Securing TCP/IP Stacks Using IP Tables

Hamisu I. Usman[1], **J. Dharani**[2]

[1]M.Tech Student, IT Department SRM University Chennai, India
[2]Associate Professor, IT Department SRM University Chennai, India

Abstract: ***The Transmission Control Protocol/Internet Protocol (TCP/IP) suite has become a widely method of interconnecting hosts, networks, and the internet at large. The TCP/IP protocol suite is vulnerable to a variety of attacks ranging from password sniffing to devastating attacks such as denial of service (DoS), and one thing that makes this attack successful is that most of the software uses to carry out those type of attacks are freely available on the internet. A DoS attack is an attempt by an attacker to flood a user's or an organization's system. There are two main categories of DoS attacks. DoS attacks can be either sent by a single system to a single target (simple DoS) or sent by many systems to a single target (DDoS). The goal of DoS isn't to gain unauthorized access to machines or data, but to prevent legitimate users of a service from using it.***

Keyword: TCP/IP, DoS, Sniffing, Internet, Flooding.

1. Introduction

The Transmission Control Protocol/Internet Protocol (TCP/IP) is the most widely used protocol suite today, which was developed under the sponsorship from DARPA (Defense Advanced Research Projects Agency). It is the de facto standard employed to interconnect computing facilities in modern network environments. The different communication and application protocols that regulate how computers work together are commonly visualized as belonging to a layered organization of protocols that is referred to as the TCP/IP protocol stack [1]. TCP/IP protocol suite is designed through a highly structured and layered approach, with each layer responsible for a different facet of communications [15]. This hierarchical architecture makes each protocol layer possible to develop independently without affecting the adjacent layer. Data encapsulation is achieved by various headers among different transportation layers, like IP header, TCP header or application header. These headers are critical in keeping the state information for each network connection facilitating the multiplexing and de-multiplexing of communication messages.

2. IP (Internet Protocol)

The IP (Internet Protocol) is the workhorse protocol of the TCP/IP protocol suite, which provides an unreliable, connectionless datagram delivery service. All TCP, UDP (User Datagram Protocol), ICMP (Internet Control Message Protocol), and IGMP (Internet Group Management Protocol) data are transmitted as IP datagrams.

Within IP header, there is important information like source IP address, destination IP address, which plays an important role in routing the packet around the network. Deliver such a packet to the correct destination host is non-trivial, especially in a large scale network system like Internet. Each intermediate routing device makes best effort to deliver the IP datagram, but there is no guarantee it will reach the destination finally. So, datagram can be lost, duplicated, or delivered out of order. It is the task of higher layer protocol to perform error control and flow control.

3. Transmission Control Protocol

Transmission Control Protocol **(**TCP) is built on top of IP layer, which is unreliable and connectionless. However, TCP provides the higher layer application, a reliable connection-oriented service [15]. As a trade-off, each TCP connection requires an establishment procedure and a termination step between communication peers. Furthermore, TCP also provides sequencing and flow control. Without options, TCP header occupies 20 bytes. The source and destination port number is used to identify the sending and receiving processes. The sequence number is essential in keeping the sending and receiving datagram in proper order. There are six flag bits with the TCP header, namely URG, ACK, PSH, RST, SYN and FIN, each of them has a special use in the connection establishment, connection termination or other control purposes. Window size is advertised between communication peers to maintain the flow control.

TCP establishes a connection in three steps, namely three-way handshake.

4. Three-Way Handshake

The TCP three-way handshake in Transmission Control Protocol (also called the TCP-handshake; three messages handshake and/or SYN-SYN-ACK) is the method used by TCP set up a TCP/IP connection over an Internet Protocol based network. TCP's three way handshaking technique is often referred to as "SYN-SYN-ACK" (or more accurately SYN, SYN-ACK, ACK) because there are three messages transmitted by TCP to negotiate and start a TCP session between two computers. The TCP handshaking mechanism is designed so that two computers attempting to communicate can negotiate the parameters of the network TCP socket connection before transmitting data such as SSH and HTTP web browser requests [13].

First, source host sends a SYN packet to destination host, telling it the wish to establish a connection and setting its own ISN (Initial Sequence Number) in sequence number field. Upon receiving the request packet, the destination host sends back a SYN_ACK packet with its own ISN and the incremented ISN from source host. Finally, the source host will transmit an ACK packet and data transfer can take place. There is one extra point need to mention. Suppose that host S did not send any SYN packet but received a SYN_ACK packet from host D, it would just send back a RST packet to reset the connection [3], [8], [11].

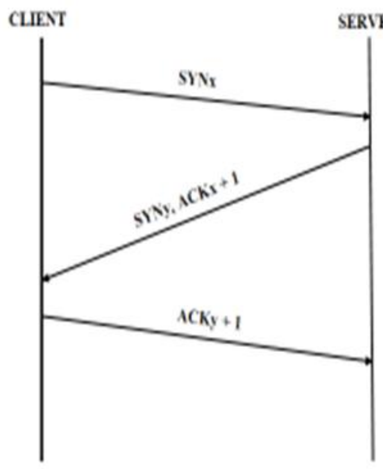


Figure 1: Three-way handshake

5. SYN Flooding

SYN flooding is a specially designed attack, which employs a flood of SYN packet to consume all available new network connections on a targeted host, resulting in delays responding to legitimate network connection requests and eventual halting the service provider. Theoretically, this attack applies to all TCP connections, such as WWW, Telnet, e-mail, and so on. Under most UNIX systems, several memory structures need to be allocated for each TCP connection establishment [16]. Take BSD system as an example, a socket structure is used to hold the communication information, like protocol used, address information, connection queues, buffers and flags.

The goal of SYN Flooding or (DoS) isn't to gain unauthorized access to machines or data, but to prevent legitimate users of a service from using it. The aims of the attacker is to achieve one of the following [18]:

- Flood a network with traffic, thereby preventing legitimate network traffic.
- Disrupt connections between two machines, thereby preventing access to a service.
- Prevent a particular individual from accessing a service.
- Disrupt service to a specific system or person

Moreover, there are two other memory structures have special meanings to a TCP connection, namely IP control block (inpcb) and TCP control block (tcpcb), which keep the TCP state information, port numbers, sequence numbers and several connection-related timers, etc. Typically, these structures will take at least 280 bytes in total. A normal scenario of TCP connection starts with the system in LISTEN State while receiving a SYN packet, which is being examined for checksum immediately.

If checksum is incorrect, the packet will be discarded silently, with the expectation that the remote site will retransmit it. Otherwise, the TCP control block associated with this connection is being searched for. If no such item is found, it means no server process is waiting for this packet, then the packet will be removed and an RST packet is used to inform remote client process [13].

On the other hand, if a server process is located and the corresponding backlog queue happens to be not full, several memory structures will be allocated for this connection and a SYN_ACK packet will be sent to continue the three way handshake. At same time, system enters SYN_RECVD State and starts a connection establishment timer. Most TCP/IP implementation uses exponential back-off algorithm and set this timer as 75 seconds. If the final ACK arrives before the timer expires, the request will leave kernel space and goes to backlog queue or application process space. Otherwise, the timer expires and three-way handshake fails [11]. Under both cases, the corresponding memory structures will be de-allocated from kernel space.

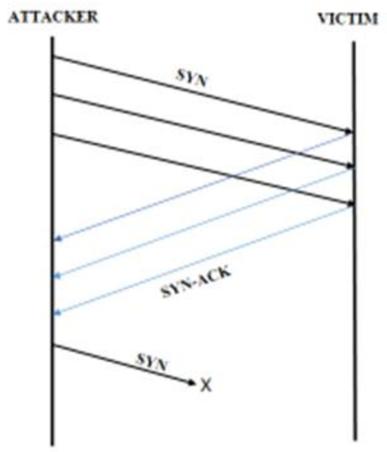


Figure 2: SYN-Flooding

6. IP Spoofing

In computer networking, the term IP address spoofing or IP spoofing refers to the creation of Internet Protocol (IP) packets with a forged source IP address, called spoofing, with the purpose of concealing the identity of the sender or impersonating another computing system. In a spoofing attack, the intruder sends messages to a computer indicating that the message has come from a trusted system. To be successful, the intruder must first determine the IP address of a trusted system, and then modify the packet headers to that it appears that the packets are coming from the trusted system. In essence, the attacker is fooling (spoofing) the distant computer into believing that they are a legitimate member of the network. The goal of the attack is to establish a connection that will allow the attacker to gain root access to the host, allowing the creation of a backdoor entry path into the target system.

7. Methods of Attacks

There are so many types of denial of service, Smurf attack involves an attacker sending a large amount of Internet Control Message Protocol (ICMP) echo traffic to a set of Internet Protocol (IP) broadcast addresses [2]. The ICMP echo packets are specified with a source address of the target victim (spoofed address). Most hosts on an IP network will accept ICMP echo requests and reply to them with an echo reply to the source address, in this case, the target victim. This multiplies the traffic by the number of responding hosts. On a broadcast network, there could potentially be hundreds of machines to reply to each ICMP packet. The process of using a network to elicit many responses to a single packet has been labelled as an "amplifier". There are two parties who are hurt by this type of attack: the intermediate broadcast devices (amplifiers) and the spoofed source address target (the victim).

The victim is the target of a large amount of traffic that the amplifiers generate. This attack has the potential to overload an entire network. SYN Flood attack is also known as the Transmission Control Protocol (TCP) SYN attack, and is based on exploiting the standard TCP three–way handshake [7]. The TCP three-way handshake requires a three-packet exchange to be performed before a client can officially use the service. A server, upon receiving an initial SYN (synchronize/start) request from a client, sends back a SYN/ACK (synchronize/acknowledge) packet and waits for the client to send the final ACK (acknowledge). However, it is possible to send a barrage of initial SYN's without sending the corresponding ACK's, essentially leaving the server waiting for the non-existent ACK's.

Considering that the server only has a limited buffer queue for new connections, SYN Flood results in the server being unable to process other incoming connections as the queue gets overloaded. UDP Flood attack is based on UDP echo and character generator services provided by most computers on a network [5]. The attacker uses forged UDP packets to connect the echo service on one machine to the character generator (chargen) service on another machine. The result is that the two services consume all available network bandwidth between the machines as they exchange characters between themselves. A variation of this attack called ICMP Flood, floods a machine with ICMP packets instead of UDP packets[15].

8. Defenses against Attacks

Many observers have stated that there are currently no successful defenses against a fully distributed denial of service attack [6]. This may be true. Nevertheless, there are numerous safety measures that a host or network can perform to make the network and neighboring networks more secure. These measures include:

Filtering Routers: Filtering all packets entering and leaving the network protects the network from attacks conducted from neighboring networks, and prevents the network itself from being an unaware attacker. This measure requires installing ingress and egress packet filters on all routers.

Disabling IP Broadcasts: By disabling IP broadcasts, host computers can no longer be used as amplifiers in ICMP Flood and Smurf attacks. However, to defend against this attack, all neighboring networks need to disable IP broadcasts.

Applying Security Patches: To guard against denial of service attacks, host computers must be updated with the latest security patches and techniques. For example, in the case of the SYN Flood attack, there are three steps that the host computers can take to guard themselves from attacks: increase the size of the connection queue, decrease the time-out waiting for the three-way handshake, and employ vendor software patches to detect and circumvent the problem.

Disabling Unused Services: If UDP echo or chargen services are not required, disabling them will help to defend against the attack. In general, if network services are unneeded or unused, the services should be disabled to prevent tampering and attacks.

Performing Intrusion Detection: By performing intrusion detection, a host computer and network are guarded against being a source for an attack, as while as being a victim of an attack. Network monitoring is a very good pre-emptive way of guarding against denial of service attacks. By monitoring traffic patterns, a network can determine when it is under attack, and can take the required steps to defend itself. By inspecting host systems, a host can also prevent it from hosting an attack on another network.

9. Yahoo! Attack

The attacks on the major Web sites began in early February 2000, with the first major attack being on Yahoo! On February 7 [15]. The surprise attack took the Yahoo! Site down for more than three hours. It was based on the Smurf attack, and most likely, the Tribe Flood Network technique. At the peak of the attack, Yahoo! was receiving more than one gigabit per second of data requests. Yahoo! has stated that it was unprepared for this magnitude of an attack, and hence, was not ready to defend itself. Yahoo! receives huge number of visitors every day, and most likely believed that the bandwidth provided to the users by their Internet service providers would defend them against any type of denial of service attacks. However, they did not account for a large distributed denial of service attack from numerous servers and networks across the Internet.

10. Implementation

8.1 Nmap

Nmap ("Network Mapper") is a free and open source (license) utility for network discovery and security auditing. Many systems and network administrators also find it useful for tasks such as network inventory, managing service upgrade schedules, and monitoring host or service uptime [19]. Nmap uses raw IP packets in novel ways to determine what hosts are available on the network, what services (application name and version) those hosts are offering, what operating systems (and OS versions) they are running, what type of packet filters/firewalls are in use, and dozens of other characteristics. It was designed to rapidly scan large networks, but works fine against single hosts.

8.2 Scapy

Scapy is a powerful interactive packet manipulation program. It is able to forge or decode packets of a wide number of protocols, send them on the wire, capture them, match requests and replies, and much more. It can easily handle most classical tasks like scanning, tracerouting, probing, unit tests, attacks or network discovery [20].

8.3 Wireshark

Wireshark is a network packet analyzer. A network packet analyzer will try to capture network packets and tries to display that packet data as detailed as possible [21]. You could think of a network packet analyzer as a measuring device used to examine what's going on inside a network cable, just like a voltmeter is used by an electrician to examine what's going on inside an electric cable (but at a higher level, of course.

8.4 Iptables

Originally, the most popular firewall/NAT package running on Linux was ipchains, but it had a number of shortcomings. To rectify this, the Netfilter organization decided to create a new product called iptables, giving it such improvements as:

- Better integration with the Linux kernel with the capability of loading iptables-specific kernel modules designed for improved speed and reliability.
- Filtering packets based on a MAC address and the values of the flags in the TCP header. This is helpful in preventing attacks using malformed packets and in restricting access from locally attached servers to other networks in spite of their IP addresses.
- System logging that provides the option of adjusting the level of detail of the reporting.
- Better network address translation.
- Support for transparent integration with such Web proxy programs as Squid.
- A rate limiting feature that helps iptables block some types of denial of service (DoS) attacks.

Packet processing in iptables

All packets inspected by iptables pass through a sequence of built-in tables (queues) for processing. Each of these queues is dedicated to a particular type of packet activity and is controlled by an associated packet transformation/filtering chain.

There are three tables in total.

The first is the mangle table which is responsible for the alteration of quality of service bits in the TCP header. This is hardly used in a home or SOHO environment.

The second table is the filter queue which is responsible for packet filtering. It has three built-in chains in which you can place your firewall policy rules [2]. These are the:

- Forward chain: Filters packets to servers protected by the firewall.
- Input chain: Filters packets destined for the firewall.
- Output chain: Filters packets originating from the firewall.

The third table is the NAT queue which is responsible for network address translation. It has two built-in chains; these are:

- Pre-routing chain: NATs packets when the destination address of the packet needs to be changed.
- Post-routing chain: NATs packets when the source address of the packet needs to be changed

Processing for Packets in iptables

You need to specify the table and the chain for each firewall rule you create. There is an exception: Most rules are related to filtering, so iptables assumes that any chain that's defined without an associated table will be a part of the filter table. The filter table is therefore the default.

To help make this clearer, take a look at the way packets are handled by iptables. In Figure below TCP packet from the Internet arrives at the firewall's interface on Network A to create a data connection.

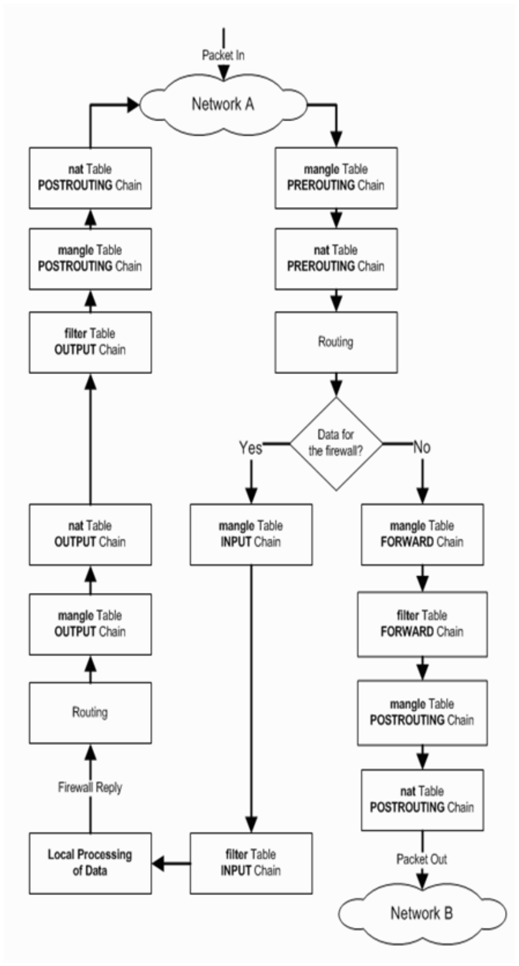

Packet In
Network A
nat Table POSTROUTING Chain
mangle Table PREROUTING Chain
mangle Table POSTROUTING Chain
nat Table PREROUTING Chain
filter Table OUTPUT Chain
Routing
Yes
Data for the firewall?
No
nat Table OUTPUT Chain
mangle Table INPUT Chain
mangle Table FORWARD Chain
mangle Table OUTPUT Chain
filter Table FORWARD Chain
Routing
mangle Table POSTROUTING Chain
Firewall Reply
nat Table POSTROUTING Chain
Local Processing of Data
filter Table INPUT Chain
Packet Out
Network B

Figure 5: Processing Packets in iptables

The packet is first examined by your rules in the mangle table's PREROUTING chain, if any. It is then inspected by the rules in the NAT table's PREROUTING chain to see whether the packet requires DNAT [22]. It is then routed.3

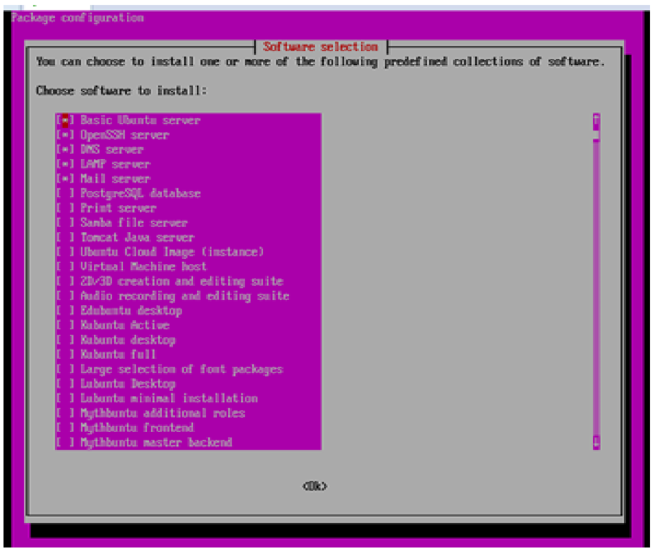
If the packet is destined for a protected network, then it is filtered by the rules in the FORWARD chain of the filter table and, if necessary, the packet undergoes SNAT in the POSTROUTING chain before arriving at Network B. When the destination server decides to reply, the packet undergoes the same sequence of steps. Both the FORWARD and POSTROUTING chains may be configured to implement quality of service (QoS) features in their mangle tables, but this is not usually done in SOHO environments.3

If the packet is destined for the firewall itself, then it passes through the mangle table of the INPUT chain, if configured, before being filtered by the rules in the INPUT chain of the filter table before. If it successfully passes these tests then it is processed by the intended application on the firewall [23].

11. Experimental Procedure

The created virtual environment using vmware workstation, two virtual machines running Ubuntu 14.04 desktop, and the victim machine running Ubuntu server 14.04 with following servers installed.

- Basic Ubuntu server
- OpenSSH server
- LAMP server
- Mail server



```
Package configuration

                    ┤ Software selection ├
 You can choose to install one or more of the following predefined collections of software.

 Choose software to install:

   [*] Basic Ubuntu server
   [*] OpenSSH server
   [*] DNS server
   [*] LAMP server
   [*] Mail server
   [ ] PostgreSQL database
   [ ] Print server
   [ ] Samba file server
   [ ] Tomcat Java server
   [ ] Ubuntu Cloud Image (instance)
   [ ] Virtual Machine host
   [ ] 2D/3D creation and editing suite
   [ ] Audio recording and editing suite
   [ ] Edubuntu desktop
   [ ] Kubuntu Active
   [ ] Kubuntu desktop
   [ ] Kubuntu full
   [ ] Large selection of font packages
   [ ] Lubuntu Desktop
   [ ] Lubuntu minimal installation
   [ ] Mythbuntu additional roles
   [ ] Mythbuntu frontend
   [ ] Mythbuntu master backend

                              <Ok>
```

Figure 6: Ubuntu server 14.04

- We identify the open ports by scanning the victim machine on a network using scanning tools like nmap to find the open ports.
- We used the opened ports identified above and generate a scapy script with spoof IP address to flood the victim machine.
- Run wireshark to detect the suspicious packets in the victim machine.
- We write shell script using iptables to block the attacks on the on the victim's machine, to reject the suspicious packets from unknown sources or ports.

12. Result and Discussions

When the open ports are identified on the victim machine using the nmap scanning tool, we generate the scapyscript to spoof the victim's machine and flood it. iptables shell script is generated on the victim's machine and block the attack.

All the procedure will be on a virtual environment on a vware workstations, and both the victim's machine and attacker/s should also be on a virtual machine, the aims of this paper is to provides the awareness on those type of attack and the countermeasures to prevent or mitigate the attack.

Lot of mechanisms are in use today to prevent those devastating type of attack but is very hard to prevent it completely, the best way is to mitigate the attack.

13. Conclusion and Future Work

In this paper, we have described how to spoof the IP address and flood the victim machine using scapy script and denial the service to the authorized, iptable script is generated to prevent or mitigate the attack, for the future work, the attack scenario is based on a specific port for example port 80 (http port) the victim can still work on some of the available ports for example port 25 (SMTP port) we want block the user completely in which a user cannot access the server at all.

Acknowledgement

Working on a paper like this requires determination, inspiration and encouragement. My deepest appreciation goes to my beloved guide J. Dharani, My warmest gratitude also goes to Mr. Venkatesh thank you a lot.

References

[1] D. DeepthiRani, T. V. Sai Krishna, G. Dayanandam, Dr. T. V. Rao, "TCP Syn Flood Attack Detection And Prevention, International Journal of Computer Trends and Technology (IJCTT), volume 4 Issue, October 2013, ISSN: 2231-2803, http://www. ijctt journal. org

[2] Lau, F. Simon Fraser Univ, Burnaby, BC, Canada; Rubin, S. H. ; Smith, M. H. ; Trajkovic, L. " Distributed Denial of Service Attacks Systems, Man, and Cybernetics", 2000 IEEE International Conference (Volume:3), http:// ieeexplore. ieee. org

[3] Cisco Systems, Inc., "Defining strategies to protect against TCP SYN denial of service attacks," July 1999, http://www. cisco. com/warp/public/707

[4] S. Bellovin, "Distributed denial of service attacks," Feb. 2000, http://www. research. att. com/~smb/talks.

[5] A. Harrison, "The denial-of-service aftermath," Feb. 2000, http://www. cnn. com/2000/TECH/computing/02/14/dos. aftermath. idg/index. html.

[6] K. Rocky, C. Chang, The Hong Kong Polytechnic University, Defending against Flooding-Based Distributed Denial-of-Service Attacks, IEEE Communications Magazine, October 2002

[7] A. Kotkar, A. Nalawade, S. Gawas, A. Patwardhan, "Network Attacks and Their Countermeasures" International Journal of Innovative Research in Computer and Communication Engineering, Vol. 1, Issue 1, March 2013, www. ijircce. com

[8] Jianxi Tao, Li Zhou, Zhou Zhou, Rong Yang, Wei Yang, Qingyun Liu, Defending Against SYN Flood Attack under Asymmetric Routing Environment Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China, International Workshop on Cloud Computing and Information Security (CCIS 2013)

[9] H. N. Wang, D. L. Zhang and K. G. Shin, "Detecting SYN flooding attacks, " in INFOCOM 2002. Twenty-First Annual Joint Conferences of the IEEE Computer and Communications Societies. New York, NY, USA, 2002.

[10] Cheng Jin and Kang G. Shin, "Defense against Spoofed IP Traffic Using Hop-Count Filtering", IEEE/ACM Trans. Networking, vol. 15 No. I, Feb 2007.

[11] Sharmin Rashid, SubhraProsun Paul, World University of Bangladesh, Dhanmondi, Dhaka, Banglades "Proposed Methods of IP Spoofing Detection & Prevention" International Journal of Science and Research (IJSR), India Online ISSN: 2319-7064, Volume 2 Issue 8, August 2013

[12] Alexander Chopra, Man In the Middle (MITM) DNS Spoofing Explained, "Root Services Evolving IT Solutions Around Your Innovative Operations" May 8, 2014, https://rootserv. com

[13] S. Cheung, K. N. Levitt, C. Ko, "Intrusion Detection for Network Infrastructures", The 1995 IEEE Symposium on Security and Privacy, Oakland, CA, May 1995.

[14] Yonghua You, Mohammad Zulkernine, Anwar Haque, " A Distributed Defense Framework for Flooding-Based DDoS Attacks", The Third International Conference on Availability, Reliability and Security, IEEE, 2008.

[15] A. Kak, TCP/IP Vulnerabilities: IP Spoofing and Denial-of-Service Attacks "Computer and Network Security" June, 2014, https://engineering. purdue. edu/kak/compsec/NewLectures/Lecture16. pdf

[16] G. Yang, Introduction to TCP/IP Network Attacks, Department of Computer Science Iowa State University Ames, IA 50011, November 18, 1997, http://seclab. cs. sunysb. edu/sekar/papers/netattacks. pdf

[17] SonaliSwetapadmaSahu, Manjusha Pandey, Distributed Denial of Service Attacks, I. J. Modern Education and Computer Science, January 2014

[18] Sean-Philip Oriyano, "CEH: Certified Ethical Hacker Version 8 Study Guide", ISBN: 978-1-118-64767-7, August 2014, E-Book

[19] nmap Ref Guide: http://nmap. org/

[20] scapy'sdocumentation:http://www. secdev. org/projects/scapy/

[21] wireshark User's Guidehttps://www. wireshark. org/docs/wsug_html_chunked/ChapterIntroduction. html#ChIntroWhatIs

[22] iptables manual guid: http://ipset. netfilter. org/iptables. man. Html, https://help. ubuntu. com/community/IptablesHowTo

Author's Profile

Hamisu I. Usman obtained his First B.Sc Degree in computer science, Department of Math and Computer Science, at Bayero University Kano, Nigeria, he attended his M.Tech (ISCF) Information Security and Cyber Forensics, at SRM University Chennai, India. He's currently researching on TCP/IP Vulnerabilities.