Assignment 4b: Socket Programming in Virtualized Environment

Group 14
CS558: Computer Systems Lab
Application #1: Client-Server Programming using TCP and UDP Sockets

---

1. Introduction

1.1 Objective

To implement and analyze a two-stage communication protocol using both TCP and UDP sockets in a virtualized environment, with three configurations:

1. WSL (Windows Subsystem for Linux) client and Ubuntu VM server on the same host

2. Host machine client and VM server

3. VM client and host machine server

1.2 Description of Application #1

- The client-server protocol is written in C.

- It consists of two stages:

- TCP Phase (Negotiation): Client connects to the server via TCP and requests a UDP port.

- UDP Phase (Data Transfer): Client switches to UDP for data exchange using the negotiated port.

1.3 Networking Modes Considered

- WSL and Ubuntu VM (same host)

- Host to VM communication

- VM to host communication

2. Environment Setup

2.1 Installing Ubuntu VM on Ubuntu (Nested Virtualization)

- Install VirtualBox:

```
sudo apt install virtualbox
```

- Download Ubuntu ISO from the official website.

- Create a new VM in VirtualBox:

- Set the name, type (Linux), version (Ubuntu 64-bit)

- Allocate at least 2048 MB memory

- Create a VDI disk (dynamically allocated)

- Mount the ISO and complete the Ubuntu installation.

2.2 Installing WSL on Windows

- Open PowerShell as Administrator and run:

```
wsl --install
```

- Restart the system if required.

- Install Ubuntu from the Microsoft Store.

- Update and upgrade WSL:

```
sudo apt update && sudo apt upgrade
```

2.3 VirtualBox Bridged Adapter

- Configure the VM to use a Bridged Adapter in VirtualBox.

- This allows the VM to obtain its own IP on the same network as the host, avoiding NAT limitations and enabling direct communication.

2.4 Increasing UDP Buffer Size

- To improve UDP performance for large messages, run the following commands on both client and server:

```
sudo sysctl -w net.core.rmem_max=26214400
```

```
sudo sysctl -w net.core.wmem_max=26214400
```

3. Experiment 1: WSL Client and Ubuntu VM Server (Same Host)

Setup

- Client runs on WSL

- Server runs on Ubuntu VM

- IPs obtained using ip addr

Challenges

- WSL operates in a NAT environment with its own virtual network.

- VM could not ping WSL due to NAT isolation.

Workaround

- Communication from WSL (client) to VM (server) was successful using the VM's bridged IP.

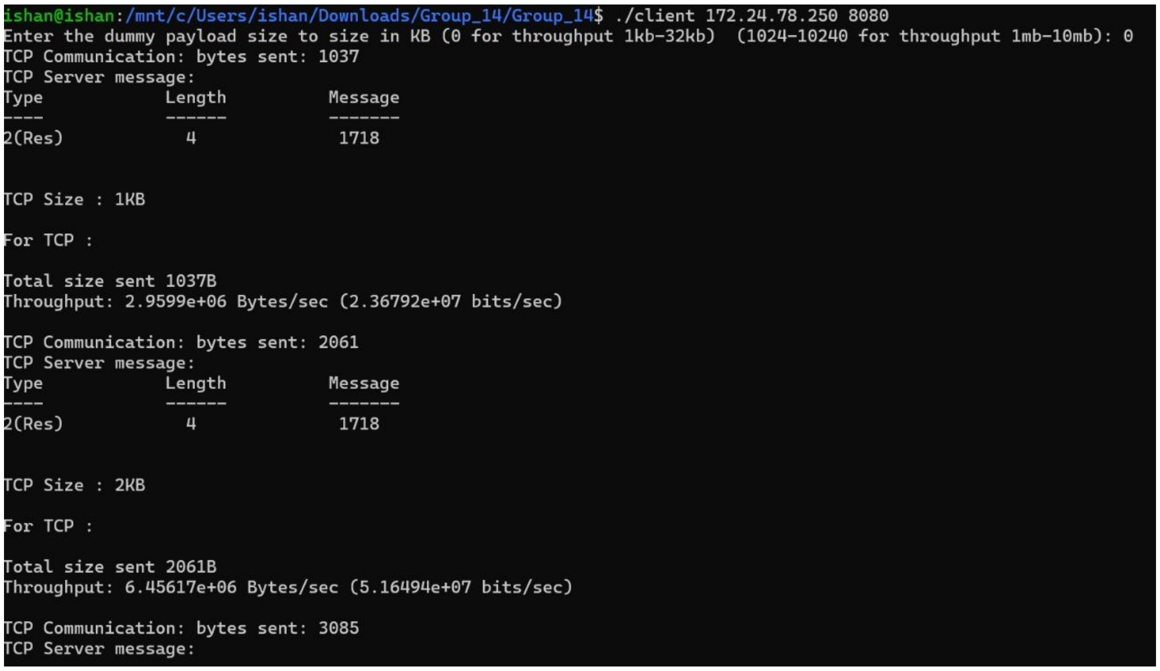
- Reverse communication (VM to WSL) was not possible by default.

Observation

- TCP and UDP connections worked from WSL to VM.

- No major packet loss was observed.

- Performance measurements such as TCP and UDP throughput and large file transfer metrics were recorded in this setup as part of the overall analysis. Three graphs were plotted specifically for this configuration: TCP throughput vs message size, UDP throughput vs message size, and TCP large file transfer throughput.



```
ishan@ishan:/mnt/c/Users/ishan/Downloads/Group_14/Group_14$ ./client 172.24.78.250 8080
Enter the dummy payload size to size in KB (0 for throughput 1kb-32kb)  (1024-10240 for throughput 1mb-10mb): 0
TCP Communication: bytes sent: 1037
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4              1718


TCP Size : 1KB

For TCP :

Total size sent 1037B
Throughput: 2.9599e+06 Bytes/sec (2.36792e+07 bits/sec)

TCP Communication: bytes sent: 2061
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4              1718


TCP Size : 2KB

For TCP :

Total size sent 2061B
Throughput: 6.45617e+06 Bytes/sec (5.16494e+07 bits/sec)

TCP Communication: bytes sent: 3085
TCP Server message:
```

4. Experiment 2: Host Client and VM Server

Setup

- Client runs on host machine

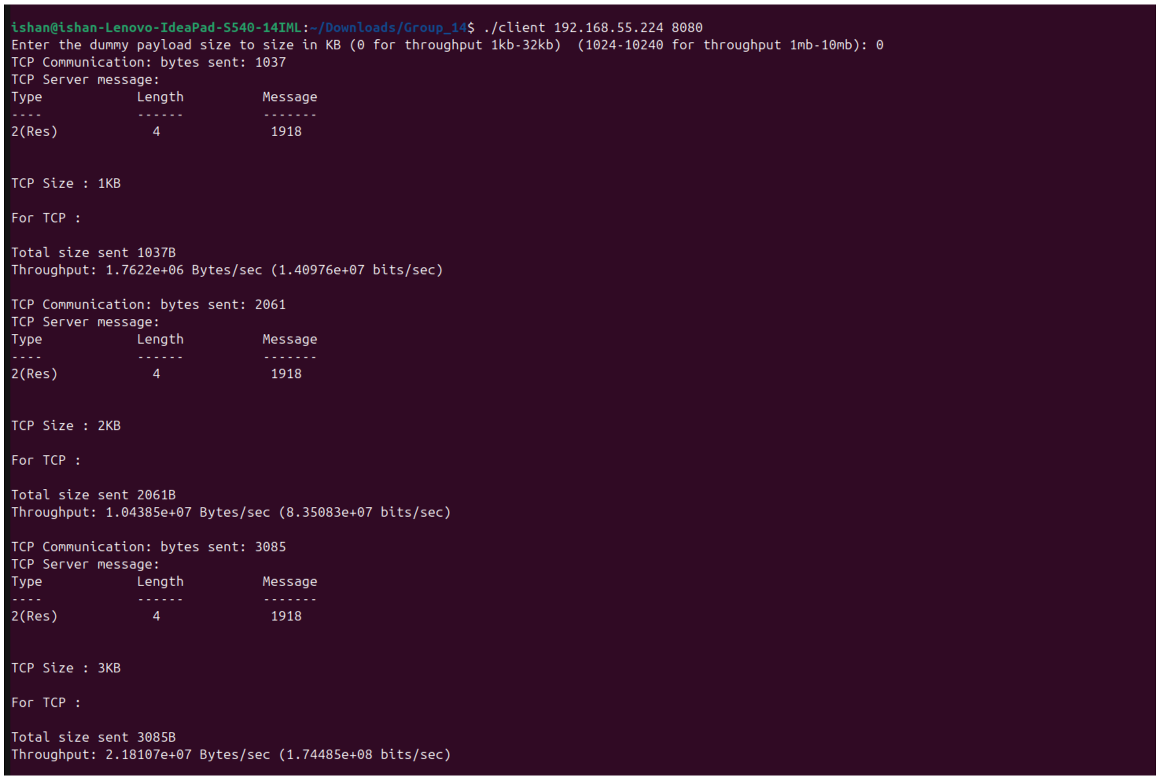
- Server runs on Ubuntu VM

- Used VM's bridged IP

Result

- Successful TCP handshake and UDP message exchange

- Performance data was gathered for this configuration as well, contributing to comparative analysis among all modes. Three additional graphs were generated to evaluate TCP and UDP performance along with large file transfer throughput in this setup.



```
ishan@ishan-Lenovo-IdeaPad-S540-14IML:~/Downloads/Group_14$ ./client 192.168.55.224 8080
Enter the dummy payload size to size in KB (0 for throughput 1kb-32kb)  (1024-10240 for throughput 1mb-10mb): 0
TCP Communication: bytes sent: 1037
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4              1918


TCP Size : 1KB

For TCP :

Total size sent 1037B
Throughput: 1.7622e+06 Bytes/sec (1.40976e+07 bits/sec)

TCP Communication: bytes sent: 2061
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4              1918


TCP Size : 2KB

For TCP :

Total size sent 2061B
Throughput: 1.04385e+07 Bytes/sec (8.35083e+07 bits/sec)

TCP Communication: bytes sent: 3085
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4              1918


TCP Size : 3KB

For TCP :

Total size sent 3085B
Throughput: 2.18107e+07 Bytes/sec (1.74485e+08 bits/sec)
```

5. Experiment 3: VM Client and Host Server

Setup

- Client runs on Ubuntu VM

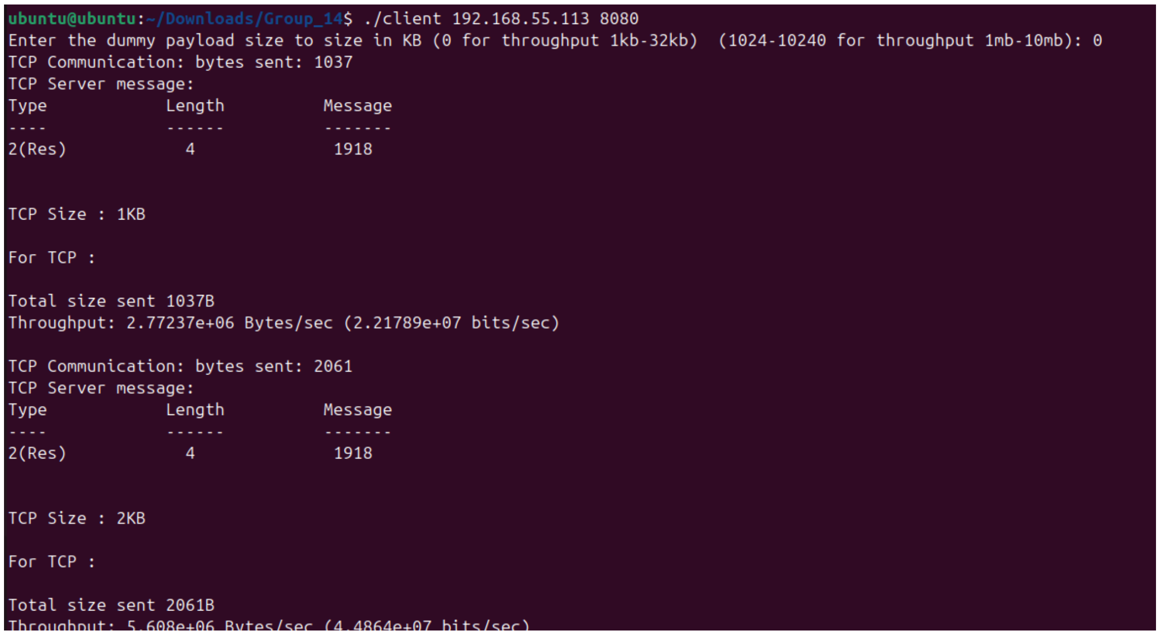
- Server runs on host machine (WSL or native server app)

Result

- Communication was successful

- Server processed incoming UDP requests smoothly

- Performance tests for this direction of communication were also conducted to ensure bidirectional data flow metrics were captured. The final three graphs were plotted for this setup covering TCP throughput, UDP throughput, and large file transfer.

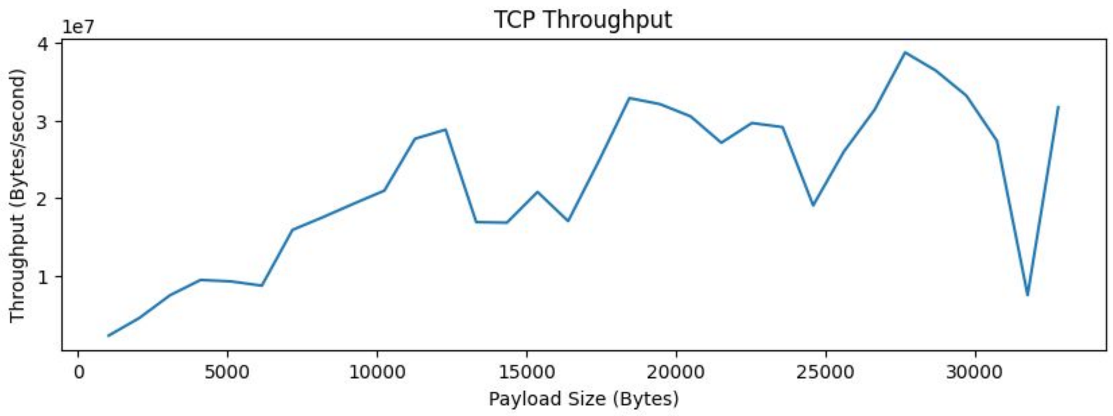


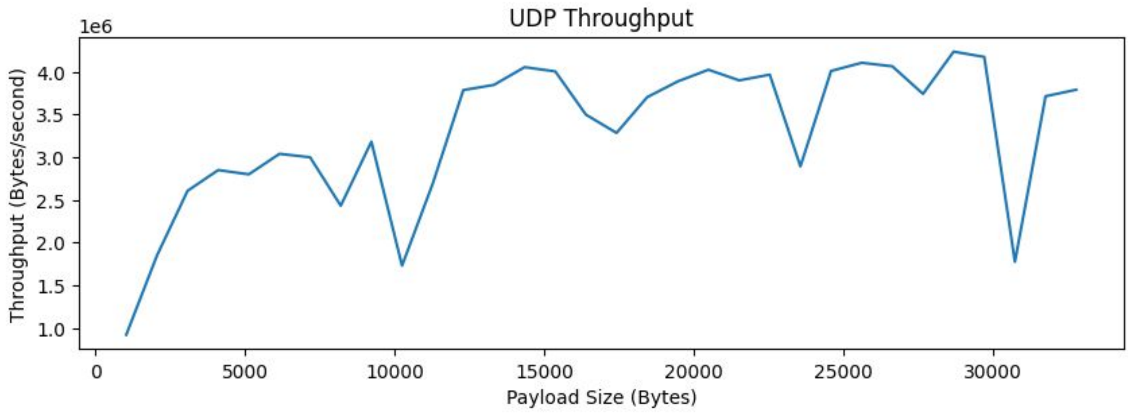
```
ubuntu@ubuntu:~/Downloads/Group_14$ ./client 192.168.55.113 8080
Enter the dummy payload size to size in KB (0 for throughput 1kb-32kb)  (1024-10240 for throughput 1mb-10mb): 0
TCP Communication: bytes sent: 1037
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4               1918


TCP Size : 1KB

For TCP :

Total size sent 1037B
Throughput: 2.77237e+06 Bytes/sec (2.21789e+07 bits/sec)

TCP Communication: bytes sent: 2061
TCP Server message:
Type            Length          Message
----            ------          -------
2(Res)            4               1918


TCP Size : 2KB

For TCP :

Total size sent 2061B
Throughput: 5.608e+06 Bytes/sec (4.4864e+07 bits/sec)
```

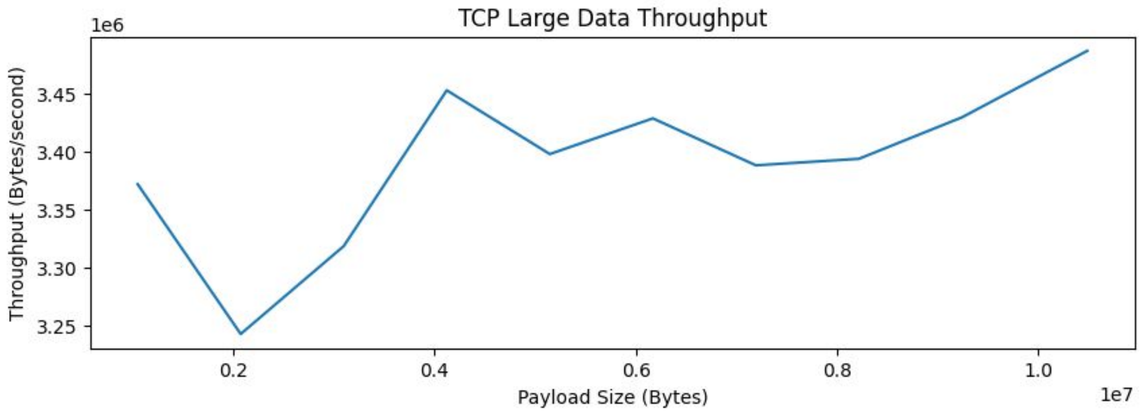
6. Performance Measurement

6.1 TCP vs UDP Throughput

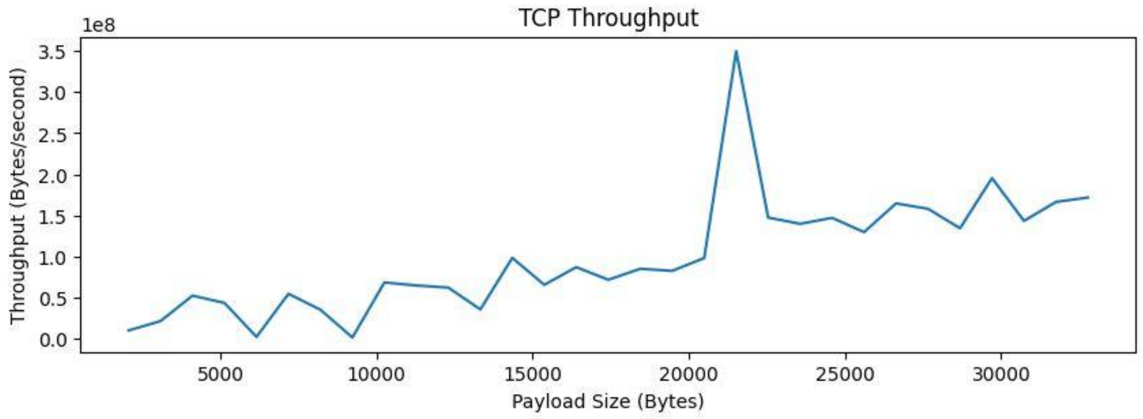
- VM to VM

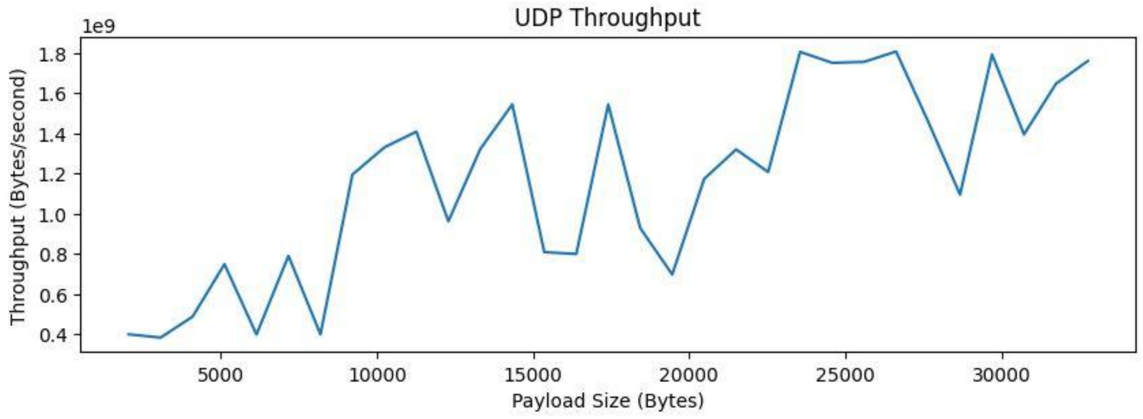


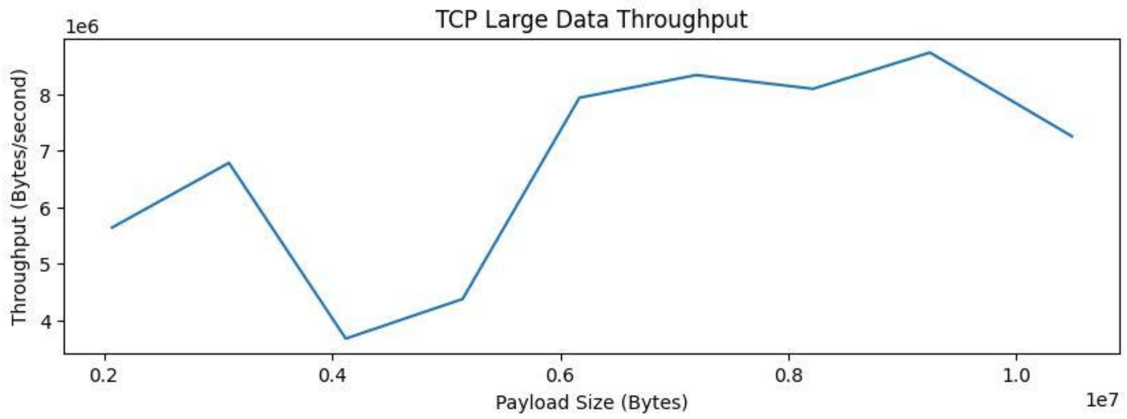




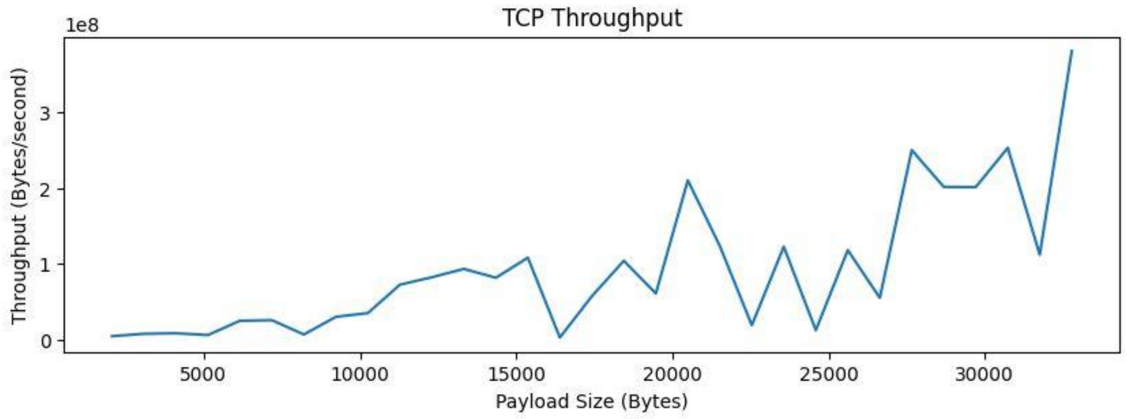
- Host to VM

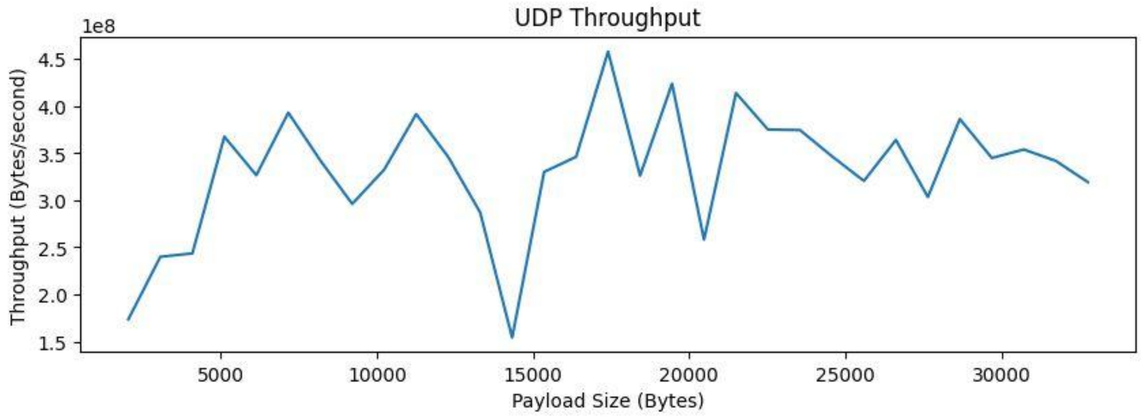

TCP Throughput
1e8
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Throughput (Bytes/second)
5000
10000
15000
20000
25000
30000
Payload Size (Bytes)

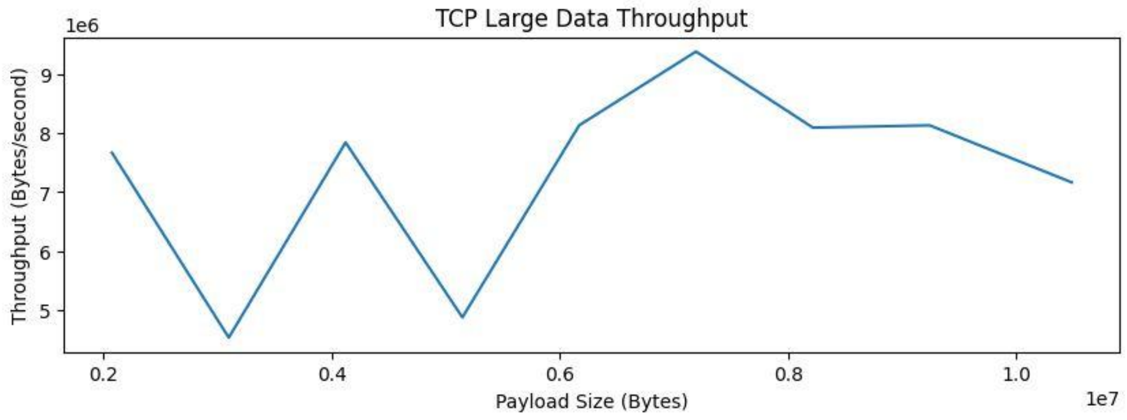

UDP Throughput
1e9
1.8
1.6
1.4
1.2
1.0
0.8
0.6
0.4
Throughput (Bytes/second)
5000
10000
15000
20000
25000
30000
Payload Size (Bytes)


TCP Large Data Throughput
1e6
8
7
6
5
4
Throughput (Bytes/second)
0.2
0.4
0.6
0.8
1.0
1e7
Payload Size (Bytes)

- VM to Host







7. Screenshot Summary

- TCP connection from WSL to VM: Successful

- UDP message received on VM: Successful

- Host to VM communication: Successful

- VM to Host communication: Successful

8. Observations and Challenges

8.1 NAT and Bridged Adapter

- WSL’s internal IP was not accessible from VM due to NAT restrictions.

- Bridged networking allowed successful communication with VM.

8.2 Client-Server Interaction

- Server in VM worked for both WSL and Host clients.

- VM client could communicate with Host server after configuring IP and firewall settings properly.

8.3 Performance

- UDP had higher throughput than TCP in most test cases.

- Round Robin scheduling maintained fairness when handling multiple clients.

9. Conclusion

- Bridged networking allowed robust communication across virtual environments.

- WSL functions like a lightweight VM but is limited by NAT.

- With appropriate configuration, bidirectional communication works seamlessly in all tested setups.

- The protocol proved effective for both control (TCP) and data (UDP) communication in virtualized environments.