Lab 1

TCP Sockets between WSL and Windows

Note: Mac users can do WSL experiments on their own machines and work with other classmates for Windows related activities or
If you have any VM installed try the same between Mac and the VM

Mouli Sankaran

Lab 1: Focus

- WSL Explained
- Server and Client Interactions
  - Using TCP Connection between WSL and Windows
  - Connection Establishment Explained
- Analyze the TCP Segments: Wireshark
- Experiments to be done in the Lab
  - Exp 1 and Exp 2

Python IDE used on Windows: Thonny

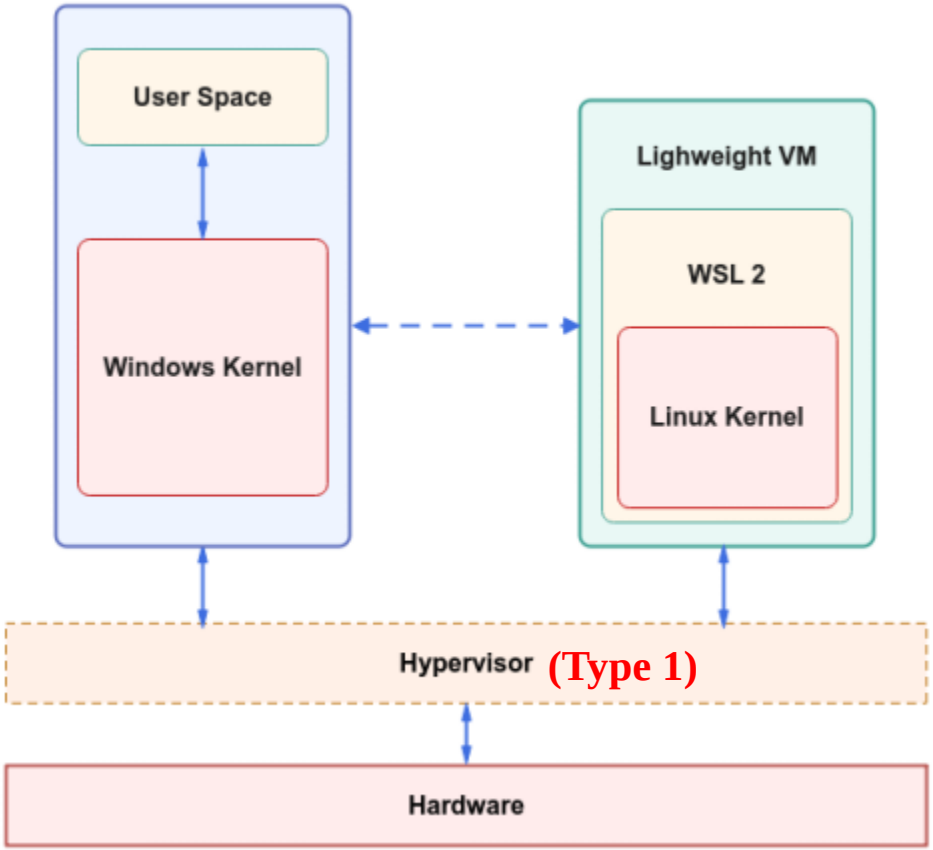
Course page where the course materials will be posted as the course progresses:

WSL Explained

WSL: Explained

Ref: Intro to WSL 2

- The Windows Subsystem for Linux (WSL) allows users to run Linux applications and command-line tools on Windows.
- WSL works by translating Linux system calls into Windows system calls.
- WSL allows Linux binaries to run on Windows without the need for a virtual machine or dual booting.
- WSL manages the Linux file system, network, and process execution.
- WSL creates a Virtual Hard Disk (VHD) to store files for each Linux distribution. In our case Ubuntu.



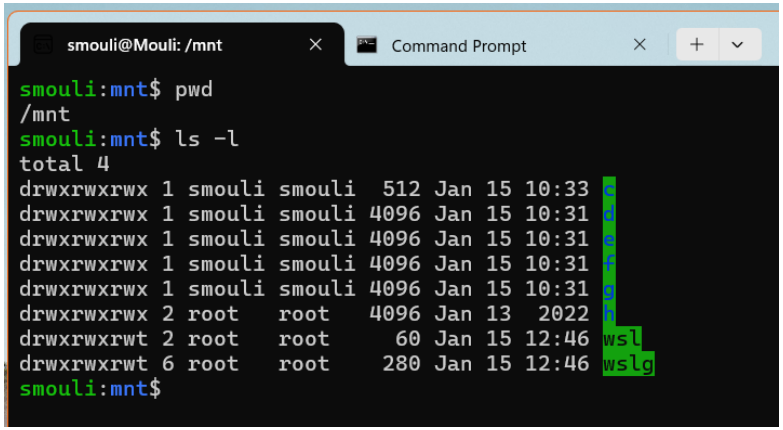
WSL: Explained

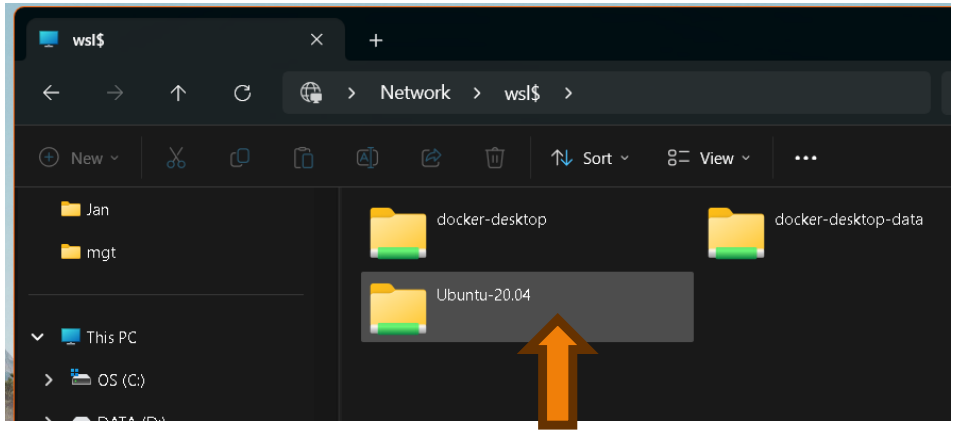
- The Lightweight Utility VM has been optimized to load the Linux kernel into the VM's address space without going through any Boot Loader process, thereby achieving fast startup time
- WSL supports only x64 and Arm CPUs.
- Using your mounted drives, you can edit code in, for example, *C:\dev\myproj* using Visual Studio or VS Code, and build/test that code in Linux by accessing the same files via */mnt/c/dev/myproj*

Accessing Linux filesystem (WSL) from Windows: **\\wsl$**

Enter **\\wsl$** in the File Explorer address bar to view the filesystem of Linux distribution on WSL **Ref link**.

Accessing Windows filesystem from WSL: **/mnt**

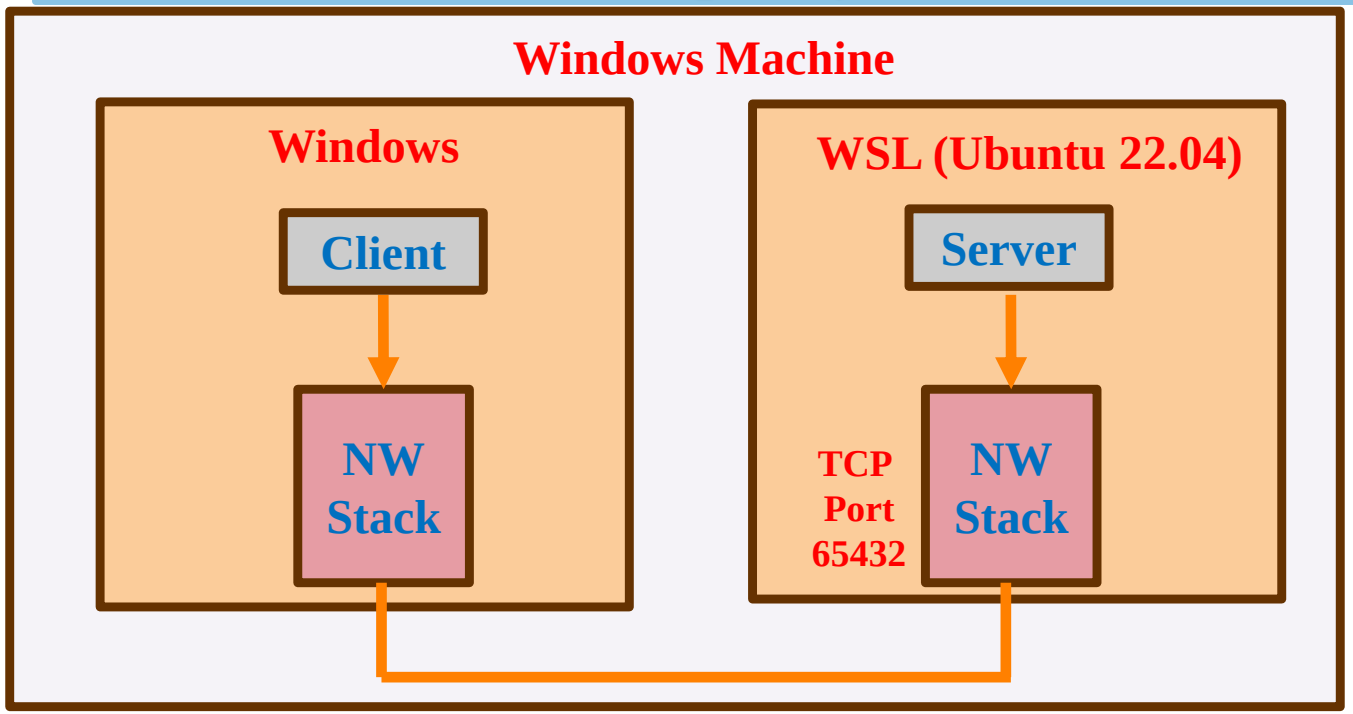




Server and Client Interactions

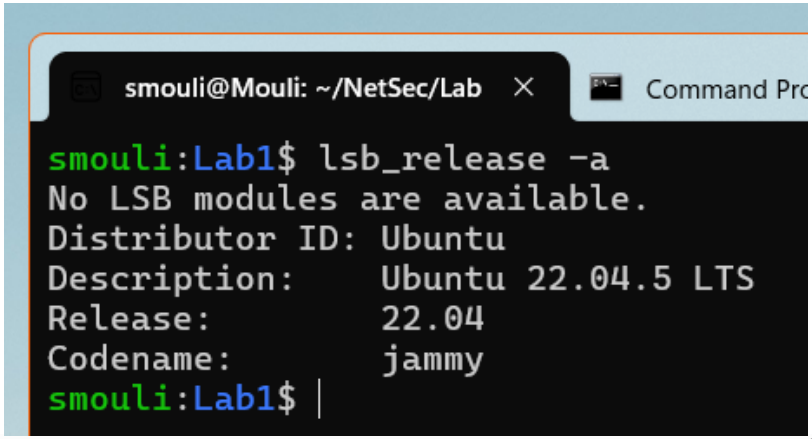
Reference: Ref1: TCP/IP Illustrated-Volume 1:
Chapter 13: TCP Connection Establishment and Termination

Exp: System Diagram
(sample Python code shared)



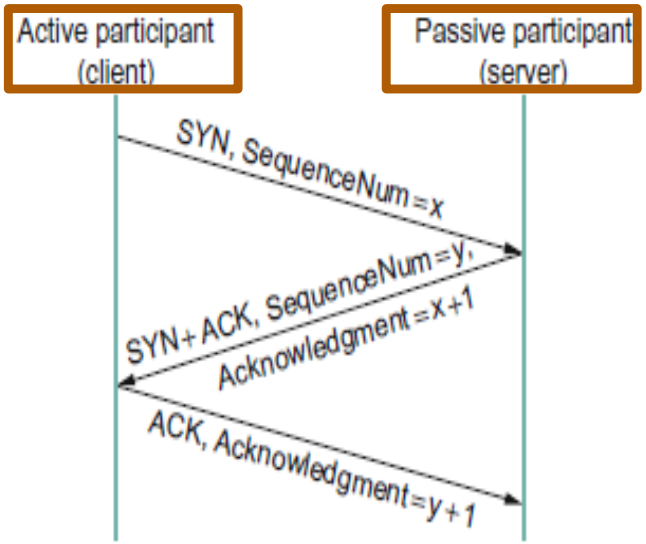
On WSL

- **Exp 0**: Run the below python programs
  - **On WSL**: **Lab1_exp0_server_WSL.py** - this should be run first
  - **On Windows: Lab1_exp0_client_Win.py** – this should be run next



Connection Establishment: Explained

Source port										Destination port
Sequence number										
Acknowledgment number (if ACK set)										
Data offset **HDR Len**	Reserved 0 0 0	N S	C W R	E C E	U R G	A C K	P S H	R S T	S Y N	F I N	Window Size
Checksum										Urgent pointer (if URG set)



- A connection typically goes through **three phases**:

1. Setup,
2. Data transfer (called *established*), and
3. Teardown (closing).

SYN: **Synch** or **Synchronization** **FIN**: **Final segment** of a **TCP connection.**

IP address: WSL
(file name: Lab1_exp0_server_WSL.py)

- IP address of the Server running on WSL: **HOST = 172.24.215.49**

smouli@Mouli: ~/NetSec/Lab | Command Prompt

```
smouli:Lab1$ ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.24.215.49  netmask 255.255.240.0  broadcast 172.24.223.255
        inet6 fe80::215:5dff:fefa:55c7  prefixlen 64  scopeid 0x20<link>
        ether 00:15:5d:fa:55:c7  txqueuelen 1000  (Ethernet)
        RX packets 7937  bytes 810731 (810.7 KB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 2346 (2.3 KB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 532 (532.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 532 (532.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0
```

inet 172.24.215.49
netmask 255.255.240.0

Note: Choose the IP addresses based on what you see in your machine.

Change the HOST value (IP addr) as shown below

Thonny - G:\Users\Mouli\RVU\Courses\NetSec\PMaterials\Labs\Lab1\code\Lab1_exp0_client_Win.py @ 4 : 77

File Edit View Run Tools Help

Lab1_exp0_client_Win.py *

```python
import socket

# Server address (replace with the server's IP)
# Replace the below IP address based on what you see on WSL for ifconfig cmd
HOST = '172.24.215.49'  # WSL: IP address assigned on Ubuntu

PORT = 65432            # Same port as the server
```

Exp 0: Server (WSL) ←→ Client (Windows)

(file names: Lab1_exp0_server_WSL.py & Lab1_exp0_client_Win.py)

- Run the server code of Exp0 on WSL and Client code of Exp0 on Windows

Server on WSL

smouli@Mouli: ~/NetSec/Lab

```
smouli:Lab1$ python3 Lab1_exp0_server_WSL.py
Lab1: Exp0:Server running on WSL: Welcome !!!
Host name:  Mouli
IP address:  127.0.1.1
FQDN:  Mouli.
Iface   Destination     Gateway         Flags   RefCnt  Use     Metric  Mask    MTU     Window  IRTT
eth0    00000000        01D018AC        0003    0       0       0       000000000       0       0

eth0    00D018AC        00000000        0001    0       0       0       00F0FFFF0       0       0


Server listening on 0.0.0.0:65432
Connected by ('172.24.208.1', 60667)
Received: Hello, server!
smouli:Lab1$
```

smouli@Mouli: ~/NetSec/Labs | Command Prompt

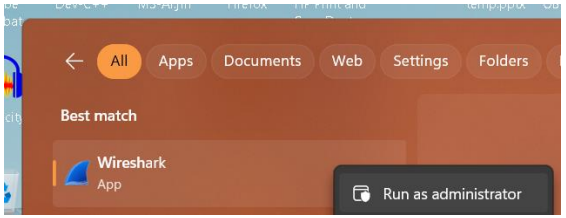
```
G:\Users\Mouli\RVU\Courses\NetSec\PMaterials\Labs\Lab1\code>py Lab1_exp0_client_Win.py
Lab1: Exp0:Client running on Windows: Welcome !!!
Host name:  Mouli
IP address:  172.16.227.191
FQDN:  Mouli
Received Hello, server! from server

G:\Users\Mouli\RVU\Courses\NetSec\PMaterials\Labs\Lab1\code>
```

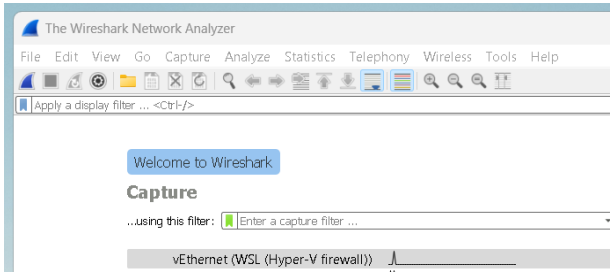
Client on Windows

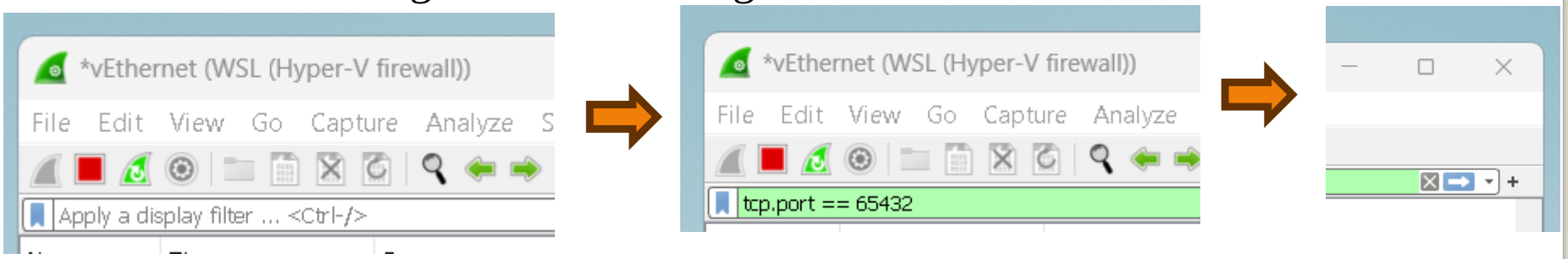
Analyze the TCP Segments: Wireshark
(exchanged between the Server and Client)

- Run the **Wireshark in Administrator mode** and capture the packets on the virtual Interface between the WSL and Windows



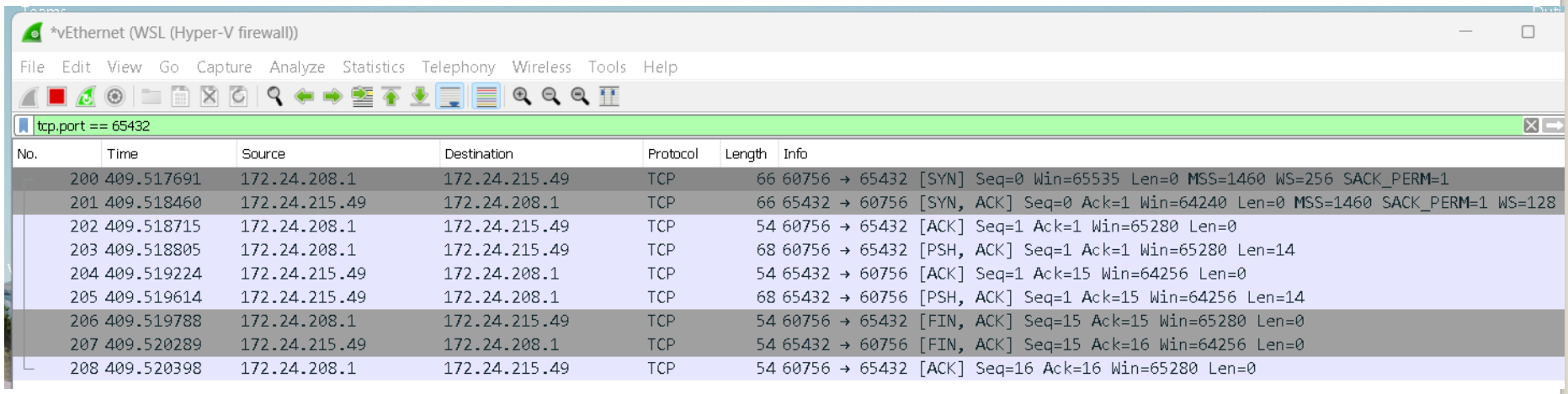
- Choose: Capture **vEthernet Interface**
- Apply a **display filter**: Type the filter cmd
  - **tcp.port == 65432**
  - Click on the right arrow at the right side





- Run the Server code on WSL first and then the Client code on the Windows side.
- You can observe TCP segments exchanged between the Server and the Client on Wireshark.

Analyze Each TCP Segment Exchanged



No.	Time	Source	Destination	Protocol	Length	Info
200	409.517691	172.24.208.1	172.24.215.49	TCP	66	60756 → 65432 [SYN] Seq=0 Win=65535 Len=0 MSS=1460 WS=256 SACK_PERM=1
201	409.518460	172.24.215.49	172.24.208.1	TCP	66	65432 → 60756 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460 SACK_PERM=1 WS=128
202	409.518715	172.24.208.1	172.24.215.49	TCP	54	60756 → 65432 [ACK] Seq=1 Ack=1 Win=65280 Len=0
203	409.518805	172.24.208.1	172.24.215.49	TCP	68	60756 → 65432 [PSH, ACK] Seq=1 Ack=1 Win=65280 Len=14
204	409.519224	172.24.215.49	172.24.208.1	TCP	54	65432 → 60756 [ACK] Seq=1 Ack=15 Win=64256 Len=0
205	409.519614	172.24.215.49	172.24.208.1	TCP	68	65432 → 60756 [PSH, ACK] Seq=1 Ack=15 Win=64256 Len=14
206	409.519788	172.24.208.1	172.24.215.49	TCP	54	60756 → 65432 [FIN, ACK] Seq=15 Ack=15 Win=65280 Len=0
207	409.520289	172.24.215.49	172.24.208.1	TCP	54	65432 → 60756 [FIN, ACK] Seq=15 Ack=16 Win=64256 Len=0
208	409.520398	172.24.208.1	172.24.215.49	TCP	54	60756 → 65432 [ACK] Seq=16 Ack=16 Win=65280 Len=0

- The first three segments seen above are related to connection establishment between the Server waiting on the port # 65432
- The next three segments are exchanging data between the Server and the Client
- The last three segments are closing the connection between them, with the Client initiating the closure of connection

Experimentes to be Done in the Lab

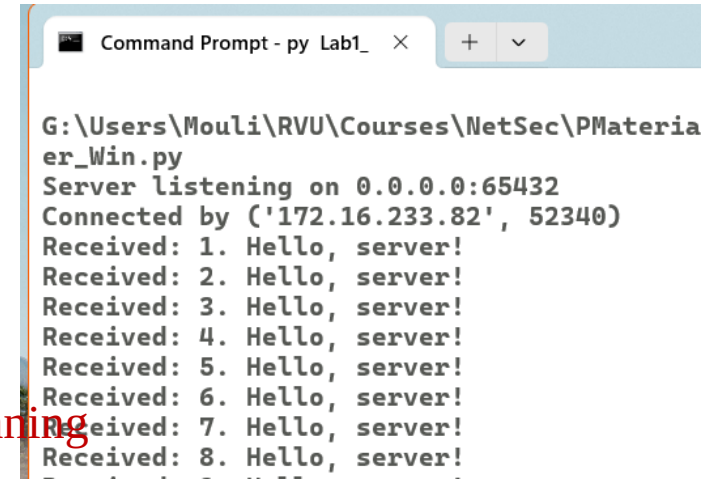
Exp 1: Server (Win) and Client (WSL)

- Copy the given Exp0 files of both Server and the Client, as Exp1 files and change the Exp1 Client file according to your IP address observed on Windows
  - Exchanging files between the WSL and Windows seamless, use the method shown on Slide #5.
- Now, run the Exp1 Server on Windows first and run the Exp1 Client code from the WSL side and observe the segments exchanged between the WSL and Windows using the Wireshark.
- Observe the exchange of segments between the server running Windows and the client running on WSL on Wireshark, by setting up the display filter as "**tcp.port == 65432**"

Exp 2a: Server (Win) and Client (WSL)

- Copy the Exp1 files both Server and Client as Exp2 and modify it to run the server and client continuously by making it exchange messages every two seconds between them, as shown below.
  - Change the name to indicate whether it is expected to be run on WSL or WIN, by changing it with **_WSL** or **_Win**
- When you give control+C the connection from the client side should disconnect with the server and both sides the program exits from the infinite loop.

Server running on Windows printing the sequence of messages sent by the Client Running on WSL:



Note: **Exp2b**: Make a connection with the server running on the machine with your classmate who is also connected to the same network as you (WiFi or hotspot of one mobile.

Exp 3: Server (Win) and Client (WSL)

- Copy the Exp2 files both Server and Client as Exp3 and modify it to run the server and client continuously by making it exchange messages every two seconds between them, as shown below.
- Server running on Windows creates max 5 sockets by creating separate threads and each waiting on the port numbers 65431 to 65435 respectively.
- Client program accepts port number as input from the user (from 65431 to 65435) and tries to establish connection with the server waiting on the port number fed as input.
- You will notice that simultaneously five TCP connections are established between the Windows and WSL and all of them run in parallel printing messages with received from the client on the Windows side.
  - Add an increasing serial number to the message printed so that we can identify messages on different TCP connections established.

Lab 1: Summary

- WSL Explained
- Server and Client Interactions
  - Using TCP Connection between WSL and Windows
  - Connection Establishment Explained
- Analyze the TCP Segments: Wireshark
- Experiments to be done in the Lab
  - Exp 1 and Exp 2