**COMPUTER NETWORKS**

**ASSIGNMENT # 3**

**REPORT**

ASHISH JUMANI 20i-0494

HAMDA ZAHIR 20i-0477

**INTRODUCTION:**

This report discusses a code snippet that utilizes the Scapy library to capture and analyze network packets. The code leverages the capabilities of Scapy to handle Ethernet, IP, TCP, and UDP protocols. The objective is to extract relevant information from the packets, such as the source and destination ports for TCP and UDP protocols.

**EXPLANATION:**

The code begins by importing the necessary modules from Scapy, including "sniff," "Ether," "IP," "TCP," and "UDP."

The "packet\_handler" function is defined to process each captured packet. It takes a packet as an argument and starts by checking if it contains an Ethernet header using the "haslayer" method. If it does, the Ethernet header is extracted and stored in the "eth\_header" variable.

Next, the code checks if the packet is an IPv4 packet by comparing the "type" attribute of the Ethernet header with the value 0x0800, which indicates IPv4. If the packet is indeed an IPv4 packet, the IP header is obtained and stored in the "ip\_header" variable.

After confirming that the packet is an IP packet, the code checks the "proto" attribute of the IP header to determine whether it is a TCP or UDP packet. If the "proto" value is 6, the packet is identified as a TCP packet, and the TCP header is extracted and stored in the "tcp\_header" variable.

The code then prints the source and destination ports of the TCP packet using the "sport" and "dport" attributes of the TCP header.

If the "proto" value is 17, the packet is identified as a UDP packet, and the UDP header is extracted and stored in the "udp\_header" variable. Similarly, the code prints the source and destination ports of the UDP packet using the "sport" and "dport" attributes of the UDP header.

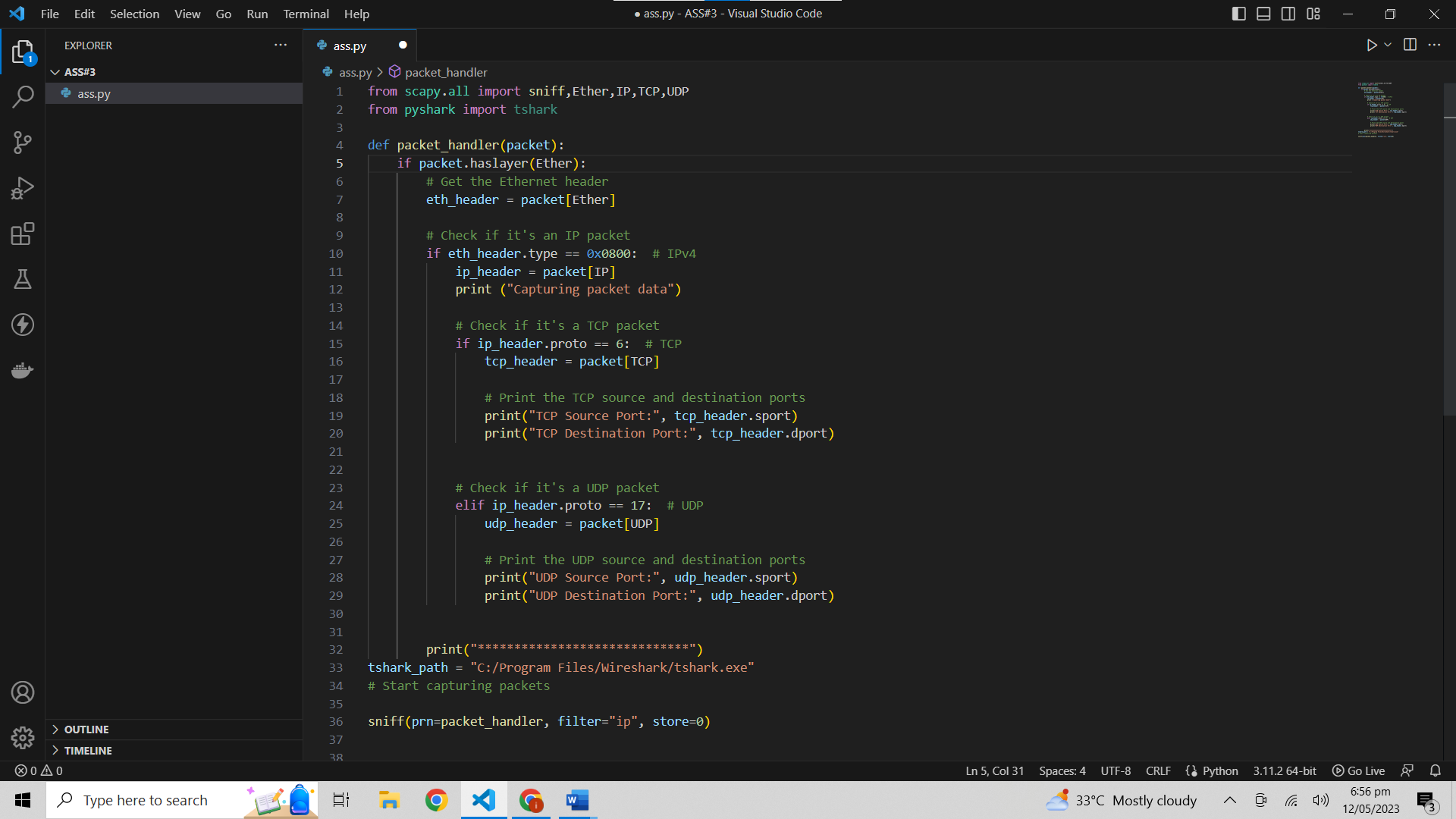
Lastly, the code prints a separator line to distinguish between different packets.

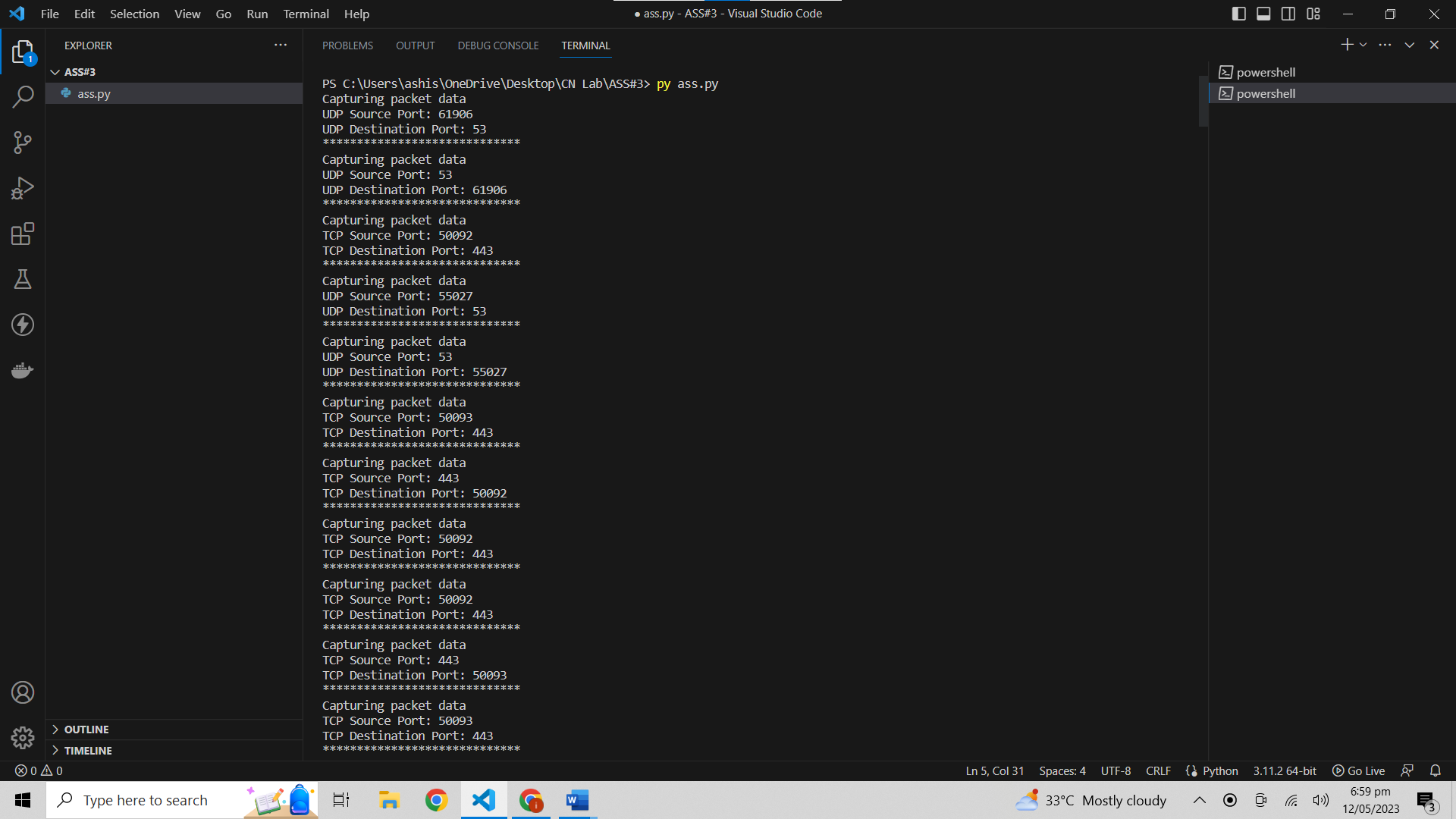
The main part of the code initiates packet capturing using the "sniff" function from Scapy. It takes three parameters: "prn" specifies the callback function to handle each captured packet, "filter" specifies a filter for capturing specific packets (in this case, only IP packets), and "store" set to 0 means that the captured packets are not stored in memory.

**CONCLUSION:**

The provided code snippet demonstrates the use of Scapy to capture and analyze network packets. It shows how to extract and print important information from Ethernet, IP, TCP, and UDP headers. This code can serve as a foundation for more advanced network packet analysis tasks, allowing for further processing or storage of captured packet data.

CODE:



OUTPUT:

A screenshot of a computer

Description automatically generated

C:\Users\ashis\OneDrive\เอกสาร\Zoom\2023-05-12 19.01.49 Ashish Jumani's Zoom Meeting