Packet Filtering Firewall

A.I. Câmpean, M. D. Deaconu, D. A. Nazemian-Poor

Contents

[1. Introduction 3](#_Toc156305738)

[2. State of Art 3](#_Toc156305739)

[3. Packet Filtering Firewall 4](#_Toc156305740)

[4. User Interface 7](#_Toc156305741)

[6. Attack 9](#_Toc156305742)

[7. Conclusion 11](#_Toc156305743)

[8. Bibliography 12](#_Toc156305744)

# Introduction

Cybersecurity has become important in this age of information, as more and more people live and work in the online medium, sharing and keeping their private information and documents on their devices or in the cloud. This represents an opportunity for people to become subjected to cybercrimes, and thus the need for cyber-defensive tools has increased in the past 20 years.

A firewall is a defensive tool that scans and controls incoming and outgoing traffic in the network, allowing only trusted information to bypass it. This can prove to be essential, as it can protect the user from most common attacks. If the data sent does not meet the requirements of the firewall, they will be denied access to the network, potentially protecting personal information and/or the health of the targeted device.

For the creation of the firewall, we will be using Kali Linux, a Linux distribution, as the operating system. It provides a wide variety of customization, and it is the most suitable for the application. It comes pre-installed with Metasploit, a software that allows attacks to be addressed. The Graphical User Interface (GUI) will be made using Whiptail, a program that enables shell scripts to present dialog boxes to users, serving informational purposes or facilitating user-friendly input. Debian comes pre-installed with Whiptail as part of its default configuration. For establishing the rules, we will use Iptables, an utility that allows the user to configure IP packet filtering rules. Lastly, we will use Python programming language to create the attacks which will help test our firewall.

# State of Art

The landscape of firewall technology is continually evolving to address the escalating challenges posed by sophisticated cyber threats. This chapter provides an in-depth exploration of the current state of the art in firewall technology, highlighting key advancements, emerging trends, and pivotal developments that shape the contemporary cybersecurity paradigm.

A next-generation firewall (NGFW) is a new generation of firewalls that integrate intrusion prevention, malware filtering, and other security functions to allow more advanced control over the data flow. The defining characteristic of a next-generation firewall is the ability to identify and control traffic at the application layer. These new firewalls look deep into the packet’s payload before planning on whether to allow or deny the traffic flow. While traditional have tended to focus on network ports and protocols, NGFWs focus more heavily on the applications and data. Application Intelligence, or awareness, is a foundational component of a NGFW that enables the identification of individual applications within network traffic, ideally irrespective of port, protocol, or evasive tactic.

Cloud firewalls are software-based, cloud deployed network devices, built to stop, or mitigate unwanted access to private networks. As a new technology, they are designed for modern business needs, and sit within online application environments. Enterprises have shifted away from running applications from on-premises servers – instead opting to use virtual machines and containers. This has led to rapid growth in endpoints, all of which need to be protected. This constant flux of endpoint exposure has necessitated a shift away from traditional network protection solutions.

Application proxy firewalls work at the application layer of the TCP/IP protocol stack, providing proxy service for specific applications. Each application proxy sits between the internal network and the world outside. There is no direct communication between the internal computer and the other end of the conversation, as is the case with packet filtering and stateful firewalls. Instead, packets travel between the external system and the proxy. The proxy examines the packets and determines which packets should be passed on to the application.

# Packet Filtering Firewall

A packet filtering firewall is a type of network security device that operates at the network layer (Layer 3) of the OSI (Open Systems Interconnection) model. Its primary function is to examine packets of data as they pass through the network and make decisions about whether to allow or to block them based on a set of predefined rules.

Packet filtering firewalls operate on a rule-based model. Network administrators define rules in access control lists (ACLs) that dictate the conditions under which a packet should be allowed or denied. These rules are crafted in accordance with the organization’s security policies, providing a customizable framework for controlling traffic. By scrutinizing packet headers, these firewalls effectively filter out undesirable or malicious packets, preventing unauthorized access and mitigating potential security threats.

One of the notable aspects of packet filtering firewalls is their simplicity and speed. Due to their straightforward rule-based approach, these firewalls can swiftly process large volumes of data without introducing significant latency into network operations. This efficiency makes them well-suited for basic filtering tasks, particularly at the network perimeter where they regulate incoming and outgoing traffic.

IP address spoofing involves the manipulation of packet headers to conceal or forge the source IP address of communication. While this can serve legitimate purposes, such as network testing and simulation, it is often associated with malicious intent, including attempts to deceive, hide, or launch cyber-attacks.

Attackers can use tools to craft packets with a forged source IP address, allowing them to send data packets that appear to originate from a different source than their actual origin. IP address spoofing is frequently employed in man-in-the-middle attacks, where an attacker intercepts and potentially alters communication between two parties. Spoofing the source address helps the attacker hide their identity. In DoS attacks, attackers may use IP address spoofing to flood a target system with a high volume of traffic that seems to come from various sources, making it challenging to block the attack by simply blocking specific IP addresses.

Internet Service Providers (ISPs) and organizations can implement ingress filtering to block incoming traffic with spoofed source addresses, preventing such packets from entering their networks. Advanced security measures involve the use of deep packet inspection to analyze packet contents for anomalous or malicious source routing information. Implementing secure communication protocols, such as HTTPS, can mitigate some forms of man-in-the-middle attacks. Additionally, strong authentication measures can enhance network security.

A Denial-of-service (DoS) attack is characterized by an explicit attempt to prevent the legitimate use of a service. These attacks overwhelm the processing or link capacity of the target sites by saturating them with bogus packets. Such attacks can seriously disrupt legitimate communications. These attacks can disrupt the availability of Internet services completely, by eating either computational or communication resources through sheer volume of packets sent from distributed locations in a coordinated manner or graceful degradation of network performance by sending attack traffic at low rate.

Preventive measures against Denial-of-Service attacks encompass pattern detection, anomaly detection, hybrid detection, and third-party detection. Mechanisms employing pattern attack detection store recognized attack signatures in a database. Each communication undergoes monitoring and comparison with database entries to identify instances of DDoS attacks. Periodically, the database is updated with new attack signatures. The primary drawback of this detection method is its limitation to known attacks, proving ineffective against novel attacks or slight variations of old attacks unmatched to the stored signature. Conversely, it excels at reliably detecting known attacks without encountering false positives.

IP fragmentation is an Internet Protocol (IP) process that breaks packets into smaller pieces, or fragments, so that the resulting pieces can pass through a link with a smaller maximum transmission unit (MTU) than the original packet size. The fragments are reassembled by the receiving host. When the fragments arrive at the destination, they are reassembled into the original packets.

Attackers deliberately craft IP fragments with sizes that are significantly smaller than the typical or expected fragment size. These fragments may be small enough to evade detection by security devices. Overlapping fragments in a way that confuses the reassembly process is another technique. By strategically overlapping fragments, attackers aim to introduce ambiguity and complicate the reassembly of the complete packet. The fragment offset field in the IP header indicates the position of the fragment within the original packet. Attackers may manipulate this field to create fragments that appear out of sequence, potentially causing confusion during reassembly.

Organizations deploy and maintain intrusion detection and prevention systems equipped to handle fragmented packets effectively. These systems utilize advanced algorithms for packet reassembly, ensuring the identification and mitigation of fragmentation attacks. Implementation of deep packet inspection is very important for analyzing both packet headers and payload content, including packets broken up into smaller pieces. Regular validation of the network protocols ensures that the devices adhere to established standards for packet fragmentation. This minimizes the risk of attackers exploiting vulnerabilities related to how specific devices handle these sorts of attacks.

# User Interface

A screenshot of a computer

Description automatically generatedThe main menu features 4 buttons: a “Start Firewall” button, which turns on and off the firewall via “ufw”, a “Settings” button, in which the user can modify and enable certain rules, a “Statistics” button, which shows the statistics of the network using “netstat”, and a “Show Rules” button to show the user the rules configured.

A screen shot of a computer

Description automatically generatedIn the “Start Firewall” button, the current selection is marked. If the user tries to select it again, it will show an error. This is to ensure that the user already acknowledges the fact that the firewall is already on and prevent any errors.

The “Statistics” menu shows the current processes using the network, along with their local address, foreign address, and the state in which they are in, as well as all active connections.

A screenshot of a computer

Description automatically generated

The rules menu shows all the implemented rules in Iptables. It respects the order in each chain, new rules being added to the bottom of the chain.

A computer screen shot of a program

Description automatically generated

1. App

A screenshot of a computer

Description automatically generatedAll the configurations are contained within the settings menu. The user can choose to either add, disable, or insert a rule, along with the main functions of the firewall.

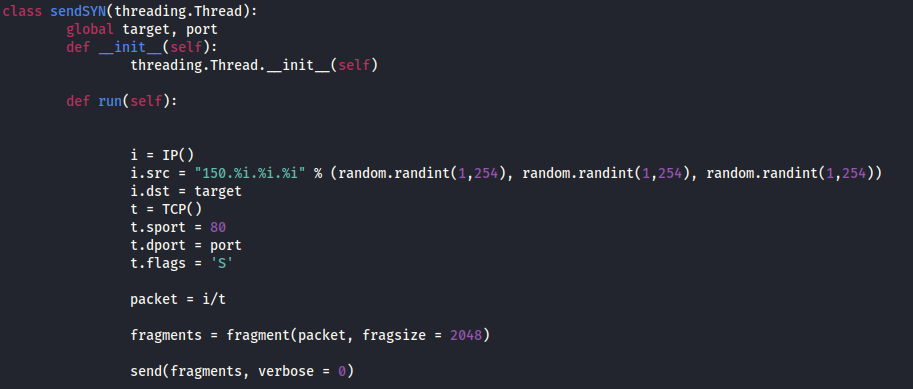
“Drop Strange Packets” will scan for packets that are sent by someone who has an IP address within the specified range. “Track and Drop SYN Packets” will refuse to accept any SYN packets sent. “Limit Number of Fragments per Packet” will try to limit the number of fragments a packet will contain to the specified amount in the GUI. All these measures should be enough to intercept and defend the machine in case any of the aforementioned attacks should happen.

# Attack

A screen shot of a computer program

Description automatically generatedWe used Python 3, together with the Scapy module, as our main programming language for the attack. Using Wireshark, a program designed to scan the network activity, we can see the incoming packets and confirm that the attack is working.

The code features a class, sendSYN, which will send SYN packets to the victim. The source IP is set as a random IP within the 150.0.0.0/8 range each time, introducing the IP Spoofing concept. Furthermore, each packet is fragmented into other fragments. At the end, the program will send all those packets.



A screenshot of a computer

Description automatically generatedIn the console, the attacker can specify the destination IP address and the port they are trying to attack. Using threading, the program will stop after a number of threads have been resolved.

Using Wireshark, we can see that we received the payload, meaning that the attack worked, but the firewall did not act as intended.

# Conclusion

While the firewall did not work as intended, the present work represents our efforts to understand how Iptables and protecting an end-device works, as well as training in ethical hacking. We will continue our efforts in order to bring this application to a working state.

# Bibliography

[1] Brian Komar, Ronald Beekelaar, and Joern Wettern. *Firewalls For Dummies, 2nd Edition*. Published in New York. Wiley Publishing, Incorporated.

[2] Barracuda Networks Incorporated. *Cloud Firewalls*.

[3] David Freet and Rajeev Agrawal. *Network Security and Next-Generation Firewalls*. ResearchGate, 2016.

[4] Ole J. Jacobsen, *A Quarterly Technical Publication for Internet and Intranet Professionals*. The Internet Protocol Journal Volume 10.

[5] Antonio Atlasis. *Attacking IPv6 Implementation Using Fragmentation*.

[6] Xuewei Feng , Qi Li, Kun Sun , Ke Xu, Baojun Liu , Xiaofeng Zheng, Qiushi Yang, Haixin Duan, Zhiyun Qian. *PMTUD is not Panacea: Revisiting IP Fragmentation Attacks against TCP*.