**CCN PROJECT-1**

**Suraj Goud Rachamalla**

**001569950**

**srachamalla@albany.edu**

**INTRODUCTION:**

In this report, we will focus on describing our methodology. We'll explain how we processed the Wireshark traces to calculate specific metrics. We'll also clarify whether we used a computer program or manual methods for this analysis.

Before capturing the Wireshark traces, it's essential to set up our environment. We've ensured that the client and cache run on your local machine, while the server operates within a virtual machine. This separation of physical mediums is crucial for a comprehensive analysis. We use the "get <file>" command to download a file from the server, with the prerequisite that the cache is empty.

Additionally, to record a packet trace for each program invocation, Wireshark is run on your personal computer. This arrangement allows us to closely examine the network traffic and its behavior during file transfers.

Moreover, our report will include a comprehensive analysis of the observed trends by calculating both the delay and throughput metrics for both TCP and SNW protocols.

**METHODOLOGY:**

1. I manually calculated the required traces using Wireshark data. We begin by using Wireshark to capture network data while your application using TCP or SNW is actively communicating. Wireshark records all the network packets, including both data and control packets, exchanged between your client and server.
2. I filter the packets based on the IP address of the server, i.e., 169.226.22.10, along with specifying whether it's SNW or TCP.
3. After filtering the packets based on the IP address and protocol, Wireshark provides timestamps for each packet, indicating when they were captured. These timestamps are used to calculate the time difference between the first and last packets in a single program session, which gives us the overall delay.
4. To determine the size of each data packet exchanged during a program session, it's crucial for calculating the total bits exchanged, which, in turn, is used to calculate the achieved throughput.
   1. Overall Delay: Calculate the time difference between the first and last data packets in a single program session using the timestamps. This quantifies the overall delay.
   2. Achieved Throughput: Sum the sizes of all data packets within a single program session. Then, divide this total by the overall delay to calculate the achieved throughput in bits per second .
5. Repeat this process for various program sessions, including different file sizes, for both TCP and SNW implementations. Collect data for different scenarios and files.
6. After calculating these metrics for multiple program sessions, analyze the data to draw conclusions and make comparisons.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Delay | File1(16kb) | File2(32kb) | File3(48kb) | File4(62kb) |
| TCP (sec) | 4 × 10⁻³ | 5 × 10⁻³ | 6× 10⁻³ | 7 × 10⁻³ |
| SNW (sec) | 1.07 | 1.09 | 1.10 | 1.17 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Throughput | File1(16kb) | File2(32kb) | File3(48kb) | File4(62kb) |
| TCP (bps) | 32 × 10⁶ | 51.2× 10⁶ | 64 × 10⁶ | 73.1 × 10⁶ |
| SNW (bps) | 117431.1 | 237037.1 | 349090.9 | 423931.6 |

As file size increases, both TCP and SNW encounter higher delays in data transmission. In the case of TCP, this delay can be attributed to factors such as the sliding window mechanism, congestion control mechanisms, processing overhead, and the round-trip time. These elements collectively contribute to the increased delay in data delivery. For SNW, larger file sizes result in extended data transmission times and the absence of pipelining. SNW employs a straightforward one-packet-at-a-time approach, which becomes more time-consuming with larger files, leading to higher delays.

On the other hand, despite the increased delay, both TCP and SNW can achieve higher throughput with larger files. TCP leverages its congestion control mechanisms to adapt more effectively to the available network bandwidth, gradually increasing its transmission rate as file size grows. This results in throughput approaching closer to the ideal efficiency. SNW also benefits from larger files, as the impact of packet transmission overhead becomes less significant relative to the total data transfer. Smaller files may suffer from the overhead associated with establishing and tearing down connections for each packet, which can lead to lower throughput. In contrast, larger files mitigate this overhead's impact, contributing to enhanced efficiency in data transfer