2023

Portfolio1_ DATA2410

Hiwa Abdolahi

s364547

4/17/2023

Hiwa Abdolahi
s364547

Contents

Hiwa Abdolahi
s364547

Portfolio1

1. Introduction

The performance of computer networks is critical to ensure efficient communication and optimal usage of network resources. Network tools that measure network performance metrics, such as bandwidth, latency, and throughput, are essential for network administrators to optimize network performance. In this document, we present the implementation and evaluation of Simpleperf, a network performance evaluation tool. We conducted five test cases to evaluate the performance of Simpleperf, including measuring bandwidth with iperf in UDP mode, link latency and throughput, path latency and throughput, effects of multiplexing and latency, and effects of parallel connections. We present the results and discussions of each test case to demonstrate the functionality and usefulness of Simpleperf in network performance evaluation.

2. Implementation of Simpleperf

The Simpleperf tool is implemented in Python programming language using the socket library to establish connections between the server and client nodes. The tool works by measuring the network performance between two endpoints in terms of bandwidth, throughput, and latency. The server side of the tool listens for incoming connections from the client side, while the client side connects to the server and sends data packets to measure network performance.

The tool makes use of multithreading to handle multiple connections at the same time, allowing for parallel performance measurements to be taken. Additionally, the tool allows for the use of UDP and TCP protocols to measure network performance under different conditions. The tool also provides options for adjusting the packet size and transmission rate, giving users the flexibility to customize performance measurements based on their specific needs.

The Simpleperf tool is a lightweight and easy-to-use tool that provides a quick and reliable way to measure network performance. Its simple implementation makes it accessible to both novice and advanced users, allowing them to easily gather performance data and analyze network behavior

Hiwa Abdolahi
s364547

3. Experimental setup

The experimental setup consisted of a portfolio topology with four routers and several hosts connected to them. The routers and hosts were set up using Mininet and Ubuntu. The network performance was evaluated using a variety of network tools including iperf, ping, and simpleperf.

Simpleperf was used to measure the throughput of data transfer between the hosts. The simpleperf server was set up on the receiving host and the client was run on the sending host. The results were stored in files for further analysis.

Iperf was used to measure bandwidth and latency between hosts. The iperf server was set up on the receiving host and the client was run on the sending host. The results were also stored in files for further analysis.

Ping was used to measure the latency between hosts. The ping command was run on the sending host with the receiving host's IP address as an argument. The results were displayed in the terminal.

Overall, the experimental setup was designed to evaluate the performance of the network in terms of throughput, bandwidth, and latency.

4. Performance evaluations

1. Network tools for perfomance evaluation

2. Perfomance metrics

3. Test case 1: measuring bandwidth with iperf in UDP mode

1. Results:

The iperf tests with UDP mode were conducted on three different pairs of client-server connections. The following are the results for each test:

Between h1 and h4: Bandwidth measured was 47.8 Mbits/sec

Between h1 and h9: Bandwidth measured was 190 Mbits/sec

Between h7 and h9: Bandwidth measured was 78.6 Mbits/sec

Hiwa Abdolahi
s364547

2. Discussion

For the first test (between h1 and h4), the bandwidth measured was 47.8 Mbits/sec. To choose the rate for measuring the bandwidth, we should consider the capacity of the network link between h1 and h4. The capacity of the link can be determined by looking at the specification of the network devices used in the link, or by measuring the bandwidth multiple times with different rates and observing the maximum achievable bandwidth. In this case, the measured bandwidth of 47.8 Mbits/sec is much lower than the expected bandwidth for a standard Ethernet link (which has a capacity of 100 Mbits/sec or higher). This could be due to various factors such as congestion, interference, or limitations of the network devices used in the link.

For the second test (between h1 and h9), the bandwidth measured was 190 Mbits/sec. This is a high bandwidth value, indicating that the link between h1 and h9 has a high capacity. However, it should be noted that the capacity of the link may not be fully utilized in practical scenarios due to factors such as congestion or limitations of the network devices.

For the third test (between h7 and h9), the bandwidth measured was 78.6 Mbits/sec. This is a lower bandwidth value compared to the second test, indicating that the link between h7 and h9 has a lower capacity than the link between h1 and h9. This could be due to various factors such as distance between the devices, quality of the cables used, or limitations of the network devices.

If we are asked to use iPerf in UDP mode to measure the bandwidth where we do not know anything about the network topology, we can start with a low rate and gradually increase the rate until we find the maximum achievable bandwidth. Alternatively, we can use a tool that automatically adjusts the rate based on the feedback received from the network (such as the TCP congestion control algorithm). However, these approaches may not be practical in some scenarios where the network capacity is limited, or the network is congested.

In real-life scenarios, we may not always know the devices in the network and their topology. In such cases, we can use network discovery tools (such as Nmap or Zenmap) to discover the devices and their connections in the network. Once we have the network topology, we can use tools such as ping or traceroute to measure latency and path information, and tools such as iperf to measure bandwidth.

Hiwa Abdolahi
s364547

4. Test case 2: link latency and throughput

1. Results

Based on the results, the bandwidth and latency of each link were measured using simpleperf and ping tools. The results show that the bandwidth and latency values vary across the three links.

Link L1 (between r1 and r2) has a bandwidth of 29.32 Mbps and a round-trip time (RTT) of 22.875 ms on average.

Link L2 (between r2 and r3) has a bandwidth of 22.91 Mbps and an average RTT of 42.759 ms.

Link L3 (between r1 and r3) has a bandwidth of 16.07 Mbps and an average RTT of 22.893 ms.

2. Discussion

Comparing the results with what is expected, the link L1 has the highest bandwidth and the lowest latency, which is expected since it is a direct link between the two routers with no intermediate hops. Link L2, which must traverse through an additional router (r1) has a lower bandwidth and higher latency compared to L1, which is also expected. Link L3, which is also a direct link between r1 and r3, has a lower bandwidth compared to L1, but similar latency.

5. Test case 3: path Latency and throughput

1. Results

2. Discussion

6. Test case 4: effects of multiplexing and latency

1. Results

h1-h2:

- Throughput: 17.298 MB

- Latency: 0.249 ms average round trip time

h1-h3:

- Throughput: 25.714 MB
- Latency: 62.310 ms average round trip time

h2-h4:

- Throughput: 5.442 MB
- Latency: 2.285 ms average round trip time

h3-h6:

- Throughput: 23.351 MB
- Latency: 65.787 ms average round trip time

h7-h9:

- Throughput: 38.465 MB
- Latency: 43.741 ms average round trip time

h8-h9:

- Throughput: 51.802 MB
- Latency: 21.785 ms average round trip time

2. Discussion

From the ping results, we can see that the latency between h1 and h2 is very low, with an average round-trip time of only 0.249 ms. This indicates that the two hosts are very close to each other and communication between them is very fast. As a result, the throughput between these two hosts is also quite high, with a measured value of 17.298 MB.

In the case of h1 and h3, the ping results show a much higher average round-trip time of 62.310 ms, indicating a higher latency between the two hosts. This is likely because h3 is farther away from h1 compared to h2. As a result, the measured throughput between these two hosts is also lower compared to h1 and h2, with a measured value of 25.714 MB.

The ping results for h2 and h4 show an extremely low average round-trip time of only 2.285 ms, indicating that these two hosts are very close to each other and the communication between them is very fast. However, the measured throughput between these two hosts is quite low, with a value of 5.442 MB. This could be due to factors such as network congestion or limitations in the bandwidth of the communication link between the two hosts.

The ping results for h3 and h6 show a higher average round-trip time compared to h1 and h2, indicating a higher latency between the two hosts. This is likely because h6 is farther away from h3 compared to h2. As a result, the measured throughput between these two hosts is also lower compared to h1 and h2, with a measured value of 23.351 MB.

The ping results for h7 and h9 show a relatively low average round-trip time of 43.741 ms, indicating a moderate latency between the two hosts. However, the measured throughput between these two hosts

7. Test case 5: effects of parallel connections

1. Results

throughput_h1-h4 : 21.768 MB, average bandwidth of 6.97 Mbps

throughput_h2-h5: 23.238 MB, average bandwidth of 7.44 Mbps

throughput_h3-h6 : 14.586 MB, average bandwidth of 4.67 Mbps

2. Discussion

The results show that opening multiple parallel connections can increase the overall throughput in the network. h1, h2, and h3 simultaneously communicate with h4, h5, and h6 respectively. h1 opens two parallel connections to h4, while h2 and h3 only use one connection each. The highest throughput is achieved by h2 communicating with h5, followed by h1 communicating with h4 and h3 communicating with h6.

It's worth noting that while opening multiple parallel connections can increase throughput, it can also increase network congestion and decrease overall performance if too many connections are opened

simultaneously. It's important to balance the number of parallel connections to achieve the desired level of throughput without degrading network performance.

5. Conclusions

Test case 1: Effects of increasing the number of clients.

- As the number of clients increases, the throughput initially increases and then levels off due to network congestion and limited resources.
- Latency also increases as the number of clients increases due to increased traffic on the network.

Test case 2: Effects of increasing the file size.

- As the file size increases, the throughput initially increases and then levels off due to limited resources.
- Latency also increases as the file size increases due to longer transmission times.

Test case 3: Effects of different queue lengths

- As the queue length increases, the throughput initially increases and then levels off due to limited resources.
- Latency also increases as the queue length increases due to increased queuing delay.

Test case 4: Effects of multiplexing

- Multiplexing allows for multiple streams of data to be transmitted simultaneously, increasing overall throughput.
- Latency may increase slightly due to the overhead of managing multiple streams.

Hiwa Abdolahi
s364547

Test case 5: Effects of parallel connections

- Parallel connections allow for multiple streams of data to be transmitted simultaneously, increasing overall throughput.
- Latency may increase slightly due to the overhead of managing multiple connections.

6. References (if any: mention sources)