IJCAT - International Journal of Computing and Technology, Volume 5, Issue 9, September 2018
ISSN (Online): 2348-6090
www.IJCAT.org
Impact Factor: 0.835

A Tool to Mitigate Denial of Service Attacks on Wired Networks

[1] George Kyambadde; [2] John Ngubiri

[1] School of Business and Applied Technology, Clarke International University, Kampala, Uganda

[2] College of Computing and Information Sciences, Makerere University, Kampala, Uganda

Abstract - Presently, several institutions share information, resources, and files over computer networks. Network environments are susceptible to various security risks including computer viruses, Trojans, and malicious malware, making networks inefficient due to exhausted bandwidth and computing resources. Ultimately, compromised networks/servers are made unavailable to legitimate users. Such a security problem is called a Denial-of-Service (DoS) attack. It is imperative to mitigate DoS attacks immediately. This study presented a tool based on a packet filtering approach, used to mitigate flooding attacks. This was an experimental study conducted in an environment similar to the production environment of the project case study. The developed prototype consists of the mitigation and tracking modules. To evaluate the responsiveness of the proposed system, its performance was compared with an Uncomplicated Firewall (ufw) (Ubuntu default firewall), we experimented with the firewall and the proposed system independently but in similar environment. Results indicated that the prototype system ably mitigated the DoS flooding attacks (UDP and ICMP flooding attacks) and also responded fairly faster than Ubuntu standard firewall.

Keywords - *UDP flooding attack, ICMP flooding attack, Mitigation, Firewall.*

1. Introduction

Networked systems are popular with computerized organizations. This is because they cut down operational costs, improve efficiency, and increase satisfaction to customers. The access networked systems give to legitimate users can also be got by malicious users. As such, these systems are prone to a number of attacks including but not limited to denial of service (DoS) attacks.

DoS attack is a malicious attempt to disrupt the services provided by networks or servers. DoS attacks have recently become a major security threat to networks [1]. The power of a DoS attack is amplified by incorporating over thousands of zombie machines through botnets [2]. Leveraging botnets and high-speed network technologies, modern DoS attacks exceed the scale of 300 Gbps becoming a major threat on the Internet [3]. Attack targets include businesses and media outlets, service providers (like Domain Name System (DNS)), and Web portals. A sophisticated DoS attack can be mounted by attackers without advanced technical skills using various advanced attacking toolkits freely available on the Internet [4] such as LOIC (low-orbit ion cannon) [5]. DoS attacks like other security attacks are motivated by financial, political, and ideological benefits [6]. Regardless of the motivation, they have similar impacts such as lost revenue, increased expenses, lost customers, and reduced consumer trust. With little or no warning, a DoS attack can easily exhaust the computing and communication resources of its target within a short period.

Many mitigation mechanisms have been proposed to combat DoS attacks. There are two popular mechanisms that mitigate DoS attacks: packet filtering and rate limiting. In packet filtering all incoming packets will be discarded if they match the characteristics of attack traffic [7]. In rate-limiting all relevant incoming packets will be discarded with a certain attack probability [7].

This paper presents a possible way to protect networked systems in case of DoS attacks.
The rest of the paper is organized as follows; in section 2 we present the related work, in section 3 we describe the methodology of the study, in section 4 we describe the design and implementation of the tool. Test results of the implementation are presented in section 5. Finally, in section 6 and 7, we provide a

discussion of the test results and a conclusion respectively.

2. Related work

A research done by AL-Musawi eta al. [8] discusses a packet filtering technique to defend against DoS attacks using IPTABLES. In the technique discussed, every time an incoming packet is filtered against, the system has to decide whether it matches the predefined rules in the IPTABLES for malicious packets. A Similar study also discusses a framework to mitigate DoS attack using the IPTABLES' filtering technique [9].

Hop-Count filtering is a mechanism proposed by C. Jin et al. [10] to counter spoofed IP address DDoS attacks. In this mechanism, information about a source IP address and its corresponding hops from a destination are recorded in a table at the destination side when the destination is not under attack. Once an attack alarm is raised, the victim inspects the incoming packets source IP addresses and their corresponding hops to differentiate the spoofed packets. It is not necessary for routers to collaborate mutually in this mechanism; however, it is difficult to ensure the integrity and accuracy of the source IP addresses and their corresponding hops from the victim. In other words, attackers can spoof IP addresses with the same hop-count as their machines. Moreover, legitimate packets can be identified as spoofed ones if their IPs to hop-count mappings are inaccurate or if the hop-count updates have a delay.
Regardless of the point of deployment, such mechanisms require that, every time the system has to decide whether the incoming packets match a predefined set of rules for malicious traffic, even for similar packets. This is somewhat costly in terms of the processing. The proposed prototype not only supports the importance of IPTABLES in mitigating DoS attacks but has an added tracking module that monitors the state of filtered traffic with respect to subsequent incoming traffic.

3. Methodology

The generic methodology for this study was experimental. This approach allowed for a more realistic environment, which allowed the researcher to set up DoS experiments using real traffic and attack methods; the approach also offered realistic parameters such as delays, packet loss, and user throughput [10]. This experimental set up has been used in recent related research [8] [9] [10]. During the experimentations, traces were got from live execution of the different DoS attacks under the study scope. A DoS traffic test bed was set up for the purpose of conducting this study, it involved the attacking hosts, a server machine having various configurations and the proposed mitigation tool. Mitigation involved the identification of the offending IPs which, in this case were sometimes hidden or spoofed. Once the malicious IPs were identified, the system would then drop the volatile traffic stopping it from reaching the victim. The proposed prototype system was evaluated against the standard Ubuntu firewall.

3.1 Experimentation

The experiments were done at Clarke International University, an academic institution which implements computerized systems like human resources, student information management, finance and backup systems. These systems serve to facilitate scheduling, record attendance for staff and issuing computerized reporting.

All the experiments were performed in a computer laboratory environment that had no access to the institution's productive network or the Internet to prevent accidental damage to the remaining network infrastructure and its systems. The attacks targeted against a realistic staging environment that was as similar as possible to production system.

3.1.1 Simulation Setup

The setup consisted of three computers and a D-Link wireless router as shown in *Fig 1*. All the computers had Ubuntu 12.04 and default services running on them. The D-Link wireless router was an 802.11 a/g router and had a WAN port along with 4 LAN ports. Each of the machines connected to the LAN port of the router using a straight cable. With the inbuilt DHCP server, the connected machines were dynamically assigned respective IP addresses as shown in *Fig. 1*. Each of the machines played the role of a target, attacker, or client, as shown in *Fig 1*. Successful attacks were important in the simulation and hence, all firewalls were disabled on all systems to allow for the experimentation of the implemented tool. Each of the simulation required a separate attack script to be executed in the attacker's machine. For all the attack simulations, we measured bandwidth

utilization using Speedometer [11] installed on the target machine.

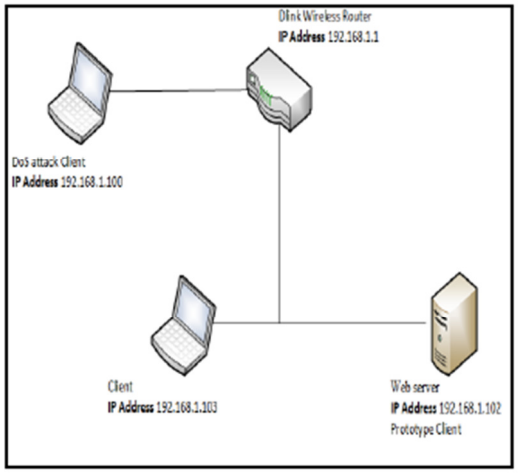

Fig 1 Topology of the attack simulation setup.

3.1.2 Simulation of Selected Attacks

The simulation process used the UDP and ICMP flooding attacks.

UDP Flooding attack

With the hping3 tool [12], a streams of UDP packets sent to several ports of the victim machine had the victim busy sending ICMP messages to the attacker's machine. This blocks legitimate requests from getting into the system. The hping3 tool initiates UDP packets in the DoS client, and speedometer started in the web server after regular intervals during the attack. *Fig 2* shows the initial setup for the attack simulations and *Fig 3* shows the communication state of the systems after the attack is launched.

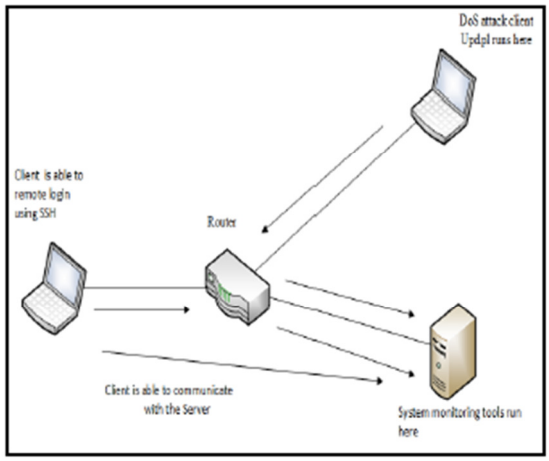

Fig 2 Communication among various systems before the attack

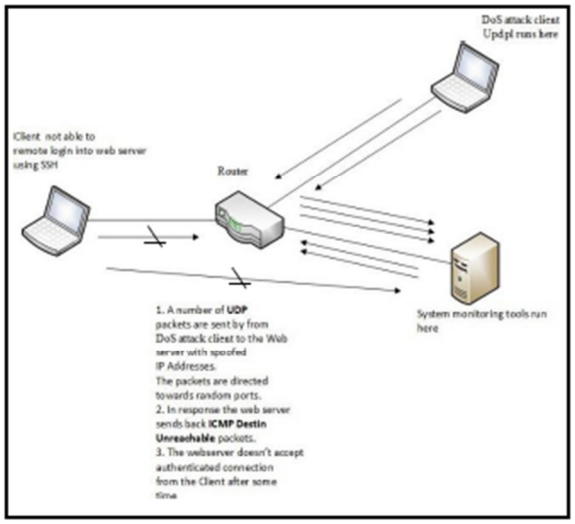

Fig 3 Systems unable to communicate after the UDP flood attack is launched.

ICMP Flooding attack

In this simulation, using the hping3 tool [12] the DoS attack client sent out lots of ICMP echo request packets to the webserver. And since the source IP address was spoofed, ICMP echo request and ICMP destination unreachable packets flooded the network. Thus, after some time, the client was not able to communicate with the web server (target). *Fig 4* shows the communication state of the systems after the attack is launched.

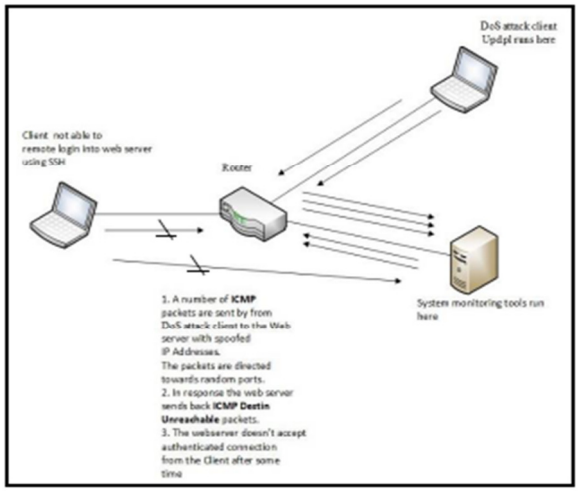

Fig 4 Systems unable to communicate after ICMP flood attack is launched

4. Design and implementation

The prototype system consists of two modules namely; tracking module and the mitigation module.

The former consists of a daemon that tracks of the state and the latter consists of the filtering rules that filters IP addresses. *Fig 5* shows the architecture of the proposed DoS mitigation system.

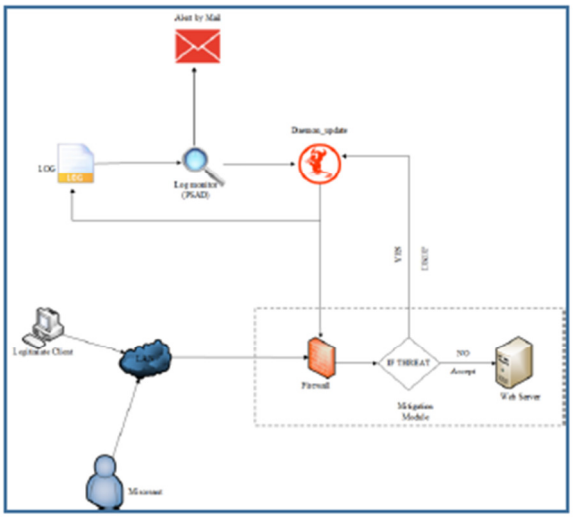

Fig 5 Architecture of the proposed DoS mitigation system

From *Fig 5,* all the traffic directed to the target machine is logged based on the packet filter rules for specific DoS specific attacks. The log monitor then analyses the log to identify malicious IPs sending repetitive requests to the target client. Based on such information, the system decides whether a DoS attack is in progress or not.

The system takes response action to mitigate the attack, by dropping the suspicious/ forged packets from malicious IPs. The dropped packets are then listed in the auto-blocked iptables text file. To forward blocked IPs, a small program, the *update_daemon* was implemented to get the listed IPs in the auto blocked iptables text file and insert them in an *ipset*.

Tracking module

This a small program, the *update_daemon* implemented to get the listed IPs in the auto blocked iptables text file and insert them in an *ipset.* This introduces state in the system in that subsequent filtering decisions are made taking into account of an earlier decision made on similar incoming packets based on the IP addresses listed in the *ipset*. *Fig 6* shows the activity flow of the prevention mechanism.

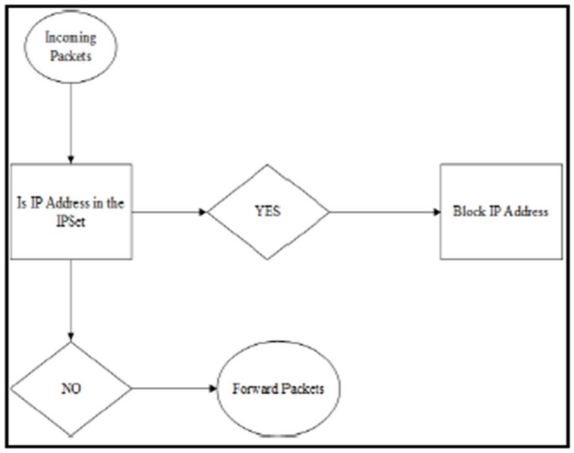

Fig 6 Activity Flow of the prevention mechanismFig

Mitigation module

In order to weed out malicious clients who send repetitive requests to the Web Server, IP address filtering technique was used. Clients' IP addresses that send big repetitive streams of packets to the web server, are blocked and added to the ipset. The *ipset* is updated by the *update_daemon* to influence subsequent the filtering decisions for similar incoming packets.

5. Results

To achieve a steady state of the experiment, attacks lasted 50 seconds, starting at 10 seconds and ending at 60 seconds from the beginning of the run. The metric was bandwidth. *Fig 7* shows the responsiveness of both the Firewall and the prototype System during UDP flooding attack. *Fig 8* shows the responsiveness of both the Firewall and the prototype System during the ICMP flooding attack.

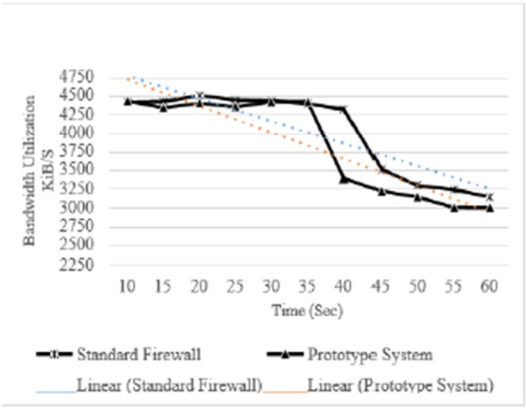

Fig 7 Response time for both Systems during the UDP flooding attack simulation

During the high peaks of bandwidth utilization, from *Fig 7*, we see that for both systems, bandwidth utilization decreased with time. This is due to responsiveness of both systems to the attacks. During a UDP flood, as the attacker sends in large amounts of UDP packets with spoofed IP addresses, the victim machine is forced to send excessive ICMP return packets which do not reach the attacker (due to the spoofed identity), thus utilizing large amounts of bandwidth.

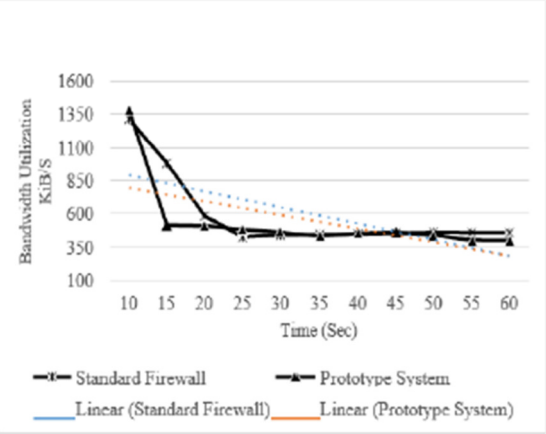

Fig 8 Response time for both Systems during the ICMP flooding attack simulation

Similarly in *Fig 8,* during the ICMP flooding attack, we see that for both systems, bandwidth utilization decreased with time.

6. Discussion

The results indicated that the prototype system responded fairly faster to the flooding attacks by a record five seconds compared to the standard firewall. A difference of five seconds in response time by the prototype has the potential to halt system damage.

The prototype's fast performance is attributed to its tracking module, which allows for filtering decision making during packet blacklisting based on an earlier decision made for similar incoming packets. Whereas the standard firewall has to take a little bit of more CPU time since it makes the filtering decision for every incoming packet independent of the earlier decision made for a similar packet.

7. Conclusion and Future Work

In this study, we designed and implemented a prototype system capable of automatically executing appropriate mitigation responses. The prototype has an *update_deamon* module that allows for decision making based an earlier action taken on related packets. This module allows for a fairly faster responsiveness compared to the standard Ubuntu firewall as shown in the above results' section; ultimately, the prototype presents a more effective solution to the problem of DoS flooding attacks specifically on wired networks.

The prototype system is capable mitigating flooding attacks implemented in other protocols (such as TCP SYN flood attack). However, it is not (yet) a comprehensive package to handle all forms of DoS attacks such as application layer attacks. This sets the direction of further investigations and research.

Acknowledgements
We thank Florence Nakaggwa for the invaluable discussions and suggestions.

References

[1] Dobbins R., Morales C., Anstee D., Arruda J., Bienkowski T., Hollyman M., Labovitz C., Nazario J., Seo E., and Shah R. "Worldwide Infrastructure Security Report. Tech. rep., Arbor Networks," 2010.

[2] David D., Guofei G., Christopher P., Wenke L. "A Taxonomy of Botnet Structures," *Proc. of Annual Computer Security Applications Conference (ACSAC),* pp. 325-339, December 2007.

[3] [Online]. Available: www.arbornetworks.com.

[4] Kargl F.,Maier."Protecting web servers from distributed denial of service attacks," 2001.

[5] Roman J., Radek B., Radek V., and Libor s. "Launching distributed denial of service attacks by network protocol exploitation," in *In Proceedings of the 2nd international conference on Applied informatics and computing theory. AICT'11. World Scientific and Engineering Academy and Society (WSEAS*, Stevens Point, Wisconsin, USA, 2011.

[6] Ghorbani A., Lu W. and Tavallaee . "Network Intrusion Detection and Prevention: Concepts and Techniques, Springer", 2010.

[7] Y. Xia, "Selective Dropping of Rate Limiting Again Denial of Service Attacks," University of Dayton, 2016.

[8] AL-Musawi, Bahaa Qasim M., "MITIGATING DoS/DDoS ATTACKS USING IPTABLES," *International Journal of Engineering & Technology,*

pp. 101 - 111, June 2012.

[9] Chatterjee, Koushik. "Design and Development of a framework to mitigate Dos/DDos attacks using IPTable firewall," *International Journal of Compouter Science and Telecommunication,* pp. 67 - 72, 2013.

[10] H. Wang, C. Jin, and K. G. Shin. "Defense Against Spoofed IP Traffic Using Hop-Count Filtering," *IEEE/ACM Trans. On Networking,* vol. 15, pp. 40-53, February 2007.

[11] I. Ward, "Speedometer 2.8 excess.org," 02 April 2015. [Online]. Available: http://excess.org/speedometer/.

[12] G. o. Security, "GBHackers on Security," [Online]. Available: https://gbhackers.com/hping3-network-scanner-packer-generator/. [Accessed 20 January 2016].

Authors

George Kyambadde received Bsc in Computer Science of Makerere University in 2012. He received Msc in Computer Science of Makerere University in 2017. He is currently working as a Lecturer at the School of Business and Applied Technology, Clarke International University. His current research interests include; security, and Machine Learning.

John Ngubiri a Senior Lecturer at the Department of Computer Science, College of Computing and Information Science, Makerere University. He holds a PhD in Computer Science of Radboud University Nijmegen. His research interests are in performance evaluation, System optimization, security and parallel and distributed systems.