Stateful Firewall

EECE 655 Final Project Presented to Dr. Imad El-hajj

Hoda Sidani
Literature Review
Simulation and Experimental Results
Beirut, Lebanon
hms62@mail.aub.edu

Samia Noaman
Abstract
Introduction
Beirut, Lebanon
sbn06@mail.aub.edu

Zeina Nasser
Theory/Design/Methodology
Conclusion
Experimental Results
Beirut, Lebanon
zan08@mail.aub.edu

***Abstract*—In today's rapidly evolving cybersecurity landscape, network security professionals face a relatively huge burden: managing the complexity of a firewall in addition to the fact of it being stateless with its increasing costs. [4] Where a statless firewall doesn't stroe information about the state or context on connections, which limits its application-level inspection. [2] Whereas, as stateful firewall holds more improved security, simplified rule configuration and an enhanced performance. When it comes for costs, Businesses that have anywhere from 15 to 100 users can expect to pay between $1,500 and$4,000 for each firewall based on its needs [5]. That's why we decided to design a stateful firewall, that is user-friendly and customizable. Our design will include a bunch of preset rules that can be activated very easily by the user based on the need, so the same design can be used on different systems because of the adjustable option so no need to pay for more than design increasing the benefit to cost ratio. Moreover, our firewall will be collecting data regarding every connection made through it, building profiles concerning every connection or host increasing the security of the system.**

Our design uses Netfilters, Python and C and Linux kernel module which we will be implementing through the stateful feature of the firewall using Netfilter. In addition to state tables that tracks active TCP and UDP connections with parameters like IP addresses, ports, protocol type, timestamp, and direction (incoming/outgoing traffic). Implementation, simulation, and comprehensive testing will be done within the project. It will implement packet filtering rules based on source and destination IP addresses, ports, and protocol types. To ensure working and performance, attack scenarios will also be tested in simulation. With flexibility and user accessibility, combined with strong security, this firewall will be an inexpensive platform for a number of systems and scenarios. .

***Index Terms*—Netfilter, NFQueue**

I. Introduction

Network security experts have a relatively heavy load in today's quickly changing cybersecurity field which is managing a firewall's complexity in addition to its statelessness and rising costs. A stateless firewall's application-level inspection is limited since it does not save information about the status or context of connections. On the other hand, stateful firewalls offer better security, easier rule configuration, and increased speed. Depending on their requirements, businesses should anticipate paying high costs for such security systems. For this reason, we chose to create a stateful firewall that is easy to use and adaptable.

Our proposed solution creates a User-Friendly Stateful Firewall that maintains a balance between simplicity, security, and adaptability. Through an easy-to-use terminal-based interface, users will be able to monitor traffic, define custom rules, and activate them thanks to the firewall's Python and Netfilter construction. In contrast to conventional firewalls, which might necessitate a high level of technical expertise, this tool will serve a wider range of users and offer an economical and effective solution.

The firewall applies filtering rules based on packet headers, including source/destination IP addresses, ports, and protocols, and uses state tables to monitor the status of connections. The firewall will process network-layer traffic using Python's socket module, allowing for real-time monitoring and decision-making. The implementation of logging features will improve the entire security architecture by documenting questionable activity and offering insights into network behavior. The project's main goals are stateful connection tracking by keeping thorough records of all active connections, use state tables. A customizable rule configuration providing a terminal interface for users to activate or deactivate rules. Traffic filtering using rules to permit or prohibit packets according to preset standards. Logging and analysis by keeping an eye on traffic trends, identify potential dangers, and improve usability, use logging. And finally, testing for validation to guarantee performance and dependability, we will simulate different traffic situations and attack patterns.

This project is inspired by previous research and practical guides, such as the use of Python and Netfilter to implement firewalls. Previous studies, like the ones conducted by Robin Marshall and other researchers, have shown that it is possible to develop a personalized firewall starting from the beginning. Our goal is to create a tool that merges practicality with ease of use by leveraging these core elements.[9]

II. Literature Review

A. Importance of Stateful Firewalls

Stateful firewalls have the potential to offer advanced traffic filtering techniques by maintaining connection states. Unlike stateless firewalls, they can track active connections to distinguish legitimate returning traffic from illegitimate ones. With the integration of tools like Netfilter, NFQueue, and Python libraries, we can implement and manage firewall rules. In addition, stateful firewalls handle the limitations of static rule-based systems by providing enhanced security when maintaining state tables to track packet flows. Article [12] shows that state tables store connection details such as IP addresses, ports, and protocol states to reduce administrative complexity and improve security. Using frameworks like Netfilter further refines the implementation of stateful firewalls. The modular architecture of Netfilter which integrates stateful filtering into the Linux Kernel, enables dynamic inspection of packets at different stages. [13]

B. Optimization Techniques and Algorithms

Article [15] introduced an algorithm based on IP flow matching to enhance Netfilter's performance in high-throughput environments. By using temporal locality, the proposed method reduced the hash table traversals, thus improving the efficiency of packet processing.

Whereas, the author of article [11] integrated machine learning into Netfilter-based systems for network anomaly detection. By utilizing statistical models and dynamic rule generation, his model adapts to emerging threats which is considered a great step towards intelligent firewalls. [11]

Similarly, a Python-based implementation was explored in [1], however, it focused on conceptual clarity and educational purposes rather than performance to simplify the concepts of stateful firewalls to the readers. They used the NFQueue library to process packets in user space and implemented a custom TCP state machine to track connections based on RFC 793, the official documentation of TCP. [1] In addition, they designed features including pot knocking for secure access and a rule structure similar to IPTables.

In article [7], the authors demonstrated the adaptability of stateful firewalls across different environments. They compared stateful and stateless firewalls using the POX SDN controller to demonstrate how stateful firewalls block malicious packets and detect anomalies.

Another approach the authors of [10] followed was the NFShunt, which is a hybrid firewall that combines Netfilter with OpenFlow-enabled hardware. Their approach balanced between security and performance which allows for forwarding packets at a higher speed while maintaining a state table, while this article has valuable information, it will not be used in our implementation due to its hardware complexity.

Article [8] highlighted the importance of stateful firewalls to monitor and manage sessions dynamically using a state table. In their tables, they included source and destination IPs, ports, and protocols. Additionally, they incorporated Application-Specific Packet Filtering (ASPF) to handle protocols including FTP and SIP.

C. Challenges and Research Gaps

Even though there has been plenty of research about stateful firewalls, challenges regarding scalability, management of the state, and encrypted traffic handling are persistent challenges. For instance, it is difficult to maintain connection states across distributed systems in software-defined networking (SDN) environments. [14] Improper state management can create vulnerabilities where attackers can exploit terminated connections or inject malicious packets. Furthermore, article [6] addressed the absence of a unified framework for analyzing and optimizing stateful firewalls. Since, without such frameworks and with the increasing complexity of networks, designing and verifying the firewall's behavior is inconsistent and prone to errors.

Additionally, we will have a trade-off between security and performance where keeping a stateful connection might introduce latency and packet loss. [10]Finally, encrypted traffic, as mentioned in [11] and [3], limits the firewall's ability to detect and mitigate attacks which will require alternative techniques for secure traffic analysis

D. Relevance to our Research Implementation

Our proposed research on implementing a stateful firewall using Netfilter and NFQueue aligns with the principles and methodologies discussed. By using Netfilter's kernel-level packet filtering and NFQueue's user-space packet processing, we can have an efficient state table management and rule application. Articles [13] and [1] provided us with valuable details about integrating Python scripts for dynamic rule management. Moreover, article [15] provided us with potential improvements for our connection tracking logic. All aforementioned limitations indicate the need for continued research about adaptive, scalable, and efficient stateful firewall designs.

III. Theory/Design/Methodology

In our project, after dowloading all the necessary libraries for netfilter, netfilterque, and the setting up requirements for the VM's Kali and Ubuntu, we developed a stateful firewall on Ubuntu virtual machine to be tested by Kali virtual machine that represents the client. First, we compiled the C code by the "make" command to employ the Netfilter which consisted of a hook function to filter and intercept the packets and extract IP headers. Following this we loaded the firewall.ko module using the command "sudo insmod firewall.ko" and then we ran the firewall of the python "scriptsudo python3 firewall3.py" to manage the rules and show logging actions while we used nfqueue library to connect C to python to capture queued packets. We tested the firewall under different scenarios, the first one was ICMP pinging "`ping <firewall_Ip_machine>`" . In the second time, we tested the http traffic by running "sudo python3 -m http.server 80" on Ubuntu and `curl http://<firewall_Ip_machine>/` on Kali. In addition to this, we tested the firewall for block or allow traffic from certain IP: `Try ping ip ftp`

IV. Simulation and Experimental Results and Analysis

Upon trying to ping Ubuntu virtual machine from Kali virtual machine we noticed that the pinging failed when the ICMP traffic was blocked where the firewall logs showed the blocked ICMP packets along with source and destination IP addresses and timestamps. On the other hand, when the ICMP traffic was allowed the ICMP packets were accepted and the pinging was achieved. Similarly, blocking the HTTP traffic on the firewall on Ubuntu prohibited Kali from accessing the HTTP server. Once again the logs showed that the HTTP traffic was blocked while showing the source and destination IP and the port. To further verify this, we then allowed the HTTP traffic and we noticed from the firewall logs that the server can be accessed. Furthermore, we tested the firewall on blocking a specific IP which is Kali's IP in our case, the logs showed the blocked connections and pinging. After permitting Kali's IP the connection and the traffic were successful. We finally tested the function of flushing or cleaning up all the rules to return the system as it was. The following link contains a demo about our firewall : Project demo

V. Conclusion

It is an undeniable fact that implementing stateful firewalls is crucial to guarantee enhanced level of security against attacks. In our project, we aimed to design a user interface that enables users to easily choose the firewall rules they want to activate by simply pressing the numbers matching to the rules. This firewall extracts the IP headers of the packets and filters based on those headers, where it takes into account the source and destination IP addresses, ports, and protocols. Our firewall uses state tables to keep track of the connections, it intercepts the packet and inspects it at the level of the netfilter which is linked to the python's code through NetfilterQueue. In python the packets are treated based to the chosen rules which can either allow or block connections and logging information about the packets will be shown in order to monitor any suspicious behavior. All the codes we used to implement this project are found on github on this link:Project Codes

References

[1] Stephen Brennan. "PyWall: A Python Firewall". In: *Unknown Journal* (Unknown).

[2] ConnectWise. *Stateful vs Stateless Firewall*. Accessed: 2024-11-18. n.d. URL: https://www.connectwise.com/blog/cybersecurity/stateful-vs-stateless-firewall.

[3] Francois De Keersmaeker, Ramin Sadre, and Cristel Pelsser. "Supervising Smart Home Device Interactions: A Profile-Based Firewall Approach". English. In: IFIP, 2024, pp. 413–422.

[4] Elisity. *The Hidden Costs of Firewall Complexity: Addressing Admin Burnout and Security Risks*. Accessed: 2024-11-18. n.d. URL: https://www.elisity.com/blog/the-hidden-costs-of-firewall-complexity-addressing-admin-burnout-and-security-risks.

[5] Fortinet. *Network Firewall Pricing*. Accessed: 2024-11-18. n.d. URL: https://www.fortinet.com/products/network-firewall-pricing.

[6] M. G. Gouda and A. X. Liu. "A model of stateful firewalls and its properties". English. In: IEEE, 2005, pp. 128–137. ISBN: 1530-0889.

[7] Vipin Gupta, Sukhveer Kaur, and Karamjeet Kaur. "Implementation of stateful firewall using POX controller". English. In: Bharati Vidyapeeth, New Delhi as the Organizer of INDIACom - 2016, 2016, pp. 1093–1096.

[8] Renshan Liu. "Firewall Technology Strategy Analysis and Application Research". English. In: IEEE, 2023, pp. 1907–1911.

[9] R. Marshall. "Norton's latest firewall more user-friendly". In: *The Press* (Oct. 2001). Retrieved from https://www.proquest.com/newspapers/nortons-latest-firewall-more-user-friendly/docview/314447885/se-2.

[10] Simeon Miteff and Scott Hazelhurst. "NFShunt: A Linux firewall with OpenFlow-enabled hardware bypass". English. In: IEEE, 2015, pp. 100–106.

[11] S. Staroletov. "Software Architecture for an Intelligent Firewall Based on Linux Netfilter". In: *2022 25th Conference on Innovation in Clouds, Internet and Networks (ICIN)*. Paris, France: IEEE, 2022, pp. 160–162. DOI: 10.1109/ICIN53892.2022.9758121.

[12] Z. Trabelsi and S. Zeidan. "Enhanced Session Table Architecture for Stateful Firewalls". English. In: IEEE, 2018, pp. 1–7.

[13] Qing-Xiu Wu. "The Research and Application of Firewall based on Netfilter". English. In: *Physics procedia* 25 (2012), pp. 1231–1235.

[14] Salaheddine Zerkane et al. "A Proactive Stateful Firewall for Software Defined Networking". English. In: ed. by Jean-Louis Lanet et al. Vol. 10158. Risks and Security of Internet and Systems. Switzerland: Springer International Publishing AG, 2017, pp. 123–138. ISBN: 0302-9743.

[15] Ke Zhang, Juan Wang, and Dasen Ren. "A Matching Algorithm of Netfilter Connection Tracking Based on IP Flow". In: *2008 2nd International Conference on Anti-counterfeiting, Security and Identification*. Guiyang, China: IEEE, 2008, pp. 199–203. DOI: 10.1109/IWASID.2008.4688360.