South Asian Journal of Engineering and Technology Research Article

Open Access Full Text Article 151(2025) 1-16 DOI:10.26524/sajet.2025.15.1

Understanding Botnets: Architecture, Attacks, and Mitigation Strategies

Prajwal B[a*], Asha Kumari A[b], Dr E Saravana Kumar[c], Mohit S[c]

[a] Student, Department of Computer Science and Engineering, The Oxford College of Engineering, Bangalore, Karnataka, India

[b] Assistant Professor, Department of Computer Science and Engineering, The Oxford College of Engineering, Bangalore, Karnataka, India

[c] Professor, Department of Computer Science and Engineering, The Oxford College of Engineering, Bangalore, Karnataka, India

[d] Student, Department of Artificial Intelligence and Machine Learning, The Oxford College of Engineering, Bangalore, Karnataka, India

Corresponding author.

Correspondence: Prajwal B

E-mail: asha.hebbar@gmail.com

Article info

Received 12th November 2024 Received in revised form 16 January 2025

Accepted 9 February 2025

Keywords

Botnet, Command-and-Control, Cybersecurity, Denial-of-Service, DDoS, Network Security, Intrusion Detection, Ping of Death, LAND Attack.

https://sajet.in/index.php/journal/article/view/304

Abstract

Botnets represents a significant threat in the cybersecurity landscape. Botnets relies on the set of compromised devices called as bots or zombies which are remotely connected and controlled by the adversary or the hacker. The adversary or the hacker controls the infected devices via a Command and Control(C&C) server. Botnets or bots are known for exploiting the set of vulnerabilities, escalating privileges and permissions in the infected systems and also establishing backdoors. Hackers or adversaries may use botnets to launch large scale cyber-attacks, in most cases it is a type of DOS (Denial of Service) attack. The article focuses on the architecture of botnets and also the way it works with a C&C server in general. A basic pythonic implementation of botnet is implemented to showcase how dangerous they may be, here it just sends back an acknowledgement message back to the user using sockets. The article also deals with various types of Denial-of-Service attacks in a detailed manner and then provides a detailed simulation of two types of DOS attacks in general which is the PoD attack and LAND attack. The article then later deals with the set of mitigation strategies to prevent and minimize the overall effects which may be caused via botnets. This article aims to give a basic foundational understanding of botnets in general with a basic comparison of the various types of botnet attacks.

1. Introduction:

Security is the most important nonfunctional aspect which needs to be consider while developing applications as well as while using them. Vulnerabilities refers to the set of issues or flaws in the system which the hacker or the adversary may use to exploit or gain access to the user's system. Post attack the user may deploy or create backdoors, change permissions as well as deploy a botnet in the user's system. This article deals with botnet in general and covers some important functionals regarding the architecture of botnets in general and also shows how they work in general.

Botnets are among the most dangerous and sophisticated forms of malware, capable of orchestrating large-scale cyberattacks with devastating consequences. A botnet comprises a network of compromised devices, often referred to as "zombies," which are remotely controlled by an adversary through a Command-and-Control (C2) infrastructure. Once an attacker successfully infiltrates a system—typically through privilege escalation,

backdoors, or exploiting security vulnerabilities—the botnet is deployed to execute malicious activities. These actions can include everything from financial fraud and cyber espionage to extensive Distributed Denial-of-Service (DDoS) attacks and data exfiltration. Botnets have an especially damaging effect because of their ability to go unnoticed for long periods of time while waging relentless and coordinated attacks.

By examining their design, methods of operation, and attack strategies, the chapter provides an in-depth analysis of botnets in general. The various types of botnet-based attacks are discussed in a brief manner targeting different layers of the OSI models in general and also explains in a general manner. Here we deal with different types of botnet-based attacks in a generalized manner. A controlled experimental setup is used to replicate botnet behaviour on a network, illustrating real-world attack situations. The actual implementation simulates two major types of DDoS attacks: the Ping of Death (PoD) attack and the LAND attack. Furthermore, the paper offers a thorough examination of both traditional and contemporary Denial-of-Service (DoS) and DDoS tactics, their effects on network security, and the changing tactics attackers employ to increase attack speed. In addition to assault strategies, this paper focuses on defences against botnet-based attacks. Intrusion detection systems (IDS), anomaly-based traffic analysis, network segmentation, and the application of strong access control policies are among the countermeasures covered. Furthermore, ethical aspects are investigated in order to highlight acceptable research approaches for investigating and comprehending cyber risks.

By the end of this article, the readers will understand botnets in general including the dangers as well as the attack mechanism they follow. The insights provided in the context of this chapter will equip cybersecurity professionals, researchers, and system administrators with the knowledge required to detect, prevent, and mitigate botnet-driven threats, ultimately strengthening overall cybersecurity resilience as well as the preparedness against botnet attacks in general [1].

Botnets:

Botnets can be defined as a group of connected devices which have already been compromised by the hacker or the adversary in general. The assumption here is that the hacker has already exploited and entered the user's system by any way which may be social engineering or may be via backdoor or by injecting some malware into the user's system or any other way. After entering the user's system, the adversary may make it a bot or a zombie in the context of botnets. Botnets from the name itself can be understood as a collection or network of bots in general. The adversary here will act as the root or the router of this botnet architecture. The adversary will have a root access terminal in the context of botnet it is called as Command and Control(C&C) server. Just like execution of scripts or commands in root mode, the bots or the affected devices execute these commands without any interruptions by overriding default permissions and security measures. The reason why these bots connected networks or botnets in general are referred to as zombies is because the users will not be able to control their own devices when adversary runs a command on the C&C server. Another reason is that the infected devices will not know they are infected until taken over by the adversary making this one of the most dangerous type of attack in the context of cybersecurity. Botnets can be used for various illegal purposes in general some examples are distributing ransomwares, initiating DDoS attacks, stealing confidential information, etc [2]. The below figure, *fig.1* shows a basic architectural implementation of botnets as well as the way they affect the targets in general is also outlined in the figure.

The figure typically shows a detailed implementation of botnet, here the botmaster refers to the system of the adversary or the hacker who has access to the set of bots as well as the command-and-control server. The botmaster may enter set of commands in the command and client server and ensure that the bots execute those commands. These affected bots may execute these scripts or commands in general to perform some malicious tasks like spam generation, DDoS, or may generate phishing links to infect other nearby or known user devices. In context of DoS the bots may be directed towards the target to execute set of commands and deny the services which may take the service down or make it unavailable to the users who are using the services in general. The next section deals with the different types of DDoS attacks in general and also a basic python implementation of a botnet server and client demonstrating how easy it is to code a botnet as well as how dangerous botnets are in the context of cyber security.

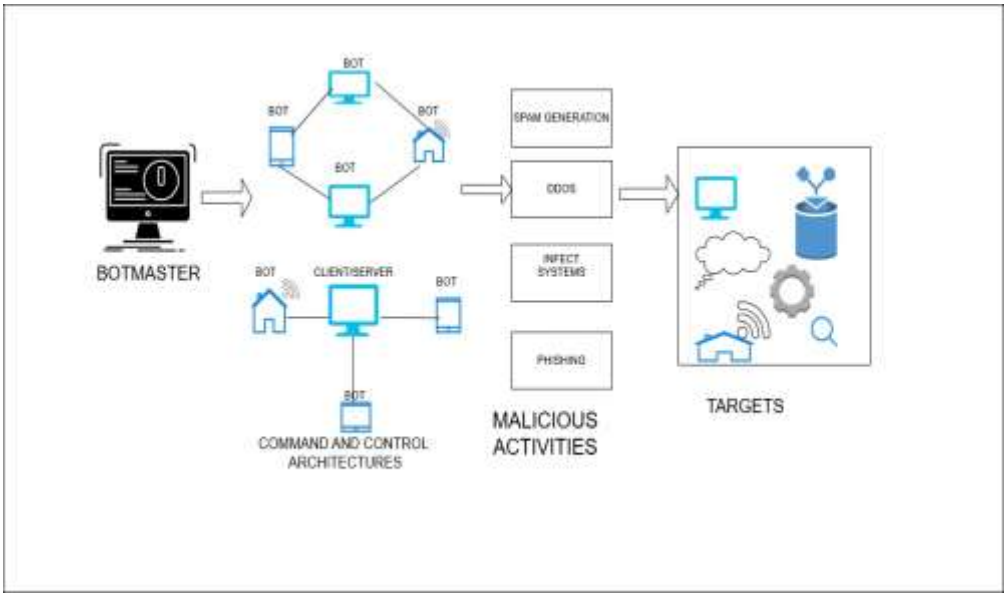

Fig 1: Botnet Architecture

Types of DDOS:

This section discusses the various types of DoS attacks in general and what they do and how they affect the user's system. Some DoS attacks are not covered as they are illegal to discuss and present in general, most of the DoS attacks discussed in this chapter are deprecated or do not work as intended or as they used to work before. DoS which stands for Denial of Service are usually set of attacks which are done with some malicious intent trying to either crash the service being targeted or support some common cause where users volunteer their systems as bots or may be cause some loss to the target due to high usage of services and resources in general. The paper only deals with ethical implementations of botnets in general and do not advice the users to implement or try it illegally as mostly almost any botnet-based attacks are illegal in general and may lead to a lot of legal troubles. The below deals with the most important classifications in the context of DoS attack in general.

- *Denial of Service (DoS) Attacks:* Denial of Service attacks are the set of attacks which aims to disrupt the normal functioning of some services which is usually run on some servers or networks in general. DoS attacks ensure that these servers or the networks on which the services run are disrupted or constrained in general that leads to the unavailability of services. There are several potential consequences for a server under a DoS attack:

1. *Resource Exhaustion:* The server may not necessarily go down, but the attacker can exhaust available resources (e.g., CPU, memory, or connection pool). This makes it difficult for the legitimate users to access the service making it difficult for legitimate users to access the server. This can lead to denial of service for other users trying to make requests [1].
2. *Server Crash*: In some cases, the attacker might overwhelm the server with a massive number of requests. This can lead to the server to crash, requiring a manual restart or a scheduled task to recover [2].
3. *High Latency:* Each server has a connection pool. Due to DoS the server's connection pool may be clogged or blocked with the attacker's requests. This makes it hard for the legitimate users as they have little to no capacity to them for logging in their requests. Thereby it introduces significant lag and delays even for genuine requests. [3].

- *Distributed Denial of Service (DDoS) Attacks:* A DDoS (Distributed Denial of Service) attack is a more complex and severe form of a DoS attack. In DDoS attacks, multiple attacking systems follow flooding mechanism. The infected systems typically flood the target server with requests, overwhelming it to the point where it cannot process legitimate requests. When a server experiences a DDoS attack, the following scenarios may occur:

1. *Hacktivism-Driven Attacks:* Hacktivism refers to hacking having some activity and vision in mind. These set of DDoS attacks are caried out by some hacker groups having some political or social agendas. These groups

may attack a server to protest a cause they disagree with or to make a political statement. Hackers here are known to use some publicly available DoS tools or private servers to conduct these attacks. The attackers may launch the DDoS attack on a specific date, such as during a protest [4]. These type of attacks are usually known to vast number of people and usually public driven in general.

2. *Botnet-Driven Attacks:* In this case, a network of compromised machines as discussed earlier called as botnets is controlled by a hacker or a group of hackers. The botnet's Command-and-Control (C&C) centre issues instructions to the bots, directing them to launch an attack on a specific target. The bots then send massive amounts of requests to the target website, flooding the server with traffic and causing it to crash [5].

- *Reflected Denial of Service (RDoS) Attacks:* RDoS (Reflected Denial of Service) attacks, also known as amplification attacks, are a variation of DoS and DDoS attacks. The main difference between RDoS and other DoS attacks is that RDoS use a more stealth based approach. In RDoS spoofing is done where the adversary spoofs or impersonates the victim's IP address. The attacker then sends a small request to a third-party server, which then responds with a much larger message. The response is sent to the victim, overwhelming them with data. This attack is harder to trace because the attack seems to be coming from the third-party server, not the attacker's own system [6].

The table below i.e., *table 1* shows the major differences between the three above mentioned types of DOS attacks

. Table 1: differences between DOS, DDOS, RDOS and BOTNET attack in detail

Feature	**DoS (Denial of Service)**	**DDoS (Distributed Denial of Service)**	**RDoS (Reflective Denial of Service)**	**Botnet Attack**
Attack Source	Originates from a single system.	Originates from multiple distributed systems, making it harder to trace.	Uses third-party servers to reflect the attack to the target.	Originates from a network of infected devices (bots), controlled by an attacker. [1]
Attack Strength	Limited by the resources of a single attacker.	Much stronger due to the involvement of multiple systems.	Amplifies the attack using reflected traffic, often with high intensity.	Varies based on the size of the botnet, can be highly potent. [2]
Scalability	Limited to the capabilities of the attacking system.	Highly scalable as the number of distributed systems increases.	Very scalable as it exploits multiple vulnerable servers for reflection.	Extremely scalable, depending on the size of the botnet. [3]
Mitigation	Can be mitigated with firewalls, rate-limiting, and traffic filtering.	Requires more advanced DDoS mitigation strategies, such as cloud-based protection.	Difficult to mitigate as it involves using legitimate third-party servers.	Complex to mitigate; requires identifying and removing infected devices from the botnet. [4]
Purpose	Typically aimed at disrupting or disabling a single system.	Designed to overwhelm and take down larger or multiple systems at once.	Intended to create a high-volume attack by reflecting traffic from legitimate servers.	Used for various malicious purposes, including data theft, DDoS attacks, and remote control of infected devices. [5]

OSI Model:

Before understanding how botnets work and also to understand at which layer the DDOS attacks typically focuses on we need to understand set of layers present in the OSI model and thereby a brief outline is provided below in context of the set of layers used in the OSI models.

The OSI model consists of seven layers, each with distinct responsibilities:

- *Physical Layer (Layer 1)*: In the context of OSI model the bottom most layer is the first layer and it is called as the physical layer. As the name suggests, Physical layers deals with the physical transmission of raw bits using some physical entity this can be the routing devices, the communication channels being used and other functionals which deal with transmitting the bits. The physical layer includes the basic hardware technological components like cables, switches, etc which defines how the data is converted into appropriate electrical signals or any other form like light pulses, radio waves, etc. Key concepts to be noted here are encoding schemes, bandwidth utilization (broadband vs. baseband), and physical topologies (bus, star, ring) [7].
- *Data Link Layer (Layer 2)*: The second layer in the context of the OSI stack is called as the Data Link layer. This layer is responsible for node-to-node data transmissions and error correction and detection. Data Link Layer can be divided into two different sublayers which are Media Access Control (MAC) layer, which is responsible for managing the physical addressing of the transmitted bits and the LLC layer which is the Logical Link Control which provides set of logic for performing synchronization, multiplexing, flow control as well as error checking functions. Various protocols like ethernet operate at this layer of the OSI stack [8].
- *Network Layer (Layer 3)*: The third layer of the OSI model, the Network layer, is responsible for logical addressing, routing, packet forwarding, fragmentation, error handling, and quality of service. It strategizes the logical addressing and routing of data packets. Protocols such as Internet Protocol (IP) are employed to facilitate the communication between different networks [9].
- *Transport Layer (Layer 4)*: The transport layer is the fourth layer present in the OSI stack.The major functionality of transport layer is to ensure reliable delivery of segments between the two parties who wants to communicate and send messages amongst themselves.. The transport can be either reliable or unreliable. The transport layer is one of the most important layer as it deals with flow control, error correction as well as segmentation in general. Here various protocols like TCP and UDP are used for enhancing reliability as well as to minimize latency in the above-mentioned protocols are used in the Transport layer [10].
- *Session Layer (Layer 5)*: The next layer is the session layer in the context of OSI model which is responsible for establishing, managing, maintaining as well as terminating sessions which are used for communication between various applications. It syncs the communication and manages sessions through parameter negotiation, ensuring that the health of connections is properly maintained [11].
- *Presentation Layer (Layer 6)*: The Presentation is the sixth layer in the OSI model stack and this layer makes sure that the data which had been sent from the application layer from the end user in one system can be understood and read by application layer of another system. It is responsible for data translation, encryption, and compression. Its job is to format data in a way that makes it more accessible (e.g., JPEG for images, ASCII for text) [12].
- *Application Layer (Layer 7)*: The Application layer is the topmost layer of the OSI model. Here, user-facing applications are handled. It provides network services to end-user applications such as email protocols (SMTP, IMAP) and file transfer protocols (FTP). This layer is often misunderstood due to its underlying services that are used by the applications rather than the applications themselves [13].

  The next section deals with the basic implementation of botnets in general and then simulates the working of PoD and LAND attacks. A basic analysis regarding which of the above-mentioned layers are exploited to carry out DoS attacks using botnets in discussed in the later sections of the article.

2. Experimental Methodology

This section gives the details of botnet implementation and working. The basic introduction and architecture of the botnet was discussed in the previous sections of the paper, this section deals with the implementation of botnet communication using a python script and the script is intentionally made without any harmless functionality in it so as to be ethical and also provide a basic understanding regarding how botnets work in general.

- The below deals with a basic code for The Command-and-control server(C&C)

- Code for C&C:

```
import socket
# Command and Control (C&C) Server Setup
def start_server(host='127.0.0.1', port=65432):
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind((host, port))
    server_socket.listen()
    print(f"C&C Server started on {host}:{port}")
    while True:
        client_socket, client_address = server_socket.accept()
        print(f"Bot connected from {client_address}")
        command = input("Enter command for bot (or 'exit' to end): ")
        client_socket.sendall(command.encode())
        if command.lower() == 'exit':
            print("Shutting down C&C server.")
            break
        # Receive response from the bot
        response = client_socket.recv(1024).decode()
        print(f"Bot response: {response}")
        client_socket.close()
    server_socket.close()
# Start the server
start_server()

```

code: command-and-control centre in botnet using socket programming in python

The above deals with the coding implementation of command-and-control centre in a botnet-based architecture and the working is explained in later part of the section.

- Bot-Client Code:

```
import socket
def start_bot(server_host='127.0.0.1', server_port=65432):
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect((server_host, server_port))
    print("Connected to C&C server")
    # Receive command from server
    command = client_socket.recv(1024).decode()
    print(f"Received command: {command}")
    # Example: Bot responds with acknowledgment instead of any action
    response = f"Executed command: {command}"  # No actual execution
    client_socket.sendall(response.encode())
    client_socket.close()
# Start the bot
start_bot()

```

code: Bot Client implementation using python sockets

The working of the code is explained below in the series of four steps in detailed manner:

1. The server listens for incoming connections from bot clients.
2. The user (administrator) can input a command, which is then sent to the bot.
3. The server waits for a response from the bot and prints it for verification.
4. The connection is closed after the interaction.

The bot client connects to the C&C server. Upon receiving a command, the bot simply responds with an acknowledgment ("Executed command") without performing any malicious activity. This example is intentionally simple, designed to demonstrate how communication between the C&C server and bots occurs in a botnet. No harmful action is taken by the bot, ensuring this is a safe, educational example [3]. The above code showcases how simple it is to write set of advanced scripts for botnets and creating a DoS service attack to overcome this set of basic mitigation strategies are discussed in the next section.

Defence methods against Botnets:

The code in the above section typically showed how simple it was to write a simple botnet script and these raises set of security questions regarding how one can protect themselves against these attacks as well as how to mitigate if one notices DoS attack against them. The below are some of the ways one can defend themselves from a potential botnet attack:

- *Intrusion Detection Systems (IDS)*: Intrusion detection Systems (IDS) typically monitors the network traffics for set of odd patterns in general which may be seen in the context of botnet-based attacks. These patterns may be set of useless requests sent by various devices, sudden increase in the Network traffic in the server or communication to dubious IP addresses. These can be made intelligent enough so that they can assist in detecting suspicious activities and block such request thereby defending against botnets and DoS attacks in general [4].
- *Firewalls and traffic Filtering*: Set of network-based filters can be used for blocking known C&C server IP addresses or some common ports used by botnets while carrying out DoS attacks. Most modern firewalls can detect and prevent or block the traffic from botnets thereby not allowing it to enter the system [5].
- *Anti-Botnet Software*: Similar to antivirus software which are used for blocking malwares and other dangerous viruses in general. Antiviruses or software which can detect botnets in general can be used to detect and protect the systems from botnets in general. This software can be an endpoint security software in general. Similar to antiviruses even these software needs to provide regular security updates and security patches in general which can be used to even deal with the most recent botnets-based attacks in real world making it fully protected against most botnet-based attacks [6].
- *Rate Limiting and Load Balancers:* Rate limiting can be another strategy that can be used in real time to prevent the bots from flooding the server in general. This works by restricting the maximum number of request an IP address can make at the given instance of time. Load balancers can be used to distribute the load to another system when the server is full. This makes it harder to the botnet to flood a single server. The major disadvantage here is that the user needs to pay high cost for load balancing in general which may be another way to make the user suffer which may be the major goal of some adversaries [14].
- *Domain Name System (DNS) Security*: DNS filtering can help to identify and block requests to known botnet C&C servers, preventing bots from receiving commands [15].
- *Botnet Takedowns*: This may not be a user step to reduce or mitigate the botnet but still this is very important in the context of prevention against botnet-based attacks in general. The set of Law enforcements and cybersecurity firms may work together to identify and dismantle the botnet infrastructures that adversaries may have built across the internet. This typically includes shutting down the known C&C servers in general and arresting the set of people responsible for it [16].

The next sections deal with PoD and LAND attacks in a detailed manner as well as the simulation of them on a virtual machine to show how these DoS attacks work in general and the effect they have on these systems.

POD and LAND DDOS attacks:

LAND Attack (Local Area Network Denial Attack) is a type of denial-of-service attack where the attacker sends a specially crafted packet that makes the victim's machine try to send the packet back to itself. This is done by setting both the source and destination IP addresses to the victim’s own IP address. As a result, the system gets stuck in a loop trying to process this packet, which causes it to crash or become unresponsive. Older systems and software were particularly vulnerable to this type of attack, but modern systems have implemented measures to prevent such attacks. While it is not as commonly used today, the LAND attack serves as a reminder of the importance of validating incoming network data [1].

Ping of Death (PoD) Attack is another older attack that exploits a vulnerability in how systems handle oversized ICMP (Internet Control Message Protocol) packets. The attacker sends a "ping" (ICMP echo request) with a payload that exceeds the maximum allowed size, which causes the target system to crash or freeze when it tries to process this oversized packet. While this type of attack was a major concern in the late 1990s, many systems have since been patched to protect against it. Still, it demonstrates how vulnerabilities in handling basic network protocols can be exploited to cause significant damage to a system [2].

hping3 is a command-line network tool often used for testing and network scanning. It is a powerful tool that can send custom-crafted packets and is commonly used for DoS and DDoS attacks, including Ping of Death attacks. With hping3, users can manipulate packet flags, sequence numbers, and other fields in the packet to simulate various types of attacks or network behaviour. It can also be used for network diagnostics and penetration testing by generating traffic and analysing responses from firewalls, routers, or other network devices. Because of its versatility, hping3 is a popular tool among security professionals and attackers alike [3].

Hands on implementation of POD attack:

- The POD typically works on systems with older architecture, thereby for this experiment we use a windows 7 virtual machine and a kali Linux virtual machine and the implementation is discussed in the series of steps as mentioned below:
- *Step 01:*
  - Run command “ping 192.168.0.105 “to see if the target system is responding.
  - open (task manager) on Windows to check for CPU utilization.
  - Open Wireshark and the pink coloured one are ICMP packets. Apply the filter ICMP in Wireshark to observe only ICMP packet.
  - Check if the Kali Linux machine sent ping requests to target system and also if the target system has responded.
- Step2:
  - Open Second terminal in Kali Linux and run the command with byte size 65000

“ping 192.168.0.105 -s 65000”. The command can be seen in figure-2 inside the kali Linux VM.

  - compare the response time of 2 terminals on kali Linux. You will find out that the latency of the 2nd terminal is quite high compared to the other.
  - Check Your Impact on Windows: open Task Manager to monitor CPU impact, you will not find much impact on CPU utilization graph and stop the attack.
- Step 3:
  - once again, we attack on victim IP using the ping command this time, we introduce the time parameter(-i) to send packets as fast as possible. Minimum frequency for sending packets is 200ms. To do it faster you will require a privileged access.
  - Run “Sudo ping 192.168.0.105 -s 65000 -i 0.1” //this elevates privilege
- Step 4:
  - Look at Windows Task Manager for effect on CPU and it should still be at little impact at this point. You will find out small spike on CPU utilization graph. Stop the attack
- Step 5:
  - Run “Ping 192.168.0.105 -s 65000 -i 0.0001” //this command will send ICMP packets with higher frequency which will affect the performance of the system.
- Step 6:
  - End the attack.

The below are some of the snippets from the experiment as outlined from steps 1 to 6.

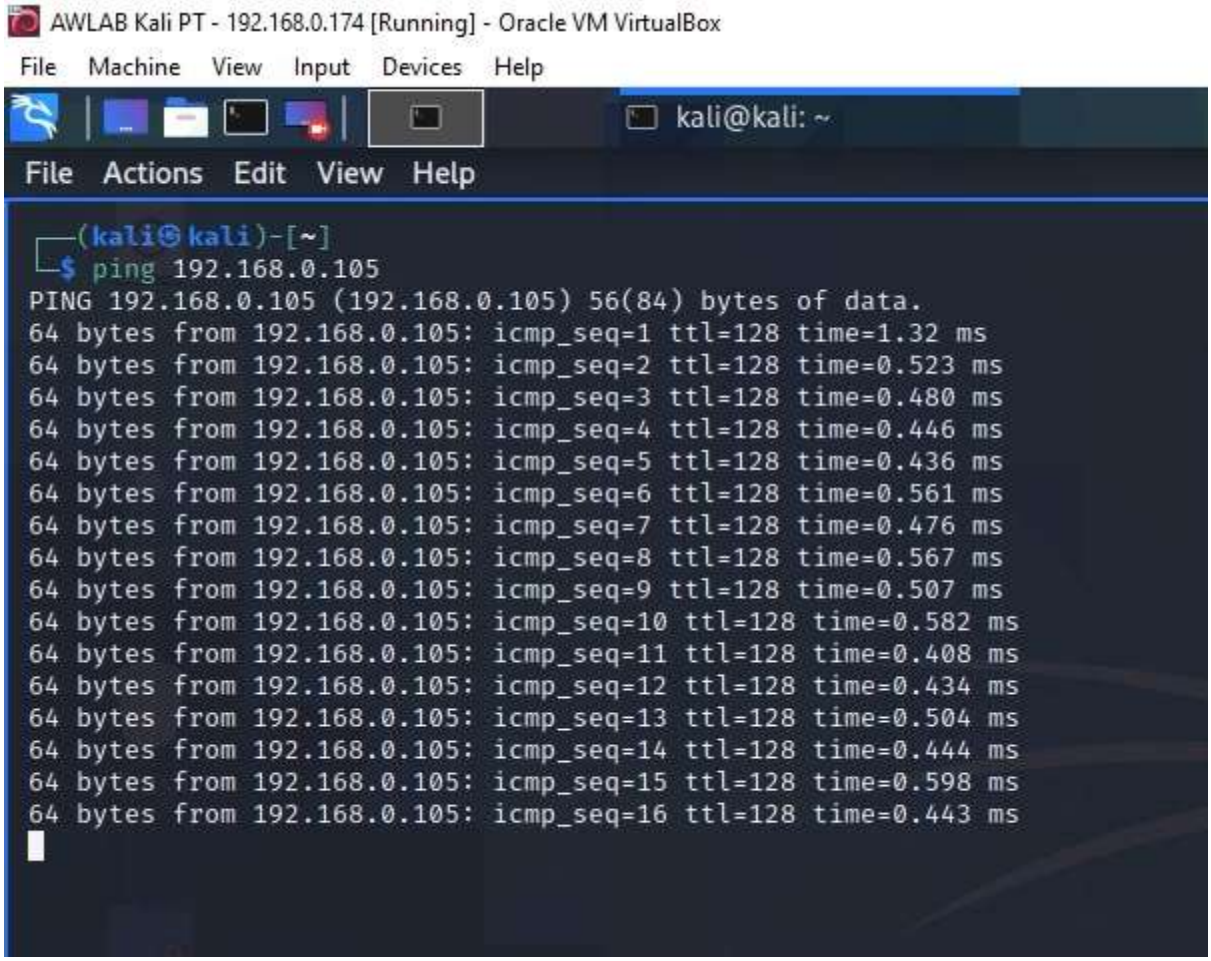

Figure-2 ping attack from kali vm to windows vm

The above figure, figure-3 showcases a basic ping attack that is done on the windows vm and the next figure shows the resource allocation graph indicating the effect this POD attack has on the victim machine

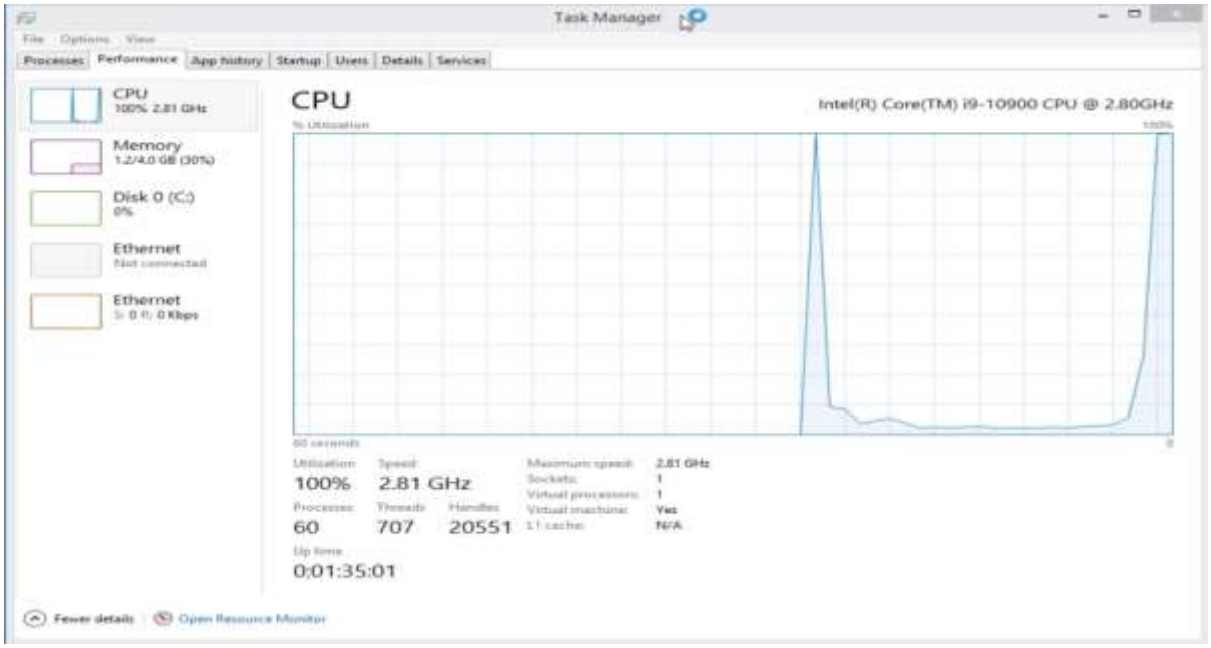

Figure 3- increased CPU utilization due to POD

The above attack is typically based on ICMP protocol as the ping sends set of ICMP packets to the targeted machine and can be seen in detail from the below figure, figure-4 which shows ICMP filter applied in Wireshark for the set of packets received.

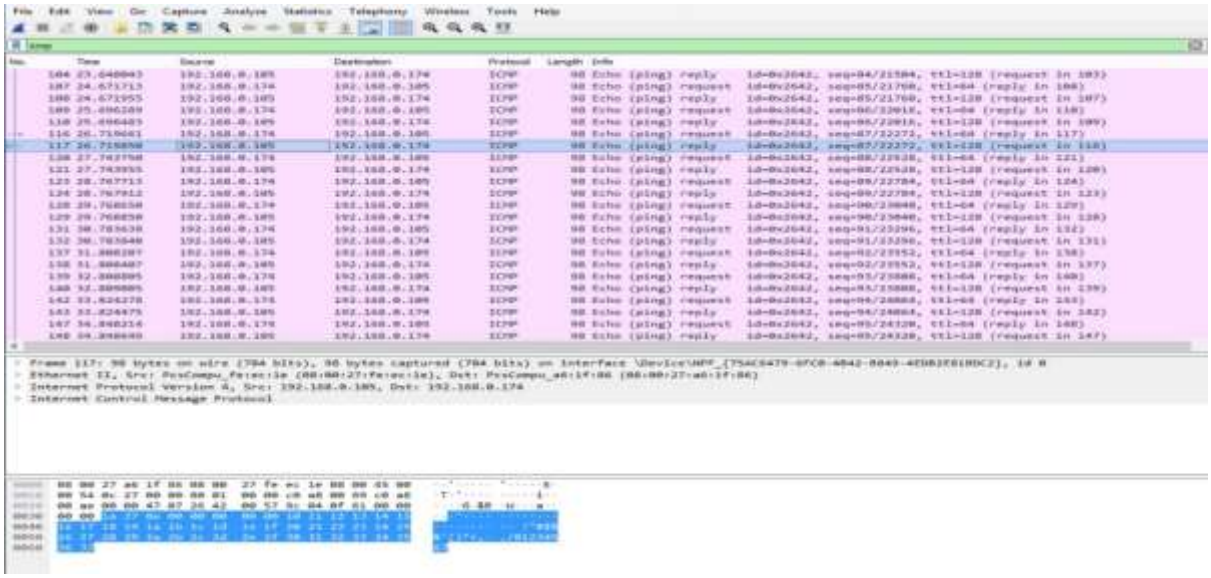

Figure 4- ICMP packets captured by windows machine

To summarize this attack does not work as of current scenario and this attack typically dealt with the network layer or layer 3 of the OSI model and worked on the basis of ICMP protocol. These typically do not work in the current scenario but in order to keep one's system protected from POD firewalls were used which would block out these pings and ensure that the system was not exhausted completely. This is one of the various types of DDOS attacks available and the rest are discussed in brief in the next section.

Hands On Implementation of LAND Attack:

LAND attack is an attack variant of the DoS attack. This attack works on 4th layer of the OSI model. Here the attacker spoofs the IP address of the victim and sends a TCP SYN packet back to the victim himself. Here the victim is now on a self-loop and keeps on sending and receiving the TCP SYN packet from himself. This leads to a DOS attack scenario. New versions of Operating Systems are not vulnerable to these attacks and firewalls can be configured to overcome such attacks in general.

For simulation of this attack Windows XP Virtual Machine and kali linux virtual machines are used.The commands used here are as follows:

For this attack we use hping3 command which is discussed in the previous subsection. Initially we need even port for carrying out the attack. The port selection can be done via nmap. hping is a basic packet crafting tool used in this attack.

The command we use is as follows:

for i in {1..100}; do hping3 192.168.0.180 -a 192.168.0.180 -p 445 -s 445 -S -c 1 -D; sleep 1;done;

here, 192,168.0.180 is the IP address of the windows machine and port chosen for the attack is 445 and -D is the debug mode which shows the detailed output on the window. This command sends the packet sleeps for 1 second and resends it in general. The output for the same can be seen in task manager of the windows machine which can be seen in *figure-6,* in a detailed manner.

Figure 5- hping command in the kali linux terminal

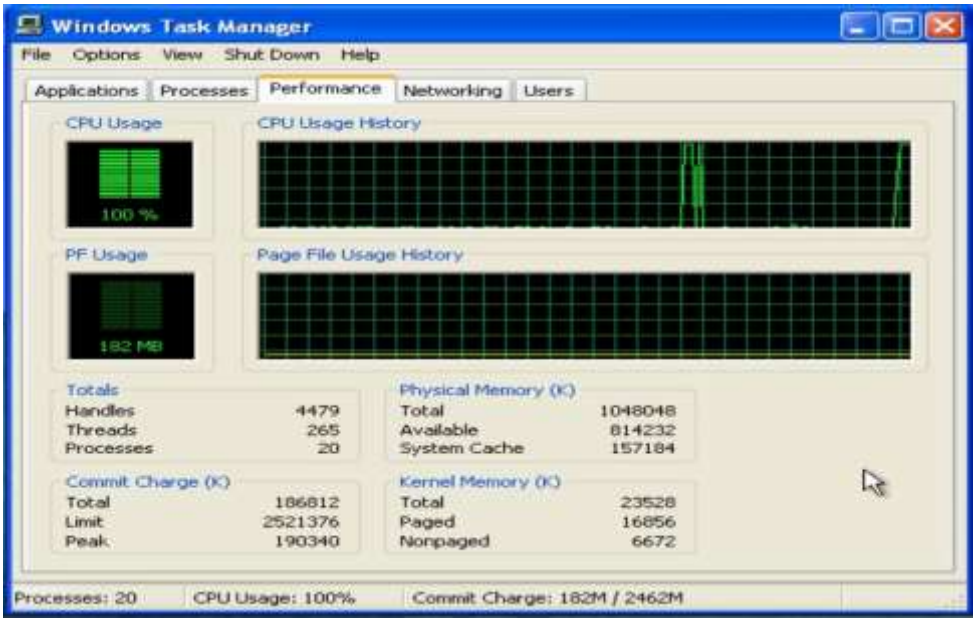

Figure 6:The task manager after executing the hping command in kali linux

The above figure, *figure 6* shows how the windows machine is typically affected by this self forwarding of packets which is carried out by the hping command in general. This attack does not work now and has been fixed by various operating systems over the years. These attacks could have been avoided on windows XP operating systems by configuring firewalls to block such packet request in general thereby making mitigating and protecting the systems against LAND attacks. The next section deals with the analysis of different types of DDOS attacks.

3. Results and Discussions:

The table below deals with the different types of DOS attacks which exists and also the OSI layer they affect in general and is given below:

Table-2: different types of DoS attacks

DDoS Attack Type	Layer Affected	Description	Unique Characteristics
UDP Flood	Transport (Layer 4)	Sends a large number of UDP packets to overwhelm the target, often exhausting network resources [7].	Spoofed source IPs make this attack challenging to trace back. Often mitigated by blocking UDP traffic from unknown sources.
SYN Flood	Transport (Layer 4)	Exploits the TCP handshake by sending multiple SYN packets, leaving the server waiting for replies [12].	Prevents legitimate connections from being established and commonly affects high-volume server applications.
ICMP (Ping) Flood	Network (Layer 3)	Overloads the network by sending a large volume of ICMP Echo requests, overwhelming bandwidth and capacity [11].	Effective in overloading network capacity; often mitigated by rate limiting ICMP traffic.
HTTP Flood	Application (Layer 7)	Mimics legitimate HTTP GET or POST requests to exhaust server resources by forcing processing of high numbers of connections [10].	Targets server resources directly, making it difficult to distinguish from legitimate traffic; common against websites.
DNS Amplification	Network/Transport (Layers 3–4)	Leverages open DNS resolvers to send amplified responses, overwhelming the target network [15].	Amplification attack; often increases data volume significantly (e.g., by 70x) and can be difficult to trace due to open resolvers.

NTP Amplification	Network/Transport (Layers 3–4)	Uses NTP servers' 'monlist' feature to send large response packets, flooding the target with traffic [11].	Produces high data volume and can be prevented by disabling 'monlist' on NTP servers.
Slowloris	Application (Layer 7)	Opens multiple HTTP connections with minimal data, keeping server connections active indefinitely [1].	Effective with low bandwidth and often bypasses traditional DDoS defenses; impacts servers with persistent connections.
Smurf Attack	Network (Layer 3)	Uses spoofed IPs in ICMP packets to flood the target with responses from other devices on the network [11].	Commonly mitigated by blocking ICMP broadcasts and implementing network ingress filtering.
Fragmentation Attack (e.g., Teardrop)	Network (Layer 3)	Sends fragmented packets that cause issues during reassembly, crashing or overloading the target system [8].	Exploits packet reassembly vulnerabilities, often in older or unpatched systems.

Botnet-Based Attack	Multiple Layers (Layers 3, 4, or 7)	A network of compromised devices simultaneously launches different attack types targeting multiple layers [2].	Highly adaptable and can be customized to target specific system weaknesses, often resulting in complex, layered attacks.
Application Layer Attack (Layer 7 DDoS)	Application (Layer 7)	Directly targets specific applications (e.g., login pages or APIs), simulating legitimate behavior to exhaust resources [4].	Requires advanced detection techniques; commonly targets high-value resources, making it difficult to distinguish from legitimate traffic.
Ping of Death (POD)	Network (Layer 3)	Sends malformed or oversized ping packets, causing the target system to crash [7].	Rare on modern systems due to updates, but effective on outdated or poorly configured devices.

The above table deals with the different types of DOS attacks and the layer they affect in the OSI model and set of unique characteristics about each one of them.

The below figure-7 deals with the graphical representation of the frequency number of attacks carried out on each layer of OSI stack. This graph is based on the well known attacks in general and the data is based on NIST findings.

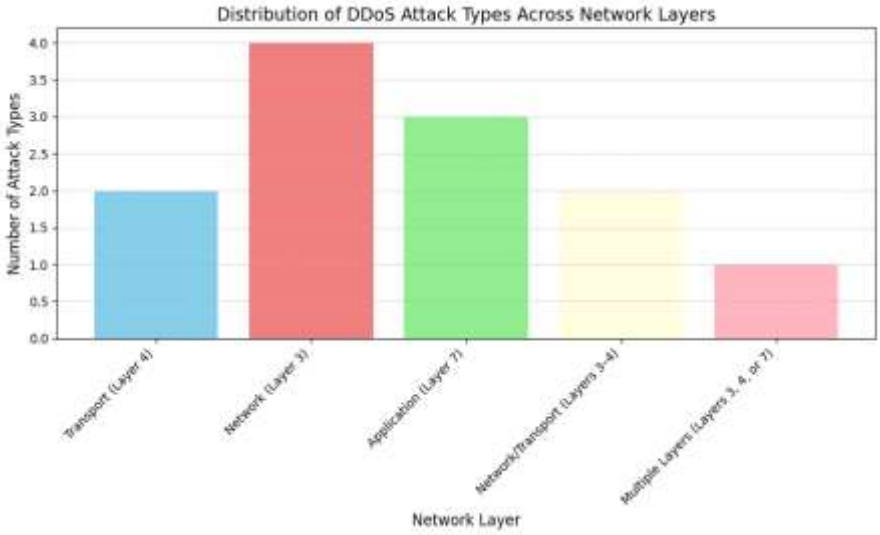

Figure 7- Distribution of DDoS attacks over the various network layers in the OSI stack

The next figure, *figure-8* deals with the characteristics analysis of various types DDoS attacks and compares it with botnets-based DDoS attacks.

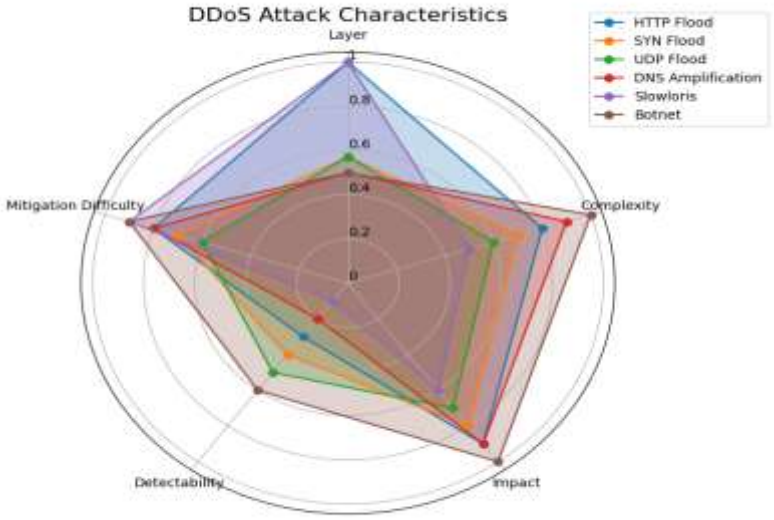

Figure-8 DDoS attacks characteristics comparison

The above figure deals with the various types of DDoS attacks and also showcases how dangerous each attack is specially botnet-based attacks as they are very difficult to mitigate and have a very high level of impact as compared to other types of attacks in general. The next section concludes the paper and also provides a basic summary of this article.

Conclusion:

The paper deals with the basic architecture of botnets and also explains how botnets work. Then the paper focuses on the various types of DOS attacks in general and deals with DDOS and RDOS in a detailed manner. The next focus was typically on the implementation of botnets via a python script and also set of preventive measures in order to prevent a DOS attacks were also discussed. The paper concludes with POD attack implementation and also deals with an in-depth analysis of the various types of DOS attacks which exists. The paper is written for just providing a basic introduction to botnet technology and also to enable the set of users to understand how they work in detail and not intended for usage in malicious and illegal applications.

Acknowledgement

Nill

Funding

No funding was received to carry out this study.

References

1. T. W. Ho, “Botnets: Overview, Control, and Prevention,” IEEE Transactions on Cybersecurity, vol. 9, no. 3, pp. 255–267, Mar. 2018.
2. S. M. Chen and Y. H. Lee, “Botnet Attacks and Defenses: A Survey,” International Journal of Computer Science and Network Security, vol. 17, no. 10, pp. 162–170, Oct. 2017.

3. D. S. Zeldovich and M. D. Kaafar, “Safe Botnet Simulation for Educational Use,” *Proceedings of the IEEE Cybersecurity Symposium*, 2019, pp. 42-47.
4. T. R. Brown and A. K. Singh, “Intrusion Detection Systems and Their Effectiveness in Mitigating Botnet Threats,” *Journal of Information Security*, vol. 12, no. 4, pp. 321–334, Jul. 2019.
5. J. W. Lee and M. C. Ha, “Effective Use of Firewalls to Block Botnet Attacks,” *Journal of Network and Computer Applications*, vol. 76, pp. 87–94, May 2019.
6. E. G. Gupta, “Anti-Botnet Software: Defending Against Modern Cyber Threats,” Cybersecurity Insights Journal, vol. 7, no. 2, pp. 54-61, Feb. 2020.
7. A. S. Tanenbaum and D. J. Wetherall, *Computer Networks*, 5th ed. Pearson, 2011.
8. W. Stallings, *Data and Computer Communications*, 10th ed. Pearson, 2014.
9. J. F. Kurose and K. W. Ross, *Computer Networking: A Top-Down Approach*, 7th ed. Pearson, 2016.
10. L. Peterson and B. Davie, *Computer Networks: A Systems Approach*, 5th ed. Elsevier, 2011.
11. C. S. R. Prabhu and K. S. Rajasekaran, “Network Layer Design Issues and Algorithms,” International Journal of Computer Science, vol. 9, no. 3, pp. 123–134, 2018.
12. R. H. Webber, *Internet and Web Technologies*, 2nd ed. Wiley, 2012.
13. H. M. Deitel and P. J. Deitel, *Internet & World Wide Web: How to Program*, 5th ed. Pearson, 2011.
14. L. H. Kim and R. J. Patel, “Rate Limiting as a Defense Against DDoS Attacks in Botnet Communication,” International Journal of Network Security, vol. 14, no. 1, pp. 75–82, Jan. 2020.
15. R. S. Jacobson, “Securing DNS Infrastructure Against Botnet Activities,” Computers & Security Journal, vol. 62, pp. 134-141, Sep. 2020.
16. F. X. Zhao and D. L. Weitzner, “Botnet Takedown and the Legal Implications of Cybercrime,” Journal of Cyber Law, vol. 10, no. 2, pp. 92-101, Aug. 2021.