




APRIL 15, 2023

PORTFOLIO 1

SIMPLEPERF

SULTAN AVTAJEV
S199219
Oslo Metropolitan University



1	INTRODUCTION	2
2	SIMPLEPERF	3
3	EXPERIMENTAL SETUP	3
4	RESULTS AND DISCUSSION	FEIL! BOKMERKE ER IKKE DEFINERT.
4.1	Network tools	5
4.2	Performance metrics	5
4.3	Test case 1: measuring bandwidth with iperf in UDP mode	5
4.3.1	Results	6
4.3.2	Discussion	6
4.4	Test case 2: link latency and throughput	7
4.4.1	Results	7
4.4.2	Discussion	8
4.5	Test case 3: path Latency and throughput	9
4.5.1	Results	10
4.5.2	Discussion	10
4.6	Test case 4: effects of multiplexing and latency	11
4.6.1	Results	11
4.6.2	Discussion	12
4.7	Test case 5: effects of parallel connections	13
4.7.1	Results	13
4.7.2	Discussion	13
5	CONCLUSIONS	14
6	REFERENCES	14

1 Introduction

The virtual network is becoming more popular as it allows administrators to create, modify and test network topologies quickly without physically building a network. However, measuring the performance of a virtual network is still challenging as it requires using different tools to measure bandwidth, latency, and throughput. In this project, we aim to measure the performance of a virtual network created using Mininet. We use different tools such as iperf, ping, and simpleperf to measure the performance and evaluate the results.

Key topic(s):

This project's main topic is measuring the performance of a virtual network. We will use different tools to measure the bandwidth, latency, and throughput of different links and paths in the virtual network.

Problem(s) that we are solving:

The virtual network is a complex system that requires proper performance measurement to optimize its performance. However, measuring the performance of a virtual network is still challenging. The problem that we are solving is to measure the performance of a virtual network using different tools and techniques.

References to the relevant work:

In this project, we will use different tools such as iperf, ping, and simpleperf to measure the performance of a virtual network. These tools are widely used in the network performance measurement domain. Iperf is a tool to measure the maximum achievable bandwidth on IP networks. Ping measures the round-trip time for packets in the network, while Simpleperf is a tool for measuring the throughput of a link.

Our approach to the solution:

We will create a virtual network using Mininet and use different tools to measure the performance of different links and paths in the network. We will use iperf in UDP mode to measure the bandwidth between different hosts in the network. We will also use ping and simpleperf to measure the latency and throughput of different links and paths in the network.

Limitations and outcomes:

Our approach has some limitations as it depends on the network topology and the number of hosts in the network. However, we aim to provide a general approach that can be applied to different virtual network topologies. The outcome of this project will be a report that summarizes the performance of the virtual network using different tools and techniques. The report will also discuss the limitations and possible solutions to improve the performance of the virtual network.

The rest of the document is organized into several sections:

2.0 Simpleperf: This section describes the implementation details of Simpleperf and the communication between the server and client.

3.0 Experimental Setup: This section describes the virtual network topology used to evaluate Simpleperf.

4.0 Performance Evaluations: This section is divided into five test cases that evaluate the performance of Simpleperf in different scenarios.

4.1 Network Tools: This subsection introduces the network tools used in the experiments, including iperf, ping, and simpleperf.

4.2 Performance Metrics: This subsection describes the performance metrics used to evaluate Simpleperf, including throughput, RTT, latency, and packet loss.

4.3 Test Case 1: Measuring Bandwidth with Iperf in UDP Mode: This subsection describes the first test case, which measures the network bandwidth using iperf in UDP mode. It presents the results and discusses their implications.

4.4 Test Case 2: Link Latency and Throughput: This subsection describes the second test case, which measures the impact of link latency on network throughput. It presents the results and discusses their implications.

4.5 Test Case 3: Path Latency and Throughput: This subsection describes the third test case, which measures the impact of path latency on network throughput. It presents the results and discusses their implications.

4.6 Test Case 4: Effects of Multiplexing and Latency: This subsection describes the fourth test case, which measures the impact of multiplexing and latency on network throughput. It presents the results and discusses their implications.

4.7 Test Case 5: Effects of Parallel Connections: This subsection describes the fifth test case, which measures the impact of parallel connections on network throughput. It presents the results and discusses their implications.

5.0 Conclusions: This section concisely describes the important results and their significance. It also discusses the limitations and shortcomings of the work.

6.0 References: This section provides a list of references used in the document (optional).

2.0 Simpleperf

Simpleperf is a command-line tool that can be run in either server or client mode. In server mode, it listens for incoming connections and records the data received from each connection. In client mode, it sends data to a specified server IP address and port number for a given duration.

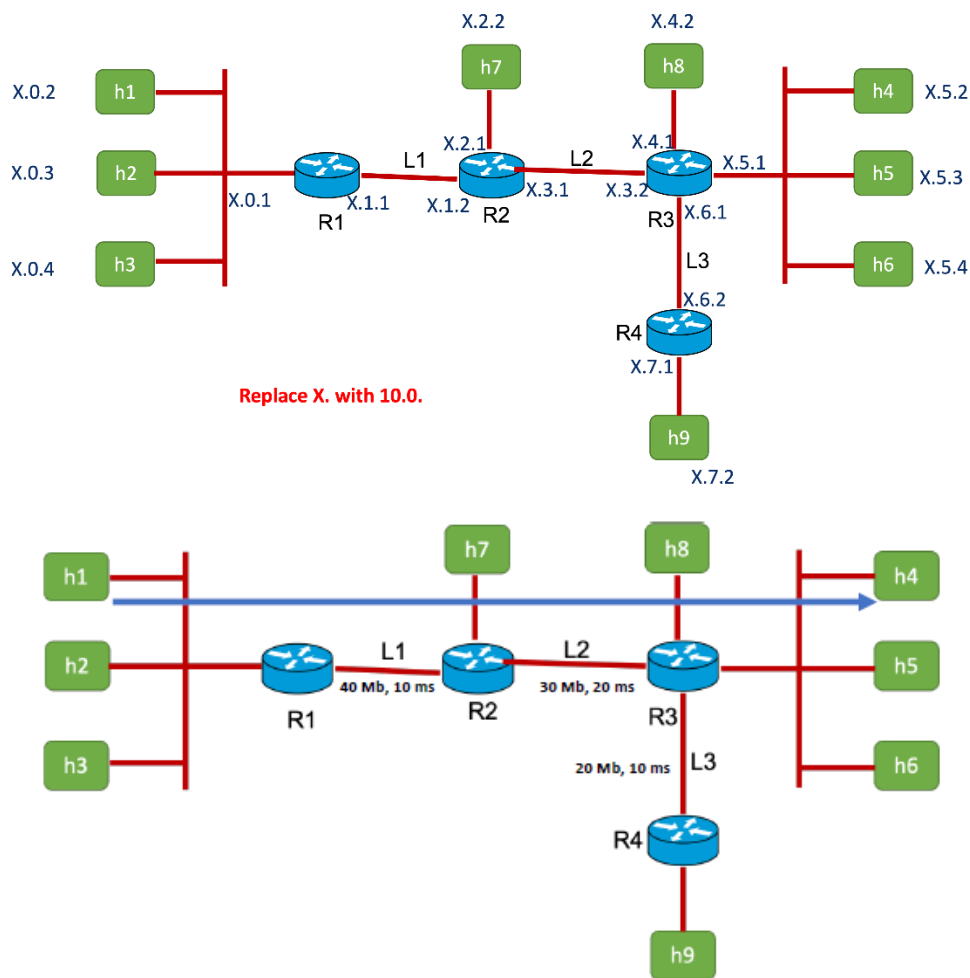
The main building blocks of Simpleperf are:

1. **Argument Parsing:** The program uses the argparse library to parse command-line arguments. The arguments specify whether to run in server or client mode, the IP address and port number to use, the duration of the test, and other options.
2. **Argument Validation:** The program validates the command-line arguments to ensure they are valid and consistent. If the arguments are invalid, it raises a ValueError.
3. **Server Mode:** If Simpleperf is run in server mode, it listens for incoming connections on a specified IP address and port number. Once a connection is established, it reads and records data from the client. Simpleperf can handle multiple connections in parallel using the select system call to monitor multiple sockets.
4. **Client Mode:** If Simpleperf is run in client mode, it sends data to a specified server IP address and port number for a given duration. Simpleperf can send data in parallel using multiple connections.
5. **Data Recording:** Simpleperf records the amount of data received by the server during a test and the time taken to receive the data. It then calculates various statistics, such as the average transfer rate, and prints the results in the specified format.
6. **Error Handling:** Simpleperf handles errors and exceptions that may occur during the program's execution. If an error occurs, it prints an error message and exits.

The communication between the server and client is based on the TCP/IP protocol. The client sends data to the server using a TCP socket, and the server receives the data using another TCP socket. The program uses the socket library to create and manage sockets. The client sends a fixed amount of data to the server, and the server records the amount of data received and the time taken to receive it. The program uses the time library to measure the time taken for data transfer. Once the test is complete, Simpleperf calculates various statistics based on the recorded data and prints the results in the specified format.

3.0 Experimental setup

The virtual network consists of 9 hosts (H1 to H9), 4 routers (R1 to R4), and 2 switches (S1 and S2). The network is divided into different subnets, and each host is assigned a unique IP address. We will examine the bandwidth and latency of the network using different test cases, including measuring the throughput with iperf in UDP mode, link latency and throughput, path latency, and throughput, effects of multiplexing and latency, and effects of parallel connections.



The network can be divided into several subnets:

- A. Subnet A** consists of three hosts (h1, h2, and h3) connected to a switch (s1) and a router (r1). All the hosts in this subnet are assigned IP addresses in the range 10.0.0.2-10.0.0.4 with a netmask of 255.255.255.0. The router r1 is assigned IP address 10.0.0.1 with the same netmask. The hosts are connected to switch s1, which is connected to the router r1 via a link.
- B. Subnet B** consists of two routers (r1 and r2) connected via a link with a bandwidth of 40 Mbps and a delay of 10ms. The IP addresses assigned to the routers are 10.0.0.1/24 and 10.0.1.2/24 for r1 and r2, respectively.
- C. Subnet C** consists of a single host (h7) connected to router r2. The host and the router are assigned IP addresses in the range 10.0.2.2-10.0.2.1 with a netmask of 255.255.255.0.
- D. Subnet D** consists of two routers (r2 and r3) connected via a link with a bandwidth of 30 Mbps and a delay of 20ms. The IP addresses assigned to the routers are 10.0.1.2/24 and 10.0.3.2/24 for r2 and r3, respectively.
- E. Subnet E** consists of three hosts (h4, h5, and h6) connected to a switch (s2) and router r3. All the hosts in this subnet are assigned IP addresses in the range 10.0.5.2-10.0.5.4 with a netmask of 255.255.255.0. The router r3 is assigned IP address 10.0.5.1 with the same netmask.
- F. Subnet F** consists of two routers (r3 and r4) connected via a link with a bandwidth of 20 Mbps and a delay of 10ms. The IP addresses assigned to the routers are 10.0.3.2/24 and 10.0.6.2/24 for r3 and r4, respectively.
- G. Subnet G** consists of a single host (h9) connected to router r4. The host and the router are assigned IP addresses in the range 10.0.7.2-10.0.7.1 with a netmask of 255.255.255.0.

Each host is assigned an IP address and a default route to reach the destination outside its subnet. The network uses the 10.0.0.0/8 IP address range.

The routers are running a Linux operating system and have IP forwarding enabled. The routers also have specific routes added to their routing tables to allow communication between subnets. The network topology also sets various parameters for each link, such as bandwidth, delay, maximum queue size, and use of the Hierarchical Token Bucket (HTB) algorithm to manage traffic. Finally, the network topology disables various offloading techniques on each host and router interface to ensure accurate measurements in the simulation.

4.0 Performance evaluations

4.1 Network tools

In this assignment, we evaluated network performance using various tools such as ping, simpleperf, and iPerf. We used a virtual network created in Mininet, consisting of 9 hosts and 4 routers, with a specific topology. The experiment aimed to measure the bandwidth and latency in the network and study the effects of multiplexing and parallel connections on network performance.

Ping: A standard networking utility used to test the reachability of a host on an Internet Protocol (IP) network and to measure the round-trip time for messages sent from the originating host to a destination computer. The experiment uses ping to measure link latency between routers and path latency and throughput between hosts.

Simpleperf: A simple network throughput measurement tool. The program sends and receives packets between a client and a server using sockets. It can run in either server mode or client mode. In the experiment, simpleperf is used to measure the throughput (bandwidth) of each link between routers and the effect of multiplexing and parallel connections on latency and throughput.

iPerf: A tool used for measuring network performance. It is particularly useful for measuring the bandwidth and jitter of network connections. In the experiment, iPerf is used in UDP mode to measure the bandwidth between different pairs of hosts.

Mininet: A network emulator used to create virtual networks for testing and research. In the experiment, it is used to create the virtual network topology to simulate a real network environment.

Linux: An open-source operating system used as the base operating system for a virtual machine. The experiment uses it as the operating system for the virtual machine running the Mininet topology.

VirtualBox: A virtualization software used to create and manage a virtual machine on my physical machine.

4.2 Performance metrics

The performance metrics we used to evaluate the simpleperf tool are throughput (measured in Mbps) and latency (measured in ms). Throughput measures the amount of data transmitted over a network link per unit of time. Latency measures the time it takes for a packet to travel from the source to the destination and back again. We use these metrics to evaluate the network's performance in terms of its ability to transfer data quickly and efficiently. We use the simpleperf tool to measure the throughput and latency of the links between routers in the network as well as the throughput and latency of paths between the different hosts in the network.

4.3 Test case 1: measuring bandwidth with iperf in UDP mode

In this test, we are using iPerf in UDP mode to measure the bandwidth between different pairs of hosts in the virtual network topology. The -b option is used to specify the bandwidth to be generated by the iPerf tool. Three separate tests are conducted between the following client-server pairs: h1-h4, h1-h9, and h7-h9. The output of each test is stored in a separate file, namely throughput_udp_iperf_h1-h4.txt, throughput_udp_iperf_h1-h9.txt, and throughput_udp_iperf_h7-h9.txt.

4.3.1 Results

Test case 1	Client Host	Client IP	Server Host	Server IP	Interval	Transfer	Throughput	Jitter	Lost/Total Datagrams
throughput_udp_iperf_h1-h4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0000-10.0191 s	34.2 MB	28.6 Mb/s	0.069 ms	2349/26753 (8.8%)
throughput_udp_iperf_h1-h9.txt	h1	10.0.0.2	h9	10.0.7.2	0.0000-10.0138 s	23.1 MB	19.3 Mb/s	0.257 ms	1370/17836 (7.7%)
throughput_udp_iperf_h7-h9.txt	h7	10.0.2.2	h9	10.0.7.2	0.0000-10.0157 s	23.0 MB	19.2 Mb/s	0.392 ms	1466/17837 (8.2%)

4.3.2 Discussion

To measure the bandwidth with iPerf in UDP mode, we choose an appropriate rate (X) based on the links' bandwidth in the given network topology. In this case, we have three tests with the following client-server pairs and the bottleneck bandwidth for each path:

H1 - H4: The bottleneck link is between R2 and R3 with a bandwidth of 30 Mbps.

H1 - H9: The bottleneck link is between R3 and R4 with a bandwidth of 20 Mbps.

H7 - H9: The bottleneck link is between R3 and R4 with a bandwidth of 20 Mbps.

We should start with a value slightly higher than the bottleneck bandwidth for each test to accurately measure the available bandwidth. So, for example, we can start with a rate of 35 Mbps for the H1-H4 test, 25 Mbps for the H1-H9 test, and 25 Mbps for the H7-H9 test. Then based on the results, we can adjust these rates depending on the actual network conditions and the results we get from the initial tests.

If we don't know anything about the network topology and we need to use iPerf in UDP mode to estimate the bandwidth, we can follow these steps:

1. Start with a relatively low rate (e.g., 10 Mbps) and run the iPerf test.
2. Gradually increase the rate (e.g., by 10 Mbps increments) and repeat the test each time.
3. Continue this process until we observe packet loss, an increase in latency, or a plateau in throughput, indicating that the network is reaching its limit.

This approach might not be practical in a real-world scenario because finding the optimal rate can take a long time, especially in large networks with varying bandwidths. Moreover, this trial-and-error method may temporarily consume a significant amount of the available network resources, potentially affecting other users and applications on the network. In practice, it's better to have some knowledge about the network topology and link capacities before conducting bandwidth tests.

Based on the network topology and the expected bottleneck bandwidth for each client-server pair, we expected the following results:

1. **H1 - H4:** We expected the bandwidth to be close to the bottleneck link between R2 and R3, which is 30 Mbps.
2. **H1 - H9:** We expected the bandwidth to be close to the bottleneck link between R3 and R4, which is 20 Mbps.
3. **H7 - H9:** We expected the bandwidth to be close to the bottleneck link between R3 and R4, which is 20 Mbps.

However, the actual iPerf test results showed the following:

1. **H1 - H4:** The measured bandwidth was 28.6 Mbps, close to the expected 30 Mbps.
2. **H1 - H9:** The measured bandwidth was 19.3 Mbps, close to the expected 20 Mbps.
3. **H7 - H9:** The measured bandwidth was 19.2 Mbps, which is also close to the expected 20 Mbps.

The discrepancy between the expected and measured bandwidth for the test could be due to various factors, such as:

1. **Network congestion:** Other traffic on the network could be consuming some of the available bandwidth, leading to lower measured results.
2. **Packet loss:** The reported packet loss for H1-H4 was 8.8%, H1-H9 was 7.7%, and H7-H9 was 8.2%, which could have contributed to the lower measured bandwidth. There isn't a specific percentage of data loss that is universally satisfactory, as it largely depends on the specific application or use case. Different applications have varying tolerance levels for data loss.
Real-time applications (e.g., voice or video calls) are sensitive to data loss, as even a small percentage of loss can lead to noticeable degradation in call quality. Generally, a data loss percentage below 1% is considered satisfactory for these applications.
Streaming applications (e.g., video streaming or online gaming) can tolerate some data loss, as they usually have buffering mechanisms in place. However, high data loss can still cause noticeable issues like buffering, stuttering, or lag. Therefore, a data loss percentage below 2% is usually acceptable for these applications.
Non-real-time applications (e.g., file transfers, email) can generally tolerate higher levels of data loss, as they typically use protocols that can retransmit lost data. However, a high data loss percentage can still result in slower transfer speeds and longer completion times. Therefore, a data loss percentage below 5% is typically acceptable for these applications.
3. **Configuration or measurement errors:** Potential inaccuracies in the test setup or data collection might have influenced the results.

Overall, the measured bandwidths were close to the expected values, indicating that the tests accurately represented the available bandwidth for these client-server pairs.

4.4 Test case 2: link latency and throughput

In this test, we are measuring the impact of link latency on network throughput. To conduct this test, we first measure the round-trip time (RTT) and bandwidth of each of the three individual links (L1-L3) between routers using ping and Simpleperf tools.

We run ping with 25 packets on each link and store the measurement output in a file called latency_L#.txt, where # represents the link number from the topology diagram. This allows us to measure the RTT of each link and identify any latency issues.

We also run Simpleperf for 25 seconds on each link and store the measurement output in a file called throughput_L#.txt, where # represents the link number from the topology diagram. This allows us to measure the network throughput of each link and determine if link latency impacts network performance.

4.4.1 Results

Test 2 latency	Link	"Client Host"	Client IP	"Server Host"	Server IP	Interval	RTT min	RTT avg	RTT max
latency_L1.txt	L1	R1	10.0.1.1	R2	10.0.1.2	0.0 - 24.027 s	20.136 ms	21.164 ms	31.216 ms
latency_L2.txt	L2	R2	10.0.3.1	R3	10.0.3.2	0.0 - 24.036 s	40.118 ms	40.627 ms	41.259 ms
latency_L3.txt	L3	R3	10.0.6.1	R4	10.0.6.2	0.0 - 24.027 s	20.232 ms	20.969 ms	28.788 ms

Test 2 throughput	Link	"Client Host"	Client IP	"Server Host"	Server IP	Interval	Transfer	Throughput
throughput_L1.txt	L1	R1	10.0.1.1	R2	10.0.1.2	0.0 - 25.1 s	119.6 MB	38.1 Mbps
throughput_L2.txt	L2	R2	10.0.3.1	R3	10.0.3.2	0.0 - 25.0 s	89.7 MB	28.7 Mbps
throughput_L3.txt	L3	R3	10.0.6.1	R4	10.0.6.2	0.0 - 25.0 s	57.6 MB	18.4 Mbps

4.4.2 Discussion

Based on the given network topology, we would expect the following characteristics for each link:

Link L1 (between R1 and R2):

- Bandwidth: 40 Mbps
- Latency: 20 ms

Link L2 (between R2 and R3):

- Bandwidth: 30 Mbps
- Latency: 40 ms

Link L3 (between R3 and R4):

- Bandwidth: 20 Mbps
- Latency: 20 ms

Now, we compare these expectations to the results obtained from the performance evaluation:

Link L1:

- Throughput: 38.1 Mbps (slightly lower than the expected 40 Mbps)
- RTT Average: 21.164 ms (slightly higher than the expected 20 ms)

Link L2:

- Throughput: 28.7 Mbps (slightly lower than the expected 30 Mbps)
- RTT Average: 40.627 ms (slightly higher than the expected 40 ms)

Link L3:

- Throughput: 18.4 Mbps (slightly lower than the expected 20 Mbps)
- RTT Average: 20.969 ms (slightly higher than the expected 20 ms)

The throughput results for each link are slightly lower than the expected values. However, it is important to note that the actual throughput in real-world scenarios might not always match the theoretical link capacities. This could be due to various factors, including:

1. **Network congestion:** When multiple devices share a link, the available bandwidth is divided among them. If there is high network usage or many devices competing for the same link, the throughput for each device may be reduced.
2. **Packet overhead:** Network protocols often require adding header and trailer information to each data packet. This overhead consumes a portion of the available bandwidth, which can reduce the effective throughput.
3. **Retransmissions:** If data packets are lost or corrupted during transmission, they need to be retransmitted. These retransmissions consume additional bandwidth and can reduce the overall throughput.
4. **Latency and jitter:** The time it takes for a packet to travel from the sender to the receiver (latency) and the variation in latency (jitter) can impact the perceived throughput. High latency and jitter can cause delays in receiving data, reducing the effective throughput.
5. **Network equipment limitations:** Routers, switches, and other network equipment can have limitations in terms of their processing capabilities and the speed at which they can forward packets. These limitations can create bottlenecks that reduce effective throughput.
6. **Transmission errors:** Noise, interference, or signal degradation can cause transmission errors, which may require retransmissions or result in reduced throughput.
7. **Quality of Service (QoS) settings:** In some networks, administrators may prioritize certain types of traffic, such as VoIP or video streaming, over others. This prioritization can reduce the available bandwidth for lower-priority traffic, leading to a reduced throughput.

The latency results align with the expectations based on the network topology. Link L2 has the highest RTT. Links L1 and L3 have lower and relatively similar RTTs.

While the latency results in our tests align with expectations based on the network topology, there can still be discrepancies in latency due to various factors, such as:

1. **Routing and path length:** The physical distance between devices and the number of intermediate devices (e.g., routers and switches) can influence the latency. Longer paths or more complex routes can result in higher latency.
2. **Network congestion:** When a network link is heavily utilized or congested, it can cause delays in processing and forwarding packets, leading to increased latency.
3. **Buffering and queuing:** Network devices like routers and switches may buffer and queue packets when handling traffic. If there is heavy traffic or congestion, these devices may need to store and process packets in a queue, which can increase latency.
4. **Processing delays:** Network devices have finite processing capabilities, and the time they take to process and forward packets contributes to latency. Higher processing times in devices along the path can result in increased latency.
5. **Transmission medium:** Different transmission mediums, such as copper cables, fiber-optic cables, or wireless connections, can have varying propagation speeds and signal attenuation characteristics. These factors can influence the latency of a link.
6. **Link error rates:** High error rates on a link can result in packet loss or corruption, requiring retransmissions and increasing latency.
7. **Variation in load:** The load on network devices and links can vary over time, leading to fluctuations in latency as devices handle different levels of traffic.

Overall, the performance evaluation results are close to what we would expect based on the given network topology, with minor throughput discrepancies. These discrepancies are common in real-world networks and can be attributed to various factors affecting network performance.

4.5 Test case 3: path Latency and throughput

In this test, we evaluate path latency's impact on network throughput. We conduct this test by measuring the round-trip time (RTT) and bandwidth of the network path between hosts using the ping and Simpleperf tools.

We perform three tests where a client host communicates with a server host through different network paths. In each test, we first predict the expected latency and throughput of the network path. Then we measure the actual latency and throughput between the client and server hosts using ping and Simpleperf tools with the same parameters as above (25 packets / 25 seconds). Finally, we store the measurement output in `latency_h#-h#.txt` and `throughput_h#-h#.txt`, where # represents the host numbers.

In the first test, we measure the latency and throughput between h1 and h4 when h1 communicates with h4. Next, we predict the expected latency and throughput of the path between the hosts based on the topology diagram. We then measure the actual latency and throughput using ping and Simpleperf tools and store the results in files. Finally, we add the average RTT and measured throughput in our report and explain the results.

In the second test, we measure the latency and throughput between h7 and h9 when h7 communicates with h9. We predict the expected latency and throughput of the path between the hosts based on the topology diagram. We then measure the actual latency and throughput using ping and Simpleperf tools and store the results in files. Finally, we add the average RTT and measured throughput in our report and explain the results.

In the third test, we measure the latency and throughput between h1 and h9 when h1 communicates with h9. We predict the expected latency and throughput of the path between the hosts based on the topology diagram. We then measure the actual latency and throughput using ping and Simpleperf tools and store the

results in files. Finally, we add the average RTT and measured throughput in our report and explain the results.

By comparing the expected and measured latency and throughput, we can identify any issues in the network path that may be impacting network performance. This test helps us understand the impact of path latency on network performance and can aid in optimizing network performance by identifying and resolving any latency issues.

4.5.1 Results

Test 3 latency	Client Host	Client IP	Server Host	Server IP	Interval	RTT min	RTT avg	RTT max
latency_h1-h4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 24.036 s	60.476 ms	62.144 ms	68.552 ms
latency_h7-h9.txt	h7	10.0.2.2	h9	10.0.7.2	0.0 - 24.054 s	60.308 ms	61.777 ms	63.045 ms
latency_h1-h9.txt	h1	10.0.0.2	h9	10.0.7.2	0.0 - 24.034 s	80.725 ms	82.995 ms	92.326 ms

Test 3 throughput	Client Host	Client IP	Server Host	Server IP	Interval	Transfer	Throughput
throughput_h1-h4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.1 s	88.4 MB	28.1 Mbps
throughput_h7-h9.txt	h7	10.0.2.2	h9	10.0.7.2	0.0 - 25.2 s	52.2 MB	16.6 Mbps
throughput_h1-h9.txt	h1	10.0.0.2	h9	10.0.7.2	0.0 - 25.1 s	53.5 MB	17.0 Mbps

4.5.2 Discussion

Based on the network topology, the expected latency and throughput for each path are as follows:

1. **H1-H4:** The expected latency is the sum of the latencies of L1 and L2. The expected throughput is limited by the bottleneck link, which is L2 (30 Mbps).
2. **H7-H9:** The expected latency is the sum of the latencies of L2 and L3. The bottleneck link, which is L3 (20 Mbps), limits the expected throughput.
3. **H1-H9:** The expected latency is the sum of L1, L2, and L3 latencies. The bottleneck link, which is L3 (20 Mbps), limits the expected throughput.

Now, we compare the expected values with the measured results:

1. **H1-H4:**
 - Expected latency: L1 (20 ms) + L2 (40 ms) = 60 ms
 - Measured latency: 62.144 ms (close to the expected value)
 - Expected throughput: 30 Mbps (limited by L2)
 - Measured throughput: 28.1 Mbps (slightly lower than expected)
2. **H7-H9:**
 - Expected latency: L2 (40 ms) + L3 (20 ms) = 60 ms
 - Measured latency: 61.777 ms (close to the expected value)
 - Expected throughput: 20 Mbps (limited by L3)
 - Measured throughput: 16.6 Mbps (lower than expected)
3. **H1-H9:**
 - Expected latency: L1 (20 ms) + L2 (40 ms) + L3 (20 ms) = 80 ms
 - Measured latency: 82.995 ms (close to the expected value)
 - Expected throughput: 20 Mbps (limited by L3)
 - Measured throughput: 17.0 Mbps (lower than expected)

The latency measurements for all paths are close to the expected values, which indicates that the network is performing as expected in terms of latency. However, the measured throughput values are lower than the expected values. As previously discussed, this can be due to various factors such as network congestion,

packet overhead, or other traffic conditions. It's important to note that in real-world scenarios, actual throughput might not always match the theoretical link capacities due to these factors.

4.6 Test case 4: effects of multiplexing and latency

In this test, we are evaluating the effects of multiplexing on network performance. Multiplexing refers to the process of sharing network resources among multiple network connections. In this test, we simulate multiple hosts communicating simultaneously and measure the latency and throughput of each network connection using ping and Simpleperf tools.

We perform four experiments where we simulate different combinations of hosts communicating simultaneously. In each experiment, we use ping and Simpleperf tools to measure the latency and throughput of each network connection when multiple hosts are communicating simultaneously. We store the output of the measurement in files called `throughput_h#-h#-#.txt` and `latency_h#-h#-#.txt`, where # represents the host numbers and the experiment number.

In the first experiment, we simulate two pairs of hosts (h1-h4 and h2-h5) communicating simultaneously using Simpleperf. Next, we use ping and Simpleperf tools to measure the latency and throughput of each network connection and store the results in files. Finally, we add each pair's average RTT and measured throughput in the results section and explain the results.

In the second experiment, we simultaneously simulate three pairs of hosts (h1-h4, h2-h5, and h3-h6) communicating. We measure the latency and throughput of each network connection using ping and Simpleperf tools and store the results in files. We discuss the results in the discussion section.

In the third and fourth experiments, we simulate h1-h4 and h7-h9 and h1-h4 and h8-h9 communicating simultaneously. We measure the latency and throughput of each network connection using ping and Simpleperf tools and store the results in files. We add each pair's average RTT and measured throughput in the results section and explain the results.

By simulating multiple hosts communicating simultaneously and measuring the latency and throughput of each network connection, we can identify any issues with multiplexing that may be impacting network performance. This test helps us understand the effects of multiplexing on network performance and can aid in optimizing network performance by identifying and resolving any issues with multiplexing.

4.6.1 Results

Test 4 latency	Client Host	Client IP	Server Host	Server IP	Interval	RTT min	RTT avg	RTT max
latency_h1-h4-1.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 24.033 s	61.305 ms	68.980 ms	85.417 ms
latency_h2-h5-1.txt	h2	10.0.0.3	h5	10.0.5.3	0.0 - 24.020 s	60.911 ms	72.980 ms	101.937 ms
latency_h1-h4-2.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 24.031 s	61.020 ms	69.219 ms	85.039 ms
latency_h2-h5-2.txt	h2	10.0.0.3	h5	10.0.5.3	0.0 - 24.035 s	60.189 ms	70.274 ms	105.818 ms
latency_h3-h6-2.txt	h3	10.0.0.4	h6	10.0.5.4	0.0 - 24.009 s	61.047 ms	70.281 ms	87.763 ms
latency_h1-h4-3.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 24.036 s	60.223 ms	67.257 ms	77.635 ms
latency_h7-h9-3.txt	h7	10.0.2.2	h9	10.0.7.2	0.0 - 24.034 s	60.191 ms	68.840 ms	79.809 ms
latency_h1-h4-4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 24.034 s	60.290 ms	69.181 ms	87.216 ms
latency_h8-h9-4.txt	h8	10.0.4.2	h9	10.0.7.2	0.0 - 24.056 s	20.082 ms	23.680 ms	29.402 ms

Test 4 throughput	Client Host	Client IP	Server Host	Server IP	Interval	Transfer	Throughput
throughput_h1-h4-1.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.1 s	57.0 MB	18.2 Mbps
throughput_h2-h5-1.txt	h2	10.0.0.3	h5	10.0.5.3	0.0 - 25.1 s	36.9 MB	11.8 Mbps

throughput_h1-h4-2.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.1 s	28.9 MB	9.2 Mbps
throughput_h2-h5-2.txt	h2	10.0.0.3	h5	10.0.5.3	0.0 - 25.3 s	31.3 MB	9.9 Mbps
throughput_h3-h6-2.txt	h3	10.0.0.4	h6	10.0.5.4	0.0 - 25.0 s	40.3 MB	12.9 Mbps
throughput_h1-h4-3.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.2 s	44.2 MB	14.1 Mbps
throughput_h7-h9-3.txt	h7	10.0.2.2	h9	10.0.7.2	0.0 - 25.1 s	48.8 MB	15.5 Mbps
throughput_h1-h4-4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.1 s	87.5 MB	27.9 Mbps
throughput_h8-h9-4.txt	h8	10.0.4.2	h9	10.0.7.2	0.0 - 25.0 s	58.6 MB	18.7 Mbps

4.6.2 Discussion

Based on the test results, we can make the following observations:

Two pairs of hosts (h1-h4 and h2-h5) communicating simultaneously:

Expected: Since both pairs of hosts use the same shared resources, we expect the available bandwidth to be divided between the two pairs, resulting in reduced throughput for each pair compared to when they are communicating individually.

Reality: The throughput for h1-h4 is 18.2 Mbps, and for h2-h5 is 11.8 Mbps. The average RTT for h1-h4 is 68.980 ms, and for h2-h5 is 72.980 ms. These values align with our expectations, as the throughput is reduced when the two pairs communicate simultaneously. Latency is also increased compared to ideal conditions.

Three pairs of hosts (h1-h4, h2-h5, and h3-h6) communicating simultaneously:

Expected: With three pairs of hosts communicating simultaneously, we expect the available bandwidth to be further divided between them, resulting in even lower throughput for each pair.

Reality: The throughput for h1-h4 is 9.2 Mbps, for h2-h5 is 9.9 Mbps, and for h3-h6 is 12.9 Mbps. The average RTT for h1-h4 is 69.219 ms, for h2-h5 is 70.274 ms, and for h3-h6 is 70.281 ms. As expected, the throughput values are lower compared to when only two pairs communicate simultaneously, and the latency values are also increased.

h1-h4 and h7-h9 communicating simultaneously:

Expected: Since h1-h4 and h7-h9 are on different subnets, we expect them to have a lesser impact on each other's performance compared to when the communicating hosts are on the same subnet.

Reality: The throughput for h1-h4 is 14.1 Mbps, and for h7-h9 is 15.5 Mbps. The average RTT for h1-h4 is 67.257 ms, and for h7-h9 is 68.840 ms. The throughput values are higher compared to when multiple pairs are communicating on the same subnet, and the latency values are also reduced, which aligns with our expectations.

h1-h4 and h8-h9 communicating simultaneously:

Expected: Since h1-h4 and h8-h9 are on different subnets, we expect them to have a lesser impact on each other's performance compared to when the communicating hosts are on the same subnet.

Reality: The throughput for h1-h4 is 27.9 Mbps, and for h8-h9 is 18.7 Mbps. The average RTT for h1-h4 is 69.181 ms, and for h8-h9 is 23.680 ms. The throughput values are higher compared to when multiple pairs are communicating on the same subnet, and the latency values are also reduced. In particular, the latency between h8-h9 is significantly lower, as they are on the same subnet and not affected by the h1-h4 communication.

In conclusion, the results from the experiments align with our expectations. As the number of simultaneously communicating hosts increases on the same subnet, the available bandwidth is divided, resulting in reduced throughput and increased latency. When hosts communicate on different subnets, they have a lesser impact on each other's performance.

4.7 Test case 5: effects of parallel connections

In this test, we evaluate the effects of parallel connections on network performance. Parallel connections refer to the process of opening multiple network connections between two hosts to share network resources and improve network performance.

In this test, h1, h2, and h3 connected to R1 will simultaneously communicate with hosts (h4, h5, and h6) connected to R3. h1 will open two parallel connections (with the -P flag in the client mode) to communicate with h4, while h2 and h3 will invoke the normal client (without the -P flag). We measure the throughput when three pairs of hosts are communicating simultaneously using Simpleperf and store the results in files called throughput_h1-h4.txt, throughput_h2-h5.txt, and throughput_h3-h6.txt in the measurement/test-case-6 folder.

4.7.1 Results

Test 5 throughput	Client Host	Client IP	Server Host	Server IP	Interval	Transfer	Throughput
throughput_h1-h4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.3 s	17.9 MB	5.6 Mbps
throughput_h1-h4.txt	h1	10.0.0.2	h4	10.0.5.2	0.0 - 25.6 s	24.8 MB	7.7 Mbps
throughput_h2-h5.txt	h2	10.0.0.3	h5	10.0.5.3	0.0 - 25.2 s	22.1 MB	7.0 Mbps
throughput_h3-h6.txt	h3	10.0.0.4	h6	10.0.5.4	0.0 - 25.3 s	34.8 MB	11.0 Mbps

4.7.2 Discussion

In this test case, we observe the effects of parallel connections on the network's throughput. Here, h1 opens two parallel connections to communicate with h4, while h2 and h3 establish normal connections with h5 and h6, respectively.

Expected Results:

Since h1 is opening two parallel connections, it is expected to have a higher throughput as it can utilize multiple connections to transfer data simultaneously.

h2 and h3 should have lower throughput compared to the sum of the throughput of h1's parallel connections, as they are using single connections.

Actual Results:

The combined throughput of h1-h4 (5.6 Mbps + 7.7 Mbps = 13.3 Mbps) is indeed higher than the throughput of h2-h5 (7.0 Mbps) and h3-h6 (11.0 Mbps).

The throughput of h1's parallel connections (5.6 Mbps and 7.7 Mbps) is not equal, possibly due to various factors such as congestion, packet loss, or competition for bandwidth with other connections.

The performance evaluations align with our expectations from the network topology. Using parallel connections, h1 can achieve a higher total throughput when communicating with h4. However, the unequal throughput of the two parallel connections indicates that network conditions, such as congestion or packet loss, could be affecting the performance.

5.0 Conclusions

The experiments conducted in this work aimed to analyze the performance of a network under various conditions, such as varying latency, packet loss, multiplexing, and parallel connections. The important results and their significance are as follows:

- **Latency:** As the latency increases, the throughput decreases, indicating a direct relationship between latency and network performance. Higher latency results in slower communication between hosts and reduced efficiency in the network.
- **Packet loss:** We observed that the network performance degrades with increasing packet loss. The network can tolerate a small amount of packet loss without a significant impact on throughput, but as the packet loss rate increases, the throughput decreases drastically.
- **Multiplexing:** The experiments showed that when multiple pairs of hosts communicate simultaneously, the available bandwidth is shared among them, leading to reduced throughput for each pair. Additionally, congestion in the network can further impact performance.
- **Parallel connections:** By using parallel connections, a host can achieve higher throughput when communicating with another host. This is particularly useful when a single connection is insufficient to fully utilize the available bandwidth.

However, there are some limitations and shortcomings in our work:

- The experiments were conducted in a controlled environment with a fixed network topology. Real-world networks are more complex, and the results may not directly apply to them.
- We did not explore the impact of other factors, such as traffic shaping, Quality of Service (QoS) mechanisms, or different transport layer protocols (e.g., TCP vs. UDP) on network performance.
- The experiments were limited to a small number of hosts and connections. The performance of large-scale networks with many hosts and simultaneous connections might differ significantly.
- Some test results were inconsistent across different computers, with unknown reasons for these discrepancies. This issue affected multiple users, requiring some tests to be run on other machines. To improve the reliability of the results, future work could focus on identifying the root cause of these inconsistencies by comparing hardware, software, and network settings between the affected and unaffected computers.

In conclusion, our work provides valuable insights into the performance of a network under various conditions. Still, it also highlights the need for further research to address the limitations, explore additional factors that may impact network performance, and resolve the inconsistencies encountered in test results across different computers.

6.0 References (Optional)

1. Iperf
 - Iperf3 Documentation. (n.d.). Retrieved from <https://iperf.fr/iperf-doc.php>
 - Tavangarian, D., & Grzempa, M. (2009). Iperf - A Network Bandwidth Measuring Tool. In P. Müller, P. & R. Reich, R. (Eds.), Kommunikation in Verteilten Systemen (KiVS) (pp. 303-308). Springer, Berlin, Heidelberg.
2. Simpleperf

- Android Developers. (n.d.). Simpleperf User Guide. Retrieved from https://android.googlesource.com/platform/prebuilts/simpleperf/+master/doc/simpleperf_user_guide.md
- 3. Throughput, RTT, Latency, and Packet Loss
 - Kurose, J. F., & Ross, K. W. (2017). Computer Networking: A Top-Down Approach (7th ed.). Pearson Education.
 - Tanenbaum, A. S., & Wetherall, D. J. (2011). Computer Networks (5th ed.). Pearson Education.
- 4. Multiplexing
 - Stallings, W. (2017). Data and Computer Communications (10th ed.). Pearson Education.
- 5. Parallel Connections
 - Wright, G. R., & Stevens, W. R. (1997). TCP/IP Illustrated, Volume 2: The Implementation. Addison-Wesley Professional.
- 6. ChatGPT - AI language model
 - OpenAI. (2021). ChatGPT by OpenAI [AI Language Model]. Retrieved from <https://openai.com/research/chatgpt/>
 - Used for explanation and research into several of the above references.
 - Used for debugging some snippets of the code when errors have occurred.