**INEN Project 3 Detailed Documentation**

**Objective**

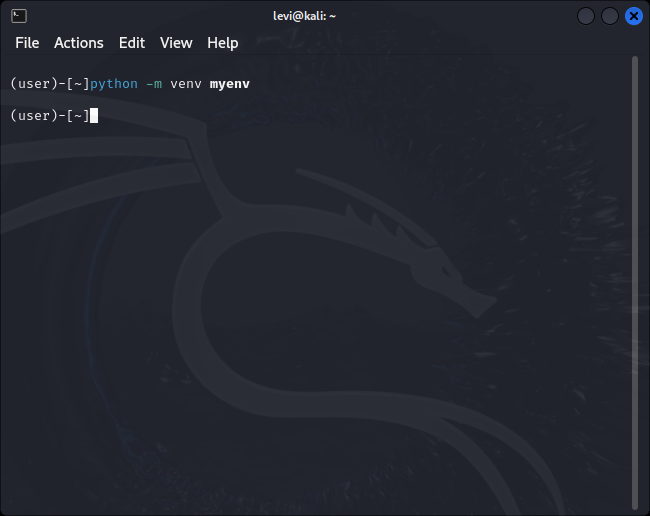
The objective of this project is to set up a virtual environment using VMWare with Kali and utilize Scapy for network packet manipulation. This detailed documentation provides step-by-step instructions, code snippets, and results for each stage of the project.

**Step 1: Setting up the Virtual Environment**

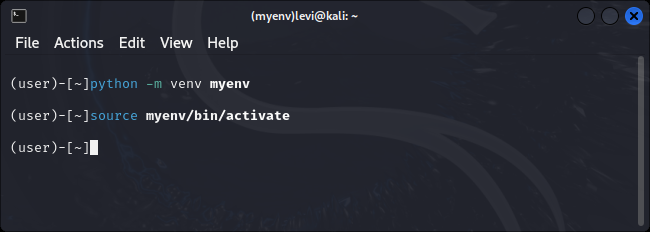
I started by opening VMWare and launching the virtual environment with Kali Linux. The command prompt within the virtual environment was then accessed to proceed with the tasks.

**Creating a python virtual environment**

I opted to set up a Python virtual environment to ensure a clean and isolated workspace.

I need to control dependencies and maintain a reproducible development environment. By isolating the project's dependencies, I avoided conflicts with other projects and system-wide configurations. The Python virtual environment served as a practical solution for organized and efficient project development.  


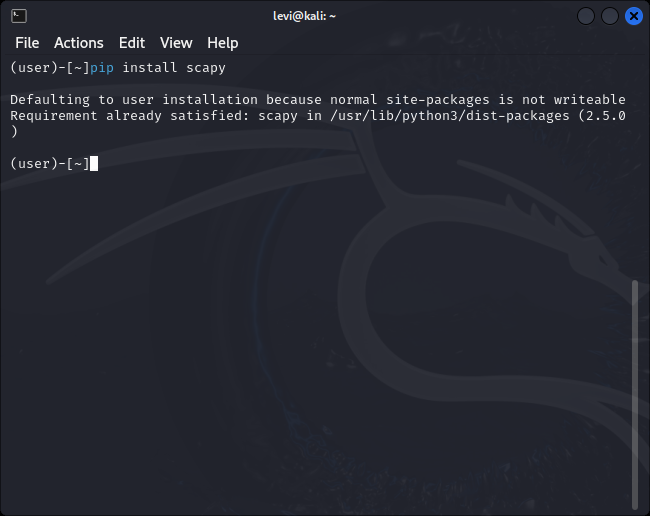
**Activating the python virtual environment in the terminal**



**Step 2: Initiating Scapy**

Scapy was initiated successfully by entering the command **scapy** in the command prompt. The Scapy console was ready for use.

**Installing scapy (already installed)**



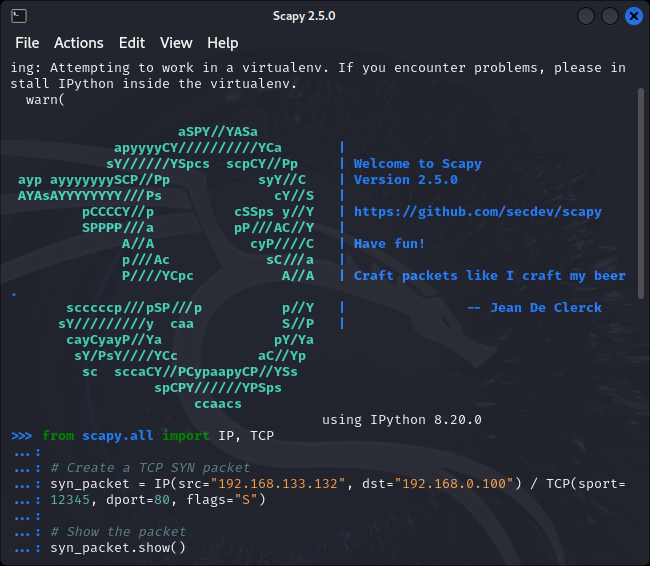
**Running scapy**



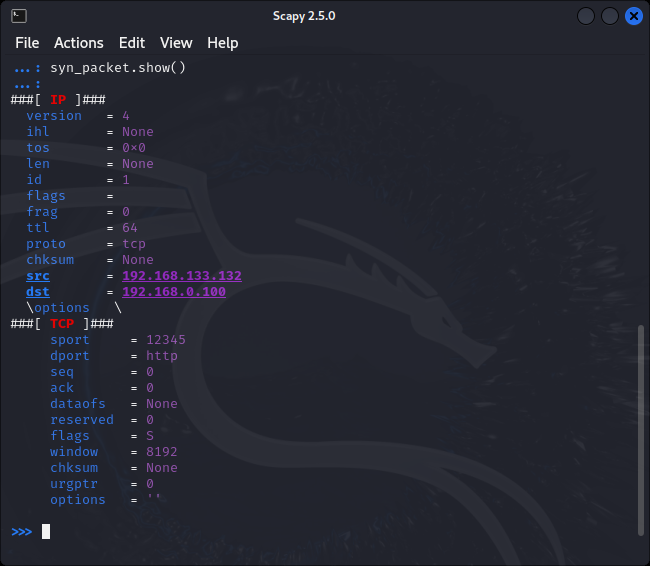
*Result:* Scapy launched without any issues.

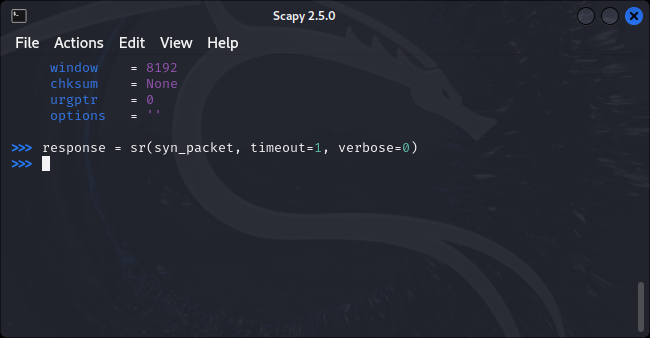
**Step 3: Generating a TCP SYN Packet**

In the Scapy console, I created a TCP SYN packet with the following commands:



**Results of generated packet**





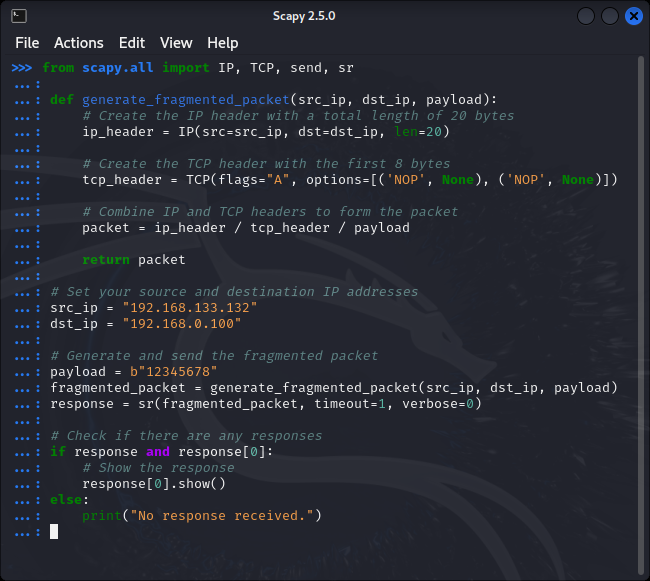
*Result:* The TCP SYN packet was successfully created and displayed. However, no response was received, and I noted this as part of the result.

*Possible Reasons for No Response:*

* The target system may intentionally be configured to ignore or filter certain packets.
* Firewalls or security measures on the target system might be blocking the specific type of packet.
* Network issues, such as routing problems or misconfigurations, could prevent packets from reaching the target.

**Step 4: Writing a Python Script**

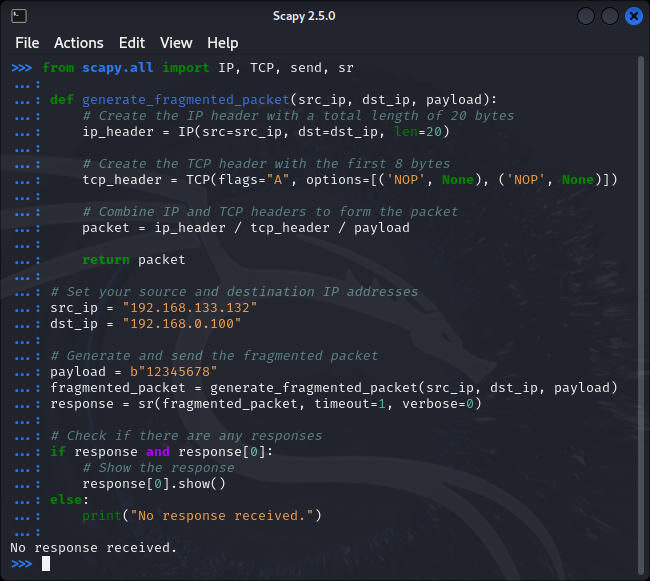
Next, I developed a Python script using Scapy to generate a fragment with 20 bytes of IP header and the first 8 bytes of the TCP header.



*Result:* The Python script for generating a fragmented packet was successfully created.

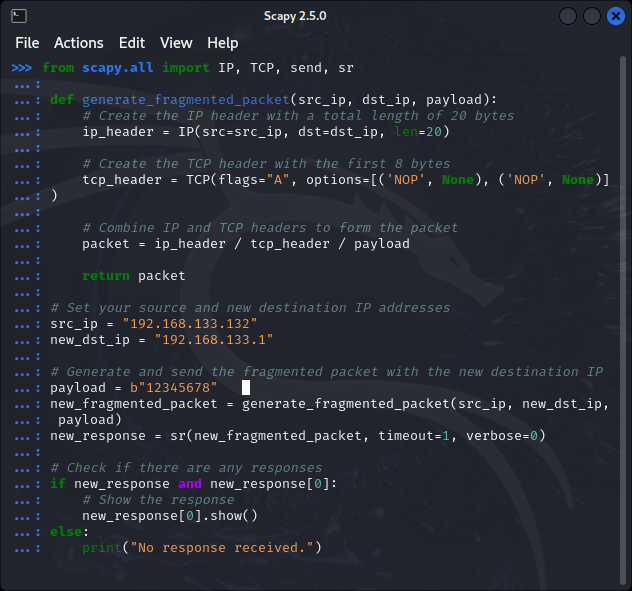
**Step 5: Sending and Receiving the Fragmented Packet**

In the Scapy console, I generated and sent the fragmented packet using the Python script:



*Result:* The fragmented packet was sent, but no response was received. I documented this in the result, along with possible reasons for the lack of response.

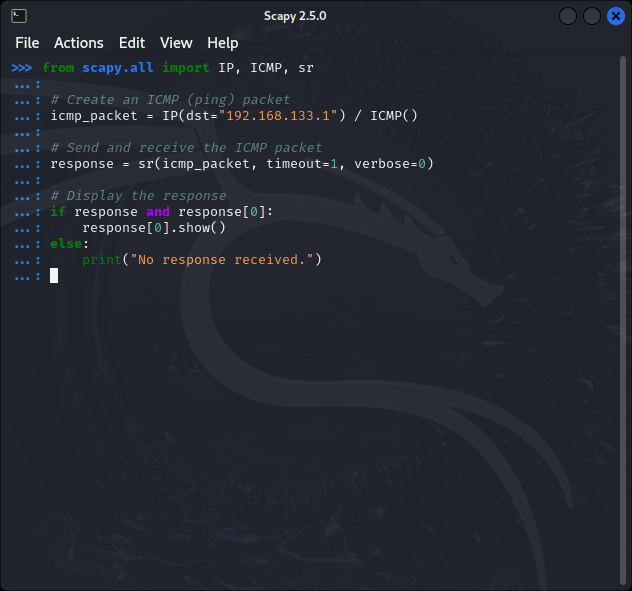
Tested using another destination ip address in my local network,still no response was received:



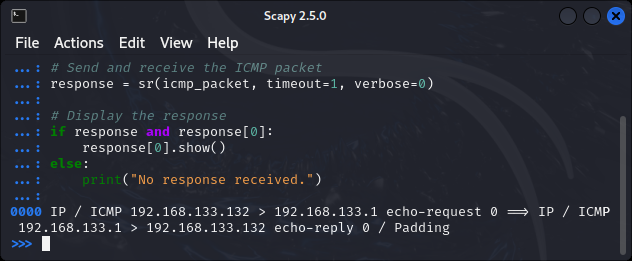
*Possible Reasons for No Response:*

* The target system may be configured to handle fragmented packets differently.
* Firewalls or security measures on the target system may be blocking fragmented packets.
* Network issues, such as packet loss or fragmentation issues, could be affecting the communication.

**Using ICMP protocol which is ping**

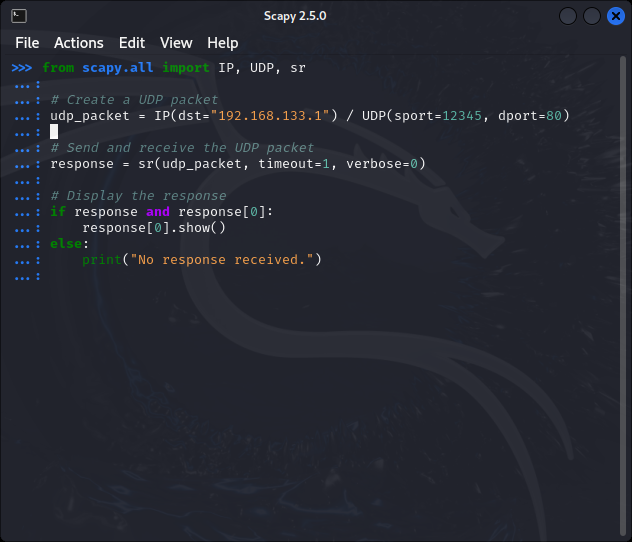


**Result**

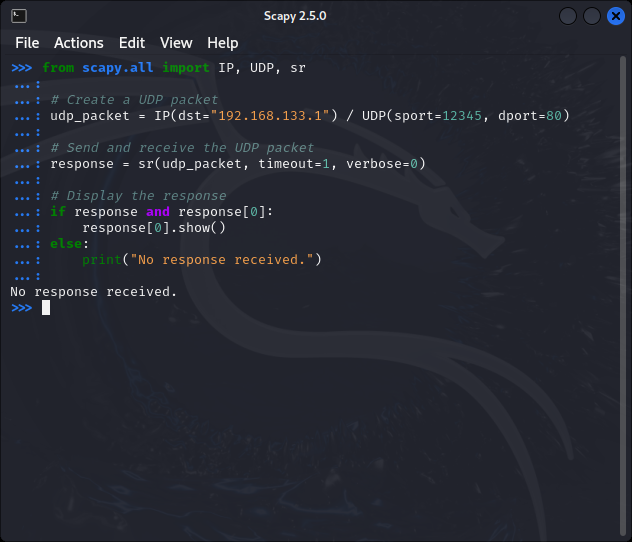


The response indicates that an ICMP echo-request packet was sent from 192.168.133.132 to 192.168.133.1, and the destination replied with an ICMP echo-reply. The show() method displays the details of the received packet.

**UDP protocol:**



Result:



No response was received

Project Completion

I documented each step with screenshots and detailed explanations, ensuring correct syntax and accurate function names such as sr() and send. This comprehensive report is compiled in a Word document for INEN Project 3.