Analysis of Linux Kernel Iptables for Mitigating DDOS Attacks; A Component-Based Approach

Ike, Uche Kingsley[1], Ononiwu, Chamberlyn[2] & Onumajuru, Joy Sonia[3]
[1-3]Department of Computer Science, Imo State Polytechnic Omuma Oru-East Nigeria
Corresponding email: uchebenzene@yahoo.com

DOI: 10.56201/ijcsmt.v9.no4.2023.pg12.22

ABSTRACT

Denial-of-Service (DoS) is a network security problem that constitutes a serious challenge to reliability of services deployed on the servers. DoS attacks aim is to exhaust a resource in the target system, reducing or completely subverting the availability of the service provided. Threat of DoS attacks has become even more severe with DDOS (Distributed Denial-of-Service) attacks. DDOS is an attempt by malicious users to carry out DoS attack indirectly with the help of many compromised computers on the Internet. Service providers are under mounting pressure to prevent, monitor and mitigate DoS/DDOS attacks directed towards their customers and their infrastructure. Defending against these types of attacks is not a trivial job, mainly due to the use of IP spoofing and the destination-based routing of the Internet. Literatures abound on DDOS attacks and mitigation strategies while many scholars attempt to demonstrate the effectiveness of one strategy against another. This paper however explores the Linux kernel firewall iptables, a defence-mechanism inherent in the Linux operating system capable of detecting intrusions and mitigating attacks. The paper is organized such that it presents the motivation for DDOS, various DDOS classifications and architecture, illustrates types of DDOS attacks and then extensively reviews iptables using a component based approach to show its efficient packet-filtering and analysis technique capable of mitigating DDOS attacks.

Keywords: *DoS, DDOS, Linux operating system, Security, Firewall, IP Tables.*

1. INTRODUCTION

The internet has enjoyed tremendous growth since it was created. Via the internet infrastructure, hosts can share files, as well as complete other tasks cooperatively by contributing their computing resources. Moreover, an end-host can easily join the network and communicate with any other host by exchanging packets. These are the encouraging features of the Internet; openness and scalability. However, persons with malicious intent can also take advantage of these internet features to launch attacks, perpetrated from numerous hosts, more powerful than those launched by a single machine. According to Fu et al. (2008), Denial-of Service Attack (DoS) is one type of such attacks. A Denial of Service (DoS) attack is a type of attack focused on disrupting availability of service. Such an attack can take many shapes, ranging from an attack on the physical IT environment, to the overloading of network connection capacity, or through exploiting application

weaknesses. In computing, a denial of service attack may be defined as an attempt to make a machine or network resource unavailable to its intended users, such as to temporarily or indefinitely interrupt or suspend services of a host connected to the internet.
Gligor et al.(1990) defined DoS as follows: "a group of otherwise authorized users of a specific service is said to deny service to another group of authorized users if the former group makes the specified service unavailable to the latter group for a period of time which exceeds the intended and advertised waiting time."
Internet and other networked infrastructure components are at risk of DoS for two primary reasons:

1. Resources such as bandwidth, processing power, and storage capacities are not unlimited and so DoS attacks target these resources in order to disrupt systems and networks.
2. Internet security is highly interdependent and the weakest link in the chain may be controlled by someone else, thus taking away the ability to be self-reliant.

In Distributed Denial of Service (DDOS) attacks, attackers do not use a single host for their attacks but a cluster of several dozens or even hundreds of computers to do a coordinated strike. The evolution of solutions to resolve the occurrence of attacks promoted the expansion of the attacks itself, and nowadays DoS attacks have been superseded by DDOS attacks (Arun and Selvakumar 2011). There are two major reasons making DDOS attacks attractive for attackers. The first is that there are effective automatic tools available for attacking any victim, i.e., expertise is not necessarily required. The second is that it is usually impossible to locate an attacker without extensive human interaction or without new features in most routers of the Internet (Abbilash and Sunil 2011).

2. DDOS MOTIVATION, CLASSIFICATION AND ARCHITECTURE

2.1. DDOS Motivation

DDOS attackers are usually motivated by various reasons. DDOS attacks based on the motivation of the attackers can be categorized into seven main classes (Prasad et al 2014):

i. Financial/economical gain: Attacks launched for financial gain are often the most dangerous and difficult to stop. These are mainly concern of corporations and require more technical skills and experience.
ii. Invariably slow network performance: The attacker launches an attack to block the resources of victim system, which slowdowns the performance of the system and in turn to the network.
iii. Revenge: Attackers of this kind are normally with lower technical skills and are frustrated individuals. They carry out these as a response to a perceived injustice.
iv. Ideological belief: Attackers in this category are inspired by their ideological beliefs to attack their targets. This category is one of the major incentives for the attackers to launch DDOS attacks.
v. Intellectual Challenge: attackers in this category attack the targeted systems for experiment and learn how to launch various attacks. They are usually young hacking enthusiasts who want to show off their competencies.
vi. Service unavailability: In this category, attacker overloads the services offered by the victim system through unwanted or fake traffic.

vii. Cyber warfare: Attackers of this class normally belongs to the military or terrorist organizations of a country and they are politically motivated to attack a wide range of critical sections of another country.

2.2. DDOS Classifications

Various classifications of DDOS attacks have been proposed in the literature. When DDOS attacks are classified based on the degree of automation; they are defined as Manual, Semi-automatic and Automatic attacks depending on the level of human effort required. Another classification of DDOS attacks is by considering attack rate i.e., how the rate of attack varies with respect to time. The classes are: Continuous Rate and Variable Rate attacks. The attack has constant flow in continuous rate after it is executed. But as in variable rate, attack changes its impact and flow with time, making it more difficult to detect and respond. Within variable rate, the attack rate can further be applied as Fluctuating or Increasing. Additionally, based on the data rate of attack, traffic in a network is also categorized as high rate and low rate DDOS attacks.

DDOS attacks are further classified 'by impact' i.e., in which the normal service is completely unavailable to users known as "disruptive", or it can be degrading the services of victim system in which it is not completely unavailable or decrease in the efficiency. In direct attacks, agents or zombie machines directly attack the victim system but in reflector attacks, zombies send request packets to a number of other compromised machines (PCs, routers etc.) called Bots and the reply generated s is targeted towards the victim system for an impact desired by the attacker. Example for this attack is sending huge amount of traffic as 'ping' request with spoofed IP address to the victim system to saturate bandwidth. The main classification of DDOS attacks is 'by exploited vulnerability' through which an attacker launches attack on the victim. In this classification, flood attack is used to block the victim's machine or network's bandwidth. This can be performed as TCP flood, UDP flood and ICMP flood. In general, all flooding attacks generated through DDOS can come as direct attacks or reflector attacks.

2.3. DDOS Attack Architecture

According to Prasad et al. (2014), DDOS attack networks use three types of architectures: the Agent-Handler architecture, Internet Relay Chat (IRC)-based architecture and the Web-based architecture.

i. Agent-Handler Architecture

The Agent-Handler architecture is also referred as Botnet-based architecture, in which the attacker uses the botnet to launch an attack. Generally a group of zombies or bots that are controlled by an attacker (also called a bot master) form a botnet. Botnets consist of master, handlers, and bots as shown in Figure 1. The handlers are means of communication that attackers use to command and control indirectly the bots. Bots are devices that have been compromised by the handlers and that will carry out the attack on the victim's system.

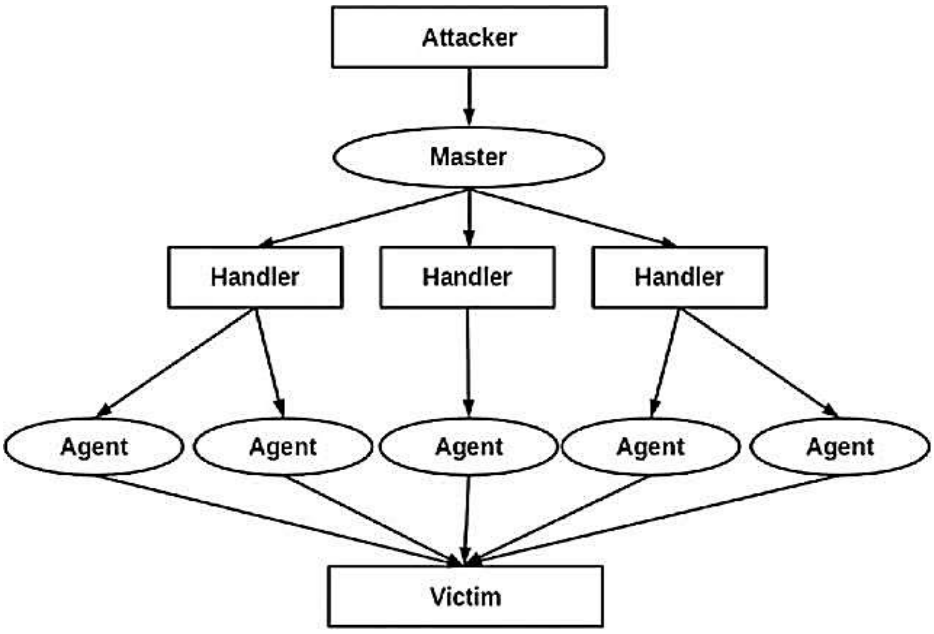

Figure 1: Agent-Handler Architecture

ii. **Internet Relay Chat (IRC) -based architecture**

The bot master or controller launches an attack through the bots by sending the commands to them which in turn behave according to the master instructions. At the other end the bot sends the response or the status information to the master. Their communication is done through public chat systems instead of doing these with their original addresses. If they use the original identity or private channels, the detection system easily track and block the location and system. Internet relay chat (IRC) is the one which allows the users to communicate without performing any authentication check and no security to user communications. IRC provides a text-based command syntax protocol to define the rules and regulations to the users and that is installed widely across the network. There is huge number of existing IRC networks available in the internet and which can be used as public exchange points, but the majority IRC networks doesn't contain any strong authentication. The wide variety of tools in the internet is available to provide anonymity on IRC networks. Therefore, IRC provides simple, low latency, widely available, and anonymous command and control channel for botnet communication.

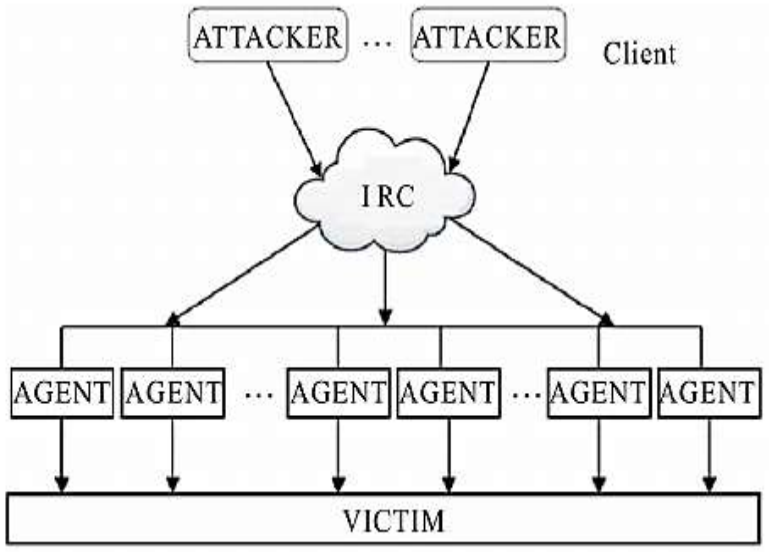

Figure 2 IRC Architecture

iii. **Web-based architecture**

Botnets are using HTTP as a communication protocol to send commands to the bots making it more difficult to track the DDOS command and control structure. Like IRC-based Botnets, web-based botnets do not maintain connections with command and control servers or handlers. The Web bots downloads the instructions using web requests periodically. Web-based Botnets are stealthier since they hide themselves within legitimate HTTP traffic. Advanced web developments languages (PHP, ASP, JSP, etc.) through encrypted communication over HTTP or HTTPS protocol are used to configure and control the bots.

2.3. DDOS Strategy

A Distributed Denial of Service (DDOS) attack consists of several elements. The steps in launching a DDOS attack are:

1. **Discover vulnerable hosts or agents**: The attacker selects the agents to perform the attack. Any system which is running with no antivirus software or pirated copies of software in internet is vulnerable and operated as a compromised system. Attackers utilized these compromised hosts or bots for further scanning and compromises. Attacker generates the attack stream by using the abundant resources of these compromised machines.
2. **Compromise:** The attacker exploits vulnerabilities and security holes of the agent machines and installs the attack code.

3. **Communication**: The attacker communicates with the handlers to identify the active agents, to schedule attacks or to upgrade agents. The communication among the attackers and handlers such as the type, length, Time to Live (TTL), and port numbers.

3. TYPES OF DOS/DDOS ATTACKS

In DOS/DDOS attacks, the attacker sends packets directly from his computer(s) to the victim's site but the source address of the packets may be forged. There are many tools available to allow this type of attack for a variety of protocols including ICMP, UDP and TCP. The common DDOS attack types are:

i. **UDP Flood Attack**

In UDP Flood attack, attacker sends large number of UDP packets to a victim system, due to which there is saturation of the network and the depletion of available bandwidth for legitimate service requests to the victim system (Lau et al. 2000). A UDP Flood attack is possible when an attacker sends a UDP packet to a random port on the victim system. When the victim system receives a UDP packet, it will determine what application is waiting on the destination port. When it realizes that there is no application that is waiting on the port, it will generate an ICMP packet of "destination unreachable" to the forged source address. If enough UDP packets are delivered to ports of the victim, the system will go down. By the use of a DoS tool the source IP address of the attacking packets can be spoofed and this way the true identity of the secondary victims is prevented from exposure and the return packets from the victim system are not sent back to the zombies (Markovic et al. 2004).

ii. **ICMP Flood Attack**

ICMP Flood attacks exploit the Internet Control Message Protocol (ICMP), which enables users to send an echo packet to a remote host to check whether it's alive. More specifically during a DDOS ICMP flood attack the agents send large volumes of ICMP_ECHO_ REPLY packets ("ping") to the victim. These packets request reply from the victim and this results in saturation of the bandwidth of the victim's network connection (Lau et al. 2000). During an ICMP flood attack the source IP address may be spoofed.

iii. **SYN Flood Attack**

In a SYN Flood attack, the victim is flooded with half open connections. The client
System begins by sending a SYN message to the server. The server then acknowledges the SYN message by sending SYN-ACK message to the client. The client then finishes establishing the connection by responding with an ACK message. The connection between the client and the server is then open, and the Service-specific data can be exchanged between the client and the server. The potential for abuse arises at the point where the server system has sent an acknowledgment (SYN-ACK) back to client but has not yet received the ACK message. This is known as half-open connection. The server has built in its system memory a data structure describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially-open connections. Creating half-open connections is easily accomplished with IP spoofing. The attacking system sends SYN messages to the victim server system; these appear to be legitimate but in fact reference a client system that is unable to respond to the SYN-ACK messages. This means that the final ACK message will never be sent to the victim server system.

iv. **Smurf Attack**

In a "smurf" attack, the victim is flooded with Internet Control Message Protocol (ICMP) "echo-reply" packets. On IP networks, a packet can be directed to an individual machine or broadcast to an entire network. When a packet is sent to an IP broadcast address from a machine on the local network, that packet is delivered to all machines on that network. In the "smurf" attack, attackers are using ICMP echo request packets directed to IP broadcast addresses from remote locations to generate denial-of-service attacks. When the attackers create these packets, they do not use the IP address of their own machine as the source address. Instead, they create forged packets that contain the spoofed source address of the attacker's intended victim. The result is that when all the machines at the intermediary's site respond to the ICMP echo requests, they send replies to the victim's machine. The victim is subjected to network congestion that could potentially make the network unusable.

v. **Teardrop Attack**

This type of denial of service attack exploits the way that the Internet Protocol (IP) requires a packet that is too large for the next router to handle to be divided into fragments. The fragment packet identifies an offset to the beginning of the first packet that enables the entire packet to be reassembled by the receiving system. In the teardrop attack, the attacker's IP puts a confusing offset value in the second or later fragment. If the receiving operating system does not have a plan for this situation, it can cause the system to crash (Oh et al 2009).

vi. **Land Attack**

The attack involves sending a spoofed TCP SYN packet (connection initiation) with the target host's IP address to an open port as both source and destination. The reason a LAND attack works is because it causes the machine to reply to itself continuously. Land attacks have been found in services like Simple Network Management Protocol (SNMP) and Windows 88/TCP (Kerberos/global services) which were caused by design flaws where the devices accepted requests on the wire appearing to be from themselves and causing replies repeatedly.(Gorodetsky et al. 2003)

3. OVERVIEW OF LINUX OPERATING SYSTEM

Linux was first released on September 12th 1991 by Linus Torvalds (Torvalds, 1991). Linux is an open source operating system (Unix-like) based on the Linux kernel (Eckert, 2012). The operating system is typically packaged as a Linux distribution which includes the kernel and supporting system software and libraries. Originally developed for personal computers based on Intel X86 architecture, Linux operating system has since been ported to more platforms than any operating system (Levine, 2013). Linux also runs on embedded systems.

The “shell”, a term used to describe the user interface is either a command line interface (CLI), a graphical user interface (GUI) or controls attached to the associated hardware, which is common for embedded systems. For desktop systems, the default user interface is usually graphical although CLI is commonly available through terminal emulator windows or a separate virtual console.

The difference between Linux and many other common modern operating systems is that Linux kernel and other components are free and open source software. Linux is not the only such operating system although it is by far the most widely used. Linux based distributions are intended by developers for interoperability with other operating system and established computing standards.

Most programming languages support Linux either directly or through third party community based ports. The original development tools used for building both Linux applications and operating system programs are found within the GNU built system. Many programming languages have a cross platform reference implementation that supports Linux. For example, PHP, Perl, Ruby, Python, Java, Rust and Haskell.

The Linux kernel is a widely ported operating system kernel, available for devices ranging from mobile phones to supercomputers; it runs on a highly diverse range of computer architectures.

4. REVIEW OF RELATED WORKS

Kolahi et al. (2015) evaluated the impact of various defence mechanisms, including access control lists (ACLs), Threshold limit, Reverse Path Forwarding (IP verifying) and Network Load Balancing. The researchers discovered that threshold limit was found to be the most effective defence. The work of Papedie et al. (2017) analysed the impact of HOIP and slowloris based DDOS attacks on both windows and Linux webservers. They also evaluated several software protection mechanisms including IIS and Dynamic IP restrictions for windows server and mod evasive, iptables, Fail2Ban for Linux server. Xuan and Wu (2015) examined the working principle of pintables and rule set and also proposed an algorithm to optimize the rule set implemented based on Linux system. In another work, Mihalos et al. (2019) examined network security threats policies and mechanisms and also analysed the firewall as a network concealing technology by elaborating the Netfilter/iptables as an implementation mechanism. Sharma & Bajaj (2011) created a virtual network environment using MS Virtual PC (SPI) and captured/analysed network packets using open source network protocol analyser Wire shark while the scripts to block/allow the network traffic were done using Iptables. Nwachukwu et al. also created a virtual network environment in which they attempted to mitigate DDOS attacks using Linux kernel Iptables and Bash scripting.

Much of related works focuses on demonstrating the effectiveness of various defence mechanisms using a comparative approach, however in this paper we explore the Linux Iptables to bring to fore its components that enables its packet filtering capabilities. .

5. LINUX KERNEL FIREWALL; IPTABLES

According to Rash (2007), the iptables firewall is developed by the Netfilter project and has been available to the masses as part of the Linux operating system since the release of the Linux 2.4 kernel in January 2001.

Iptables is a user-space application program that allows a system administrator to configure the tables provided by the Linux Kernel firewall (implemented as different Netfilter modules) and the chains and rules it stores. Different kernel modules and programs are currently used for different protocols; iptables apply to IPV4, ip6tables to IPV6, arptables to ARP and ebtables to Ethernet frames. Over the years, iptables has matured into a formidable firewall with most of the functionality typically found in proprietary commercial firewalls (Rash, 2007). For example, iptables offer comprehensive protocol state tracking, packet application layer inspection, rate limiting and a powerful mechanism to specify a filtering policy. All Linux distributions include iptables and many prompt the user to deploy an iptables policy right from the installer.

Iptables requires elevated privileges to operate and must be executed by user root, otherwise it fails to functions. On most Linux systems, iptables is installed as /usr/sbin/iptables and documented in its mans page which can be opened using man iptables when installed. It may also be found in /sbin/iptables but since iptables is more like a service rather than an "essential binary", the preferred location remains /usr/sbin.

The term iptables is also used to inclusively refer to the kernel level components. Xtables is the name of the kernel module carrying the shared code portion for use by all, for modules that also provides the API used for extension; subsequently, Xtables is more or less used to refer to the entire firewall (v4, v6, ARP, and eb) architecture.

The successor of iptables is nftables, which was merged into the Linux kernel mainline in kernel version 3.13, which was released on 19 January 2014. Xtables allows the system administrator to define tables containing chains of rules for the treatment of packets. Each table is associated with a different kind of packet processing. Packets are processed sequentially traversing the rules in chains. A rule in a chain can cause a Goto or Jump to another chain and this can be repeated to whatever level of nesting is desired. (A jump is called a "call" i.e the point that was jumped from is remembered). Every network packet arriving at or leaving from the computer traverses at least one chain.

The origin of the packet determines which chain it traverses initially. There are five predefined chains (mapping to the five Netfilter hooks), though a table may not have all chains. Predefined chains have a policy, for example DROP, which is applied to the packet if it reaches the end of the chain. The system administrator can create as many other chains as desired. These chains have no policy, if a packet reaches the end of the chain, it is returned to the chain which called it. A chain may be empty.

5.1 IP Packet Filtering

Iptables is an ipfilter which is shipped with kernel. Technically speaking, an IP filter will work on network layer in TCP/IP stack but actually iptables work on data link and transport layer as well. In a broad sense, iptables consists of tables; constructs that outlines broad categories of functionality such as packet filtering or Network Address Translation and which consists of chain which is further comprised of rules as shown in the figure 3.

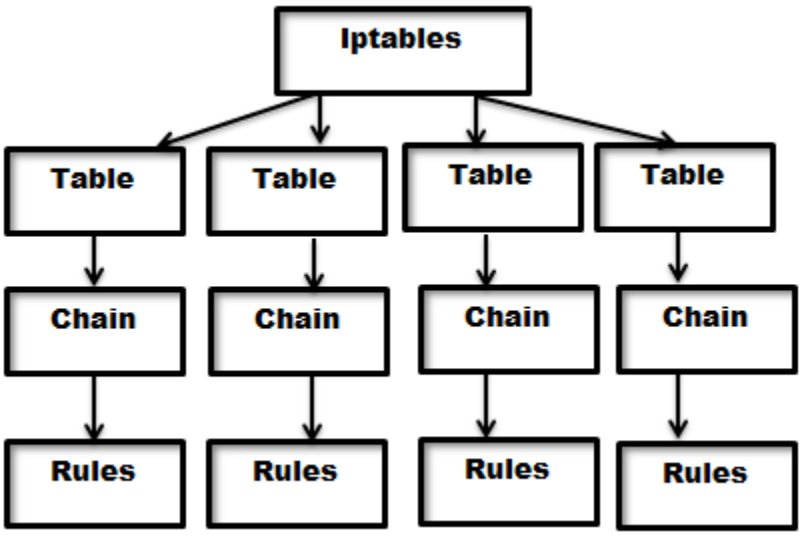

Figure 3: Iptables Components

Default tables are:

i. Raw: this table is used for mainly configuring exemptions from connection tracking with the NOTRACK target. It registers at the Netfilter hooks with higher priority and thus called before ip_conntrack or any other iptables. It provides the following built in chains: PREROUTING AND OUTPUT.
ii. Mangle: this table is used for specialized packet alteration. It consists of five built-ins: PREROUTING, INPUT, FORWARD, OUTPUT and POSTROUTING.
iii. NAT: this table is consulted when a packet that creates a new connection is encountered. It consists of three built-ins: PREROUTING, OUTPUT and POSTROUTING.
iv. Filter: this is the default table (if no t-option is passed). It contains the built-in chains: INPUT, FORWARD and OUTPUT. Among all the tables, the filter table (default) is the one which is most generally used. Filter table uses three of the predefined chains INPUT, FORWARD and OUTPUT.

5.1.1 The default chains

1. PREROUTING: used by raw, mangle, and nat tables. Packets will enter this chain before a routing decision is made.
2. INPUT: used by mangle, and filter tables. Packet is going to be locally delivered; it does not have anything to do with processes having an open socket. Local delivery is controlled by the “local delivery” routing table

3. FORWARD: used by mangle and filter tables. All packets that have been routed and were not for local delivery will traverse this chain.
4. OUTPUT: used by raw, mangle, nat, and filter tables. Packets sent from the machine itself will be visiting this chain.
5. POSTROUTING: used by mangle and nat tables. Routing decision has been made. Packets enter this chain just before handing them off to the hardware.

Each rule in a chain contains the specification of which packets it matches. It may also contain a target (used for extension) or verdict (one of the built-in decisions). As a packet traverses a chain, each rule in turn is examined. If a rule does not match the packet, the rule takes the action indicated by the target/verdict, which may result in the packet being allowed to continue along the chain or it may not. Matches make up the large part of rule sets, as they contain the conditions packets are tested for. These can happen for about any layer in the OSI model, as with for example the mac-source and the –p tcp – dport parameters and there are also protocol-dependent matches such as –m time.

The packet continues to traverse the chain until any one of the following happens:

i. A rule matches the packet and decides the ultimate fate of the packet. For example by calling one of the ACCEPT OR DROP.
ii. A rule calls the return verdict in which case processing returns to the calling chain.
iii. The end of the chain is reached; traversal either continues in the parent chain (as if RETURN was used), or the base chain policy which is an ultimate fate is used.

Below is a list of the most common targets which is executed when it is matched against criteria:

i. ACCEPT: packet is accepted and goes to the application for processing.
ii. DROP: packet is dropped. No information regarding the drop is sent to the sender.
iii. REJECT: packet is dropped and information (error)message is sent to the sender
iv. LOG: packet details are sent to the syslodg for logging.
v. DNAT:rewrites the destination IP of the packet
vi. DNAT: rewrites the source IP of the packet.

Of all the targets described the ACCEPT, DROP, REJECT and LOG verdicts are used in filter tables a lot.

Some of the common criteria:

- -p <protocol>: it matches protocol like TCP, UDP, ICMP and all.
- -s <ip_addr>: it matches source ip address.
- -d <ip_addr>: it matches destination ip address.
- -sport <port>: it matches the source port.
- -dport <port>: it matches the destination port.
- -I <interface>: it matches the interface from which the packet entered.
- -o <interface>: it matches the interface from the packet exit.

6.0 CONCLUSION

Developing mitigation strategies for DoS/DDOS attacks is a part of an overall risk management strategy for an organization. Each organization must identify the most important DoS/DDOS risks, and implement a cost-effective set of defence mechanisms against those attack types causing the highest risk for business continuity. Many different defence mechanisms are typically needed to mitigate DoS/DDOS attacks, but it is not cost-effective to blindly choose a large set of defence mechanisms against DoS/DDOS attacks. In this paper, the components of Linux kernel iptables which makes the Linux OS an efficient platform for packet filtering and matching to mitigate these attacks are explored. To determine whether the network traffic is legitimate or not, an iptables relies on a set of rules it contains that are predefined by a network or system administrator. These rules tell the iptables whether to consider as legitimate and what to do with the network traffic coming from a certain source, going to a certain destination, or having a certain protocol type.

REFERENCES

Abhilash, C.S. & Sunil, K.P. (2011) "Mitigation of Distributed Denial of Service (DDOS) Threats". Conference on advances in computational techniques (CACT)

Arun, R. & Selvakumar, S. (2011) "Distributed Denial of Service Attack Detection Using an Ensemble of Natural Classifier". International Journal of Computer Communication. Elsevier publication United Kingdom vol. 34

Gligor, V.D., & Yu, C.F. (1990) "A specification and verification method for presenting Denial of Service. IEEE trans. Software, vol.6 issue 6.

Gorodestsky, V., Kotenko, I. & Michael, B. (2003) "Multi-Agent Modeling And Simulation Of Distributed Denial Of Service Attacks On Computer Networks". Third international conference on navy and ship building nowadays (NSN), Kyrlov ship building research institute, st. Petersburg.

Lau, F., Rubin, S., Smith, M., & Trajkovic, L. (2000) "Distributed Denial Of Service Attack" IEEE International Conference On Systems, Man, And Cybernetics". Nashville.

Lei-Fei Xuan and Pei-Fei Wu (2015) 2nd International Conference On Information Science and control Engineering 988-991.

M.G Mihalos, S.I Nalmpantis, K. Ovaliadis (2019), "Design and Implementation of firewall security policies using Linux Iptables", Journal of Engineering Science and Technology Review 12(1) (2019)

Markovic, J. & Reiher, P. (2004) "A Taxonomy of Solutions to DDOS and DDOS Defense Mechanisms" ACM SIGCOMM computer communication Review vol 34 No.2 pg 39-53.

Micheal Rash (2007), "Linux Firewalls: Attack Detection and Response". Books.google.com

Nwachukwu V.C, Ikerionwu C.O, John-Otumu A.M (2021) "An Enhanced Model for Mitigating DDOS Attacks on Linux Servers using IPTables and Bash scripts. International journal of Advanced Trends in Computer Applications.

Oh, H., Doh, I. & Chae, K. (2009) "Attack Classification Based On Data Mining Technique And Its Application For Reliable Medical Sensor Communication". International journal of computer application. Vol 6 No.3

Prasad, M., Reddy, A. & Rao, K. (2014) "DoS and DDOS attacks: Defense, Detection and Traceback mechanism-A survey. Global journal of computer Science and technology: E network, web and security vol 14 issue 7.

Roxana Papadie, Loana Apostol (2017) "Analysing websites protection mechanisms against DDOS attacks. 9th International Conference On Electronics, Computer And Artificial Intelligence.

Simad, S Kolahi, Kiattikul Treseangrat, Bahman Sarrafpour (2015) 'Analysis of UDP DDOS flood cyber-attack and defence mechanisms on webserver with Linux Ubuntu 13. Conference on communications, signal processing and their applications.