LONDON METROPOLITAN UNIVERSITY

Islington college
(इस्लिङ्टन कलेज)

Module Code & Module Title

CC5004NI Security in Computing

Assessment Weightage & Type

30% Individual Coursework 2

Year and Semester

2023 -24 Spring

Student Name: Prithak Babu Shrestha

London Met ID: 22067084

College ID: np01nt4a220180

Assignment Due Date: May 7th, 2024

Assignment Submission Date: May 7th, 2024

Word Count (Where Required): 5598

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Table of Contents

Contents

Table of Figures

Abstract

This Assignment was provided to us as a Coursework Assignment by our Security in Computing module teacher Mr.Sushil Phuyal. This assignment proved to be very helpful as it has helped me improve my research and report-writing abilities. I would like to express my heartfelt gratitude to our module leader, Mr. Sushil Phuyal for giving us the chance to write this report which is essential for academic purposes.

I would like to appreciate my friends, teachers, websites, journals and articles for providing me with the required information and skill as well as access to all the necessary materials to accomplish this coursework.

1. Introduction

Denial-of-service (DoS) attacks are security risks in which an attacker prevents legitimate users from accessing computer systems, networks, services, or other IT resources. These attacks often flood web servers, systems, or networks with traffic, overloading the victim's resources and making it difficult or impossible for others to use them. Flooding attacks are more harder to recover from than distributed denial of service DDoS attacks, which exploit flaws in networking protocols and how they manage network traffic (Ferguson, 2021).

DoS and DDoS attacks target one or more of the OSI model's seven levels. The most popular OSI goals are Layer 3, Layer 4, Layer 6, and Layer 7. Protocol handshakes started by Internet of Things (IoT) devices are regularly used to launch attacks on Layers 6 and 7, which can be difficult to detect and prevent due to their recurrent existence and each being a separate intelligent client (Ferguson, 2021).

A DoS attack can be identified by slower or otherwise poor network performance, trouble accessing a website, or an increase in spam email. Researchers offer many techniques for preventing DoS and DDoS attacks, beginning with creating an incident response strategy ahead of time (Ferguson, 2021).

DoS and DDoS attacks use a number of techniques, including application layer, buffer overflow, DNS amplification, ping of death, state exhaustion, teardrop, and volumetric DoS attacks. Common DoS techniques include application layer, buffer overflow, DNS amplification, ping of death, state depletion, teardrop, and volumetric attacks (Ferguson, 2021).

2. Background

Denial of service attacks or DoS are cybercrimes where an internet site or a system is made unavailable by using a computer or computers to repeatedly make requests that tie up the site and prevent it from responding to legitimate users (Dennis, 2024).

The history of denial-of-service attacks on internet-connected systems has a long history, starting with the Robert Morris worm attack in 1988. A graduate student at Massachusetts Institute of Technology (MIT) released a self-reproducing piece of malware called the worm, which quickly spread through the internet and triggered buffer overflows and DoS attacks on the affected systems. Damage was estimated to be as high as $10 million, according to the U.S. General Accounting Office (GAO), now known as the Government Accountability Office. Morris was sentenced to 400 community service hours and three years' probation, as well as a $10,000 fine (Ferguson, 2021).

On February 7, 2000, a 15-year-old Canadian hacker, first orchestrated a series of DoS attacks against several e-commerce sites, including Amazon and eBay. These attacks were particularly designed to disrupt and overwhelm Internet commerce. Later, the U.S. Federal Bureau of Investigation (FBI) estimated that the affected sites suffered $1.7 billion in damages (Dennis, 2024).

Whereas, Distributed Denial of Service or DDoS attacks are those attacks where a hacker takes control over an array of computers known as bots or zombie net with the help of computer programs called Trojan horses. After the Trojan horse enters a user's system unknowingly. Then the malicious program at a predesignated time begins to send messages to a predetermined site, potentially tying up the selected site so effectively that little or no legitimate traffic can reach it (Dennis, 2024).

In October 2016 a botnet named "Mirai" brought down the servers of Dyn, an American company responsible for much of the Internet's domain name system (DNS). The attack interrupted much of North American Internet traffic and required security authorities to protect against such an environment.
In today's age software is insecure, making it easy for even unskilled hackers to exploit the vulnerabilities and compromise a vast number of machines. Although software companies regularly offer patches to fix vulnerabilities, not all users implement them, leaving their computers vulnerable to criminals wanting to launch DoS attacks (Dennis, 2024).

2.1 Types of DoS Attacks

2.1.1 UDP Flood Attacks

The User Datagram Protocol (UDP) is a protocol that transmits data packets without establishing a two-way connection with a server, making it vulnerable to flood attacks. Attackers transmit enough packets to overload a host that monitors its ports for genuine UDP traffic. The server uses the IP address and port encoded in UDP packets to detect applications, initiate automated operations, and target computers inside a network (Kime, 2022).

UDP flood attacks also include UDP Fragmentation Flood which sends bigger, fragmented packets to the target server, the server tries to assemble the unconnected, forged, and fragmented UDP packets, but may get overloaded in the process (Kime, 2022).

Specific UDP Amplification Attacks make a genuine UDP request to a large number of legitimate servers, spoofing the target server's IP address. The responses that come from these genuine servers quickly overload the targeted device. NTP, SNMP, and SSDP are some of the protocols commonly utilized in amplification attacks (Kime, 2022).

2.1.2 ICMP (Ping) Flood Attacks

ICMP attacks consume both incoming and outgoing bandwidth since all impacted servers constantly attempt to respond with ICMP echo reply packets, causing the entire system to crash or slow down. It is similar to UDP attacks, except it approaches and impacts the target through an ICMP echo request packet and sends at a fast transmission rate rather than waiting for a response.

2.1.3 Slowloris

This attack can cause a significant slowdown by enabling one web server to bring down another without affecting other host network services. It establishes multiple connections to the host server, transmitting only partial HTTP headers, and persistently transmits more headers. The host system maintains open ports or services for this false connection, affecting space for legitimate requests. This slowdown results in a system slowdown by overwhelming the concurrent connection range.

2.1.4 Ping of Death

Ping of Death attack exploits the maximum IP packet length of 65,535 bytes. Attackers send multiple IP fragments that comply with Ethernet's frame size but

assemble into a packet exceeding the maximum length. The recipient computer can overflow memory buffers or crash the computer as it reorganizes the fragments (Kime, 2022).

2.1.5 HTTP Flood Attacks

HTTP Flood attacks exploit HTTP commands to overflood websites, servers, and bandwidth. Botnets send multiple requests simultaneously, increasing traffic for the target website (Kime, 2022).

GET attacks involve sending large concurrent GET requests for large files,

POST attacks involve sending large POST requests containing large files for storage on the target server.

Low-and-Slow POST attacks involve using the R-U-Dead-Yet? tool, send HTTP Post requests that indicate they will send large amounts of data but send tiny bits of data slowly, avoiding DDoS defenses and tying up resources on the server.

Single Session or Single Request Attacks involve exploiting a loophole in HTTP 1.1 to include multiple different requests within a single HTTP packet.

Fragmented HTTP Flood attacks involve splitting HTTP packets into tiny fragments sent as slowly as the server allows, using a packet rate safe for DDoS defenses but consuming reserved bandwidth.

Recursive GET Flood attacks involve requesting long lists of pages or images, chewing up resources that cannot be used for legitimate traffic.

Random Recursive GET Flood involves randomizing requested pages to avoid detection.

2.1.6 SYN Flood Attacks

A SYN flood attack involves an attacker sending a large number of SYN packets to all available ports on a victim server, often using a fake IP address. The server, unaware of the attack, receives multiple legitimate requests to establish communication and responds with a SYN-ACK packet from all open ports. The malicious client either fails to send the expected ACK or never receives the SYN-ACK. The server waits for acknowledgment of its SYN-ACK packet, unable to close down the connection, leaving the connection open. As the server's connection overflow tables fill, service to legitimate clients is denied, and the server may even malfunction or crash (Yadav, 2020).

3. Demonstration

3.1 Tools Used

3.1.1 Wireshark

Wireshark is an open-source network analyzer that provides real-time network traffic details for troubleshooting issues, analyzing protocols, and ensuring network security. It is popular among academic institutions, government agencies, corporations, and nonprofits. Key features of Wireshark include packet capture (PCAP), real-time analysis, filtering capabilities, and a user-friendly GUI. PCAP converts network traffic into a human-readable format, providing immediate insights into ongoing activities. Real-time analysis offers a live view of network traffic, allowing users to focus on specific types of traffic for more efficient analysis. The GUI is designed for ease of use, making it suitable for both beginners and experts (Hanna, 2024).

3.1.2 System Monitor

The gnome-system-monitor lets you see and control the processes running on your system. We can examine detailed memory maps, send signals, and stop processes. Furthermore, gnome-system-monitor gives an overview of your system's resource utilization, such as memory and CPU allocation.

3.1.3 bWAPP

bWAPP, a buggy web application, is a free and open-source platform that has been designed to be unsafe, making it a useful tool for security enthusiasts, developers, and students. Its major purpose is to help with the discovery and mitigation of online vulnerabilities. bWAPP provides a hands-on learning environment that prepares learners with practical skills required for effective penetration testing and ethical hacking projects. Users may learn to recognize and manage possible security concerns by practicing with simulated vulnerabilities, thereby improving their capacity to protect online applications (Nagaraj, 2023).

3.2 UDP Flood

Step 1: Confirming the Attacker assigned IP Address.

Figure 1: Screenshot of checking Attacker IP Address.

Step 2: Confirming the victim's assigned IP Address

Figure 2: Screenshot of checking Victims IP Address.

Step 3: Checking the system monitor before UDP Flood.

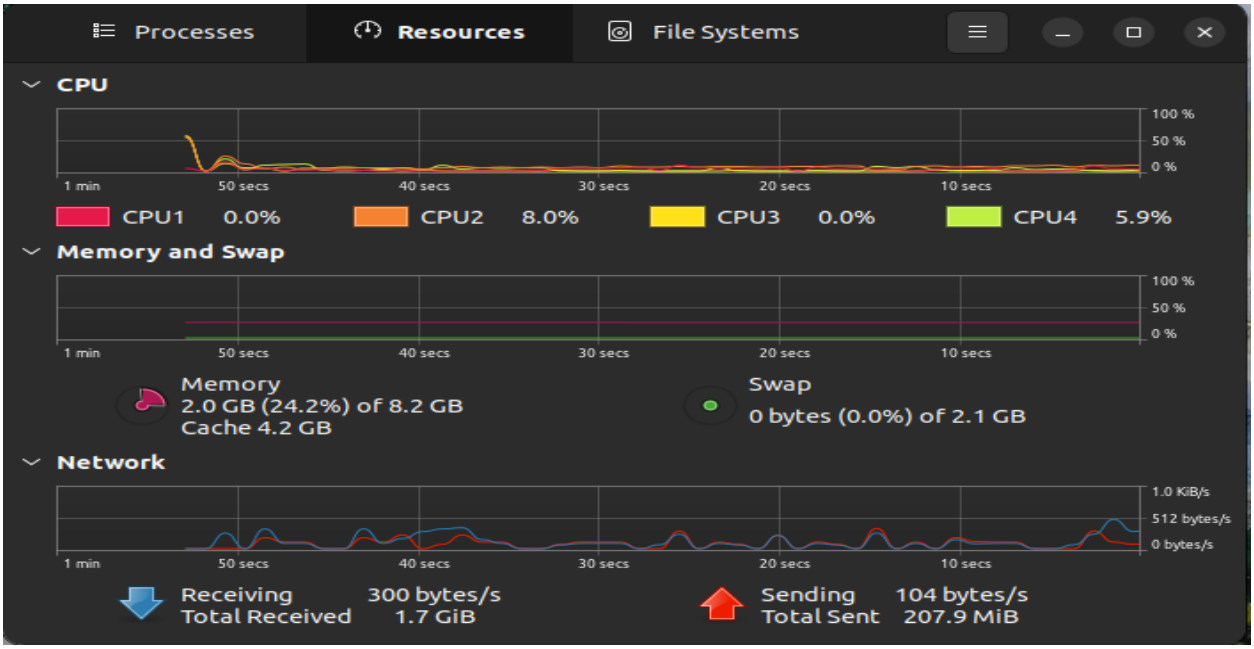

Figure 3: Screenshot of System Monitor before UDP Flood.

Step 4: Checking Wireshark for network traffic before UDP attack.

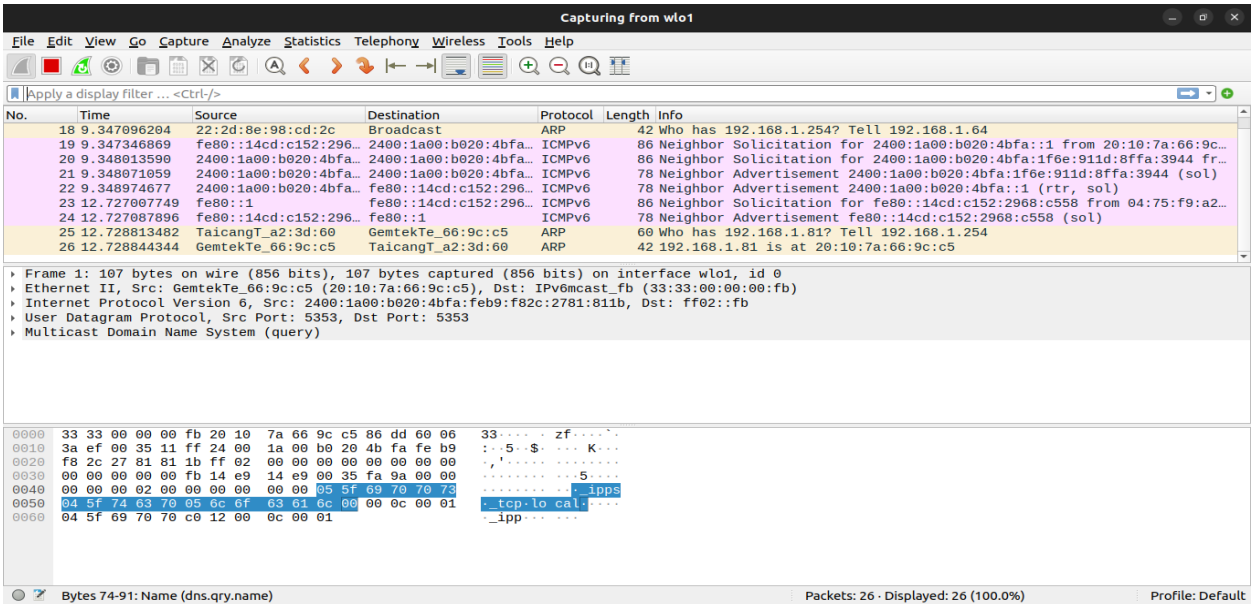

Figure 4: Screenshot of Wireshark before UDP Flood.

Step 5: Starting the UDP Flood by assigning the victim's IP and port.

Figure 5: Screenshot of starting the UDP Flood.

Step 6: Checking the System Monitor, if the network resources are being consumed abnormally.

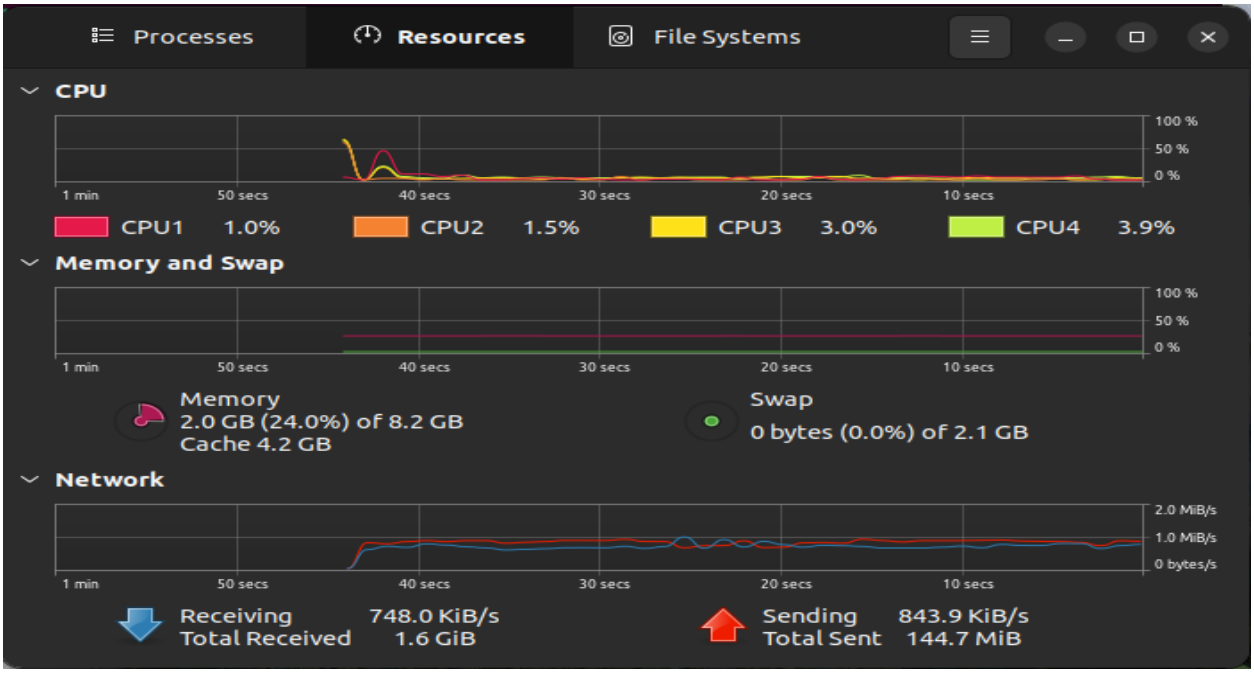

Figure 6: Checking the System Monitor During UDP Flood.

Step 7: Checking Wireshark, if there is network traffic from the attackers IP Address.

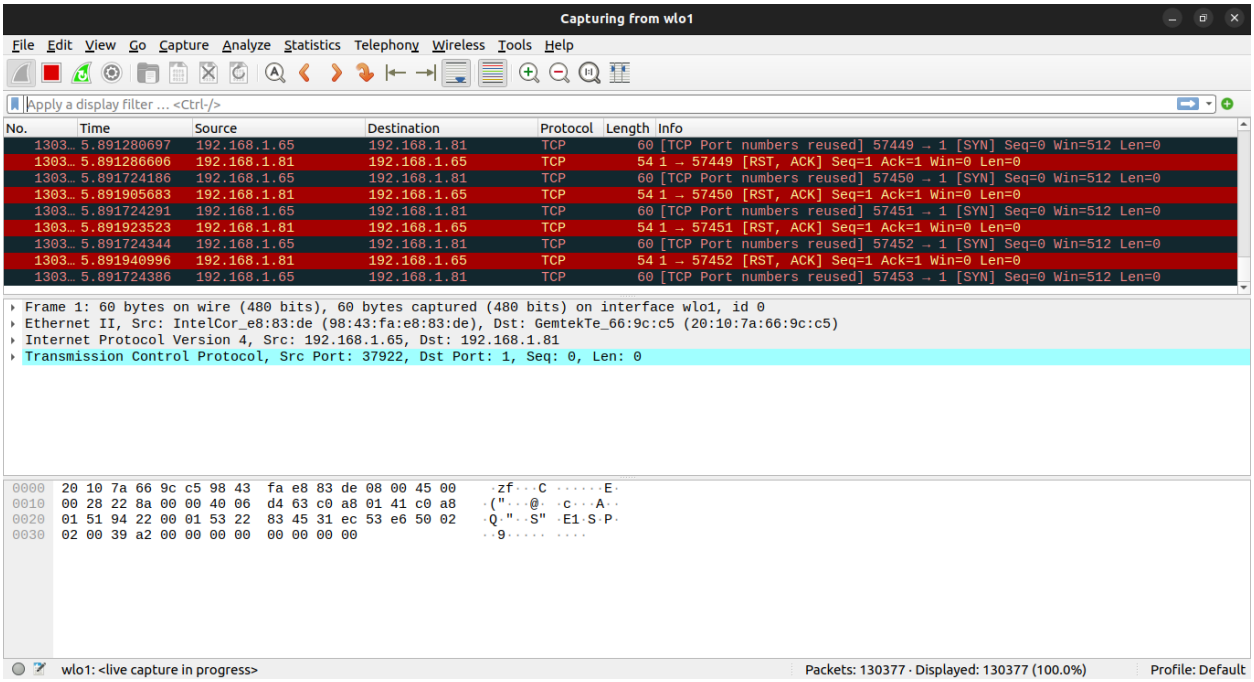

Figure 7: Screenshot of Wireshark during UDP Flood.

Step 8: Stopping the UDP Flood Attack.

Figure 8: Screenshot of stopping the UDP Flood.

Step 9: Checking the System Monitor.

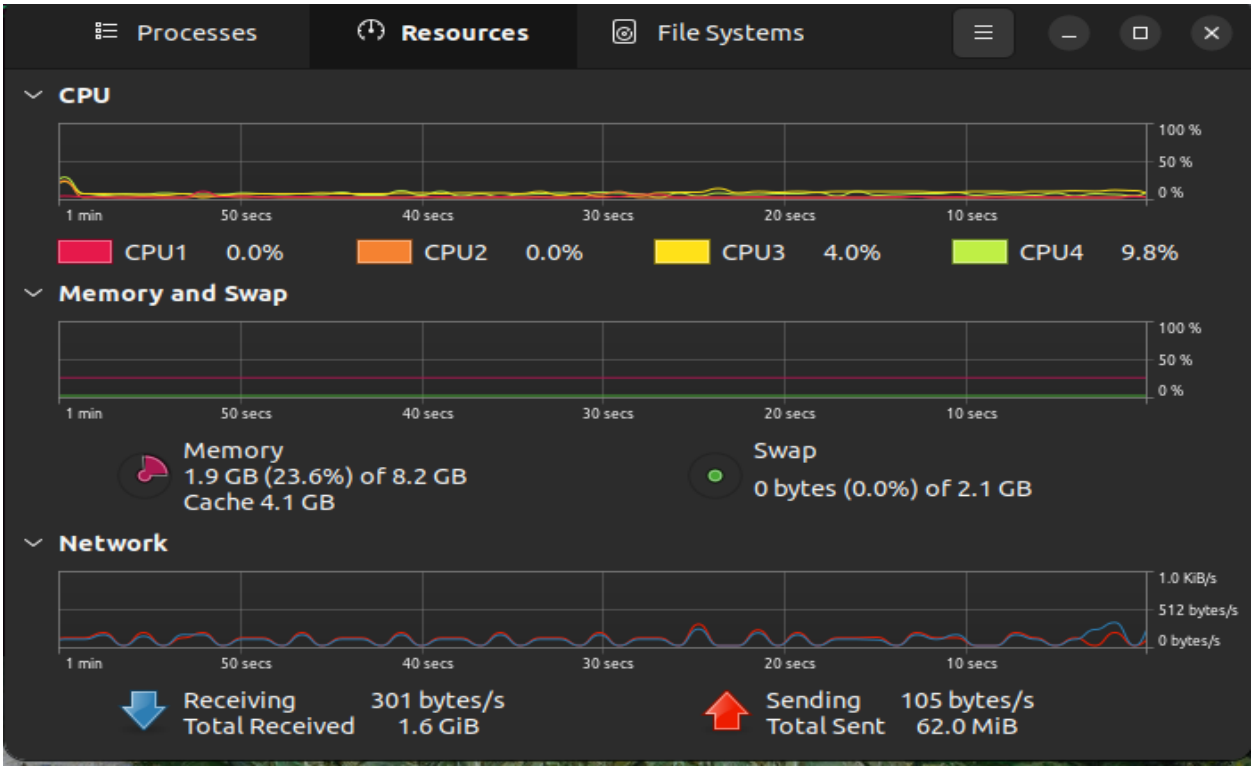

Figure 9: Screenshot of System Monitor after UDP Flood.

Step 10: Checking Wireshark.

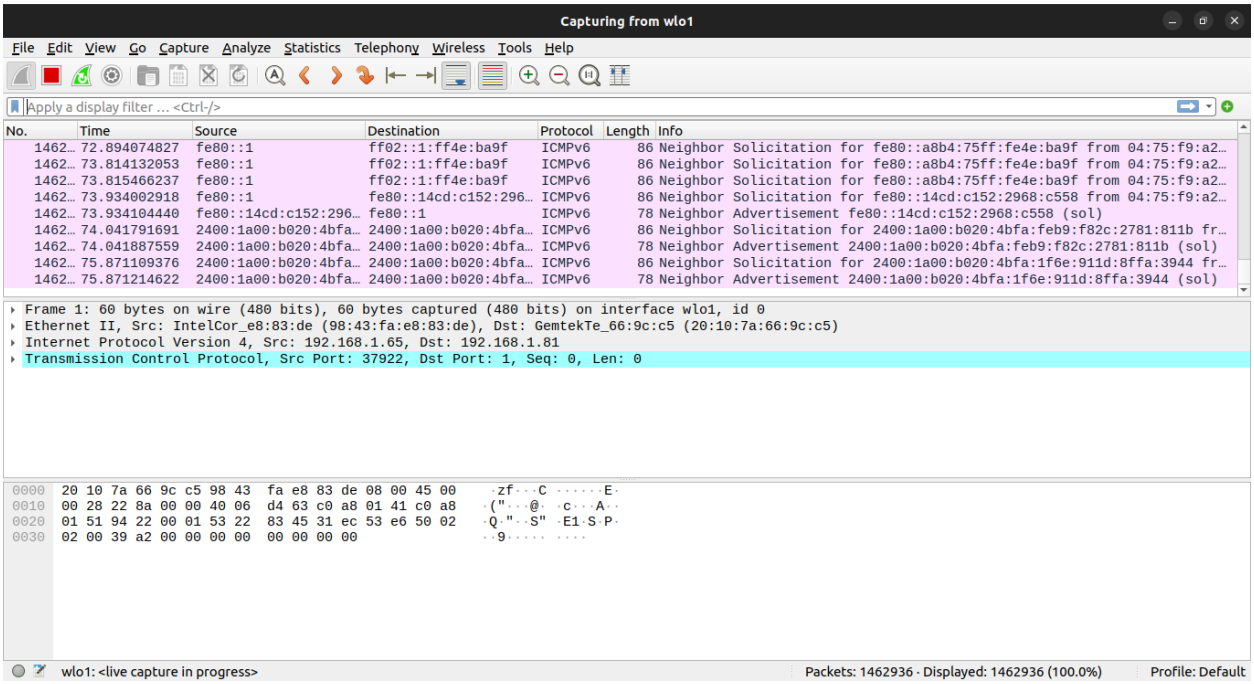

Figure 10: Screenshot of Wireshark after UDP Flood.

3.3 Slowloris

Step 1: Confirming the Attacker assigned IP Address.

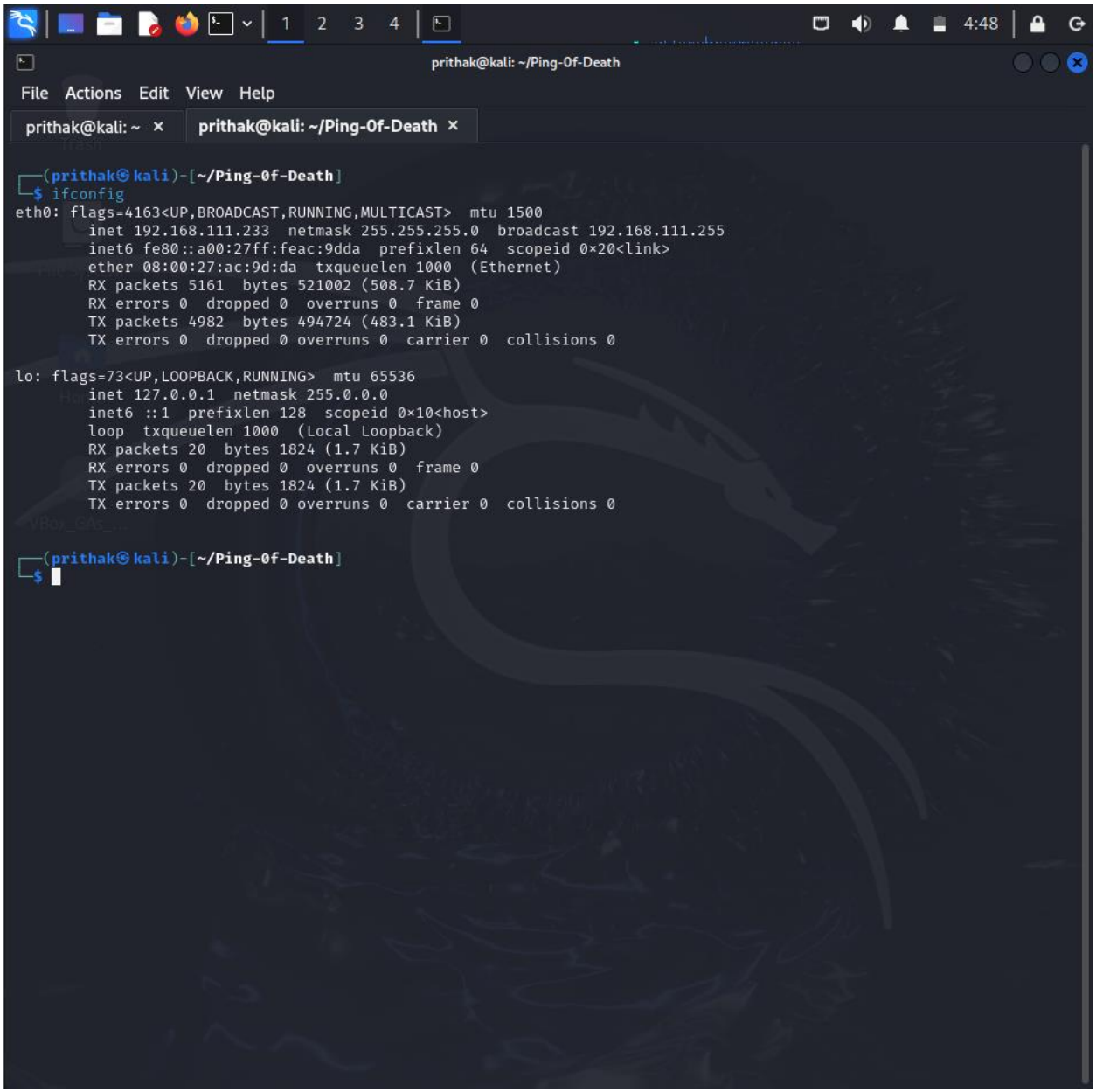

Figure 11: Confirming the Attacker assigned IP Address.

Step 2: Confirming the victim's assigned IP Address

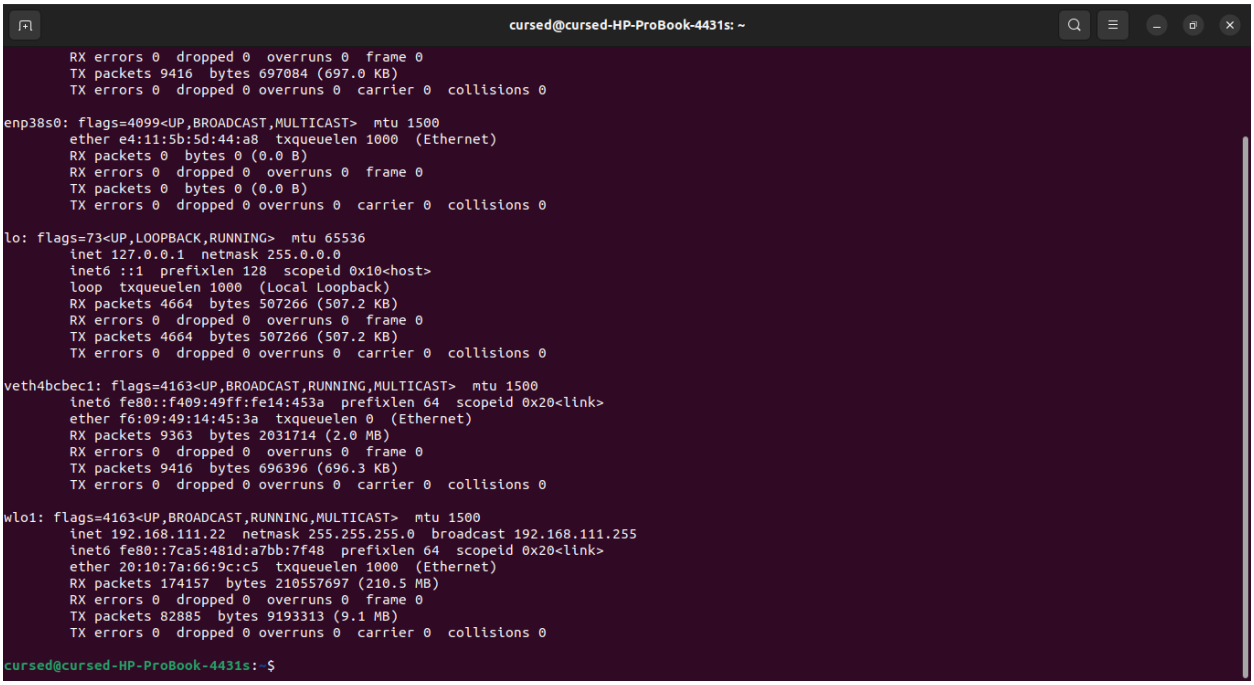

Figure 12: Screenshot of checking Victims IP Address.

Step 3: Hosting a Website Using bWAPP for testing.

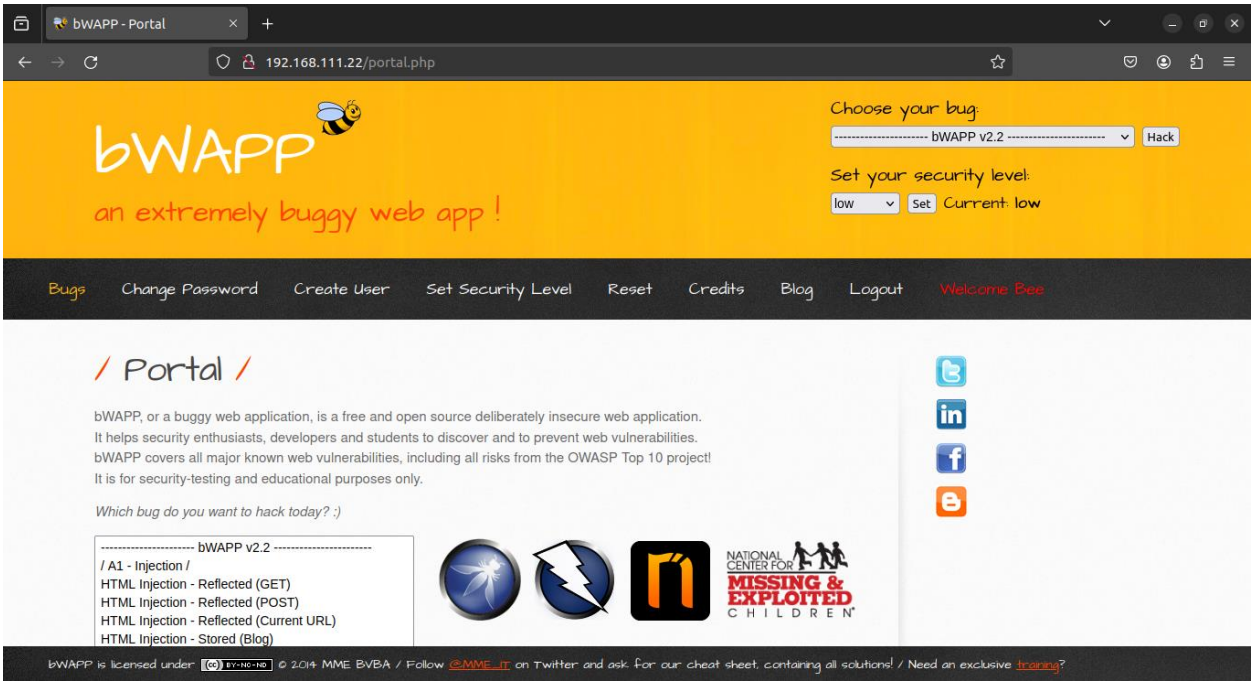

Figure 13: Screenshot of the website before Slowloris.

Step 5: Starting the Slowloris by assigning the victim's IP Address.

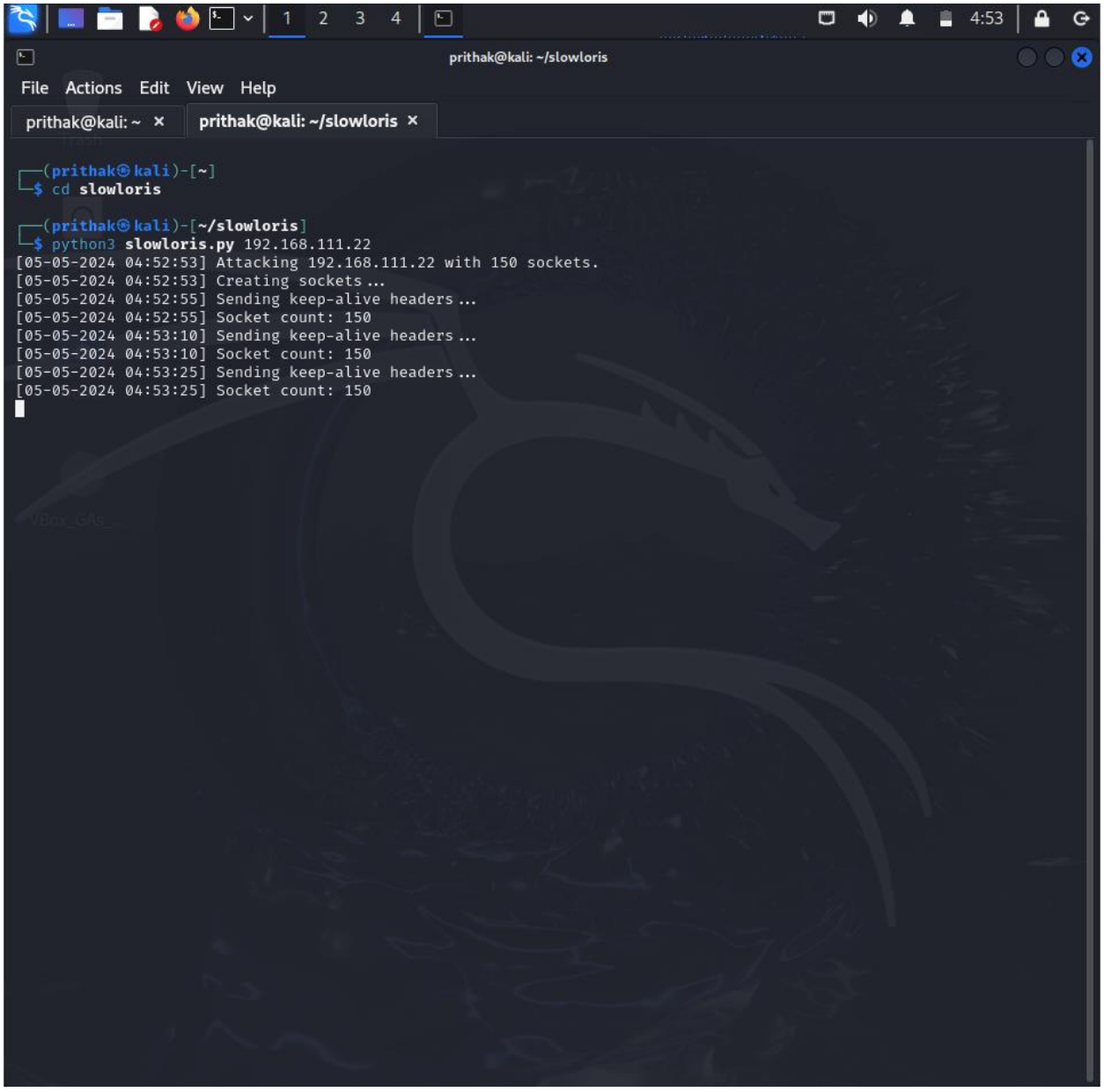

Figure 14: Starting the Slowloris Attack.

Step 6: Checking the website, if website is working or not.

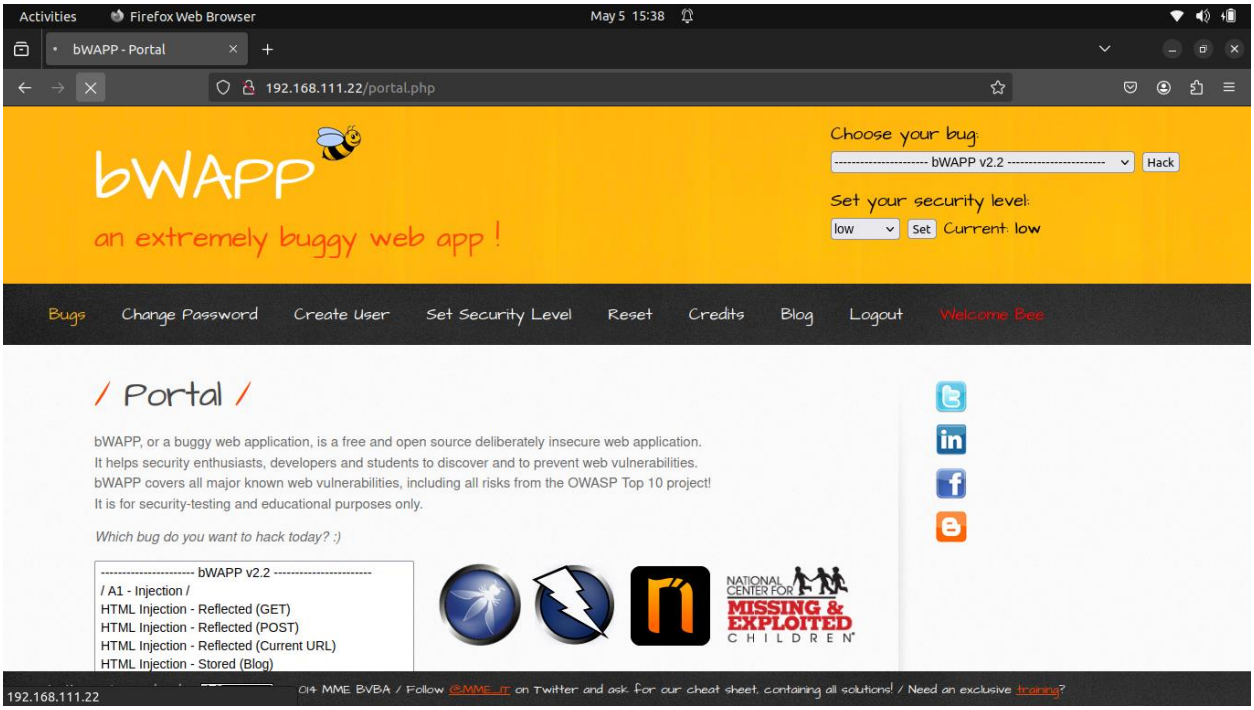

Figure 15: Screenshot of website during Slowloris Attack.

Step 7: Due to the Slowloris Attack the website is not functioning.

Figure 16: Screenshot of the website during Slowloris to check.

Step 8: Stopping the Slowloris.

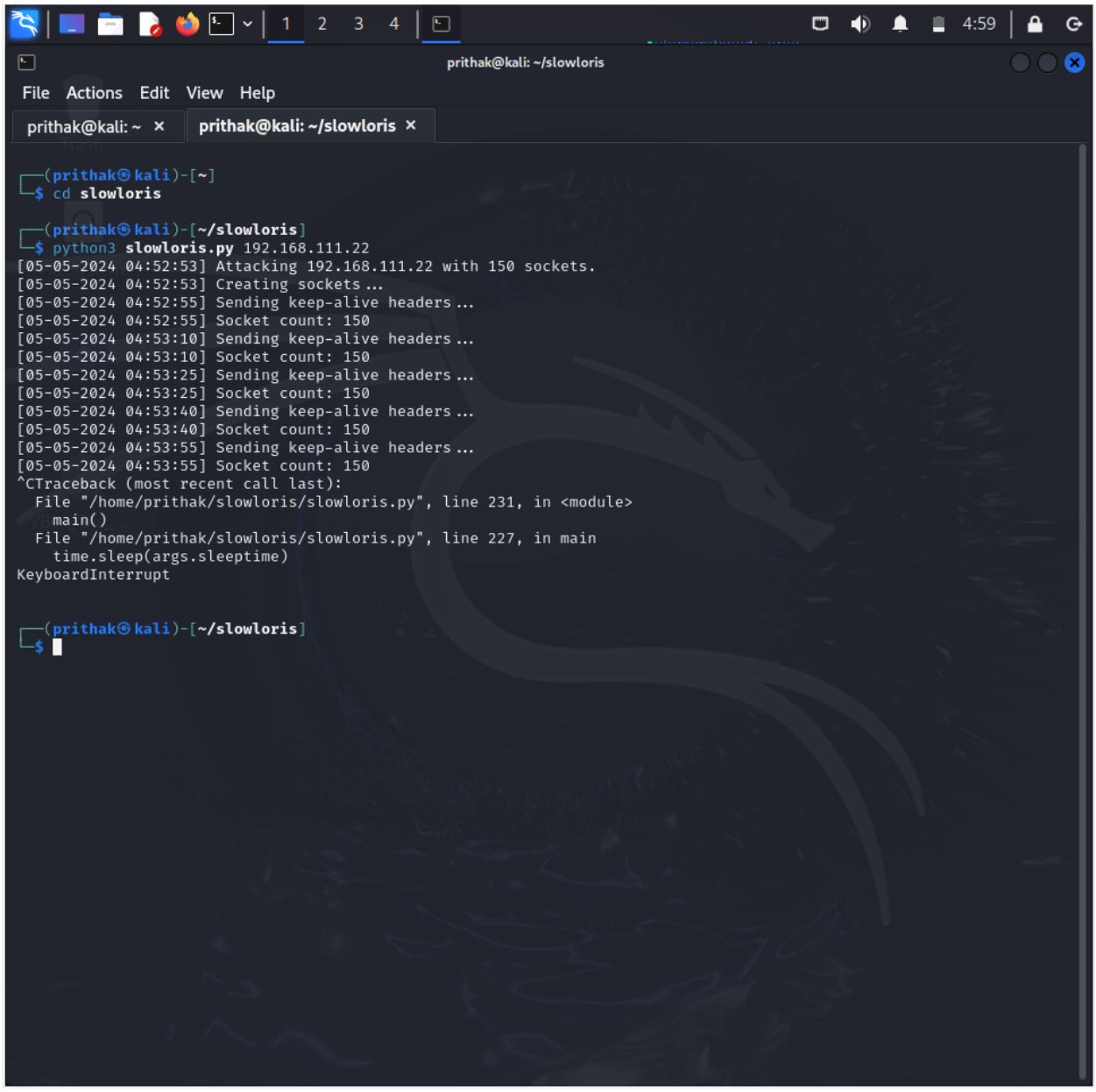

Figure 17: Screenshot of stopping Slowloris.

Step 9: Checking the website after stopping Slowloris.

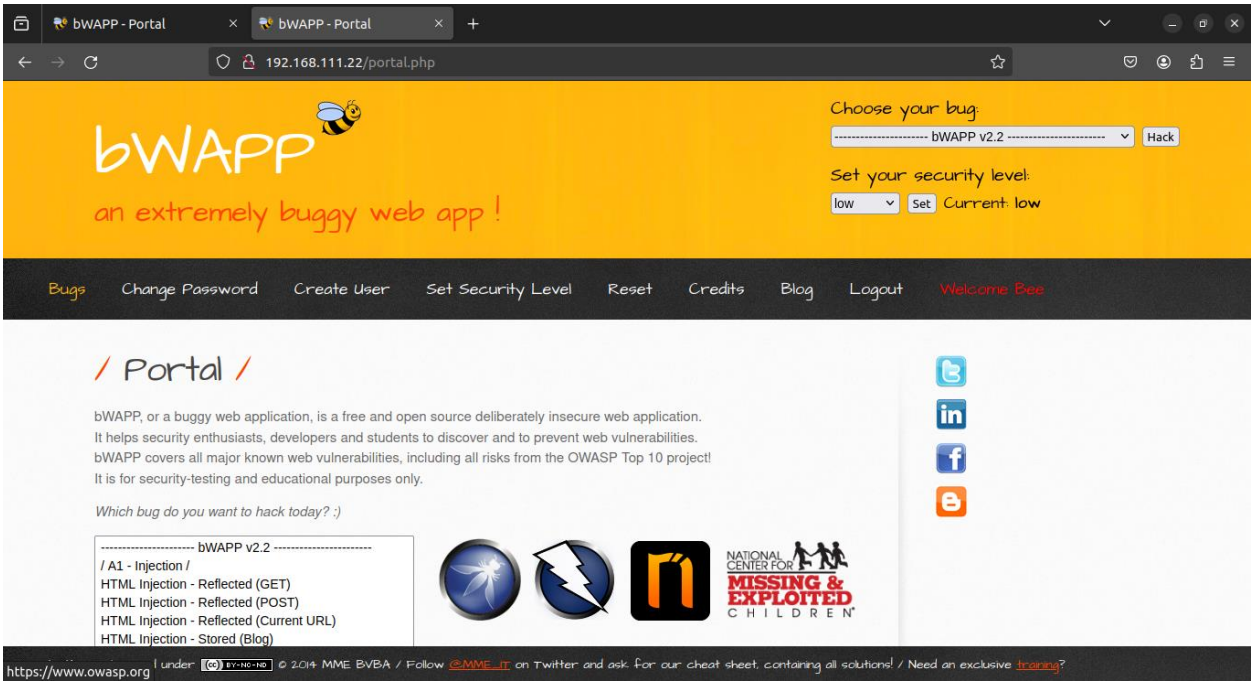

Figure 18: Screenshot of Website working after stopping Slowloris.

3.4 Ping of Death

Step 1: Confirming the Attacker assigned IP Address.

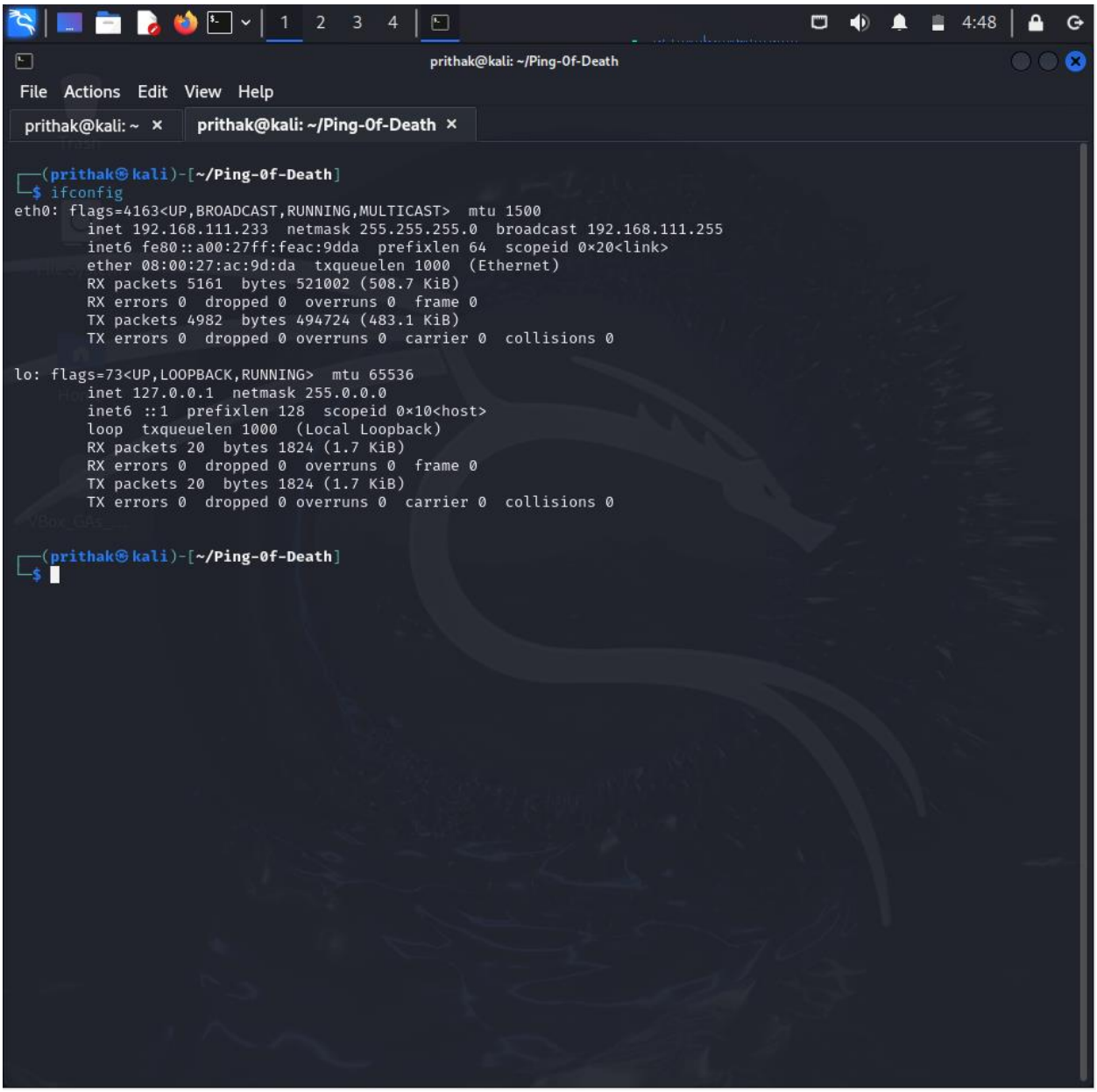

Figure 19: Screenshot of checking Attacker IP Address.

Step 2: Confirming the victim's assigned IP Address

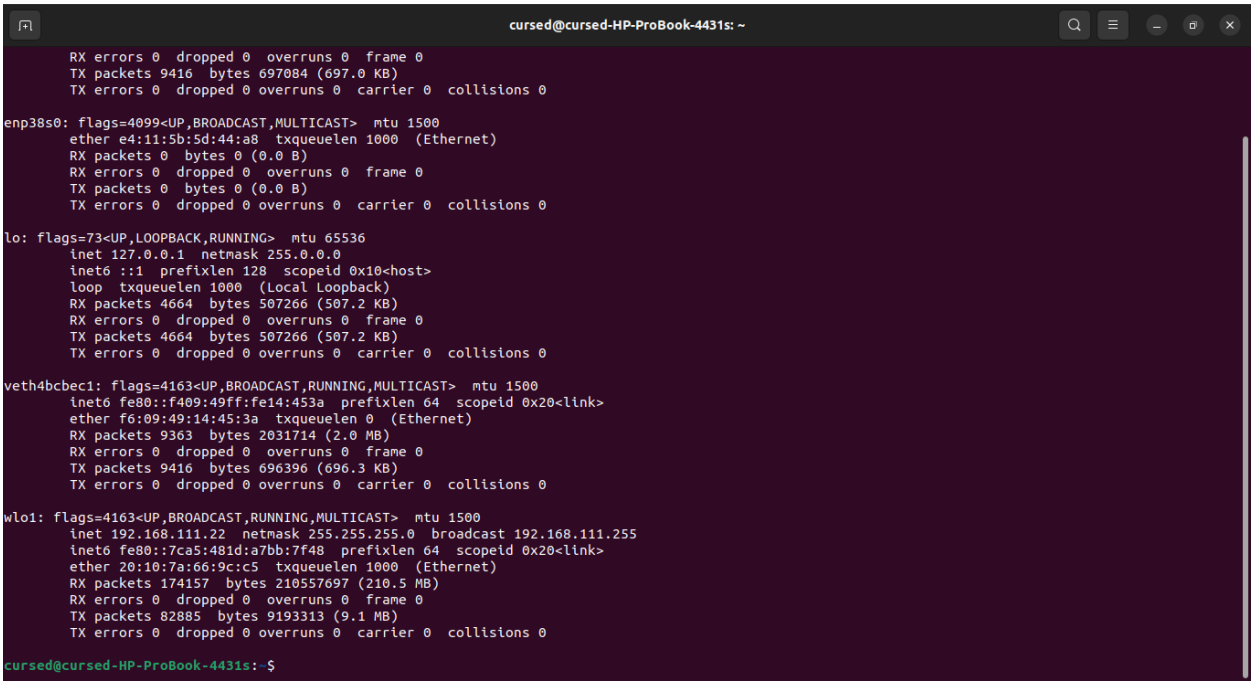

Figure 20: Screenshot of checking Victims IP Address.

Step 3: Checking the system monitor before PoD.

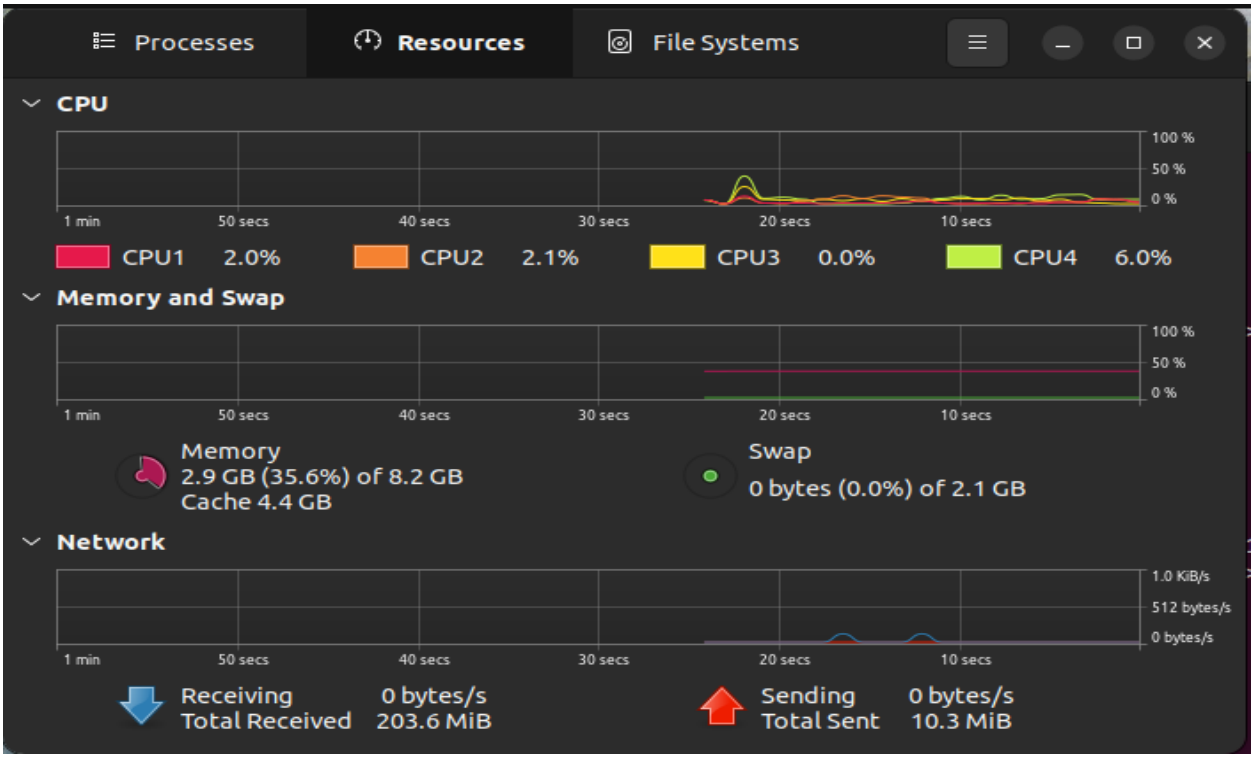

Figure 21: Screenshot of System Monitor before PoD.

Step 4: Checking Wireshark for network traffic before PoD.

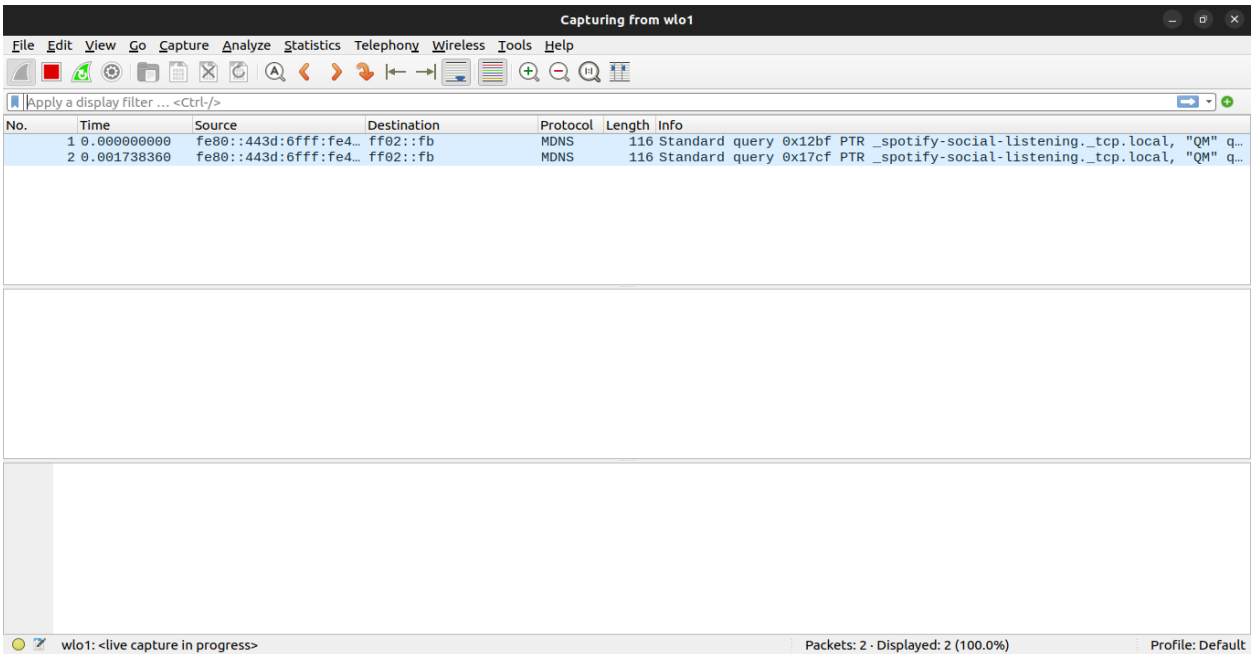

Figure 22: Screenshot of Wireshark before PoD.

Step 5: Starting PoD by assigning the victim's IP and severity of the attack.

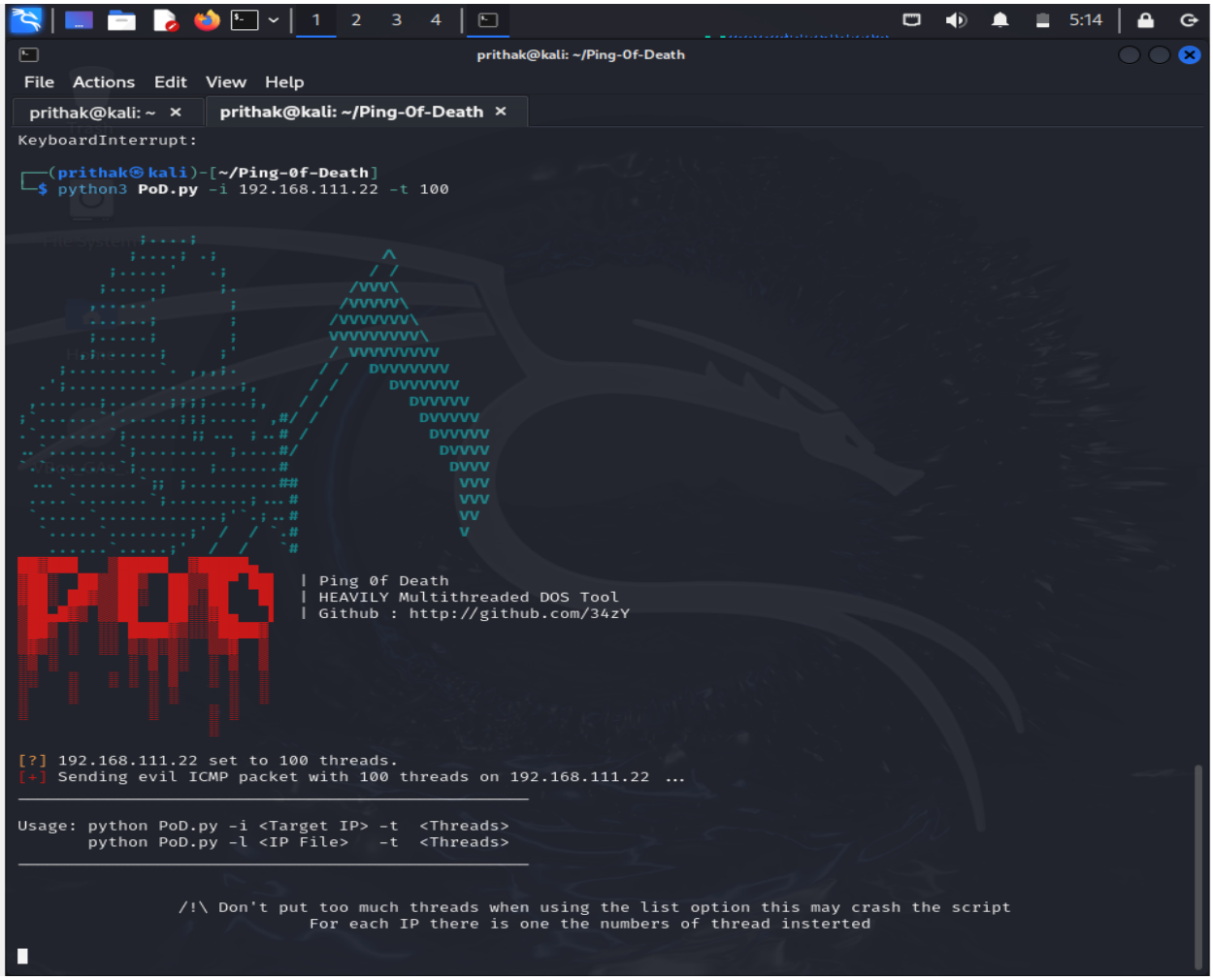

Figure 23: Screenshot of starting PoD.

Step 6: Checking the System Monitor, if the network resources are being consumed abnormally.

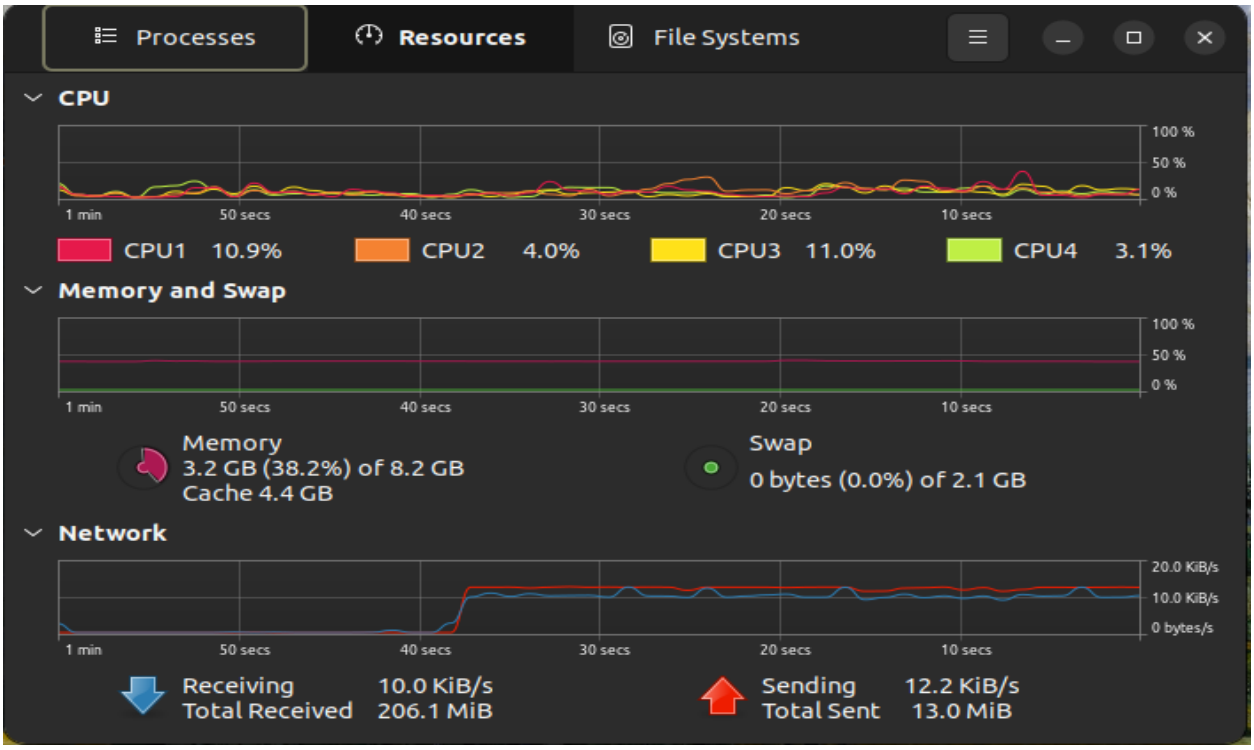

Figure 24: Screenshot of checking the System Monitor during PoD.

Step 7: Checking Wireshark, if there is network traffic from the attackers IP Address.

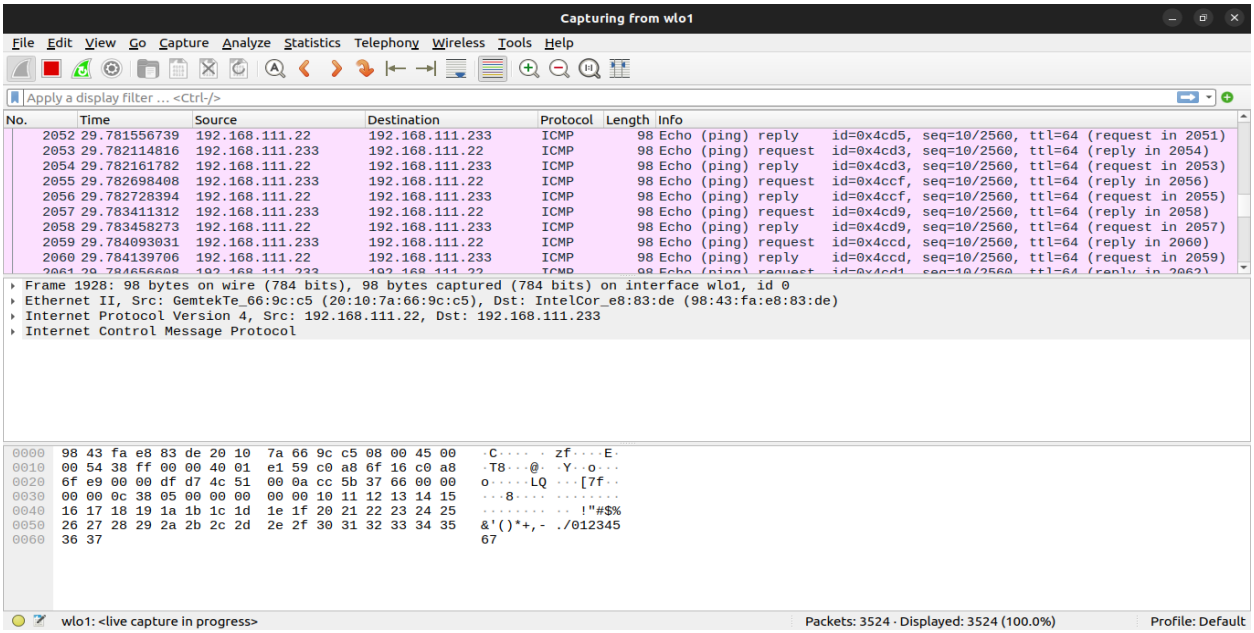

Figure 25: Screenshot of Wireshark during PoD.

Step 8: Stopping the PoD.

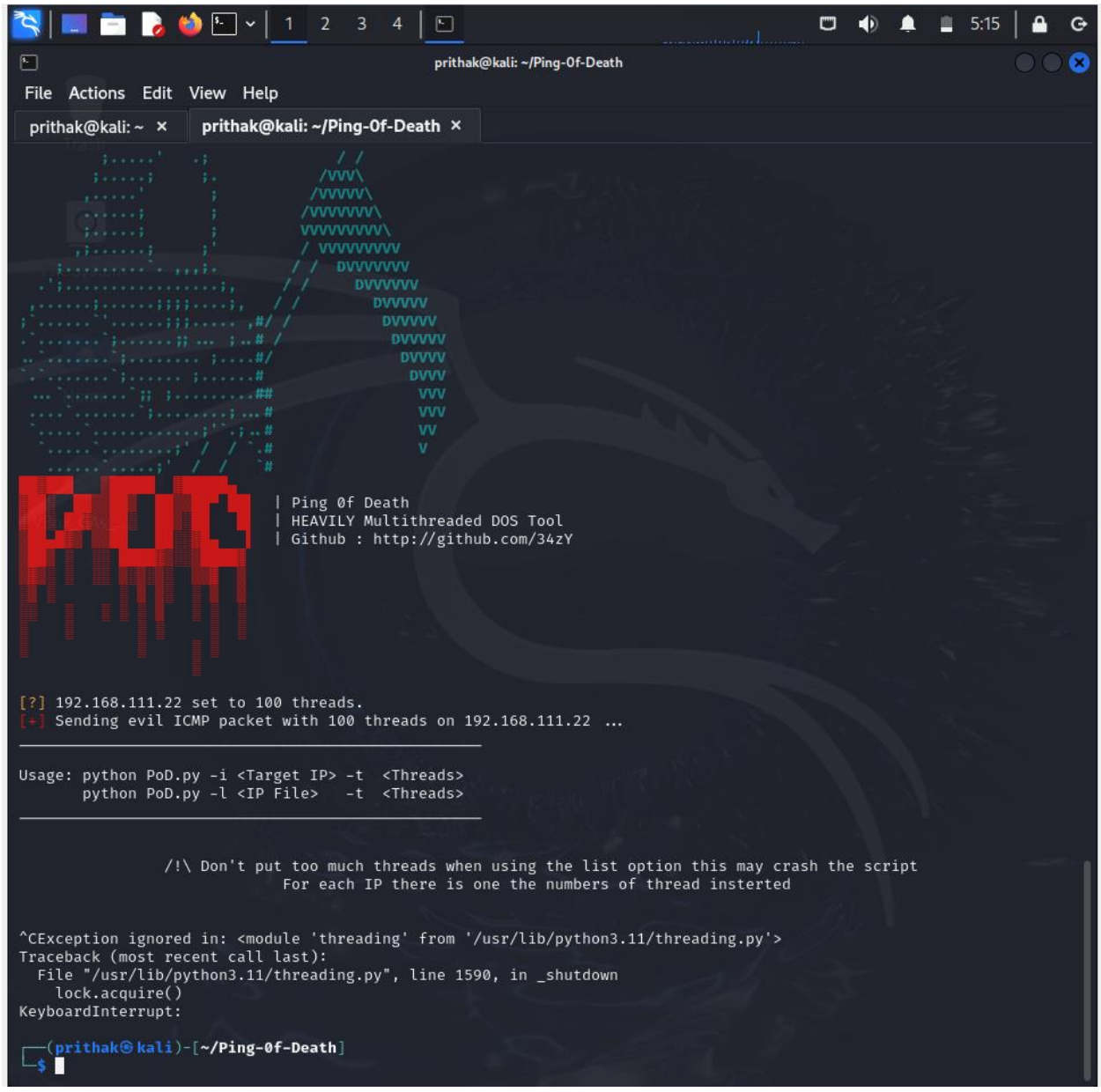

Figure 26: Screenshot of stopping PoD.

Step 9: Checking the System Monitor.

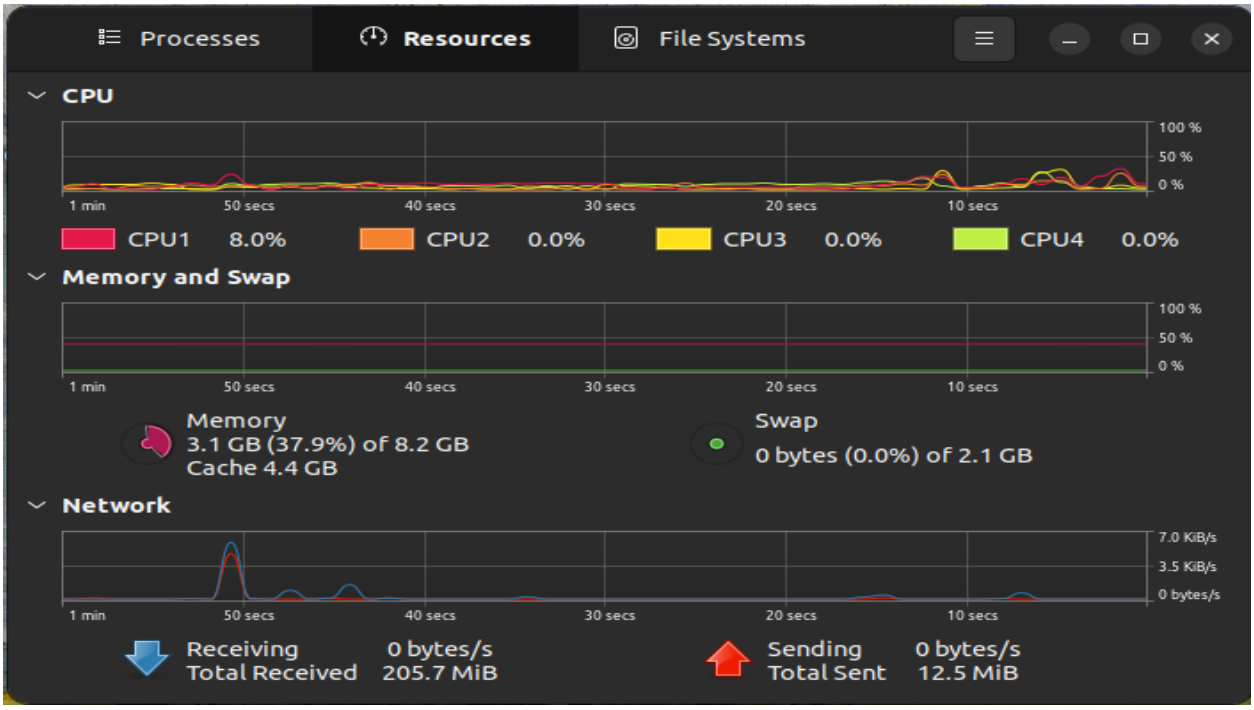

Figure 27: 9: Screenshot of System Monitor after PoD.

Step 10: Checking Wireshark.

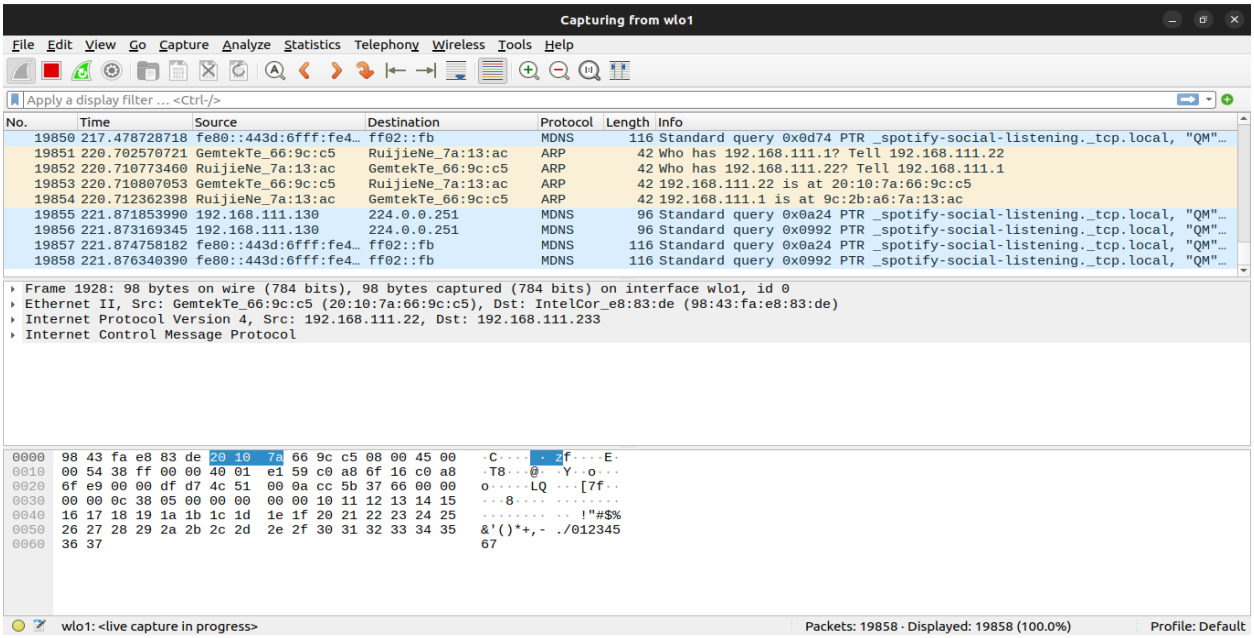

Figure 28: Screenshot of Wireshark after PoD.

3.5 HTTP Flood

Step 1: Confirming the Attacker assigned IP Address.

Figure 29: Screenshot of checking Attacker IP Address.

Step 2: Confirming the victim's assigned IP Address.

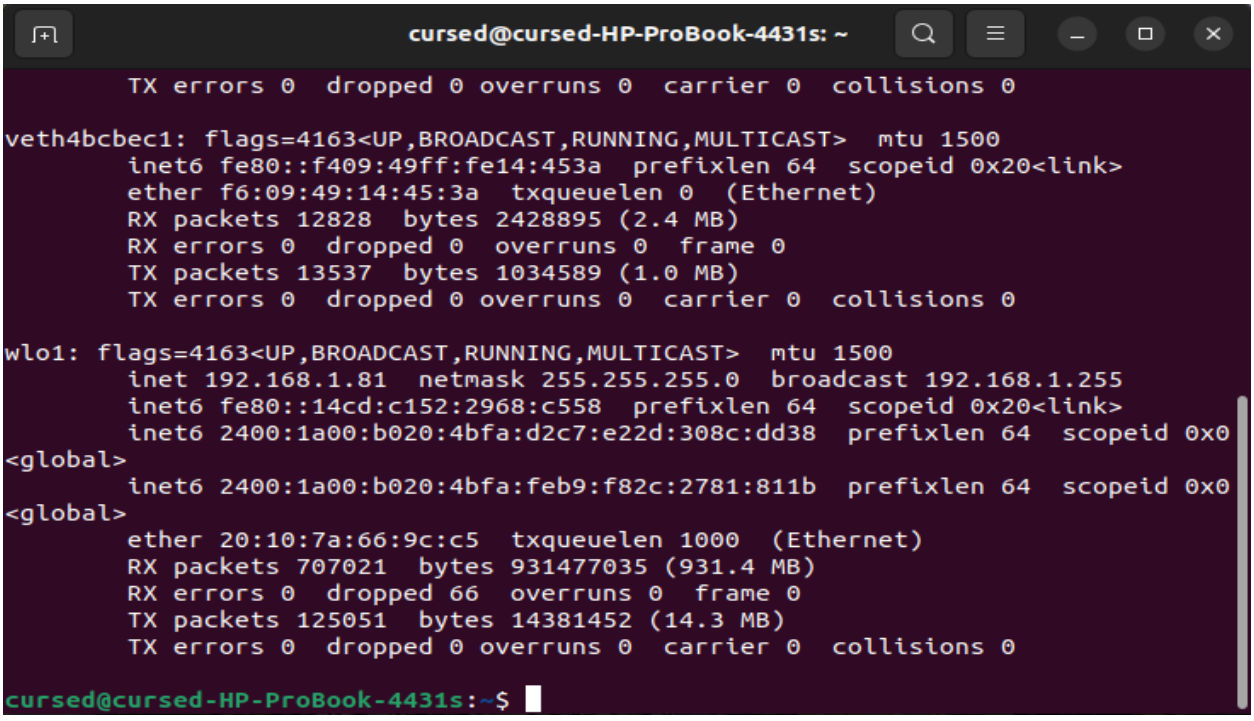

Figure 30: Screenshot of checking Victims IP Address.

Step 3: Checking the system monitor before HTTP Flood.

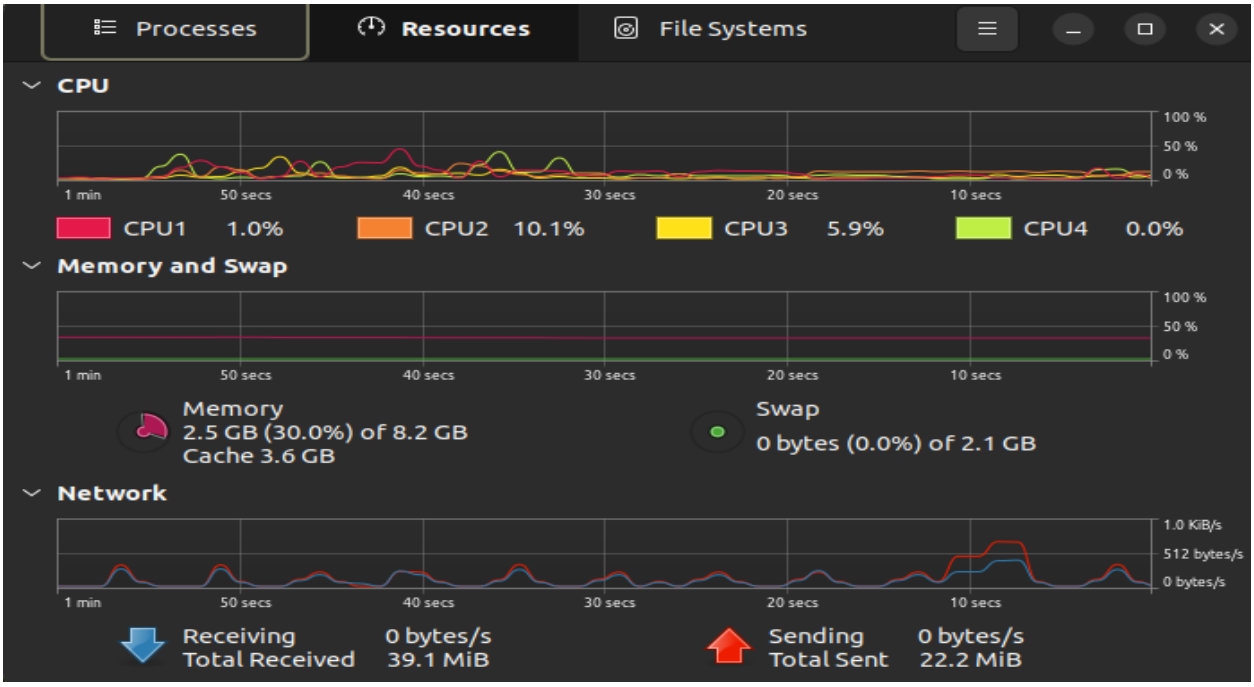

Figure 31: Screenshot of System Monitor before HTTP Flood.

Step 4: Checking Wireshark for network traffic before HTTP attack.

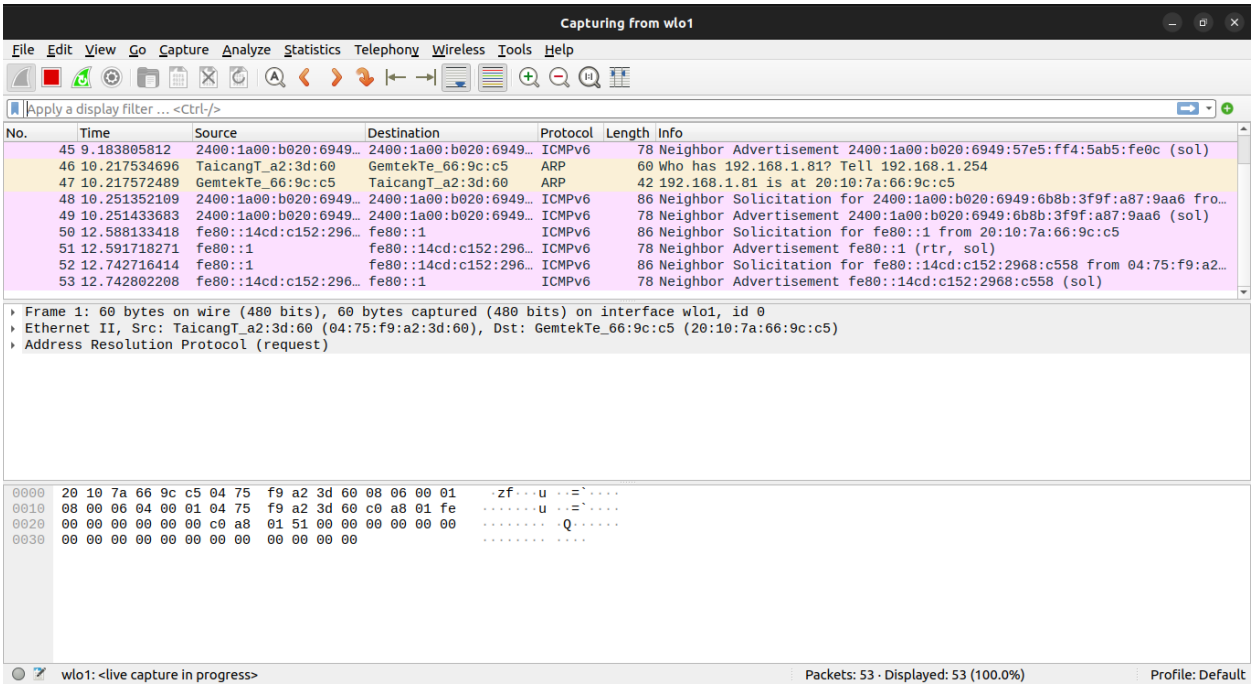

Figure 32: Screenshot of Wireshark before HTTP Flood.

Step 5: Starting the UDP Flood by assigning the victim's IP and the number of request to send.

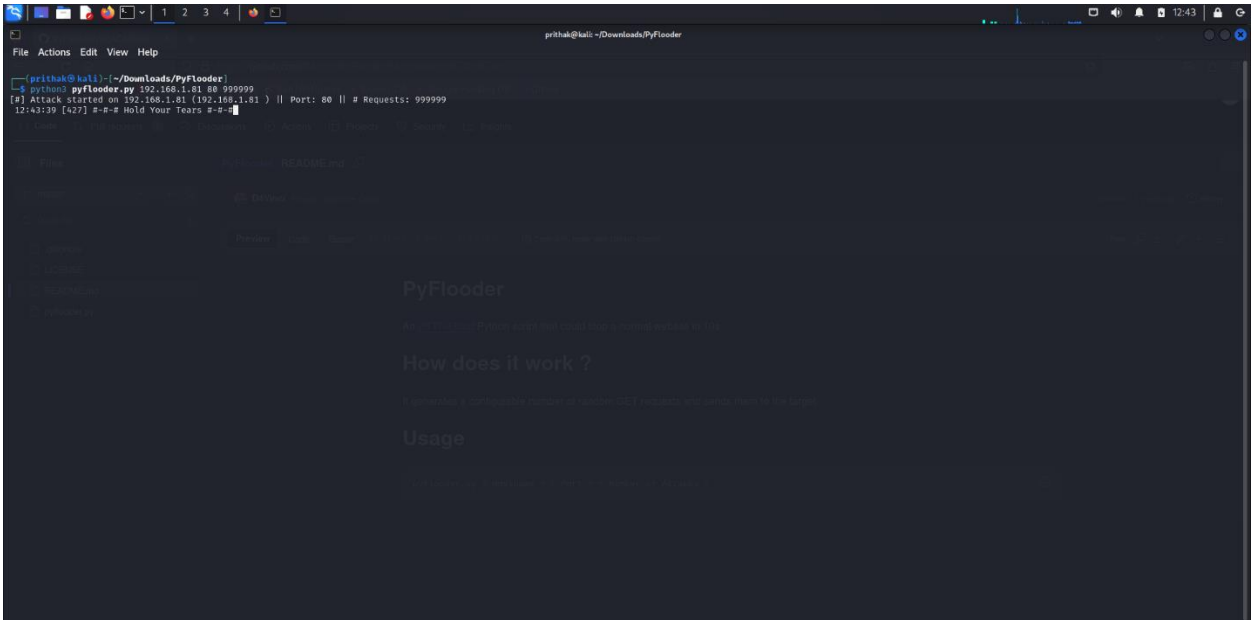

Figure 33: Screenshot of starting HTTP Flood.

Step 6: Checking the System Monitor, if the network resources are being consumed abnormally.

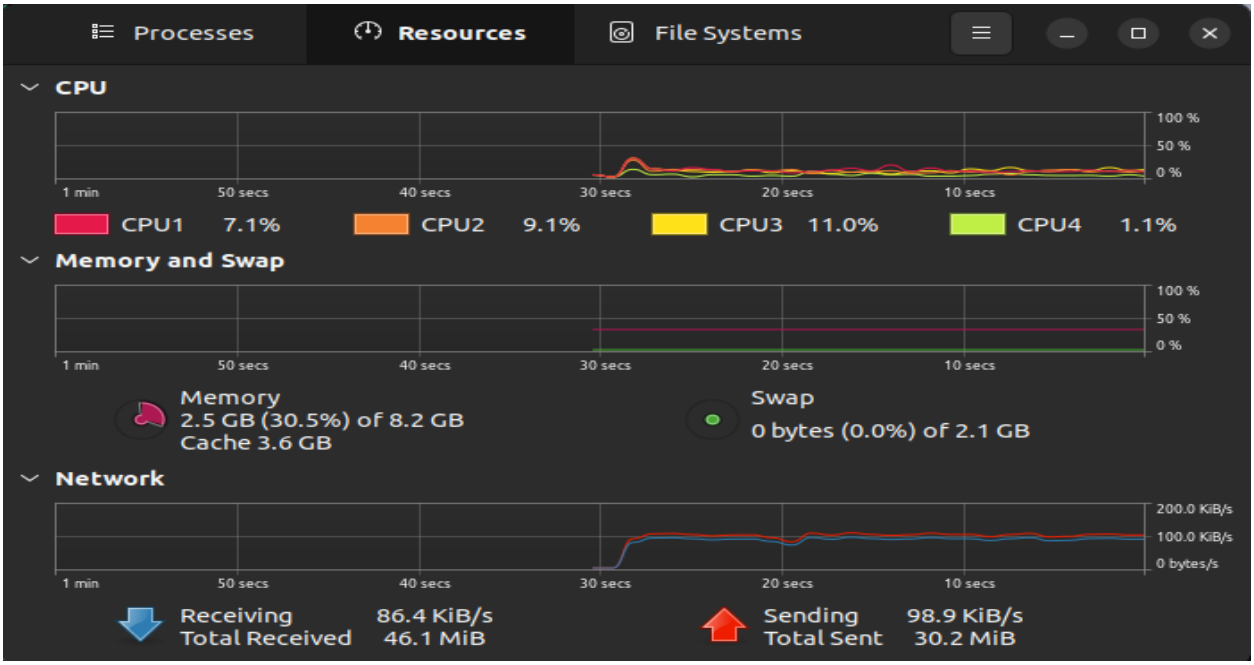

Figure 34: Screenshot of checking the System Monitor during HTTP Flood.

Step 7: Checking Wireshark, if there is network traffic from the attackers IP Address.

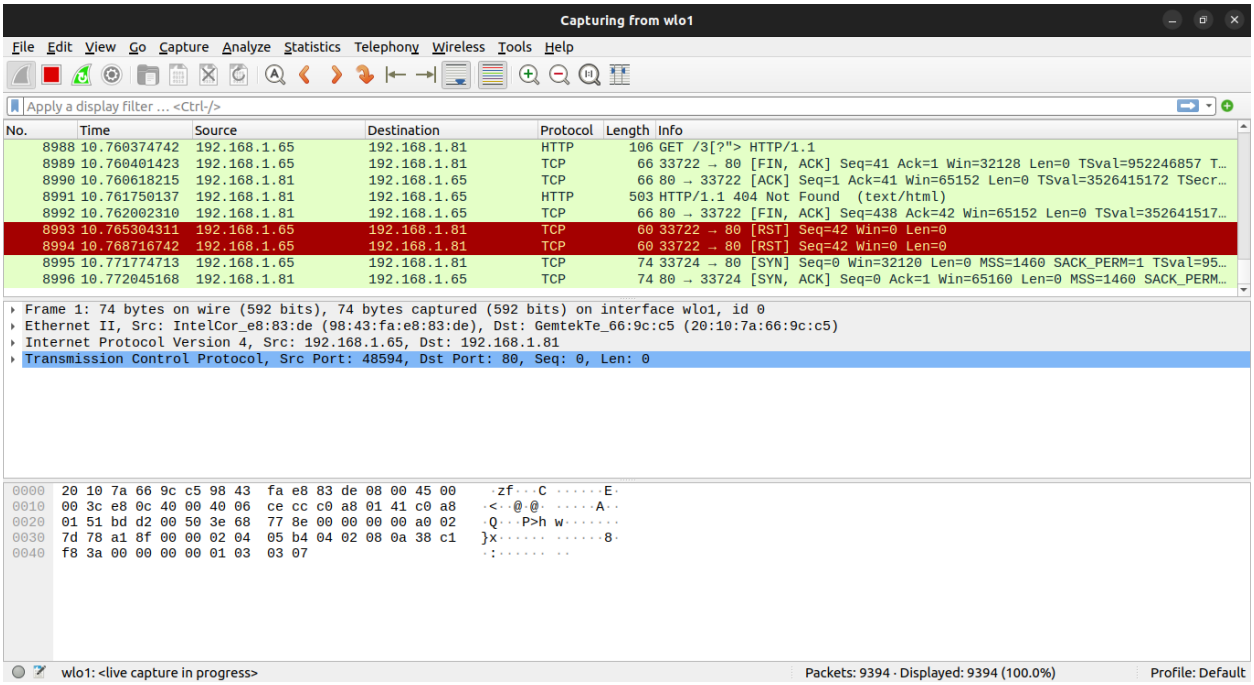

Figure 35: Screenshot of Wireshark during HTTP Flood.

Step 8: Stopping the HTTP Flood.

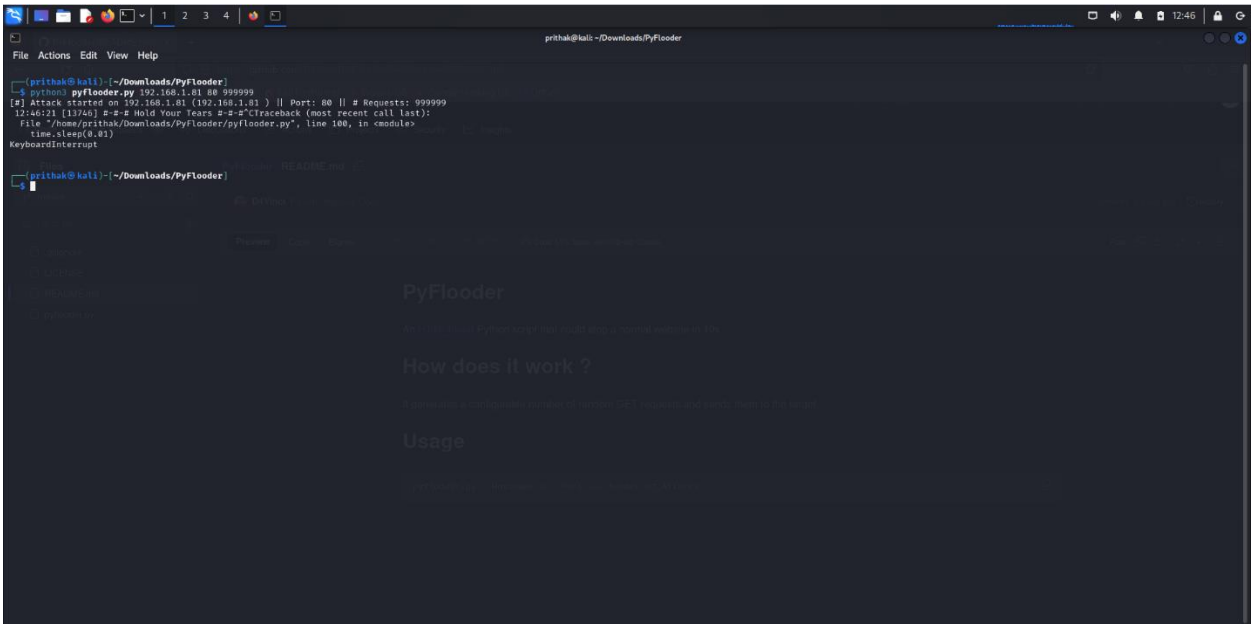

Figure 36: Screenshot of stopping the HTTP Flood.

Step 9: Checking the System Monitor.

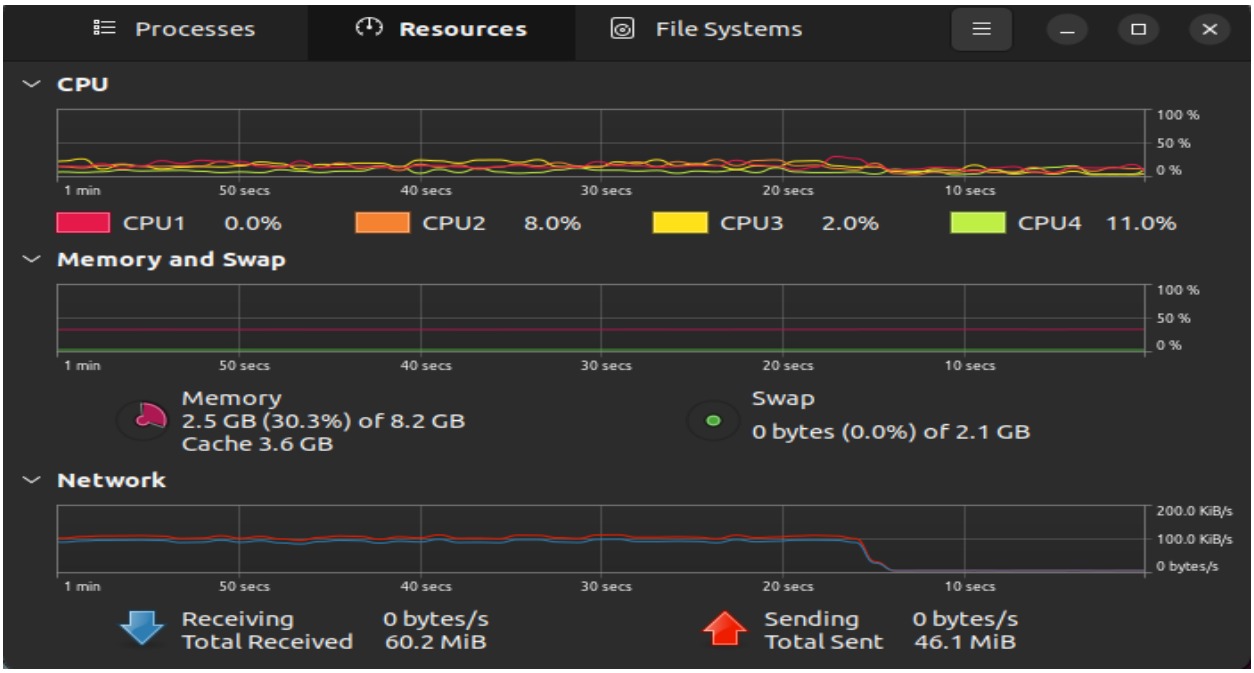

Figure 37: Screenshot of System Monitor after HTTP Flood.

Step 10: Checking Wireshark.

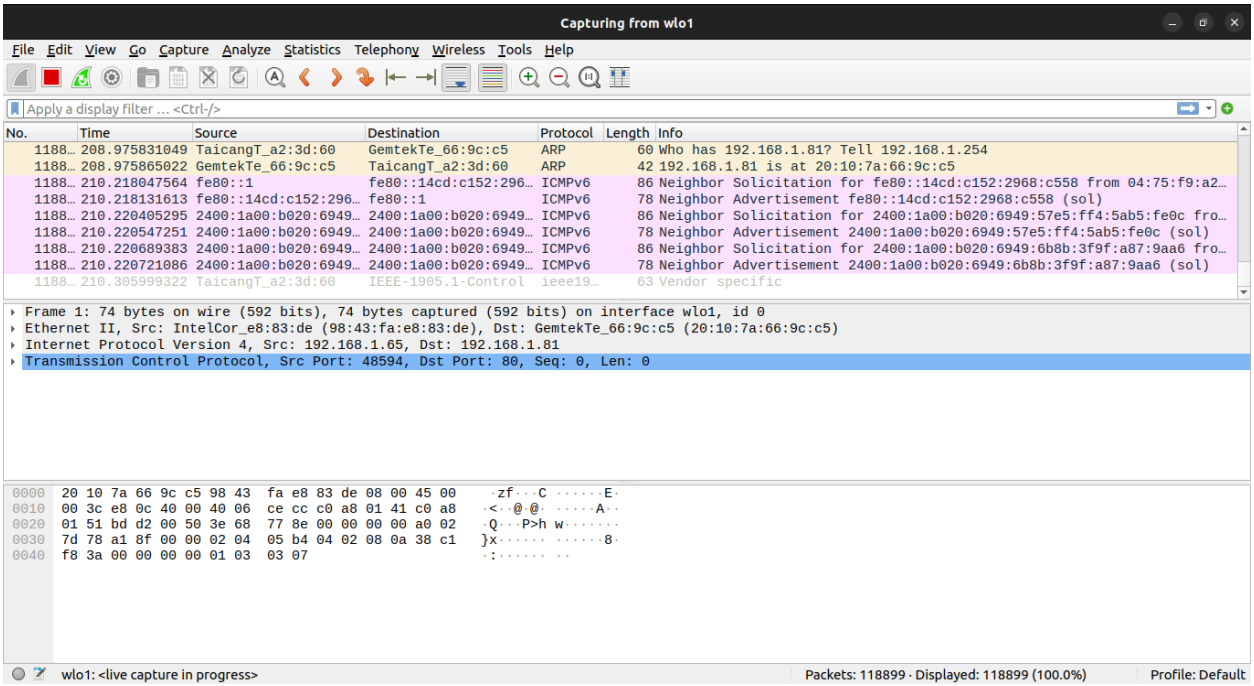

Figure 38: Screenshot of Wireshark after HTTP Flood.

3.6 SYN Flood Attack (Hping3 Attack)

Step 1: Confirming the Attacker assigned IP Address.

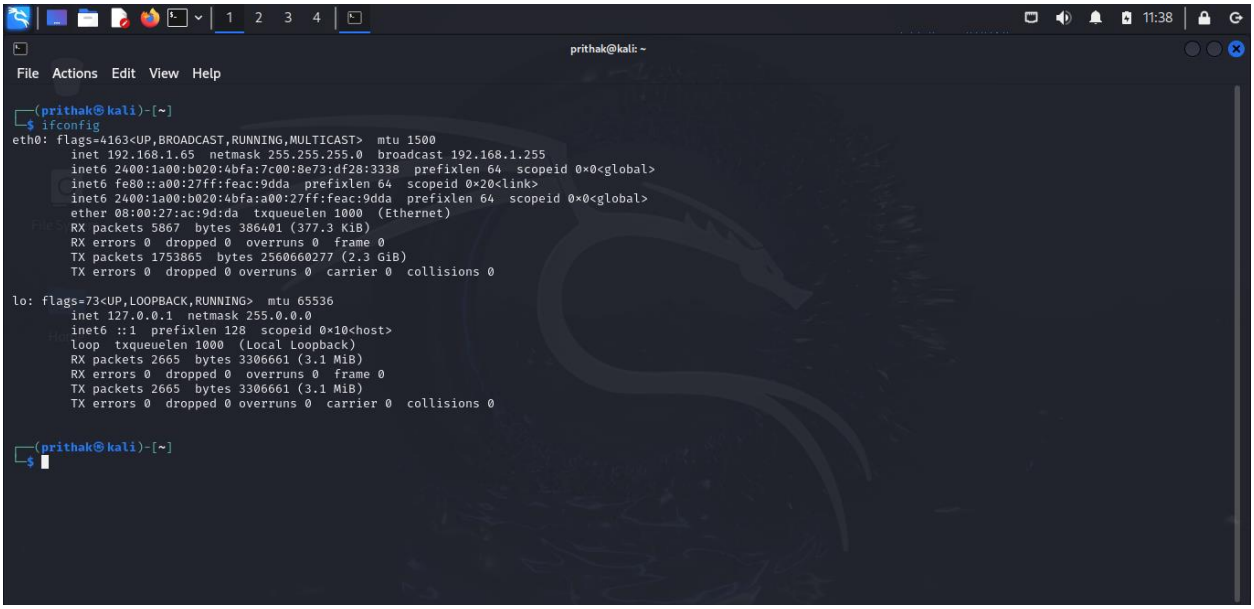

Figure 39: Screenshot of checking Attacker IP Address.

Step 2: Confirming the victim's assigned IP Address.

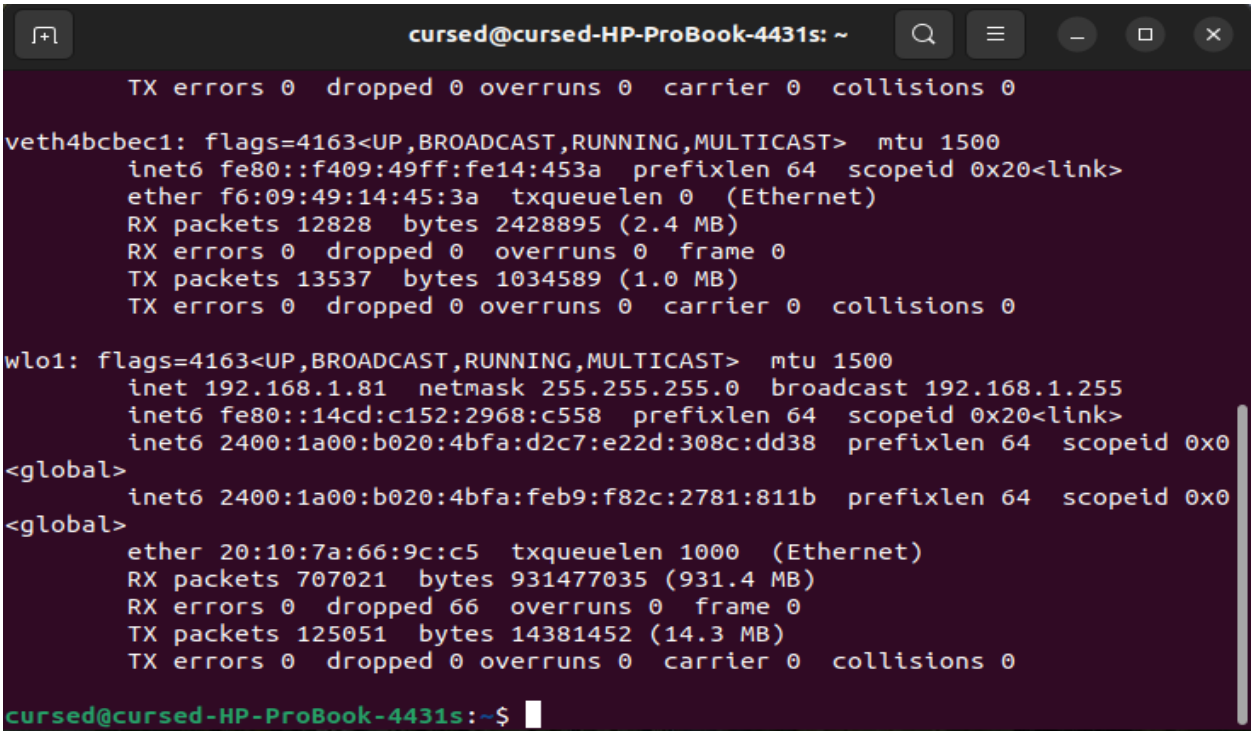

Figure 40: Screenshot of checking Victims IP Address.

Step 3: Checking the system monitor before UDP Flood.

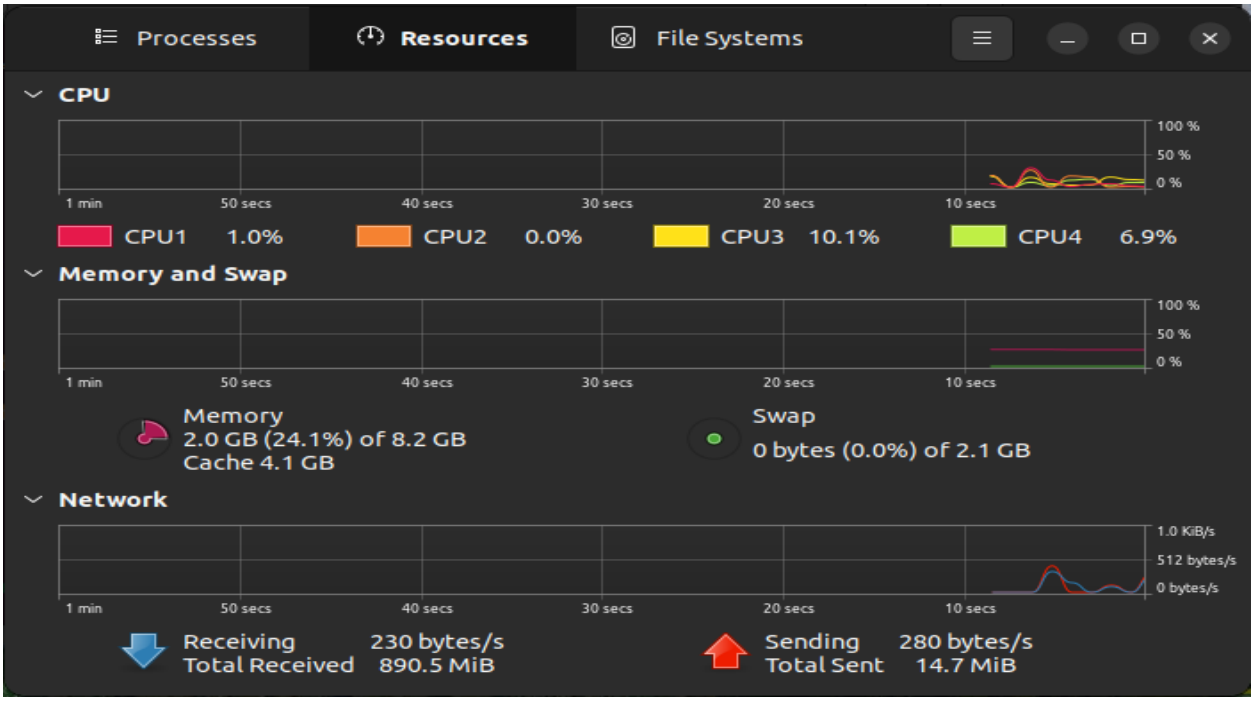

Figure 41: Screenshot of System Monitor before Hping3.

Step 4: Starting the Hping3 by assigning the victim's IP and the amount data to send.

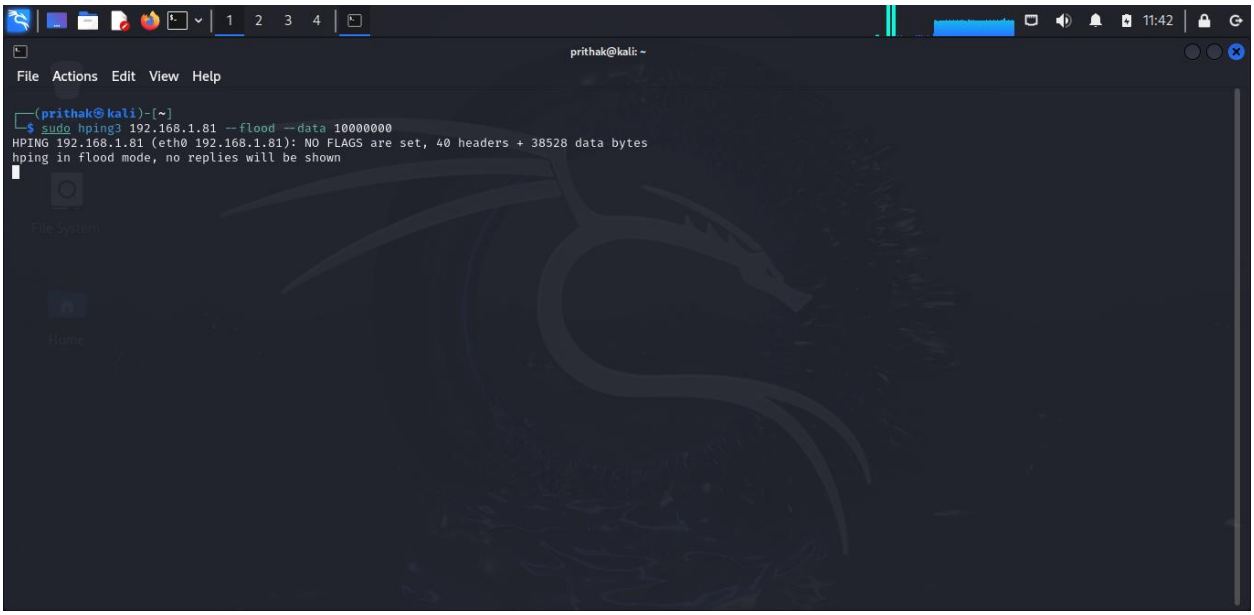

Figure 42: Screenshot of starting Hping3.

Step 5: Checking the System Monitor, if the network resources are being consumed abnormally.

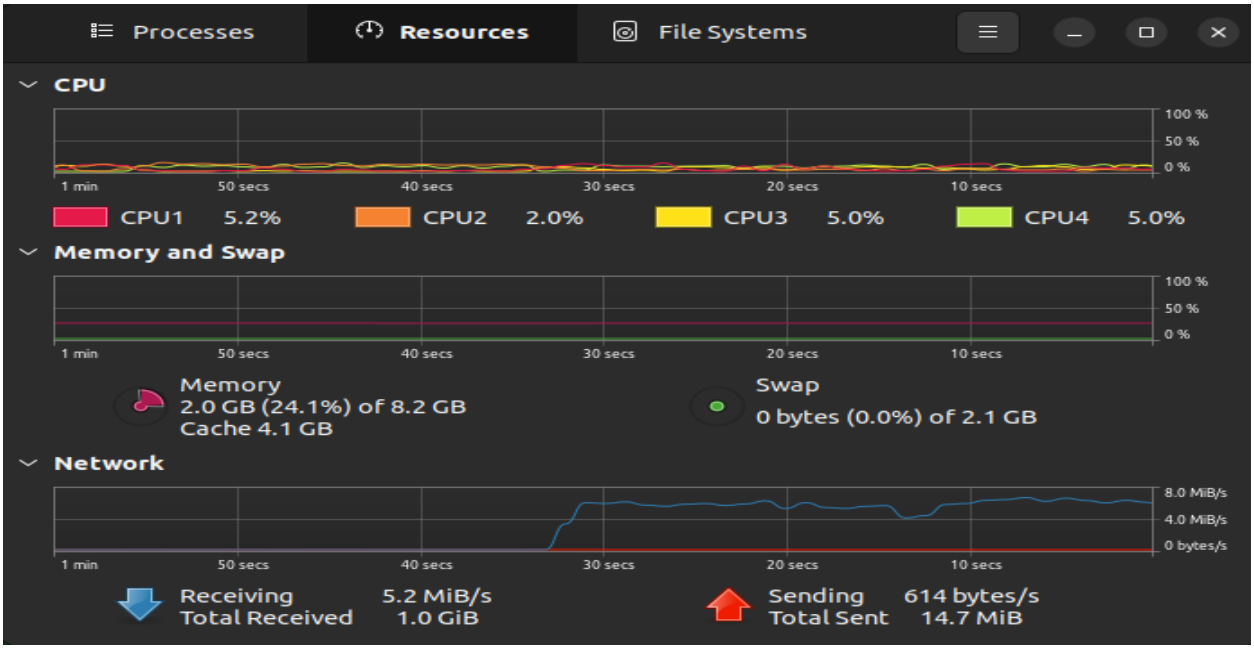

Figure 43: Checking the System Monitor during Hping3.

Step 6: Checking Wireshark, if there is network traffic from the attacker's IP Address.

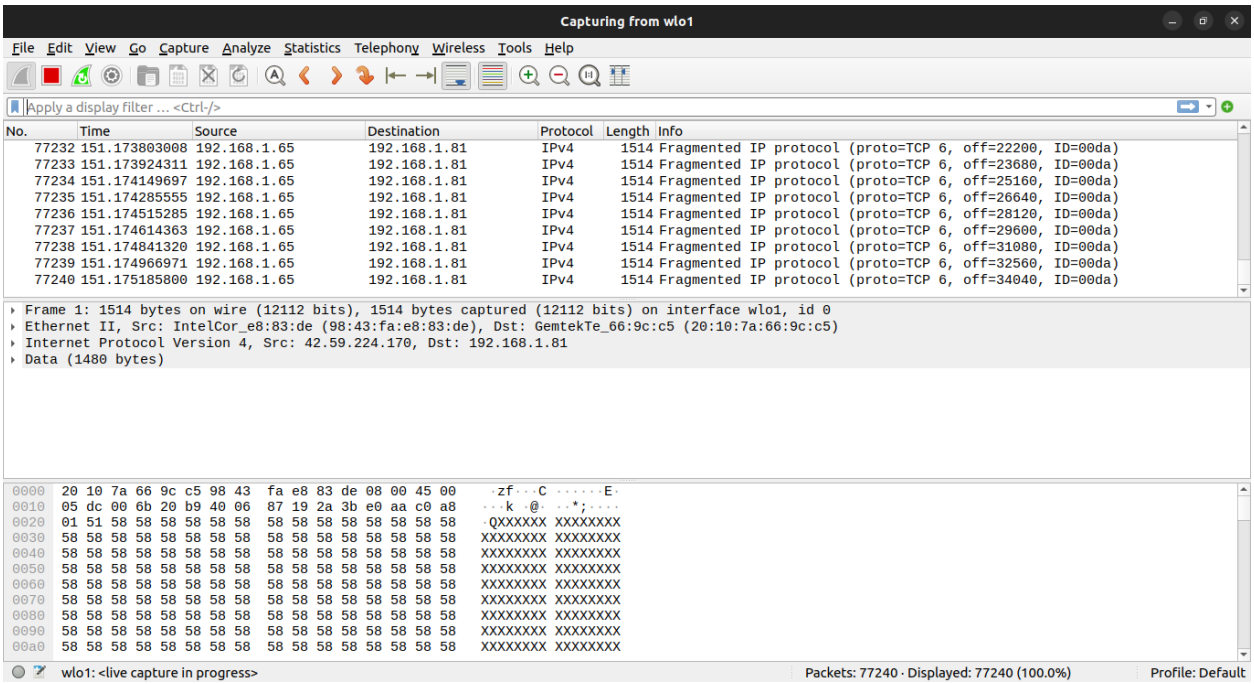

Figure 44: Screenshot of Wireshark during Hping3.

Step 7: Stopping the UDP Flood Attack.

Figure 45: Screenshot of stopping Hping3.

Step 8: Checking the System Monitor.

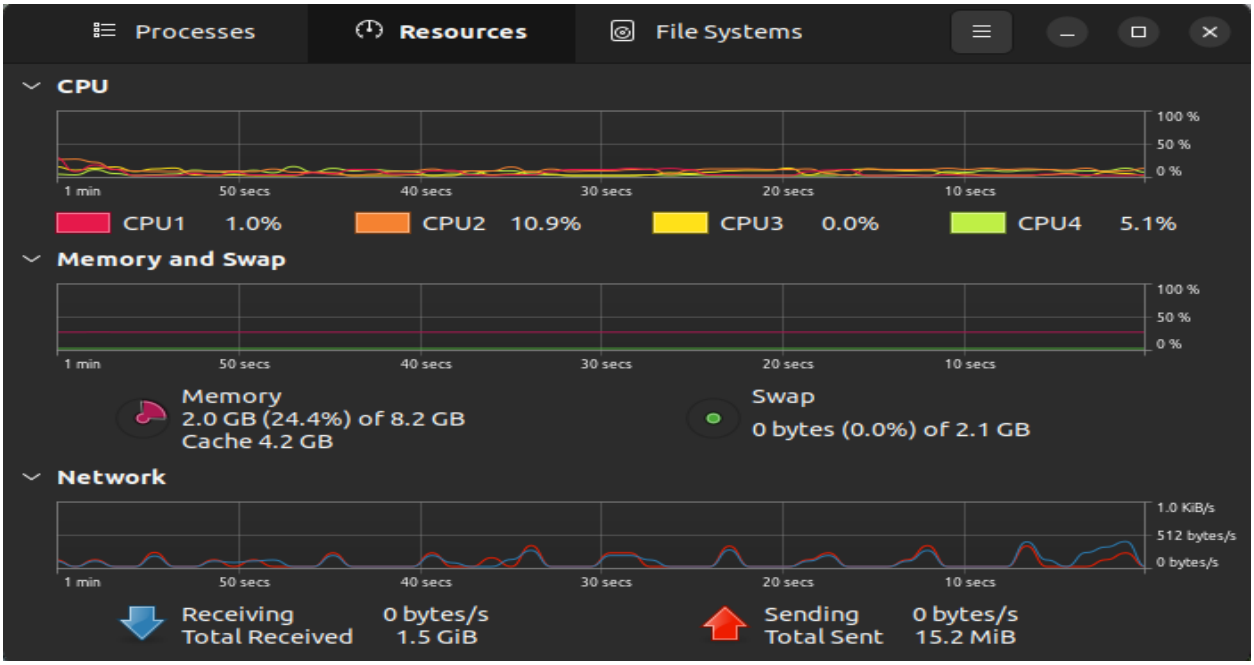

Figure 46: Screenshot of System Monitor after Hping3.

Step 10: Checking Wireshark

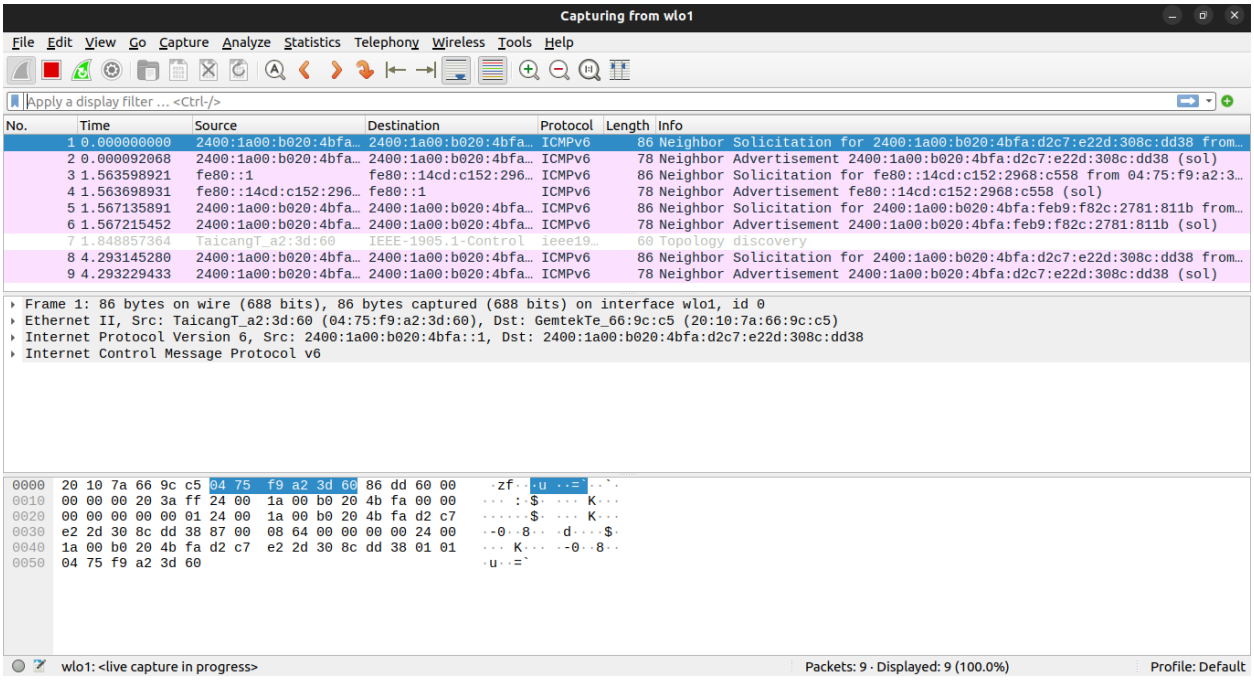

Figure 47: Screenshot of Wireshark after Hping3..

4. Mitigation

4.1 UDP Flood Attacks

UDP is a connectionless protocol, making source detection technology in SYN flood attack defense unsuitable. Firewalls initially limit UDP packet rates based on destination IP addresses, security zones, and sessions, but this may discard some packets. Attack packets have unique features, making feature-based filtering, also known as fingerprint filtering, necessary to protect against UDP flood attacks. Fingerprint filtering includes static and dynamic learning (Jie, 2022).

4.1.1 Static Fingerprint Filtering

The attack packets can be hidden in UDP packets' data segment, source IP address, source port, destination IP address, and destination port. These features can be added to the device's filter parameters, and after static fingerprint filtering is configured, the device discards or rate-limits traffic matching the attack features (Jie, 2022).

4.1.2 Dynamic Fingerprint Learning

Dynamic fingerprint learning is a method for defending against attacks that collect statistics from regular UDP flood attack packets. When the number of similar packets reaches the alarm limit fingerprint learning begins. If the same feature appears constantly, it is recognized as a fingerprint. The packets that match the fingerprint are identified as attack packets and either discarded or rate-limited (Jie, 2022).

4.1.3 Port Filtering

Port filtering is a security measure that involves selectively allowing or blocking specific UDP ports to minimize potential attacks on a network or system. The system should filter unknown and Real-Time Signature ports, along with known UDP ports like 53, 68, and 88 (Reich, 2023).

4.1.4 Cloud-Based Mitigation Services

Cloud-based mitigation services focus on detecting and mitigating DoS attacks. These services can handle and filter DDoS traffic while leaving legitimate traffic unaffected. Combining a hybrid solution based on powerful machine learning with a hybrid DoS mitigation solution provides us with minimal human intervention and maximum peace of mind (Kime, 2022) (Reich, 2023).

4.1.5 Packet Scrubbing and Anomaly Detection

Packet scrubbing techniques and a scrubbing center are used to defend against malicious or suspicious UDP packets, as well as advanced anomaly detection and RFC compliance algorithms to validate packet structure and filter out non-legitimate sources (Reich, 2023).

4.2 ICMP (Ping) Flood Attacks

4.2.1 Disabling ICMP Functionality

Disable the ICMP functionality of the targeted device by accessing its administrative interface. This will disable all network activities involving ICMP, making the device unresponsive to ping requests, traceroute requests, and other network activities. It is important to note that all network activities involving ICMP will be disabled (Dasan, 2021).

4.2.2 Limiting the processing of incoming ICMP messages

ICMP attacks can be prevented by implementing rate-limiting measures such as using tools like Iptables. These rules, such as dropping address mask and timestamp requests and limiting echo requests per second, allow legitimate ICMP traffic to proceed while controlling malicious or excessive traffic. This approach maintains network stability and security by preventing ICMP flood attacks and ensuring essential ICMP functionalities remain available for legitimate network operations (Dasan, 2021).

4.3 Slowloris

4.3.1 Using Reverse Proxy

Reverse proxy acts as a firewall between a server and its clients, effectively protecting the server from potential threats such as Slowloris attacks. The reverse proxy can closely monitor incoming requests as it is placed in front of the server. If it detects suspicious patterns indicating a Slowloris attack, such as incomplete request headers or long connections with little data transfer, it can quickly intervene by terminating those connections. This protects the server's availability and performance by reducing the impact of Slowloris attacks, allowing legitimate client requests to reach the server while intercepting and eliminating malicious packets (Sundar, 2023).

4.3.2 Limit Connection per IP

Implementing connection limitations based on particular IP addresses is an effective method for reducing the impact of Slowloris attacks. This can be done using cloud Web Application Firewalls (WAFs) such as AppTrana or load balancers. These solutions enable prevention against Slowloris attacks by limiting the number of connections allowed from a single IP address in a certain timeframe. By limiting the number of connections, legitimate users maintain access to services, while hackers are prevented from overloading servers with large, resource-intensive connections. This method improves overall system security and assures high efficiency while denial-of-service attacks such as Slowloris (Dasan, 2021).

4.3.3 Reduce the Maximum Request Duration

Connection timeout mechanisms help to prevent Slowloris attacks. These mechanisms, implemented through cloud Web Application Firewalls like AppTrana or load balancers, limit the duration a connection can remain open. Implementing timeouts allows servers to disconnect connections that remain open without completing requests, reducing the risk of service disruption and improving system resistance to denial-of-service attacks (Dasan, 2021).

4.3.4 Implementing Rate Limiting

Monitoring server connections and request volumes is essential for preventing Slowloris attacks. This includes establishing thresholds and enforcing rate limitations with software and hardware solutions such as load balancers, firewalls, and intrusion prevention systems. By monitoring traffic volumes, these systems may immediately detect and eliminate threats, protect server uptime, and assure ongoing service delivery to legitimate users (Dasan, 2021).

4.4 Ping of Death

4.4.1 Filtering Traffic

The common approach for dealing with increases in traffic is rate limitation, which allows only reasonable quantities while maintaining host availability. Advanced security methods include packet analysis, which enables selective approved communication. The first phase consists of recognizing legal traffic and establishing an initial foundation for packet authenticity. This ensures accuracy in defending against harmful activities and allowing for uninterrupted data flow (Smith, 2023).

4.4.2 Reducing the Probable Attack, Surface Attack

One of the major methods is to reduce the attack surface, limit attacker options, and put countermeasures in a centralized area to effectively prevent DOS attacks. This includes removing unnecessary ports, protocols, or programs, which reduces possible attack sites and allows for a more targeted approach to protection. Processing resources should be deployed behind Content Distribution Networks (CDNs) or Load Balancers, with direct Internet access limited to specialized infrastructure parts such as database servers. Firewalls and Access Control Lists can also be used to regulate traffic in certain situations (Smith, 2023).

4.4.3 Use a Buffer

To effectively prevent PoD attacks, change the recipient's TCP stack, confirm IP packet size, and set aside buffer space for reassembly. Truncating IP queries such that they fit within allotted buffers decreases the risk of heap overflow. While banning all ping requests is not practicable, selectively filtering out fragmented requests while processing unfragmented queries can improve network resilience and maintain operational continuity in the face of growing cyber threats. This defensive method improves network resilience and operational continuity (Smith, 2023).

4.5 HTTP Flood Attack

4.5.1 Authenticate Users

It is more effective to restrict web application access to just authenticated users using various authentication protocols such as Username/Password, OAuth 2.0, OpenId Connect, and API Keys. The application checks the identity of the person contacting the API and confirms that they have the required permissions to access the requested information. The authorization procedure can be based on the user's attributes, therefore restricting CPU and memory utilization. Enforcing multi-factor authentication (MFA) can improve overall security.

Authentication alone cannot protect against HTTP flood attacks since attackers require access to user credentials or authentication tokens to conduct resource-intensive queries. Unauthenticated queries can still cause network overload, although they take more effort on the attacker's part (Sisaka Baro, 2021).

4.5.2 Rate Limiting

Rate limitation is a precautionary measure used to avoid server and network resource depletion in the case of an HTTP flood attack. It limits the amount of

queries that may be made by a client or user, for example, to 100 per minute. When this limit is reached, the service will issue an error to the API caller. Most rate limitations algorithms identify the caller by IP address, although others can impose rate limitations based on an HTTP header (Sisaka Baro, 2021).

4.5.3 Reduce Maximum Request Size

HTTP flood attacks aim to overwhelm web servers and network resources by sending a large amount of data through API calls. It is important to set limits on the amount of data that can be accepted to prevent such attacks (Sisaka Baro, 2021).

4.5.4 Leverage Caching

Website caching is an optimization strategy that minimizes the CPU time required to execute HTTP requests, hence increasing overall website speed. It can help during an attack with DoS by reducing resource use. The most frequent caching approach is sending packets to the client through the cache-control header. However, server-side caching is also required for optimal speed. There are two types of server-side caching: in-memory cache, which temporarily saves CPU-intensive computation results, and static files cache, which is best served from a content delivery network (CDN) to reduce bandwidth needs. The server-side page or content cache improves resource use by limiting the amount of requests that a web server can process in an amount of time (Sisaka Baro, 2021).

4.5.5 Stop Processing a Request when the client closes the TCP Connection

Servers should avoid unnecessary resource usage by preventing CPU-intensive HTTP requests. If a client closes the TCP connection before receiving the response, the server should stop processing the request (Sisaka Baro, 2021).

4.5.6 Implement a Challenge-Response mechanism for Specific Operations

Challenge-response authentication is a mechanism in which a user gets notified before processing their requests, which typically necessitates manual intervention. This method reduces botnet devices' ability to overwhelm servers with HTTP requests. One example is the Completely Automated Public Turing Test (CAPTCHA), which tests if a human is behind a computer by forcing a human to solve a task, raising the likelihood that the operation is not a machine (Sisaka Baro, 2021).

4.5.7 Use Exponential Backoff When Retrying Backend Operations

Web servers frequently need to contact external resources, such as databases, and developers frequently use retry techniques to deal with temporary failures. To implement retry mechanisms in programming, utilize open-source frameworks such as Polly (Sisaka Baro, 2021).

Set intervals might increase the system burden, perhaps resulting in a self-inflicted DoS. To remedy this, an exponential back-off method is proposed. This includes waiting 15 seconds before retrying a request, followed by 30 seconds, 1 minute, or 2 minutes after each failure. The aim is to double the time between failures until a specific limit is met. The initial retry interval might vary between 5 and 15 seconds. This approach has the potential to greatly increase server performance during periods of overload (Sisaka Baro, 2021).

4.5.8 Installing Security Update on Servers

To keep your server secure, make sure you have the most recent security updates loaded. Microsoft's. Net Core 3.1 release in June 2020 fixed a denial-of-service vulnerability that could be abused remotely without authentication. If you do not utilize the most recent version, your system may be vulnerable to these issues. It is recommended to upgrade the runtimes and software on your web server (Sisaka Baro, 2021).

4.5.9 Use of Web Application Fire (WAF)

A web application firewall (WAF) is a reverse proxy that analyzes and filters HTTP, HTTPS, or TLS traffic between a web application and the internet to protect it against application-layer threats such as SQL injections and XSS. WAF can create an IP reputation library, identify malicious activity, and ban clients based on real-time traffic analysis, making it an effective tool for mitigating HTTP flood and DoS attacks (Sisaka Baro, 2021).

4.5.10 Leverage the Cloud

Cloud hosting provides various benefits for DoS avoidance, including free services from companies such as Amazon and Microsoft. Cloud providers also offer WAF components for your solution architecture. Cloud networks often have higher capacity than on-premise networks, which makes them ideal for DoS assaults. Scaling out your solution depending on traffic might be cost-effective, but make sure to establish the maximum number of instances (Sisaka Baro, 2021).

4.6 SYN Flood Attack

4.6.1 Increasing backlog

Expanding the backlog queue improves server resilience by allowing it to handle a greater amount of incoming SYN requests, thereby preventing flooding attempts. By increasing queue capacity, the server can manage more simultaneous connections during a SYN Flood attacks, preventing resource depletion and ensuring operational stability (Chinnasamy, 2024).

4.6.2 Filtering

Filtering is the process of establishing rules within network devices to detect and prevent harmful SYN requests by recognizing patterns or identifying known malicious IP addresses. This proactive solution serves as a first line of protection against SYN Flood attacks, effectively stopping malicious traffic before it reaches the server, hence improving overall network security and limiting any interruption or exploitation (Chinnasamy, 2024).

4.6.3 Reducing SYN-RECEIVED Timer

Shortening the delay duration for ACK replies following the transmission of a SYN-ACK releases resources assigned to half-open connections more quickly. By using a shorter period, the server optimizes resource consumption during a SYN Flood, reducing the amount of time spent on unfinished connections. This proactive change improves the server's capacity to handle incoming requests quickly, strengthening its resilience against SYN Flood assaults and ensuring operational efficiency even under strain (Chinnasamy, 2024).

4.6.4 SYN Cache

Using a cache to hold brief data about incoming SYN requests, rather than dedicating significant resources to each and every request, improves resource use. This technique prioritizes legitimate connection requests and manages incoming traffic effectively. By taking this strategy, the server optimizes its resource consumption, assuring responsiveness while reducing the danger of overload, thus improving its capacity to handle incoming connections efficiently and reliably (Chinnasamy, 2024).

4.6.5 SYN Cookies

SYN Cookies enable the server to reply to SYN queries with a SYN-ACK without immediately committing resources to the connection. Resource allocation occurs only after receiving a valid ACK answer. This agile strategy prevents resource depletion by deferring resource allocation until the connection's validity is confirmed. By using SYN Cookies, the server improves resource management, lowering the danger of exhaustion during high-volume connection attempts while ensuring that genuine connections are processed effectively without impacting security or performance (Chinnasamy, 2024).

4.6.6 Load Balancers

Load balancers are critical for avoiding SYN flood assaults, particularly when handling the TCP SYN/SYN-ACK/ACK handshake. They allocate incoming network traffic across numerous servers, ensuring that no server is overloaded. Load balancers automatically handle connection requests, assess the handshake process, and prevent SYN flooding before they reach backend servers. They serve as a frontline defense, reducing the burden on backend infrastructure and ensuring system availability during SYN flood attacks (Chinnasamy, 2024).

4.6.7 Firewalls and Proxies

Implementing firewalls and proxies provides an additional layer of security by intercepting and inspecting incoming traffic before it reaches the target server. These security mechanisms act as gatekeepers, examining inbound connections and either allowing or prohibiting access depending on specific requirements. By filtering out potentially dangerous traffic, firewalls and proxies significantly minimize the target server's vulnerability to SYN Flood attacks, improving overall network resilience and protecting against potential exploitation or interruption (Chinnasamy, 2024).

4.6.8 Honeypots and Honeynets

Honeypots and honeynets are proactive measures used to study malicious activity, including SYN flood attacks. Honeypots are strategically placed decoy systems that absorb malicious traffic, diverting attacks from production servers. Honeynets, on the other hand, deploy interconnected honeypots, providing a comprehensive view of attacker behavior across multiple systems. They contribute to threat intelligence and DDoS or Dos protection (Chinnasamy, 2024).

4.6.9 Rate Limiting

Implementing rate-limiting for incoming connection requests is a critical step in preventing SYN flood attacks. This method entails imposing a specified restriction on the number of connection requests that a server can accept during a given duration. Any queries that surpass this threshold are either declined outright or purposely delayed. By imposing such limits, servers may successfully reduce the impact of SYN flood attacks, avoiding massive influxes of connection requests while maintaining operational stability and responsiveness (Chinnasamy, 2024).

4.6.10 Hybrid Approaches

Using a multifaceted strategy creates a strong defense mechanism against TCP SYN Flood assaults, increasing resilience in the case of individual technique failure. Organizations may create a layered security plan by combining methods like filtering, backlog modification, SYN-RECEIVED timer optimization, SYN caching, and SYN Cookies. This comprehensive technique improves overall security by diversifying protection methods, guaranteeing that even if one method fails, others remain vigilant in protecting against SYN Flood assaults, hence increasing network resilience and successfully reducing possible vulnerabilities (Chinnasamy, 2024).

5. Evaluation

5.1 Advantages

Denial of Service (DoS) attacks, distinguished by their simplicity and cost-effectiveness, pose significant threats to online services and organizations. Unlike Distributed Denial of Service (DDoS) attacks, which involve multiple sources coordinating an attack, DoS attacks can be executed with relative ease by a single individual or a small group. This simplicity makes them accessible even to less skilled attackers, who can employ readily available tools or scripts to flood a target with excessive traffic or requests (SafeAeon Inc., 2024).

What separates attacks using DoS is their cost-effectiveness. They need fewer resources and infrastructure, making them a tempting target for fraudsters looking to disrupt or deactivate internet services. A DoS assault may be initiated with only one attacker and basic tools, resulting in a broad interruption of targeted systems. This interruption is more than just an annoyance; it may cause significant financial loss and reputational harm to the impacted companies (SafeAeon Inc., 2024).

A DoS attack aims to interfere with the availability of targeted services by overloading servers or network infrastructure. Legitimate users are effectively denied access to websites, apps, and online services when the target gets flooded with traffic or requests. This disruption has a quick effect on the service's performance, but it also reduces user trust and confidence, potentially sending them away to rivals. In summary, the simplicity, cost-effectiveness, and disruptive potential of DoS attacks highlight the crucial need for companies to deploy strong security measures to protect themselves against such threats (SafeAeon Inc., 2024).

5.2 Disadvantages

In comparison to DDoS attacks, which need coordination from several sources, DoS attacks are easier and more accessible, relating to less skilled attackers. These attacks need few resources and infrastructure, frequently simply one person with basic tools or scripts. This cost-effectiveness attracts cybercriminals who want to disrupt web services without committing substantially (SafeAeon Inc., 2024).

DoS attacks attempt to interrupt certain services by flooding servers or network infrastructure with excessive traffic or requests. This interruption goes beyond

apprehension resulting in financial losses and harming the reputations of impacted firms. DoS attacks, which make legitimate websites, applications, or online services unavailable, can have far-reaching implications, weakening confidence and pushing consumers away (SafeAeon Inc., 2024).

Organizations must identify the threat posed by DoS attacks and put in place strong security measures to effectively limit risks. Preventive techniques such as network monitoring, traffic filtering, and capacity planning can help in the defense against these assaults and assure the continuous availability of online services (SafeAeon Inc., 2024).

6. Conclusion

Denial-of-Service (DoS) attacks are a continuous and evolving concern in the field of cybersecurity. From the early days of the Robert Morris worm to the more current Mirai botnet, attackers have repeatedly exploited weaknesses in digital infrastructure to disrupt services and cause financial loss. The growth of tactics like UDP Flood, Slowloris, and HTTP Flood shows the adaptability and persistence of malicious parties attempting to disrupt online processes.

Companies should take preventative measures to reduce the risks posed by DoS assaults. Businesses should strengthen their defenses and reduce the effect of possible interruptions by employing methods like rate limitations, cloud-based mitigation services, and Web Application Firewalls (WAFs). Furthermore, continuing investment in security infrastructure, along with thorough incident response strategies, may improve resilience and ensure operational continuity even in the face of advanced cyber attacks.

Lastly, countering the threat of DoS attacks requires an integrated approach that includes both technology solutions and organizational readiness. Businesses can defend against DoS attacks and safeguard their digital assets by staying informed about emerging threats, implementing best practices in network security, and developing an alert society, thereby protecting their reputation and preserving customer trust in an increasingly interconnected world.

7. References

Chinnasamy, V. (2024, january 25). *SYN Flood Attack: The What, Impact, and Prevention Methods*. Retrieved from indusface: https://www.indusface.com/blog/what-is-syn-synchronize-attack-how-the-attack-works-and-how-to-prevent-the-syn-attack/

Dasan, S. (2021, Feburary 20). *ICMP Flood Attack – How to Mitigate*. Retrieved from bobcares: https://bobcares.com/blog/icmp-flood-attack-how-to-mitigate/

Dennis, M. A. (2024, March 29). *denial of service attack*. Retrieved from britannica: https://www.britannica.com/technology/denial-of-service-attack

Ferguson, K. (2021, April). *denial-of-service attack*. Retrieved from TechTarget: https://www.techtarget.com/searchsecurity/definition/denial-of-service

Hanna, K. T. (2024, January). *DEFINITION Wireshark*. Retrieved from TechTarget: https://www.techtarget.com/whatis/definition/Wireshark

Jie, Z. (2022, August 9). *What Is UDP Flood?* Retrieved from IP Encyclopedia: https://info.support.huawei.com/info-finder/encyclopedia/en/UDP+Flood.html

Kime, C. (2022, December 19). *Complete Guide to the Types of DDoS Attacks*. Retrieved from esecurityplanet: https://www.esecurityplanet.com/networks/types-of-ddos-attacks/

Lutkevich, B. (2021, June). *distributed denial-of-service (DDoS) attack*. Retrieved from techtarget: https://www.techtarget.com/searchsecurity/definition/distributed-denial-of-service-attack

Nagaraj, K. (2023, January 3). *bWAPP: A Vulnerable Web Application for Practicing Vulnerabilities - Installation Guide*. Retrieved from infosecwriteups: https://infosecwriteups.com/bwapp-a-vulnerable-web-application-for-practicing-vulnerabilities-installation-guide-146637e2da92#:~:text=bWAPP%20%E2%80%94%20bWAPP%2C%20a%20buggy%20web,testing%20and%20ethical%20hacking%20projects.

Reich, E. (2023, October 12). *Protect your environment from UDP Flood Attacks*. Retrieved from radware: https://www.radware.com/blog/ddos-protection/2023/10/protect-your-environment-from-udp-flood-attacks/

SafeAeon Inc. (2024, April 18). *What are the advantages and disadvantages of DoS and DDoS attacks?* Retrieved from medium: https://medium.com/@safeaeon-inc/what-are-the-advantages-and-disadvantages-of-dos-and-ddos-attacks-238e952fedf0

Sisaka Baro. (2021, Februrary 15). *10 tips to mitigate HTTP flood attacks*. Retrieved from siakabaro: https://www.siakabaro.com/10-tips-to-mitigate-http-flood-attacks/

Smith, J. (2023, August 9). *How to Avoid A Ping of Death Attack: Never Let Your Guard Down*. Retrieved from fastestvpn: https://fastestvpn.com/blog/how-to-avoid-a-ping-of-death-attack/

Sundar, V. (2023, February 22). *What is Slowloris DDoS Attack and How Does it Work?* Retrieved from indusface: https://www.indusface.com/blog/what-is-slowloris/#:~:text=By%20monitoring%20the%20number%20of,firewalls%2C%20or%20intrusion%20prevention%20systems.

Yadav, S. (2020, December 24). *linkedin*. Retrieved from How to perform SYN flooding Attack with Metasploit: https://www.linkedin.com/pulse/how-perform-syn-flooding-attack-metasploit-shubham-yadav/