node 8 $Throughput = 0.0691Mbps$

Sink Node 9 $Throughput = 0.00004Mbps$

Sink Node 10 $Throughput = 0.069Mbps$

Sink Node 11 $Throughput = 0.069Mbps$

Sink Node 12 $Throughput = 0.065Mbps$

Sink Node 13 $Throughput = 0.00004Mbps$

Sink Node 14 $Throughput = 0.066Mbps$

Sink Node 15 $Throughput = 0.07Mbps$

Source Node $AverageThroughput = 0.017Mbps$

UDP 5 Sinks The following figure displays the total throughput graph for a system using the UDP protocol, consisting of a single source node, and five sink node.

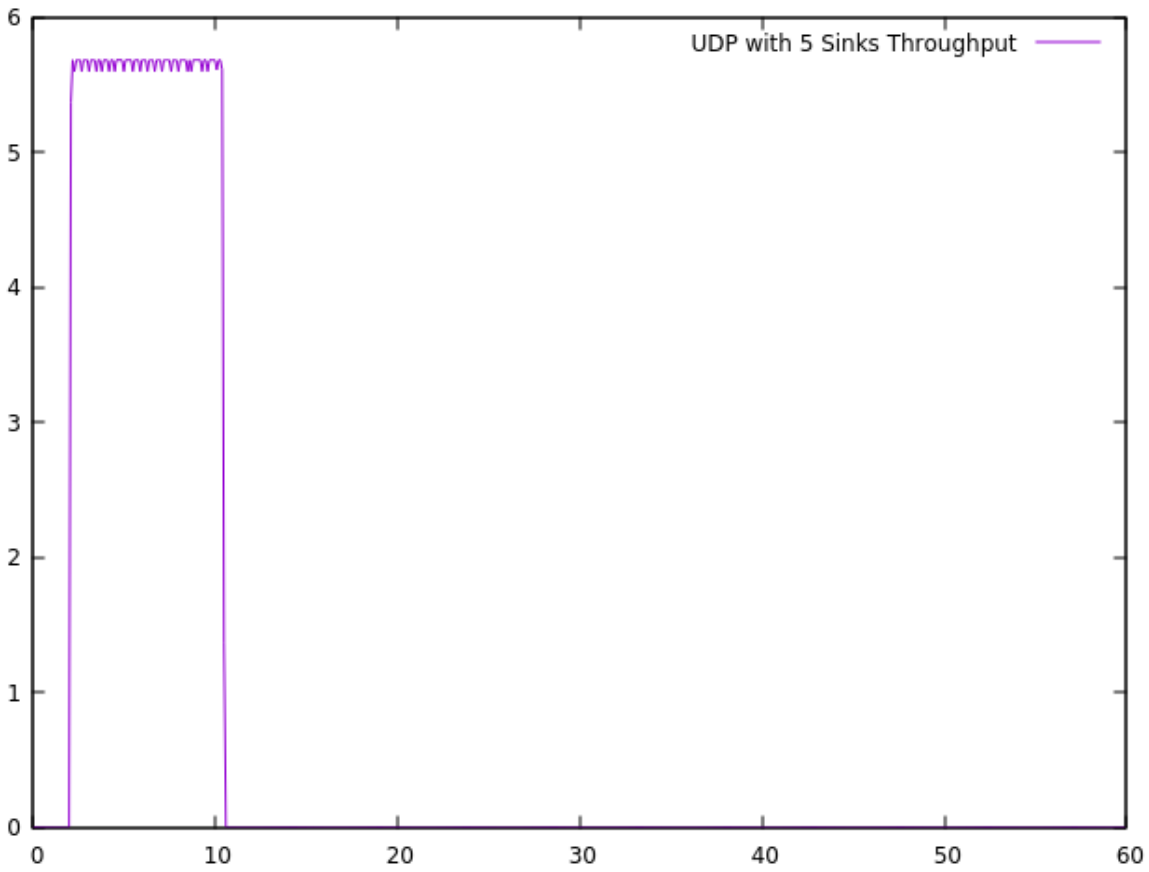

Figure 8: Total Throughput for UDP System with 1 Source and 5 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 0Mbps$

Sink Node 2 $Throughput = 5.62Mbps$

Sink Node 3 $Throughput = 0Mbps$

Sink Node 4 $Throughput = 0Mbps$

Sink Node 5 $Throughput = 0.598Mbps$

UDP 10 Sinks The following figure displays the total throughput graph for a system using the UDP protocol, consisting of a single source node, and ten sink node.

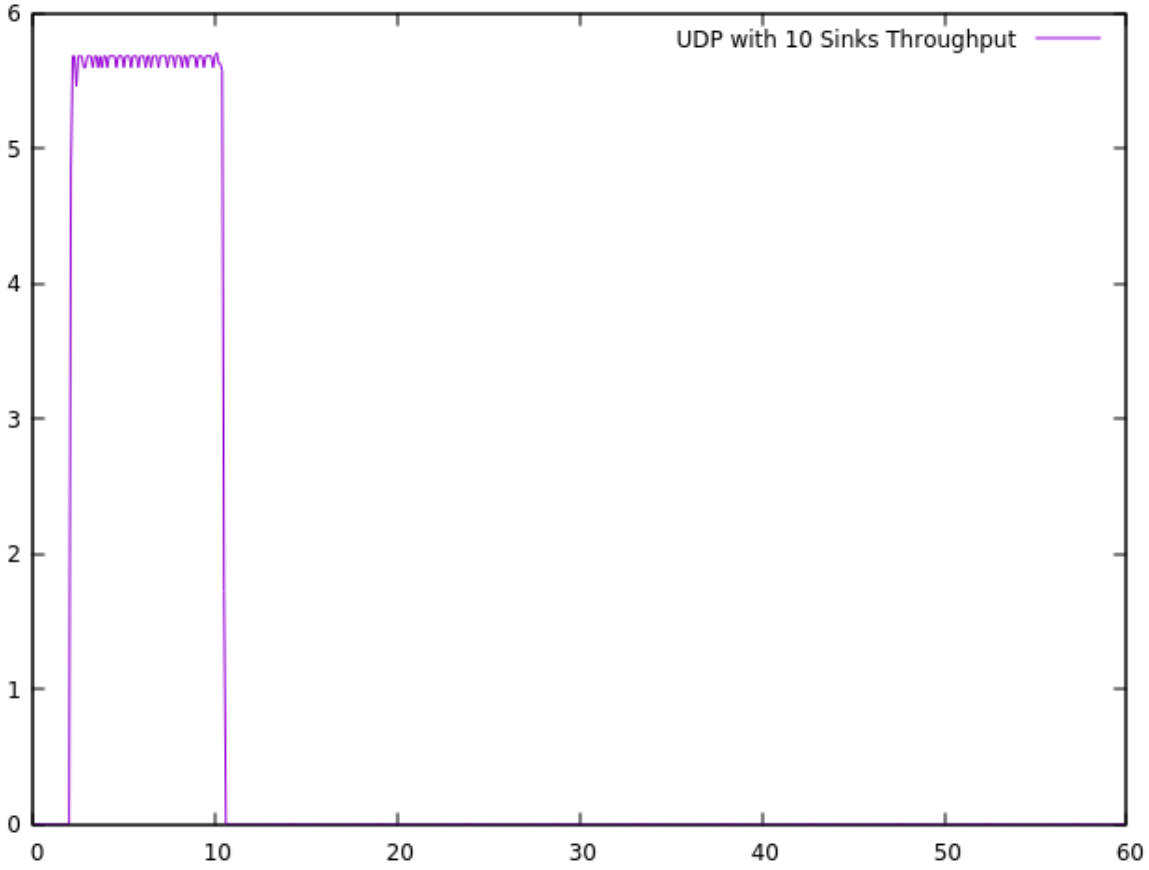

Figure 9: Total Throughput for UDP System with 1 Source and 10 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 2.3Mbps$

Sink Node 2 $Throughput = 1.68Mbps$

Sink Node 3 $Throughput = 1.61Mbps$

Sink Node 4 $Throughput = 0.454Mbps$

Sink Node 5 $Throughput = 5.9Mbps$

Sink Node 6 $Throughput = 0.1Mbps$

Sink Node 7 $Throughput = 5.46Mbps$

Sink Node 8 $Throughput = 0Mbps$

Sink Node 9 $Throughput = 0Mbps$

Sink Node 10 $Throughput = 0Mbps$

UDP 15 Sinks The following figure displays the total throughput graph for a system using the UDP protocol, consisting of a single source node, and fifteen sink node.

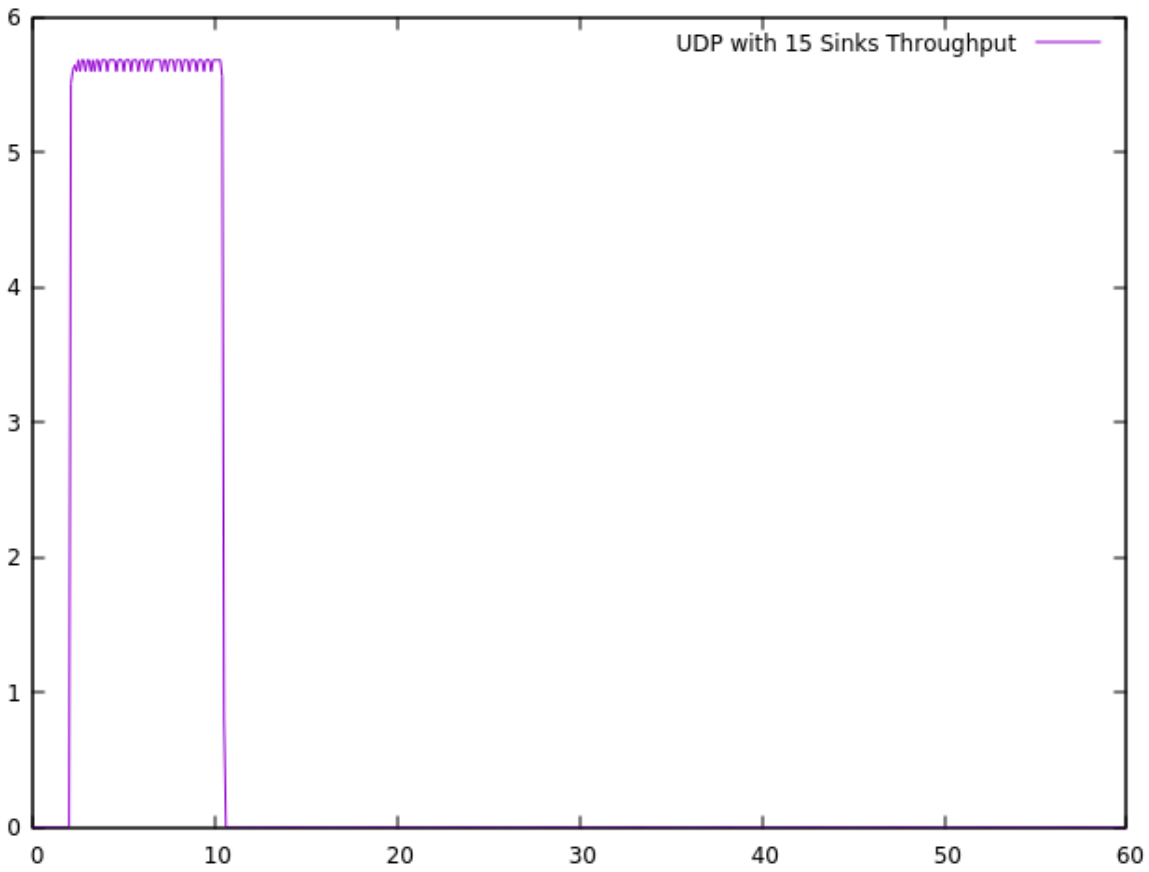

Figure 10: Total Throughput for UDP System with 1 Source and 15 Sink Nodes

The individual node throughputs were calculated as shown below:

Sink Node 1 $Throughput = 2.1Mbps$

Sink Node 2 $Throughput = 1.61Mbps$

Sink Node 3 $Throughput = 1.6Mbps$

Sink Node 4 $Throughput = 0.517Mbps$

Sink Node 5 $Throughput = 6.03Mbps$

Sink Node 6 $Throughput = 0.085Mbps$

Sink Node 7 $Throughput = 0Mbps$

Sink Node 8 $Throughput = 5.73Mbps$

Sink Node 9 $Throughput = 3.04Mbps$

Sink Node 10 $Throughput = 0Mbps$

Sink Node 11 $Throughput = 5.77Mbps$

Sink Node 12 $Throughput = 0Mbps$

Sink Node 13 $Throughput = 5.77Mbps$

Sink Node 14 $Throughput = 0Mbps$

Sink Node 15 $Throughput = 0Mbps$

3.2.2 Comparison of Results

5 Sink The highest throughput experienced in the TCP system was approximately 4.7Mbps, with the lowest throughput experienced being 0.00004Mbps. The highest throughput in the UDP system was approximately 5.6Mbps, with the lowest throughput experienced being 0Mbps. While the highest throughput experienced was in the UDP system, there were three nodes in the UDP system which had zero throughput. The TCP system experienced better throughput per node than the UDP system.

The total throughput of the TCP system was approximately 5.7Mbps, giving an average throughput of 0.95Mbps. The total throughput of the UDP system was approximately 6.2Mbps, giving an average throughput of 1.03Mbps.

10 Sink The highest throughput experienced in the TCP system was approximately 4.72Mbps, with the lowest throughput experienced being 0.00004Mbps. The highest throughput in the UDP system was approximately 5.9Mbps, with the lowest throughput experienced being 0Mbps. The maximum experienced throughput in UDP was 1.18Mbps higher than that in the TCP system. Again, the TCP system had more consistent throughput, with the UDP system having 0Mbps on three nodes.

The total throughput of the TCP system was approximately 5.05Mbps, giving an average throughput of 0.46Mbps. The total throughput of the UDP system was approximately 17.5Mbps, giving an average of throughput of 1.59Mbps.

15 Sink The highest throughput experienced in the TCP system was approximately 4.7Mbps, with the lowest throughput experienced being 0.00004Mbps. The highest throughput in the UDP system was approximately 6.03Mbps, with the lowest throughput experienced being 0Mbps. The maximum throughput in the UDP system was 1.31Mbps higher than in the TCP system, however, the TCP system experienced more consistent throughput, with the UDP system having 0Mbps on five nodes.

The total throughput of the TCP system was approximately 5.4Mbps, giving an average throughput of 0.34Mbps. The total throughput of the UDP system was approximately 32.3Mbps, giving an average throughput of 2.02Mbps.

It can be seen that the number of nodes in a system causes more nodes in the UDP system to receive no throughput, while in the TCP system, increases the number of nodes receiving very low throughput. As can be seen from the average throughput per node, the TCP system reduces as more nodes are added. The UDP system average throughput increases as more nodes are added, however, this is due to certain nodes having higher throughput, however, there is less consistency to the throughput, as mentioned above, with an increasing number of nodes receiving no throughput.

Conclusion

Within the single node systems, it can be seen that there is higher throughput experienced within the UDP system. The lower throughput of the TCP system may be due to the acknowledgements which must be transmit from the sink to the source node. There are also a number of factors, such as distance or different forms of interference which cause the simulated throughput to fall below the advertised throughputs for 802.11ac.

From this results shown above, it is obvious that the number of sink nodes has an impact on the throughput of the system. Increasing the number of nodes in the TCP system reduces the average throughput, however the total throughput of the system remains

relatively constant. Within the UDP system, both the total and average throughput increases, however, there is no consistency to which nodes receive data.

The lack of throughput for certain nodes of the TCP and UDP systems may be partially explained by the collision avoidance utilised in the 802.11ac standard. This may be blocking transmission to nodes while transmitting to others.

Appendix

5.1 Code

The code used in the completion of this assignment can be found in the .zip file containing this report, or at the following address: https://github.com/mLenehanDCU/EE452/tree/master/Assignments/Code/Assignment2

5.2 Throughput Calculations

All values used in the following throughput calculations can be found in the .zip file containing this report, or at the following address: https://github.com/mLenehanDCU/EE452/tree/master/Assignments/Data/Assignment2

5.2.1 TCP

TCP 1 Node

Source Node

$$rxBytes = 221680$$

$$timeLastRxPacket = 10\,247\,203\,463 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,016\,057\,160 \times 10^{-9}$$

$$Throughput = 215\,454.8(bps)$$

Sink Node

$$rxBytes = 4821824$$

$$timeLastRxPacket = 10\,245\,900\,448 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,009\,436\,112 \times 10^{-9}$$

$$Throughput = 4\,683\,392.1(bps)$$

TCP 5 Nodes

Sink Node 1

$$rxBytes = 147636$$

$$timeLastRxPacket = 18\,011\,184\,774 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,014\,264\,069 \times 10^{-9}$$

$$Throughput = 73\,832.209\,45(bps)$$

Sink Node 2

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,152\,003 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,005\,074\,053 \times 10^{-9}$$

$$Throughput = 39.826\,578\,41(bps)$$

Sink Node 3

$$rxBytes = 4862432$$

$$timeLastRxPacket = 10\,241\,978\,392 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,007\,382\,021 \times 10^{-9}$$

$$Throughput = 4\,723\,905.611(bps)$$

Sink Node 4

$$rxBytes = 145340$$

$$timeLastRxPacket = 17\,944\,828\,787 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,024\,382\,139 \times 10^{-9}$$

$$Throughput = 73\,033.126\,88(bps)$$

Sink Node 5

$$rxBytes = 152056$$

$$timeLastRxPacket = 17\,765\,945\,047 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,003\,436\,028 \times 10^{-9}$$

$$Throughput = 77\,173.500\,65(bps)$$

Source Node

$$rxBytes1 = 6868$$

$$rxBytes2 = 0$$

$$rxBytes3 = 223552$$

$$rxBytes4 = 6712$$

$$rxBytes5 = 6556$$

$$timeLastRxPacket1 = 17\,747\,449\,044 \times 10^{-9}$$

$$timeLastRxPacket2 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket3 = 10\,243\,938\,080 \times 10^{-9}$$

$$timeLastRxPacket4 = 18\,010\,222\,772 \times 10^{-9}$$

$$timeLastRxPacket5 = 17\,924\,059\,780 \times 10^{-9}$$

$$timeFirstRxPacket1 = 2\,014\,752\,080 \times 10^{-9}$$

$$timeFirstRxPacket2 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket3 = 2\,008\,976\,030 \times 10^{-9}$$

$$timeFirstRxPacket4 = 2\,023\,724\,124 \times 10^{-9}$$

$$timeFirstRxPacket5 = 2\,046\,052\,184 \times 10^{-9}$$

$$Throughput1 = 3492.344\,645(bps)$$

$$Throughput2 = 0(bps)$$

$$Throughput3 = 217\,173.557(bps)$$

$$Throughput4 = 3358.8343(bps)$$

$$Throughput5 = 3303.185\,219(bps)$$

TCP 10 Node

Sink Node 1

$$rxBytes = 152056$$

$$timeLastRxPacket = 18\,274\,599\,526 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,011\,658\,214 \times 10^{-9}$$

$$Throughput = 74\,798.769\,59(bps)$$

Sink Node 2

$$rxBytes = 4850024$$

$$timeLastRxPacket = 10\,233\,756\,865 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,005\,641\,069 \times 10^{-9}$$

$$Throughput = 4\,715\,562.221(bps)$$

Sink Node 3

$$rxBytes = 148724$$

$$timeLastRxPacket = 18\,495\,646\,619 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,016\,226\,308 \times 10^{-9}$$

$$Throughput = 72\,198.656\,11(bps)$$

Sink Node 4

$$rxBytes = 148724$$

$$timeLastRxPacket = 18\,494\,739\,612 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,014\,044\,268 \times 10^{-9}$$

$$Throughput = 72\,193.070\,45(bps)$$

Sink Node 5

$$rxBytes = 148724$$

$$timeLastRxPacket = 18\,473\,200\,603 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,016\,947\,323 \times 10^{-9}$$

$$Throughput = 72\,300.297\,02(bps)$$

Sink Node 6

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,152\,004 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,012\,460\,229 \times 10^{-9}$$

$$Throughput = 39.833\,117\,22(bps)$$

Sink Node 7

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,434\,012 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,003\,436\,028 \times 10^{-9}$$

$$Throughput = 39.824\,879(bps)$$

Sink Node 8

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,707\,021 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,014\,801\,288 \times 10^{-9}$$

$$Throughput = 39.834\,698\,69(bps)$$

Sink Node 9

$$rxBytes = 148724$$

$$timeLastRxPacket = 18\,171\,333\,499 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,017\,596\,342 \times 10^{-9}$$

$$Throughput = 90\,452.772\,91(bps)$$

Sink Node 10

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,001\,061\,031 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,021\,719\,407 \times 10^{-9}$$

$$Throughput = 39.840\,512\,01(bps)$$

Source Node

$$rxBytes1 = 0$$

$$rxBytes2 = 222928$$

$$rxBytes3 = 6868$$

$$rxBytes4 = 0$$

$$rxBytes5 = 6712$$

$$rxBytes6 = 0$$

$$rxBytes7 = 6712$$

$$rxBytes8 = 6712$$

$$rxBytes9 = 6712$$

$$rxBytes10 = 0$$

$$timeLastRxPacket1 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket2 = 10\,234\,208\,867 \times 10^{-9}$$

$$timeLastRxPacket3 = 18\,170\,949\,495 \times 10^{-9}$$

$$timeLastRxPacket4 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket5 = 18\,473\,434\,611 \times 10^{-9}$$

$$timeLastRxPacket6 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket7 = 18\,494\,394\,607 \times 10^{-9}$$

$$timeLastRxPacket8 = 18\,457\,199\,585 \times 10^{-9}$$

$$timeLastRxPacket9 = 18\,106\,577\,503 \times 10^{-9}$$

$$timeLastRxPacket10 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket1 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket2 = 2\,015\,379\,299 \times 10^{-9}$$

$$timeFirstRxPacket3 = 2\,022\,198\,418 \times 10^{-9}$$

$$timeFirstRxPacket4 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket5 = 2\,028\,614\,480 \times 10^{-9}$$

$$timeFirstRxPacket6 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket7 = 2\,028\,287\,470 \times 10^{-9}$$

$$timeFirstRxPacket8 = 2\,026\,280\,459 \times 10^{-9}$$

$$timeFirstRxPacket9 = 2\,025\,492\,447 \times 10^{-9}$$

$$timeFirstRxPacket10 = 0.0 \times 10^{-9}$$

$$Throughput1 = 0(bps)$$

$$Throughput2 = 216\,992.4544(bps)$$

$$Throughput3 = 3402.368\,377(bps)$$

$$Throughput4 = 0(bps)$$

$$Throughput5 = 3265.222\,701(bps)$$

$$Throughput6 = 0(bps)$$

$$Throughput7 = 3261.001\,496(bps)$$

$$Throughput8 = 3267.985\,168(bps)$$

$$Throughput9 = 3339.078\,166(bps)$$

$$Throughput10 = 0(bps)$$

TCP 15 Nodes

Sink Node 1

$$rxBytes = 152056$$

$$timeLastRxPacket = 18\,861\,420\,105 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,021\,949\,484 \times 10^{-9}$$

$$Throughput = 72\,237.900\,31(bps)$$

Sink Node 2

$$rxBytes = 4775576$$

$$timeLastRxPacket = 10\,215\,291\,769 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,003\,983\,077 \times 10^{-9}$$

$$Throughput = 4\,652\,681.982(bps)$$

Sink Node 3

$$rxBytes = 148724$$

$$timeLastRxPacket = 18\,341\,489\,046 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,028\,024\,648 \times 10^{-9}$$

$$Throughput = 72\,933.128\,79(bps)$$

Sink Node 4

$$rxBytes = 152056$$

$$timeLastRxPacket = 20\,136\,770\,746 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,022\,823\,502 \times 10^{-9}$$

$$Throughput = 67\,155.324\,22(bps)$$

Sink Node 5

$$rxBytes = 148764$$

$$timeLastRxPacket = 20\,249\,145\,860 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,026\,770\,625 \times 10^{-9}$$

$$Throughput = 65\,310.475\,98(bps)$$

Sink Node 6

$$rxBytes = 157696$$

$$timeLastRxPacket = 20\,137\,902\,760 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,010\,512\,304 \times 10^{-9}$$

$$Throughput = 69\,594.573\,09(bps)$$

Sink Node 7

$$rxBytes = 148724$$

$$timeLastRxPacket = 20\,061\,665\,667 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,029\,296\,673 \times 10^{-9}$$

$$Throughput = 65\,980.903\,58(bps)$$

Sink Node 8

$$rxBytes = 157736$$

$$timeLastRxPacket = 20\,283\,688\,899 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,017\,822\,384 \times 10^{-9}$$

$$Throughput = 69\,084.486\,03(bps)$$

Sink Node 9

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,152\,005 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,001\,436\,035 \times 10^{-9}$$

$$Throughput = 39.823\,358\,54(bps)$$

Sink Node 10

$$rxBytes = 157696$$

$$timeLastRxPacket = 20\,337\,504\,967 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,018\,543\,404 \times 10^{-9}$$

$$Throughput = 68\,866.785\,69(bps)$$

Sink Node 11

$$rxBytes = 157736$$

$$timeLastRxPacket = 20\,336\,945\,955 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,012\,159\,344 \times 10^{-9}$$

$$Throughput = 68\,862.357\,13(bps)$$

Sink Node 12

$$rxBytes = 148764$$

$$timeLastRxPacket = 20\,203\,527\,820 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,024\,756\,568 \times 10^{-9}$$

$$Throughput = 65\,467.131\,06(bps)$$

Sink Node 13

$$rxBytes = 224$$

$$timeLastRxPacket = 47\,000\,425\,017 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,014\,909\,455 \times 10^{-9}$$

$$Throughput = 39.835\,044\,18(bps)$$

Sink Node 14

$$rxBytes = 152056$$

$$timeLastRxPacket = 20\,338\,130\,978 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,021\,273\,468 \times 10^{-9}$$

$$Throughput = 6641.391\,76(bps)$$

Sink Node 15

$$rxBytes = 157696$$

$$timeLastRxPacket = 20\,136\,002\,728 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,014\,042\,420 \times 10^{-9}$$

$$Throughput = 69\,615.426\,73(bps)$$

Source Node

$$rxBytes1 = 0$$

$$rxBytes2 = 219548$$

$$rxBytes3 = 7128$$

$$rxBytes4 = 7180$$

$$rxBytes5 = 7128$$

$$rxBytes6 = 0$$

$$rxBytes7 = 7180$$

$$rxBytes8 = 7128$$

$$rxBytes9 = 6868$$

$$rxBytes10 = 6868$$

$$rxBytes11 = 6868$$

$$rxBytes12 = 6764$$

$$rxBytes13 = 6764$$

$$rxBytes14 = 6712$$

$$rxBytes15 = 6712$$

$$timeLastRxPacket1 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket2 = 10\,215\,826\,771 \times 10^{-9}$$

$$timeLastRxPacket3 = 19\,940\,296\,569 \times 10^{-9}$$

$$timeLastRxPacket4 = 20\,311\,571\,935 \times 10^{-9}$$

$$timeLastRxPacket5 = 19\,868\,978\,577 \times 10^{-9}$$

$$timeLastRxPacket6 = 0.0 \times 10^{-9}$$

$$timeLastRxPacket7 = 20\,169\,734\,778 \times 10^{-9}$$

$$timeLastRxPacket8 = 20\,328\,542\,941 \times 10^{-9}$$

$$timeLastRxPacket9 = 20\,336\,256\,948 \times 10^{-9}$$

$$timeLastRxPacket10 = 18\,756\,475\,991 \times 10^{-9}$$

$$timeLastRxPacket11 = 19\,879\,104\,578 \times 10^{-9}$$

$$timeLastRxPacket12 = 20\,061\,182\,664 \times 10^{-9}$$

$$timeLastRxPacket13 = 20\,113\,158\,689 \times 10^{-9}$$

$$timeLastRxPacket14 = 18\,245\,016\,940 \times 10^{-9}$$

$$timeLastRxPacket15 = 19\,735\,197\,626 \times 10^{-9}$$

$$timeFirstRxPacket1 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket2 = 2\,023\,320\,513 \times 10^{-9}$$

$$timeFirstRxPacket3 = 2\,025\,885\,599 \times 10^{-9}$$

$$timeFirstRxPacket4 = 2\,025\,361\,591 \times 10^{-9}$$

$$timeFirstRxPacket5 = 2\,023\,889\,541 \times 10^{-9}$$

$$timeFirstRxPacket6 = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket7 = 2\,028\,566\,661 \times 10^{-9}$$

$$timeFirstRxPacket8 = 2\,027\,249\,639 \times 10^{-9}$$

$$timeFirstRxPacket9 = 2\,036\,703\,733 \times 10^{-9}$$

$$timeFirstRxPacket10 = 2\,033\,011\,684 \times 10^{-9}$$

$$timeFirstRxPacket11 = 2\,040\,222\,769 \times 10^{-9}$$

$$timeFirstRxPacket12 = 2\,052\,118\,856 \times 10^{-9}$$

$$timeFirstRxPacket13 = 2\,051\,531\,842 \times 10^{-9}$$

$$timeFirstRxPacket14 = 2\,042\,882\,780 \times 10^{-9}$$

$$timeFirstRxPacket15 = 2\,056\,135\,886 \times 10^{-9}$$

$$Throughput1 = 0(bps)$$

$$Throughput2 = 214\,389.0947(bps)$$

$$Throughput3 = 3183.135\,638(bps)$$

$$Throughput4 = 3141.164\,786(bps)$$

$$Throughput5 = 3195.501\,008(bps)$$

$$Throughput6 = 0(bps)$$

$$Throughput7 = 3166.279\,02(bps)$$

$$Throughput8 = 3115.845\,37(bps)$$

$$Throughput9 = 3002.477\,676(bps)$$

$$Throughput10 = 3285.443\,673(bps)$$

$$Throughput11 = 3080.013\,68(bps)$$

$$Throughput12 = 3004.709\,216(bps)$$

$$Throughput13 = 2995.964\,896(bps)$$

$$Throughput14 = 3314.131\,303(bps)$$

$$Throughput15 = 3037.265\,257(bps)$$

5.2.2 UDP

UDP 1 Node

Sink Node

$$rxBytes = 5962736$$

$$timeLastRxPacket = 10\,499\,950\,632 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,011\,588\,311 \times 10^{-9}$$

$$Throughput = 5\,619\,680.946(bps)$$

UDP 5 Nodes

Sink Node 1

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 2

$$rxBytes = 5282092$$

$$timeLastRxPacket = 10\,350\,144\,473 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,830\,805\,797 \times 10^{-9}$$

$$Throughput = 5\,619\,741.02(bps)$$

Sink Node 3

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 4

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 5

$$rxBytes = 629096$$

$$timeLastRxPacket = 10\,422\,673\,852 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,005\,588\,227 \times 10^{-9}$$

$$Throughput = 597\,922.8707(bps)$$

UDP 10 Nodes

Sink Node 1

$$rxBytes = 2158704$$

$$timeLastRxPacket = 10\,345\,247\,176 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,838\,753\,882 \times 10^{-9}$$

$$Throughput = 2\,300\,625.781(bps)$$

Sink Node 2

$$rxBytes = 1696876$$

$$timeLastRxPacket = 10\,429\,399\,243 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,343\,259\,613 \times 10^{-9}$$

$$Throughput = 1\,678\,799.603(bps)$$

Sink Node 3

$$rxBytes = 1331832$$

$$timeLastRxPacket = 10\,109\,976\,026 \times 10^{-9}$$

$$timeFirstRxPacket = 3\,509\,337\,193 \times 10^{-9}$$

$$Throughput = 1\,614\,185.577(bps)$$

Sink Node 4

$$rxBytes = 368200$$

$$timeLastRxPacket = 10\,218\,544\,119 \times 10^{-9}$$

$$timeFirstRxPacket = 3\,731\,402\,303 \times 10^{-9}$$

$$Throughput = 454\,067.4589(bps)$$

Sink Node 5

$$rxBytes = 19988$$

$$timeLastRxPacket = 10\,375\,109\,541 \times 10^{-9}$$

$$timeFirstRxPacket = 10\,348\,026\,523 \times 10^{-9}$$

$$Throughput = 5\,904\,216.436(bps)$$

Sink Node 6

$$rxBytes = 98888$$

$$timeLastRxPacket = 10\,396\,974\,893 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,718\,324\,791 \times 10^{-9}$$

$$Throughput = 103\,026.4421(bps)$$

Sink Node 7

$$rxBytes = 225128$$

$$timeLastRxPacket = 2\,335\,194\,470 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,005\,588\,227 \times 10^{-9}$$

$$Throughput = 5\,464\,168.347(bps)$$

Sink Node 8

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 9

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 10

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

UDP 15 Nodes

Sink Node 1

$$rxBytes = 2045088$$

$$timeLastRxPacket = 10\,371\,878\,030 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,537\,698\,188 \times 10^{-9}$$

$$Throughput = 2\,088\,374.831(bps)$$

Sink Node 2

$$rxBytes = 1614820$$

$$timeLastRxPacket = 10\,033\,606\,514 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,022\,176\,564 \times 10^{-9}$$

$$Throughput = 1\,612\,516.128(bps)$$

Sink Node 3

$$rxBytes = 1444396$$

$$timeLastRxPacket = 10\,205\,138\,589 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,960\,660\,382 \times 10^{-9}$$

$$Throughput = 1\,595\,031.094(bps)$$

Sink Node 4

$$rxBytes = 457620$$

$$timeLastRxPacket = 10\,275\,035\,984 \times 10^{-9}$$

$$timeFirstRxPacket = 3\,193\,419\,499 \times 10^{-9}$$

$$Throughput = 516\,966.7134(bps)$$

Sink Node 5

$$rxBytes = 14728$$

$$timeLastRxPacket = 10\,394\,119\,389 \times 10^{-9}$$

$$timeFirstRxPacket = 10\,374\,566\,044 \times 10^{-9}$$

$$Throughput = 6\,025\,772.061(bps)$$

Sink Node 6

$$rxBytes = 87316$$

$$timeLastRxPacket = 10\,412\,710\,066 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,216\,792\,806 \times 10^{-9}$$

$$Throughput = 85\,228.776\,46(bps)$$

Sink Node 7

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 8

$$rxBytes = 62068$$

$$timeLastRxPacket = 2\,535\,602\,174 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,448\,903\,103 \times 10^{-9}$$

$$Throughput = 5\,727\,212.463(bps)$$

Sink Node 9

$$rxBytes = 78900$$

$$timeLastRxPacket = 2\,210\,883\,695 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,003\,588\,234 \times 10^{-9}$$

$$Throughput = 3\,044\,929.189(bps)$$

Sink Node 10

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 11

$$rxBytes = 44184$$

$$timeLastRxPacket = 2\,384\,957\,981 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,323\,677\,899 \times 10^{-9}$$

$$Throughput = 5\,768\,138.496(bps)$$

Sink Node 12

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 13

$$rxBytes = 43132$$

$$timeLastRxPacket = 2\,446\,834\,081 \times 10^{-9}$$

$$timeFirstRxPacket = 2\,387\,027\,010 \times 10^{-9}$$

$$Throughput = 5\,769\,485.016(bps)$$

Sink Node 14

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Sink Node 15

$$rxBytes = 0$$

$$timeLastRxPacket = 0.0 \times 10^{-9}$$

$$timeFirstRxPacket = 0.0 \times 10^{-9}$$

$$Throughput = 0(bps)$$

Bibliography

[1] E. H. Ong, J. Kneckt, O. Alanen, Z. Chang, T. Huovinen, and T. Nihtilä, "IEEE 802.11ac: Enhancements for very high throughput wlans," in *2011 IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*, 2011, pp. 849–853.

[2] O. Bejarano, E. W. Knightly, and M. Park, "IEEE 802.11ac: From channelization to multi-user mimo," *IEEE Communications Magazine*, vol. 51, pp. 84–90, 2013.

[3] Z. Chang, O. Alanen, T. Huovinen, T. Nihtila, E. H. Ong, J. Kneckt, and T. Ristaniemi, "Performance analysis of ieee 802.11ac dcf with hidden nodes," in *2012 ieee 75th vehicular technology conference (vtc spring)*, 2012, pp. 1–5.