Article

Implementation of Wireshark and IP tables Firewall Collaboration to Improve Traffic Security on Network Systems

Abdul Wahid[1], **Muhammad Eka Firdaus**[2*], **Jumadi Mabe Parenreng**[3]

[1]Department of Informatics and Computer Engineering, Faculty of Engineering, State University of Makassar,
[2]Department of Informatics and Computer Engineering, Faculty of Engineering, State University of Makassar,
[3]Department of Informatics and Computer Engineering, Faculty of Engineering, State University of Makassar

* Corresponding author: ekafirdaus0403@gmail.com

Abstract: Along with the development of the internet era which is very fast today, the network security system becomes a very urgent matter and needs attention. The number of criminal activities and cyber attacks that attack servers through the network makes a server administrator need to make extra efforts in maintaining and monitoring data traffic that enters or leaves the server system. One of the efforts often made by server admins is to monitor server activity and then immediately secure the server from attacks that they identify from the monitoring results. data packets. Here an algorithm is built where the output of the Wireshark application is an analysis result that will distinguish the presence of a malicious accessing IP and then notify the server admin to set the firewall and block the IP that is considered dangerous, or analyze the port that is temporarily under attack and then notify the admin to close the port. . From the results of this algorithm research by simulating attacks using Synflood Attack on the server, it can be seen that the level of effectiveness of the algorithm in dealing with attacks can make RAM and CPU lighter so that it does not burden the hardware when compared to without using the algorithm and also makes system network traffic more efficient.

Keywords: IP tables, Firewall, Wireshark, Network Traffic Security

Citation: Wahid.A, Firdaus, M.E, Parenreng J.M, Implementation of Wireshark and IP tables Firewall Collaboration to Improve Traffic Security on Network Systems. *Iota*, **2021**, ISSN 2774-4353, Vol.01, 04. https://doi.org/10.31763/iota.v1i4.509

Academic Editor : P.D.P.Adi

Received : 14 September 2021

Accepted : 15 October 2021

Published : 15 November 2021

1. Introduction

The development of network and internet technology is directly proportional to the development of the number of attacks and crimes that occur on computer network systems. Various efforts have also been made to secure this network system, starting with security on the network side, security on the server-side, and also security on the client-side (end-user). The security techniques themselves also vary, such as the use of firewalls, network traffic monitoring applications, intrusion detection systems (IDS), the use of cryptographic protocols to the technique of strengthening server configurations. On the end-user side, security campaigns and data protection efforts continue to be carried out, such as the importance of using antivirus, the introduction and prevention of phishing or social engineering, and others.

One part that is currently considered the most vulnerable in this online era is security on the application server-side. Some of the efforts made are by monitoring network traffic that accesses the server continuously and also building a strong firewall application. Generally in a firewall, all incoming and outgoing communications from the server are controlled. In the server configuration, unimportant or unused ports can be blocked (closed) and important but dangerous ports should also be blocked. However, this sometimes creates new problems, for example, if you want to access a web server securely

from outside the network, you need to use SSH for remote system configuration purposes, while the SSH port on the server is usually set to be closed by the firewall system because it is considered very dangerous. could be a security hole against our server. Of course, it will be very disturbing for a server administrator.

Another problem that is often faced by server admins is the difficulty of identifying any activities carried out by clients that can harm the server. An admin must at all times monitor his server which then analyzes the results of the monitoring whether there are activities that endanger the server or not. Furthermore, they must close or block access from the IP that is identified and considered dangerous.

In this article, we offer solutions to the problems above by establishing a Wireshark collaboration and iptables firewall techniques to establish communication with server computers through network-based applications. With this application, a server admin can quickly open and close a port, block and unblock an IP address based on the notification given by the Wireshark application based on the analysis of the traffic that accesses the server. The technique used in Wireshark is an intrusion detection system (IDS) technique that will identify if there is traffic that is out of the ordinary or known as an anomaly. The results of this Wireshark analysis will then provide a notification to the server admin which then the server admin can do port blocking or IP filtering as desired through the application we made.

The ability to do port blocking and IP filtering is on the iptables firewall system available on the Linux server operating system. However, the application that we have made is expected to run on various operating system platforms. The purpose of this research entitled "Implementation of Wireshark and IPtables Collaboration in improving Traffic Security on network systems" is to make a configuration that fits the TCP port of the new TCP line, the UDP port of the new UDP line, and the new TCP and UDP ports. The new TCP string and UDP string will create rules on the iptables Linux server. Filtering can be enabled on the host computer, and filtering on the host computer involves blocking network ports and IP filtering or access in the configuration on iptables by using applications.

2. Theory

2.1. Computer network

A computer network is a collection of autonomous computers that are connected. It can be explained in everyday language that a computer network is a collection of various computers and other devices, such as printers, hubs, etc. Which are connected through an intermediary medium. The intermediary media can be wired or wireless. Wireless media information is data that is sent from one computer to another so that each computer can exchange data or share hardware devices.[1]–[3], moreover, wireless computer networks and WSNs that are ad-hoc in nature are also developing with several studies discussing the Quality of Services (QoS) side [26-30]

2.2. Iptables

Iptables Linux module that directly supports Linux kernel version 2.4 or higher to ensure system security and various other network requirements [4]. IPTables can also be used to select incoming, outgoing, and forwarded data packets based on IP address, network ID, port number, source (source), destination (destination) and protocol used, which also depends on each respective connection type. package [5]. Iptables can perform packet counting and apply traffic priority based on service type. Iptables can be used to

define many port-based security rules to protect specific hosts [6]. Iptables can also be used to create routers or gateways, of course only for Linux operating systems.[2],[7],[8]

2.3. Firewall

The security system that can protect your computer from various Internet threats is a Firewall. This firewall acts as a barrier or wall barrier against computers on the Internet through a "firewall".[9], [10], you can control the data, information, and operations that can be sent from the Internet network to the computer, and vice versa. There are two types of firewalls: hardware and software. They all have different attitudes or attitudes. However, they both perform the same basic function: protect network security. A hardware firewall is a piece of hardware that resides in a network system, such as a router. This type of firewall requires configuration to operate effectively [11], [12]. To function properly, the firewall uses filtering methods to determine the primary, source, and destination data packets.

The system compares the data internally according to the set rules. Then, you decide which data should be deleted or sent to the destination. A software firewall is a firewall solution for home Internet users. These firewalls are usually built as standalone applications or as additional antivirus features to protect incoming and outgoing traffic, and protect you from Trojan horses and worms.[13]–[15].

2.4. Wireshark

An open-source packet data retrieval application that can be used to scan and capture traffic on the Internet. This application is usually used as a troubleshooting tool on problematic networks, and since it can read the content of any packet traffic, it is also widely used for testing software. The app was previously called Ethereum but was renamed Wireshark due to branding issues. Wireshark supports many packet capture/tracking file formats, including .cap and .erf. In addition, the built-in decryption tool can display encrypted data packets from various protocols widely used on the Internet today, including WEP and WPA/WPA2[16],[17]. One of Wireshark's strengths is cross-platform development and distribution, so even if Ian is positive, Linux and Macintosh users can install and use this feature of the Wireshark application. Wireshark is useful for network analysis. Its principle of operation is to "capture" data packets of various protocols of various types of networks that are common in Internet network traffic[18], [19]. These data packets are "captured" and then displayed in a real-time window to get the capture results. At the start of the network analysis process using Wireshark, all recorded packets will be displayed without being scanned (promiscuous mode). Use the sort and filter commands to process all the packets again. On the downside, some hackers usually use Wireshark for tracking. The term tracking is not much different from capturing parcel data, but it has a negative connotation because it can harm others, especially in terms of privacy. For Wireshark to work properly, you need a WinPcap or Npcap application as a base. WinPcap can still be used in the highest versions of Windows 7, and Windows 10 no longer supports WinPcap, which is why Npcap was developed. Compared to pcap as the libcap library on Linux systems, Windows only uses one port of the libcap library, namely Npcap . Pcap is an API (application programming interface) used to capture network traffic from the Internet. Pcap is not new, it is a major part of its predecessor, TCPDUM P (Package Retrieval Program)[20],[21]. Wireshark uses pcap to collect data packets, so network analysts using Wireshark can only collect data packet types that only pcap supports.

2.5. IPTables Chain

In general, the flow of the package description on iptables is like Figure 1 with the respective processes and functions in order to create good packet filtering.

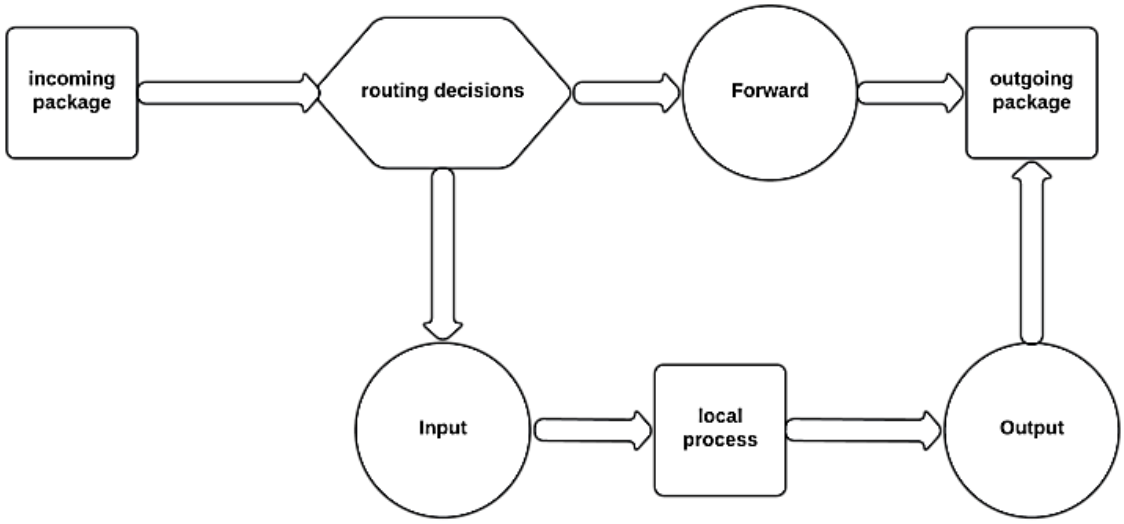

Figure 1. Filtering system on Iptables

- Filter – Specifies the packet to be DROP, LOG, ACCEPT or REJECT.
- NAT – Translate (change) the origin or destination address of a packet.
- Mangle – Perform refinement (mangle) on data packets such as TTL, TOS, and MARK.

Each table above has a rule called a chain. The filter has 3 chains:

- FORWARD: Performs a packet filter that will be forwarded from one NIC to another.
- INPUT: Perform packet filters intended for firewalls.
- OUTPUT: Perform a packet filter that will come out of the firewall.

NAT has 3 chains:

- PRE-ROUTING: Used to translate the address before the routing process occurs, namely changing the destination IP.
- POST-ROUTING: Used to translate addresses.
- OUTPUT: Used to translate the address of data packets coming from the firewall itself.

2.6. Anomali – Based IDS (Intrusion Detection System)

IDS or Intrusion Detection System is a process or activity to monitor events that occur on computer systems or networks, analyze possible incidents of attacks, violations or threats to computer security, use of law or standard security practices [22].

This system monitors the state of data traffic on the network and monitors various activities or suspicious activities on the network system. If any suspicious activity related to data traffic on the network is found, this IDS will alert the system or network administrator. In many cases, the IDS will also respond to abnormal/abnormal traffic by blocking users or source Internet Protocol (IP) addresses from trying to access the network [23].

This IDS model offers several types and systems approaches whose main purpose is to detect suspicious data traffic on a network. In general, there are two types of IDS, namely network-based (NIDS) and host-based (HIDS). There are IDSs that track the peculiarities of frequently performed experiments. This method is almost the same as antivirus software, which detects the system and protects it from threats. Then there is also IDS which recognizes the existing normal traffic patterns based on comparisons and then looks for existing traffic anomalies. There are IDSs that only function as supervisors and warnings in the event of an attack, and there are IDSs that not only function as supervisors and whistleblowers, but can also carry out activities that react to attempted attacks on networks and computer systems [24], [25].

Anomaly-based detection is the process of comparing the state of activity that is considered normal with the observed events to be able to detect anomalies through visual analysis and can detect anomalies in traffic flow on the network, of course using network analysis software such as Wireshark.

3. Method

3.1. System Design

To collaborate with the Wireshark application and the iptables firewall system as described in the introductory point above, a system is designed that will be the interface or liaison between the Wireshark application as a traffic monitoring and analysis application on a network system connected to the application server and an iptables-based firewall system as a The application located in front of the server is in charge of maintaining the security of the server behind it. The system will analyze the output of Wireshark if it is considered a malicious IP then the system will order the addition of a new rule on iptables to block that IP. Or if the analysis results detect a port that is in an exploited condition, the system will order the addition of a new rule to close the corresponding port. Likewise, if the opposite happens, then the IP or port will be unblocked again. Figure 2 shows a system design designed to collaborate with Wireshark and iptables by utilizing the algorithm that we have created.

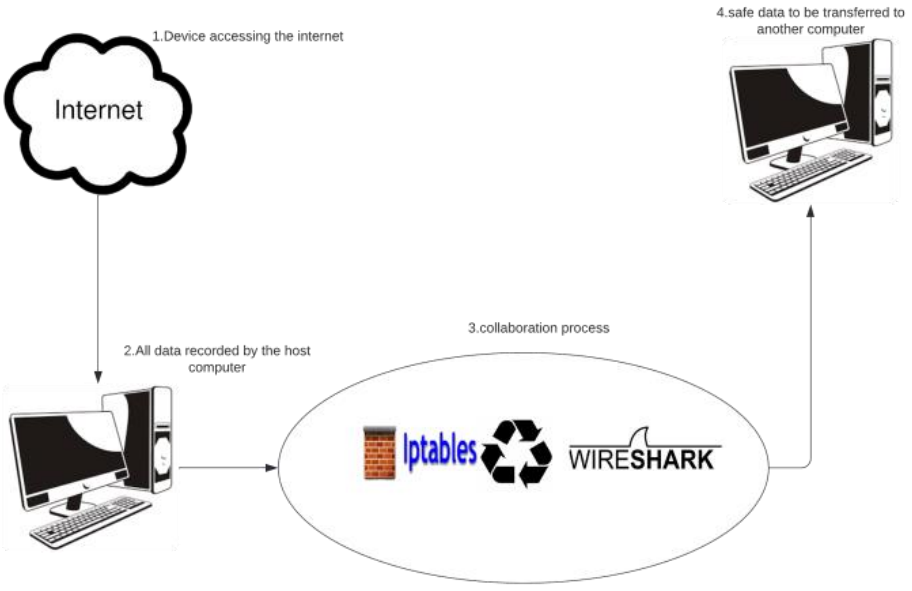

Figure 2. Wireshark-Iptables Collaboration System Design

Flow Outline:

1. Incoming data from a device that does or uses the network
2. The data obtained is entered into the host computer for processing
3. At this stage, Wireshark and Iptables collaborate to produce Firewall rules that can block threats from outside parties based on algorithms.
4. After being processed in stage 3, the IP which is considered safe can access or pass through the Firewall for the data to be used by Device clients

```
IPtables_Wireshark_Algorithm
Declaration
        VARCHAR Output Wireshark
Begin

IF (SPECIFIC PORT YOU WANT TO CLOSE OR OPEN);
ECHO " CLOSE OR OPEN A SPECIFIC PORT " ;
        ELSE IF (RESTORE IPtables RULES);
        ECHO " RESTORE THE RULES";
                ELSE IF ( IP TARGET SPECIFIC BLOCK );
                ECHO " BLOCK TARGET IP "
                        ELSE IF (MANAGING RULES BASED ON RESOURCES(SAME IP ACCESS MULTIPLE PORTS WITH 200 TIMES RANGE));
                        ECHO " MANAGING RESOURCES "
END
```

Figure 3. Pseudecode Algorithm

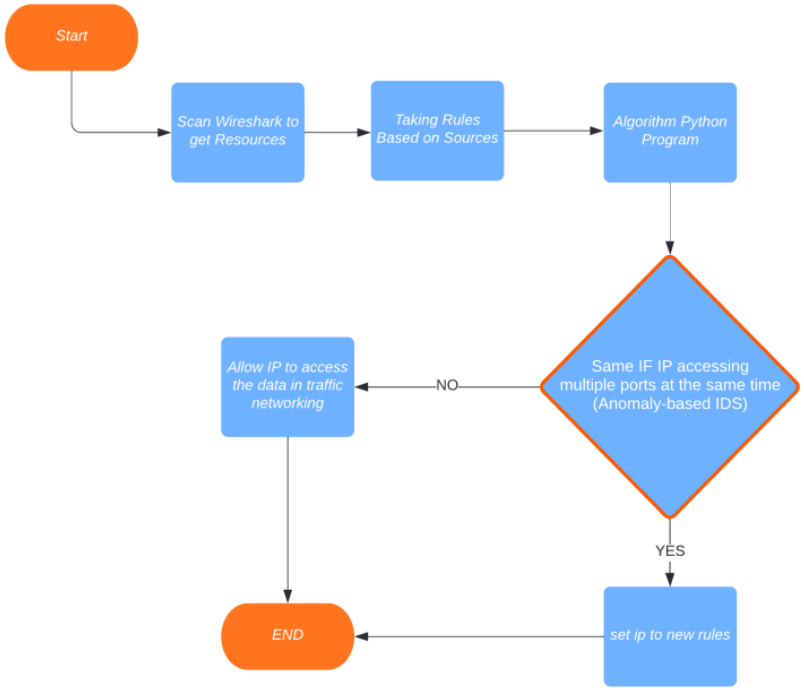

Figure 4. Flowchart Algorithm

The collaboration of iptable as IP filtering and Wireshark as an analyzer of data packets and IP that is implemented will provide a guarantee of security for the server because the firewall will be adaptive by always making new rules based on the condition of the data packets being analyzed. It is created in the form of a device management system. Figure 3 shows a flowchart of how the wireshark and iptables collaboration system application works to improve traffic security on the network.

3.2. System Requirement

In testing this Wireshark and iptables collaboration application, the host computer specifications are used for data retrieval. Table 1. shows the system specifications used as the host computer during the test as a minimal system.

Table 1. System Requirement

RAM	**4GB Memory - Min**
Operation System	Linux Server
Processor	Intel Core i5-3350 3.10GHz
Architecture	64-bit

The system requirements used are adjusted to the minimum system limits that can be used by useful supporting software so that when conducting experiments they do not experience bugs or errors.

4. Result and Discussion

4.1. Scanning process by Wireshark application

The application works by running Wireshark to detect data traffic going to and out of the server over the network. Wireshark continuously scans data that passes according to the desired network adapter, in this case, the adapter that goes to our server. If the scanning results indicate anomaly activity, namely activity that is out of the ordinary, then Wireshark will provide a notification asking to immediately close the port passed by the anomaly activity or enter the IP as a blocked IP. Figure 5 shows the continuous scanning activity carried out by Wireshark until it gets anomaly data which will then be processed by an application that is useful for writing rules on IPtables on the firewall.

4.2. Wireshark Output Analysis Process for basic iptables rule creation

The scanning results from the Wireshark application will then be saved into a file that will be processed by the application to be used as useful resources for rewriting rules on iptables. Figure 5 below shows the output data from Wireshark in the form of a file that will be the input for iptables reconfiguration.

4.3. The process of writing a new iptables rule based on the results of the analysis

After reading the Wireshark output file that contains recommendations for reconfiguring iptables rules, the server administrator can configure directly to iptables using the application we created. Figure 6 shows an application on the admin server-side that is used to write new rules on iptables.

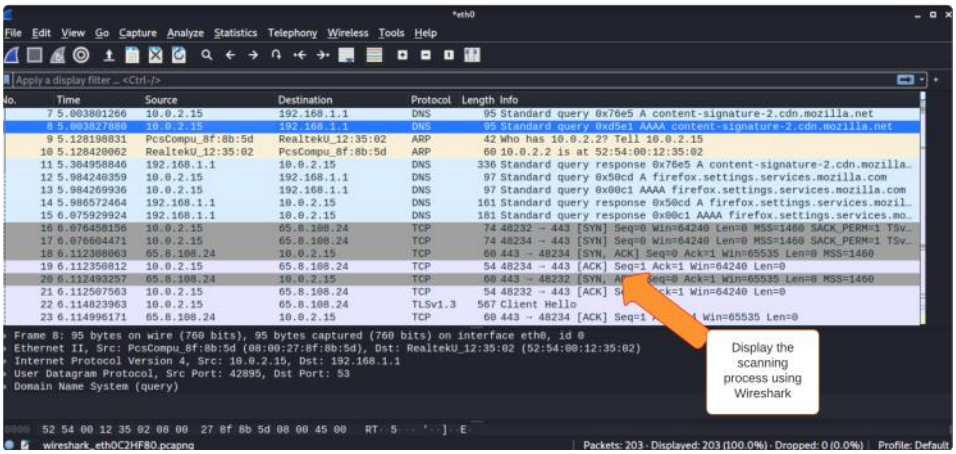

Figure 5.Wireshark Scanning process

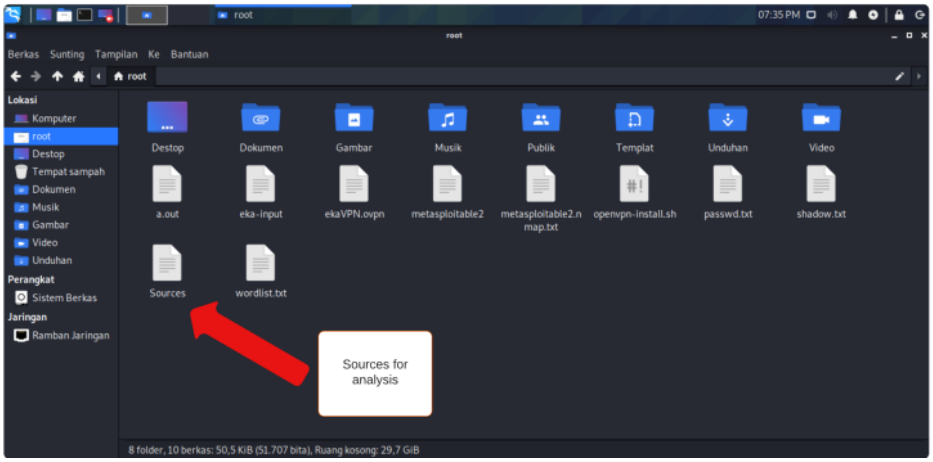

Figure 6. Data display of the output wireshark file

To process the output data from Wireshark and to make writing rules faster and easier for the server administrator, an interface application is provided using the Python GUI program Tkinter. There are several menus provided as follows:

- Open and close ports
- Restoring rules
- Target Specific Block
- Block IP based on Source

4.4. The result of writing a new rule on iptables

The results of writing rules using the above application, you can see the results directly on the iptables rules as shown in Figure 7. The application has successfully updated the rules as entered by the server administrator.

Figure 7. Application of writing new rules on Iptables using python

Figure 8. Display of new IP tables rules

Based on the results of the analysis of the output of Wireshark, it can be seen in Figure 7 that several IPs that were detected carrying out attack and threat activities could be blocked using the application and based on the results of data analysis by Wireshark.

4.5. Effectiveness and Efficiency Test

To find out how much benefit and usability of the application and the algorithm scheme we offer, we will test the effectiveness and efficiency of the application by creating an attack simulation scheme on a general topology model. Here we try to use several attack tools such as Hping3, Nmap, and Synflood Attack.

Effectiveness here is the result of how successful the system is in dealing with attacks when the application is used compared to when not using the application. We did the test by simulating the attack on the server or host with the application installed and not installing the application. The design of the simulation scheme

uses 1 computer (host/server), and a laptop (client). Host and client connections use the internet via an indihome wifi modem. The complete topology can be seen in Figure 9.

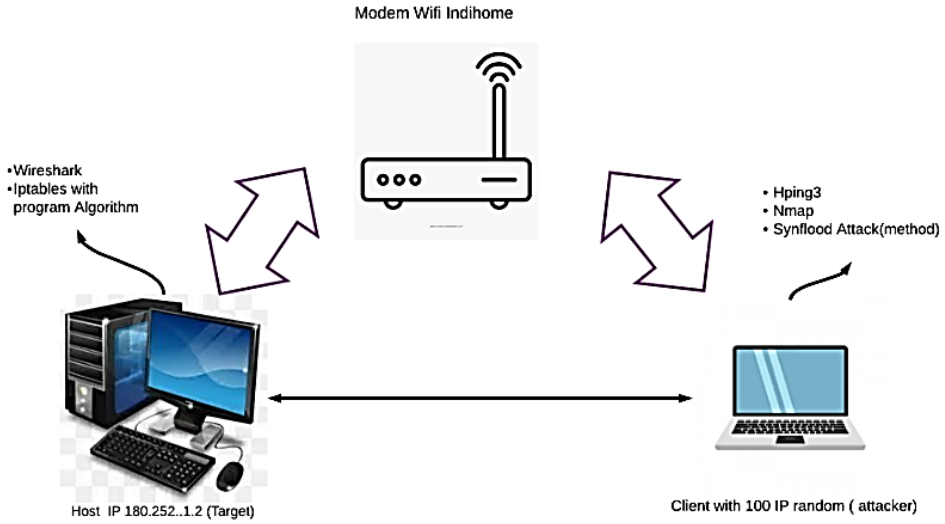

Figure 9. Attack Simulation Schematic

From the simulation scheme image, it is explained the use of each of these devices, the Indihome WiFi modem is used as an internet network provider to serve as a communication medium between devices. The computer is used as a target which will be equipped with Wireshark, iptables firewall, and applications to monitor anomalies in network traffic and update rules. iptables quickly. Then the laptop is used as an attacker who is equipped with Hping3, Nmap, and Synflood attack (method) to attack the target host with 100 IPs that have been prepared.

```
Common
  -d  --data       data size                  (default is 0)
  -E  --file       data from file
  -e  --sign       add 'signature'
  -j  --dump       dump packets in hex
  -J  --print      dump printable characters
  -B  --safe       enable 'safe' protocol
  -u  --end        tell you when --file reached EOF and prevent rewind
  -T  --traceroute traceroute mode             (implies --bind and --ttl 1)
  --tr-stop        Exit when receive the first not ICMP in traceroute mode
  --tr-keep-ttl    Keep the source TTL fixed, useful to monitor just one hop
  --tr-no-rtt       Don't calculate/show RTT information in traceroute mode
ARS packet description (new, unstable)
  --apd-send       Send the packet described with APD (see docs/APD.txt)

┌──(root💀kali)-[~]
└─# hping3 -1 --flood 180.252.1.2
HPING 180.252.1.2 (eth0 180.252.1.2): icmp mode set, 28 headers + 0 data bytes
hping in flood mode, no replies will be shown
```

Figure 10. Simulation of attack using Hping3

The attack is carried out on the client-side of the attacker using a Synflood Attack (Method) against the target to make the host computer burdened and also make network traffic crowded with attacks. on the computer side, the host itself will scan using Wireshark.

It can be seen in Figure 11 that we have done Synfloodattack to make network traffic very busy so that on the Wireshark side it will read an anomaly where the scanning output of the Wireshark itself will be entered into iptables through the application to be processed so that new rules are formed on iptables which will secure the server/ host of the currently running attacking process.

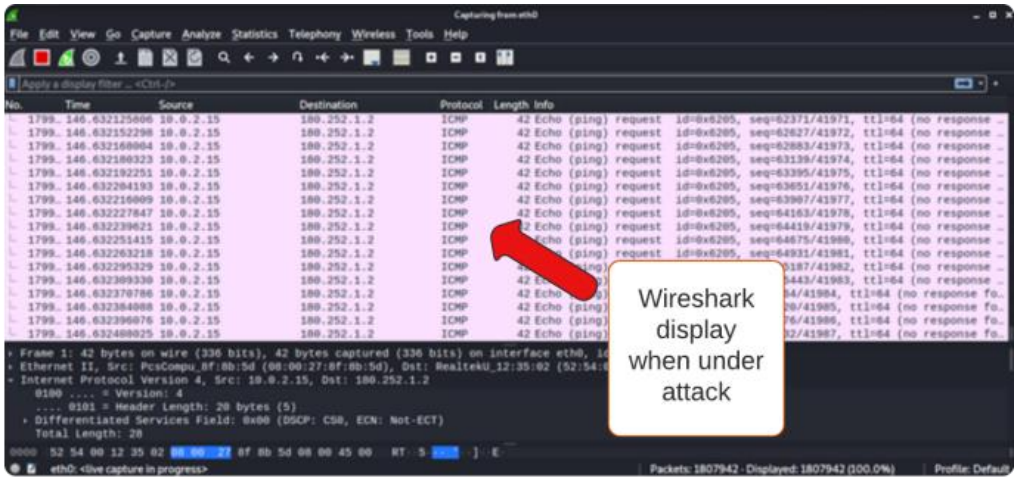

Figure 11. Scanning report showing an attack

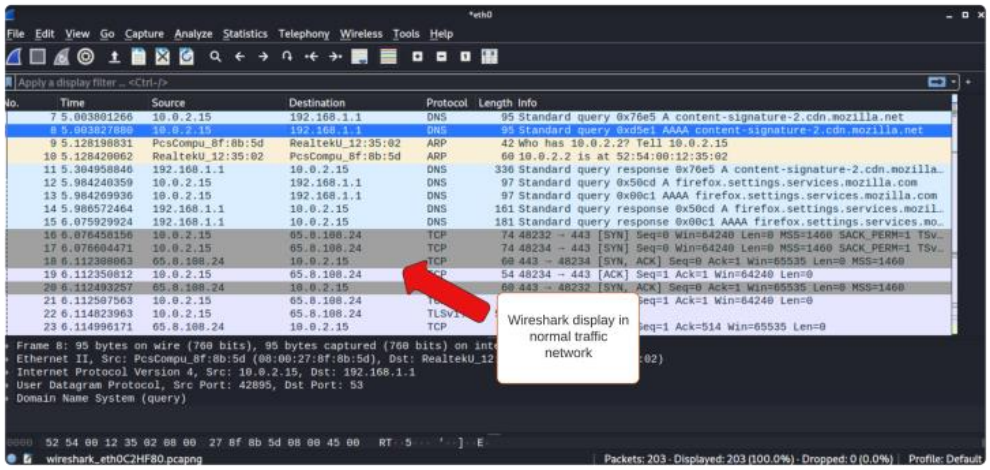

Figure 12. Scanning report under normal conditions

From the simulation process as above, which is carried out repeatedly over a predetermined period of time, we draw conclusions and put them in graphical form as shown in Figure 12.

Based on Figure 12, it can be explained that the Wireshark-iptables collaboration algorithm can run effectively as desired where about 70% of IP threats that access the server using Nmap (open port search) and Hping3 (attack) can be blocked successfully. There are a few problems experienced when the application has been running for about 2 hours then the effectiveness will decrease and we have to reset or restart the python program and turn it back on. This deficiency will be our next research to be able to improve the application.

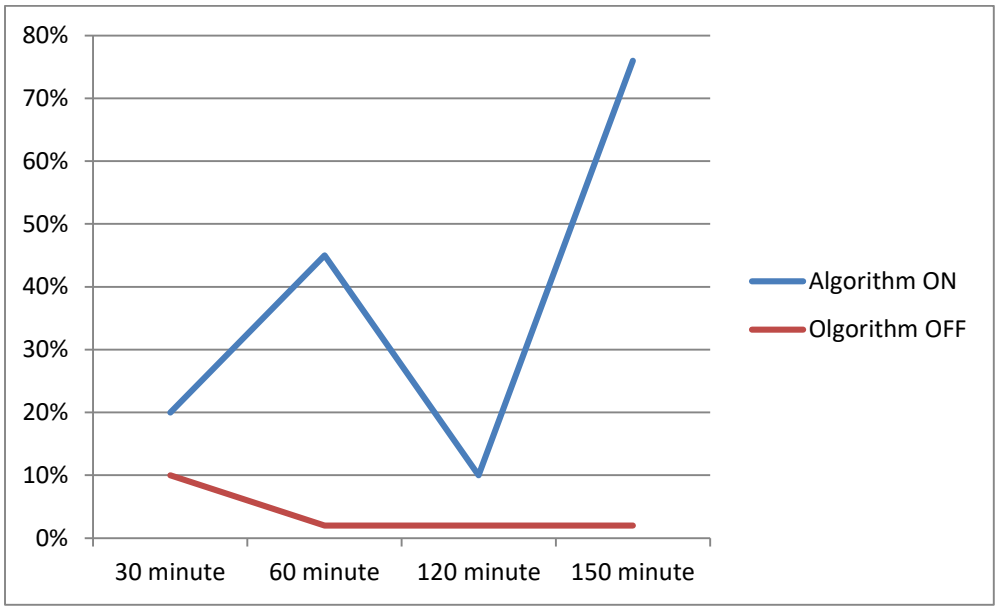

Figure 13. System effectiveness test results

Furthermore, in terms of application efficiency, we try to record differences in resource usage activities, both CPU and RAM by the server when the server is attacked but using this application or the server is attacked without using this application. The results can be seen in Figure 14 and Figure 15.

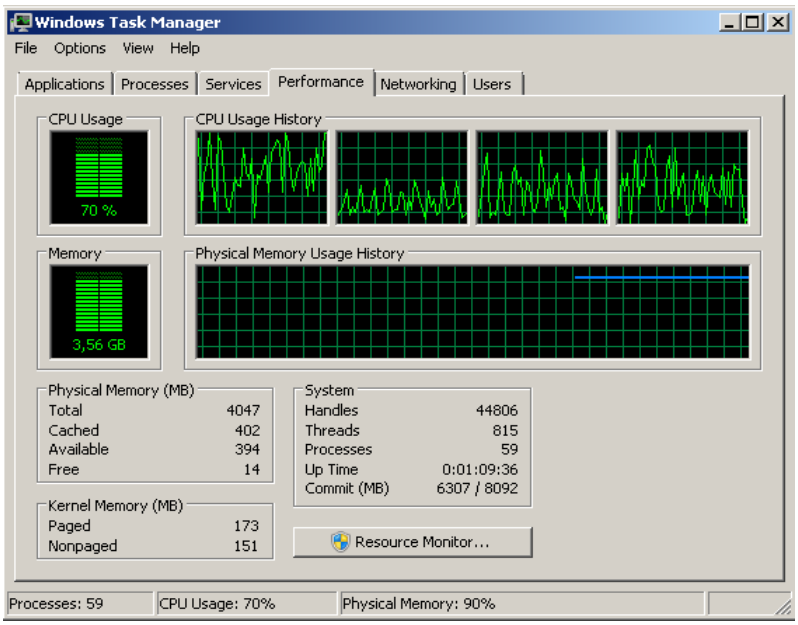

Figure 14. The use of resources by the server when attacked without using the application

When the host computer is attacked with no application that can detect the attack, the server condition experiences a CPU Usage condition that is above 70% which

will burden the host computer and RAM memory above 50% of the available memory.

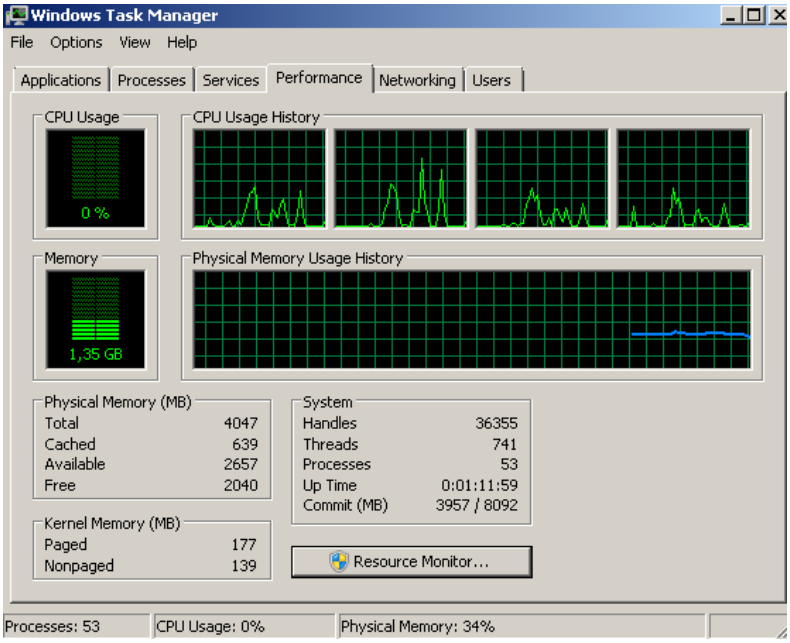

Figure 15. The use of resources by the server when attacked by using the Application

When the host computer is attacked but uses our application to create new rules based on the results of the Wireshark analysis, the server load is as shown in Figure 13 with the characteristics that occur are CPU Usage below 30% which does not burden the host computer and RAM below 50 % available memory.

Thus it can be concluded that by using this application, the server will work more efficiently with an efficiency level of 40% better in terms of CPU usage, although it is still the same in terms of RAM usage.

5. Conclusions and Suggestion

We have created an application that can increase server security from attacks by utilizing the capabilities of Wireshark as a tool for monitoring and analyzing data traffic and also the ability of the iptables firewall to perform port closures and IP attacks. This application is made in the form of a collaboration between Wireshark and iptables firewall.

After testing and simulating attacks, we can see that iptables is a firewall that can block an IP or prioritize IP in traffic based on the type of service, while Wireshark is a tool for analyzing network traffic so that it will create more detailed network security so that IP rules on the firewall can work better. In terms of effectiveness, we can conclude that by utilizing this application, about 70% of attacks can be blocked by the application compared to if not using this application, only about 2% of attacks can be overcome. In terms of its efficiency, this application can reduce the CPU workload 40% more efficiently than if you don't use this application.

Author Contributions: Conceptualization; Wahid.A (W.A), Firdaus, M.E (F.M.E), Parenreng J.M (P.J.M) ; methodology; (W.A),(F.M.E),(P.J.M), validation; (W.A),(F.M.E),(P.J.M), formal analysis; (W.A),(F.M.E),(P.J.M), investigation; (W.A),(F.M.E),(P.J.M), data curation; (W.A),(F.M.E),(P.J.M); writing—original draft preparation; (W.A),(F.M.E),(P.J.M), writing—review and editing; (W.A),(F.M.E),(P.J.M), visualization; (W.A),(F.M.E),(P.J.M), supervision; (W.A),(F.M.E),(P.J.M), project administration (W.A),(F.M.E),(P.J.M), funding acquisition; (W.A),(F.M.E),(P.J.M), have read and agreed to the published version of the manuscript.

Acknowledgments: I'am very Grateful for the supervisor who has taught and helped during the process of working on the article.

Conflicts of Interest: The authors declare no conflict of interest.

References

1.] B. Yi, X. Wang, K. Li, S. k. Das, and M. Huang, "A comprehensive survey of Network Function Virtualization," Computer Networks, vol. 133, pp. 212–262, Mar. 2018, doi: 10.1016/j.comnet.2018.01.021.
2. B. Wang, K. Lu, and P. Chang, "Design and implementation of Linux firewall based on the frame of Netfilter/IPtable," in 2016 11th International Conference on Computer Science & Education (ICCSE), Nagoya, Japan, Aug. 2016, pp. 949–953. doi: 10.1109/ICCSE.2016.7581711.
3. "IEEE Std 802.1X-2020, IEEE Standard for Local and Metropolitan Area Networks—Port-Based Network Access Control," p. 289, 2020.
4. P. Asghari, A. M. Rahmani, and H. H. S. Javadi, "Internet of Things applications: A systematic review," Computer Networks, vol. 148, pp. 241–261, Jan. 2019, doi: 10.1016/j.comnet.2018.12.008.
5. A. Boukerche and R. E. De Grande, "Vehicular cloud computing: Architectures, applications, and mobility," Computer Networks, vol. 135, pp. 171–189, Apr. 2018, doi: 10.1016/j.comnet.2018.01.004.
6. C. Diekmann, J. Michaelis, M. Haslbeck, and G. Carle, "Verified iptables firewall analysis," in 2016 IFIP Networking Conference (IFIP Networking) and Workshops, Vienna, Austria, May 2016, pp. 252–260. doi: 10.1109/IFIPNetworking.2016.7497196.
7. Yang Yang and Wang Yonggang, "A Software Implementation for a Hybrid Firewall Using Linux Netfilter," in 2010 Second World Congress on Software Engineering, Wuhan, Dec. 2010, pp. 18–21. doi: 10.1109/WCSE.2010.124.
8. D. Melkov, A. Saltis, and S. Paulikas, "Performance Testing of Linux Firewalls," in 2020 IEEE Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, Apr. 2020, pp. 1–4. doi: 10.1109/eStream50540.2020.9108868.
9. S. Khummanee, A. Khumseela, and S. Puangpronpitag, "Towards a new design of firewall: Anomaly elimination and fast verifying of firewall rules," in The 2013 10th International Joint Conference on Computer Science and Software Engineering (JCSSE), Khon Kaen, Thailand, May 2013, pp. 93–98. doi: 10.1109/JCSSE.2013.6567326.
10. A. X. Liu and M. G. Gouda, "Firewall Policy Queries," IEEE Trans. Parallel Distrib. Syst., vol. 20, no. 6, pp. 766–777, Jun. 2009, doi: 10.1109/TPDS.2008.263.
11. X. S. Hu, R. C. Murphy, S. Dosanjh, K. Olukotun, and S. Poole, "Hardware/software co-design for high performance computing: challenges and opportunities," p. 2.
12. G. Roquier, E. Bezati, R. Thavot, and M. Mattavelli, "Hardware/software co-design of dataflow programs for reconfigurable hardware and multi-core platforms," in Proceedings of the 2011 Conference on Design & Architectures for Signal & Image Processing (DASIP), Tampere, Finland, Nov. 2011, pp. 1–7. doi: 10.1109/DASIP.2011.6136875.

13. N. Thamsirarak, T. Seethongchuen, and P. Ratanaworabhan, "A case for malware that make antivirus irrelevant," in 2015 12th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), Hua Hin, Jun. 2015, pp. 1–6. doi: 10.1109/ECTICon.2015.7206972.
14. C. C. Zou, Weibo Gong, D. Towsley, and Lixin Gao, "The monitoring and early detection of Internet worms," IEEE/ACM Trans. Networking, vol. 13, no. 5, pp. 961–974, Oct. 2005, doi: 10.1109/TNET.2005.857113.
15. J. Kinder, S. Katzenbeisser, C. Schallhart, and H. Veith, "Proactive Detection of Computer Worms Using Model Checking," IEEE Trans. Dependable and Secure Comput., vol. 7, no. 4, pp. 424–438, Oct. 2010, doi: 10.1109/TDSC.2008.74.
16. Yong Yu, Qun Wang, and Yan Jiang, "Research on security of the WLAN campus network," in 2010 International Conference on E-Health Networking Digital Ecosystems and Technologies (EDT), Shenzhen, China, Apr. 2010, pp. 175–178. doi: 10.1109/EDT.2010.5496402.
17. S. Lepaja, A. Maraj, I. Efendiu, and S. Berzati, "The impact of the security mechanisms in the throughput of the WLAN networks," in 2018 7th Mediterranean Conference on Embedded Computing (MECO), Budva, Jun. 2018, pp. 1–5. doi: 10.1109/MECO.2018.8406067.
18. Jiliang Zhang, Yaping Lin, Yongqiang Lyu, and Gang Qu, "A PUF-FSM Binding Scheme for FPGA IP Protection and Pay-Per-Device Licensing," IEEE Trans.Inform.Forensic Secur., vol. 10, no. 6, pp. 1137–1150, Jun. 2015, doi: 10.1109/TIFS.2015.2400413.
19. Y. Jimenez, C. Cervello-Pastor, and A. Garcia, "Dynamic Resource Discovery Protocol for Software Defined Networks," IEEE Commun. Lett., vol. 19, no. 5, pp. 743–746, May 2015, doi: 10.1109/LCOMM.2015.2403322.
20. C. A. Fowler and R. J. Hammel, "Converting PCAPs into Weka mineable data," in 15th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), Las Vegas, NV, USA, Jun. 2014, pp. 1–6. doi: 10.1109/SNPD.2014.6888681.
21. W. Yu, Z. Mi, D. Niu, and C. Dong, "PCAP: Proportional Channel Access Probability Fairness in Multi-rate IEEE 802.11 DCF," in 2012 Second International Conference on Intelligent System Design and Engineering Application, Sanya, China, Jan. 2012, pp. 448–451. doi: 10.1109/ISdea.2012.585.
22. S. Pan, T. Morris, and U. Adhikari, "Developing a Hybrid Intrusion Detection System Using Data Mining for Power Systems," IEEE Trans. Smart Grid, vol. 6, no. 6, pp. 3104–3113, Nov. 2015, doi: 10.1109/TSG.2015.2409775.
23. M. Kumar and A. K. Singh, "Distributed Intrusion Detection System using Blockchain and Cloud Computing Infrastructure," in 2020 4th International Conference on Trends in Electronics and Informatics (ICOEI)(48184), Tirunelveli, India, Jun. 2020, pp. 248–252. doi: 10.1109/ICOEI48184.2020.9142954.
24. J. Yang, X. Chen, X. Xiang, and J. Wan, "HIDS-DT: An Effective Hybrid Intrusion Detection System Based on Decision Tree," in 2010 International Conference on Communications and Mobile Computing, Shenzhen, China, Apr. 2010, pp. 70–75. doi: 10.1109/CMC.2010.73.
25. A. R. Hakim, J. Rinaldi, and M. Y. B. Setiadji, "Design and Implementation of NIDS Notification System Using WhatsApp and Telegram," in 2020 8th International Conference on Information and Communication Technology (ICoICT), Yogyakarta, Indonesia, Jun. 2020, pp. 1–4. doi: 10.1109/ICoICT49345.2020.9166228.
26. M. Niswar et al., "Performance evaluation of ZigBee-based wireless sensor network for monitoring patients' pulse status," 2013 International Conference on Information Technology and Electrical Engineering (ICITEE), 2013, pp. 291-294, doi: 10.1109/ICITEED.2013.6676255.
27. Mukti, Fransiska Sisilia, Adi, Puput Dani Prasetyo , Prasetya, Dwi Arman, Sihombing, Volvo, Rahanra, Nicodemus, Yuliawan, Kristia and Simatupang, Julianto (2021) Integrating Cost-231 Multiwall Propagation and Adaptive Data

Rate Method for Access Point Placement Recommendation. International Journal of Advanced Computer Science and Applications (IJACSA), 12 (4). pp. 772-777. ISSN 2156-5570 (e) ; 2158-107X (p)

28. P. D. P. Adi and A. Kitagawa, "Performance Evaluation of Low Power Wide Area (LPWA) LoRa 920 MHz Sensor Node to Medical Monitoring IoT Based," 2020 10th Electrical Power, Electronics, Communications, Controls and Informatics Seminar (EECCIS), 2020, pp. 278-283, doi: 10.1109/EECCIS49483.2020.9263418.
29. P D P Adi, A Kitagawa, V Sihombing, G J Silaen, N E Mustamu, V M M.Siregar, F A Sianturi, W Purba, A Study of Programmable System on Chip (PSoC) Technology for Engineering Education, WEAST 2020, doi:10.1088/1742-6596/1899/1/012163
30. Y. A. Liani et al., "The Broiler Chicken Coop Temperature Monitoring Use Fuzzy Logic and LoRAWAN," 2021 3rd International Conference on Electronics Representation and Algorithm (ICERA), 2021, pp. 161-166, doi: 10.1109/ICERA53111.2021.9538771.