Research Article

Simulation and Analysis of Network Security using Port Knocking and Intrusion Prevention System on Linux Server

Jumadi Mabe Parenreng[1], Fahru Rizal[2*], Maya Sari Wahyuni[3]

[1,2,3] Department of Informatics and Computer Engineering Faculty of Engineering, Makassar State University
* Corresponding Author: frizalichal@ymail.com

Abstract: This research aims to design and simulate a network security system using port knocking and an intrusion prevention system (IPS) on a Linux-based server and analyze its security using port scanning, brute force, and DoS attacks. IPS uses inline snort mode with DAQ NFQ. The test results show that port knocking successfully opens and closes the port according to the knock sequence so that only those who know the knock sequence can access the port. The port scanning, SSH, and FTP Brute Force test results were successfully detected by IPS so that the attacker could not obtain any information. DoS testing with LOIC increased server CPU and memory usage, but after IPS rules were applied, usage stabilized. DoS testing with slowhttptest makes the webserver inaccessible to users, but after the IPS rule is applied, web access runs normally. In conclusion, IPS was successful in preventing all attacks because the attack packets complied with IPS rules so they were detected as threats and dropped by IPS. Test results of the Telegram monitoring system show that the system succeeded in sending real-time attack notifications with an average time difference of 2.9 seconds, and the report, start, and stop features worked as expected.

Keywords: Network Security; Port Knocking; Intrusion Prevention System; Snort; Linux-based server

Citation: J.M.Parenreng, F.Rizal, M.S.Wahyuni, "Simulation and Analysis of Network Security using Port Knocking and Intrusion Prevention System on Linux Server", *Iota*, **2024**, ISSN 2774-4353, Vol.04, 02. https://doi.org/10.31763/iota.v4i2.726

Academic Editor : Adi, P.D.P

Received : March, 17 2024

Accepted : April, 28 2024

Published : May, 5 2024

1. Introduction

These current technological developments have developed very rapidly. Technology, especially the internet, has made it easier for people to get unlimited information. Technology makes things easier for society in various fields, be it economic, educational, social, and so on[1].

Along with the development of technology, especially computer networks, there is one thing that needs to be considered, namely network security. Network security is a concept that aims to prevent unauthorized users from entering a computer network system. Network security functions to anticipate risks that can occur on a computer network that can disrupt activities that are occurring on the computer network system[2]. There are three main aspects of network security, namely confidentiality, integrity, and availability. Confidentiality is related to access rights in viewing data or information, integrity relates to access rights in changing data or information, and availability is related to data availability[3].

If vulnerabilities in computer networks are not paid attention to, they have the potential to cause losses such as data loss and server damage. The server is a vital component in the network because this component provides services needed by clients and is tasked with processing data on the network.

Several potential attacks can threaten servers, including Distributed Denial of Service (DDoS), attacks from hackers, viruses, and trojans[4]. Apart from that, port scanning attacks also often occur, where unauthorized parties try to obtain information about ports on the network [5]. Based on data from the National Cyber and Crypto Agency (BSSN), 714,170,967 cyber-attacks occurred in Indonesia throughout 2022. Of the types of cyber-attacks that occurred, there were attacks of brute force, DoS (Denial of Service), and DDoS

[6]. This certainly raises concerns so a method is needed that can protect the server. One of the network security methods used is port knocking and IDS/IPS.

Port knocking is an authentication system to protect a communication port. This method can add a layer of security to the server because only the user knows the combination sequence of requests that can access ports on the server [7]. Apart from that, there is an IDS (Intrusion Detection System) that can detect attacks and threats that occur on a computer network. IDS will provide a warning to the administrator when there is unusual activity on the computer network[8]. IPS (Intrusion Prevention System) is a development of IDS where the IPS system, apart from being able to detect attacks, can also prevent attacks from occurring. One of the open-source IDS/IPS tools is Snort [9].

This research will design a network security system using the method of port knocking and IPS and telegram as media for notification of attacks. Port knocking is used as an authentication mechanism to open blocked SSH and FTP ports while IPS is used to monitor network traffic and detect and take action to prevent attacks.

Table 1 shows some of the literature on the study. The difference between this research and previous research lies in the method of handling attacks. In previous research, the Intrusion Detection System method was used so that attacks could only be detected and there was no prevention of attacks, whereas, in this research, the IPS (Intrusion Prevention System) method was used so that the system could prevent attacks when an attack was detected. IPS prevents attacks by dropping attack packets automatically so that administrators do not need to block IPs manually via the terminal.

Table 1. Literature of Study

No	Previous Study	Research Focus	Weakness
1	Implementation of Network Security with Iptables as a Firewall Using the Port Knocking Method	Used port knocking to secure the SSH port and carry out stealth scanning testing with Zenmap	The focus is only on securing the SSH port
2	Implementation of port knocking method on virtual ubuntu server security system based on web monitoring	Implemented port knocking on Virtual Private Server (VPS) and created web monitoring to monitor knock activity on the server	
3	Analisis kinerja Intrusion Detection System dalam mendeteksi serangan siber pada Apache Web Server [12]	Focuses on analyzing the performance of Snort IDS in detecting SSH brute force, SYN Flood, DoS ICMP, DoS UDP, and DoS TCP on web servers.	The research focus is only on attack detection
4	Network Security Monitoring Model Through Telegram App With Snort [13]	Proposed a network monitoring system with snorts and attack notifications sent to Telegram	
5	Design and Implementation of NIDS Notification System Using WhatsApp and Telegram [14]	Focuses on creating a notification system by using and utilizing WhatsApp and Telegram as notification media and using snort as an IDS in detecting ping of death, SYN Flood, and SSH brute force	
6	Implementation of Intrusion Detection Systems (IDS) and Intrusion Prevention Systems (IPS) methods based on Snort Server for LAN network security [15].	Proposed IDS method with Snort and IPS with iptables.	The IPS method used is to block IPs, but IP blocking is done manually

2. Theory

2.1 Port Knocking

Port knocking is a security method used to open access to certain ports that are closed by a firewall by sending certain packets or connections so that network devices can access certain ports even though they have been restricted by the firewall [16]. The way of work port knocking is to close all ports and only certain clients can access that port. Port knocking possible server to rewrite rules firewall so that the client gets permission to access the port. After it finishes, the server can re-close the port by deleting the rules firewall previously [17].

2.2 Intrusion Detection System and Intrusion Prevention System

An intrusion detection system (IDS) is a system in the form of software or hardware that is used to monitor traffic on a computer network. IDS can provide warnings to administrators when there is unusual or suspicious activity on a network. IDS works by monitoring activity and checking for gaps in the system, file integrity, and conducting pattern analysis based on attacks [18].

Intrusion prevention system (IPS) is a development of Intrusion detection system. The difference is that IPS, apart from being able to monitor the network and detect threats, like IDS, can also take action to deal with suspicious threats in the network [5]. An intrusion prevention system consists of two types including [19]: Host-Based Intrusion Prevention System (HIPS) is a type of IPS configured directly on the protected system to monitor its internal system activity. HIPS is tied to the operating system kernel and operating system services so that HIPS can monitor and intercept suspected systems to prevent host intrusion. Network-based Intrusion Prevention System (NIPS) is a type of IPS that carries out monitoring and protection in one network and is not limited to just one host. NIPS is a system that combines IPS features with a firewall and is often referred to as inline IDS.

2.3 Snort

Snort is one of the rule-based IDS software (rule-driven) that can monitor network traffic passively and provide warnings when a threat is detected [20]. Snort developed by Martin Roesch, was initially launched as a device sniffing crossplatform and then released with IDS features in 2003. Snort is open-source software in which users can contribute to the development of the system by suggesting modifications in the source code, reporting bugs, and suggesting bug fixes. Snort combines the benefits of signature-based, protocol, and anomaly-based inspection methods [9].

3. Method

The research model used in this research is a model Security Policy Development Life Cycle (SPDLC). SPDLC is a method that determines a strategy for updating an organization from a network system. This SPDLC model consists of five stages, namely Requirement Analysis, Design, Implementation, Enforcement (testing), and Enhancement (evaluation) [21].

Requirements analysis is the problem formulation stage, analyzing the concept of port knocking and intrusion prevention systems, and identifying system needs or requirements that will be used by the server and client. Design is the design stage system topology and system flowchart to provide an overview of the security system to be built including the IP address of each device used.

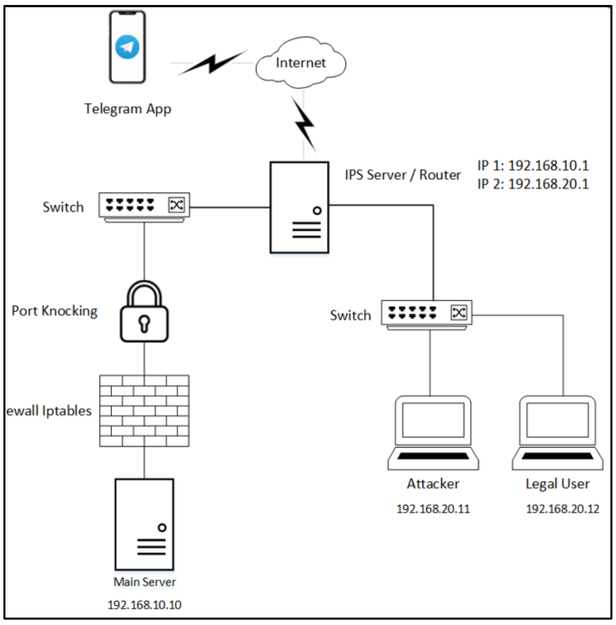


Figure 1. System Topology

Fig.1 shows the topology of the system to be built, where there is an attacker, legal user, main server, IPS Server/router, and smartphone that has Telegram installed. An attacker is a party who tests the system by carrying out port scanning, brute force, and DoS attacks on the server. IPS server/router is a server that has been configured with IPS to detect and prevent attacks on the network. IPS will also provide logs to Telegram so that administrators can find out that an attack has occurred on the server and act as a router to manage data traffic while connecting different networks. On the main server, the port knocking method and iptables firewall are applied. The iptables firewall on the main server acts to block ports, open ports, and close ports. Port knocking is used to regulate access rights to the server by requiring the correct knock sequence so that when the correct knock sequence is received, the firewall will rewrite the rules so that access to the server is opened.

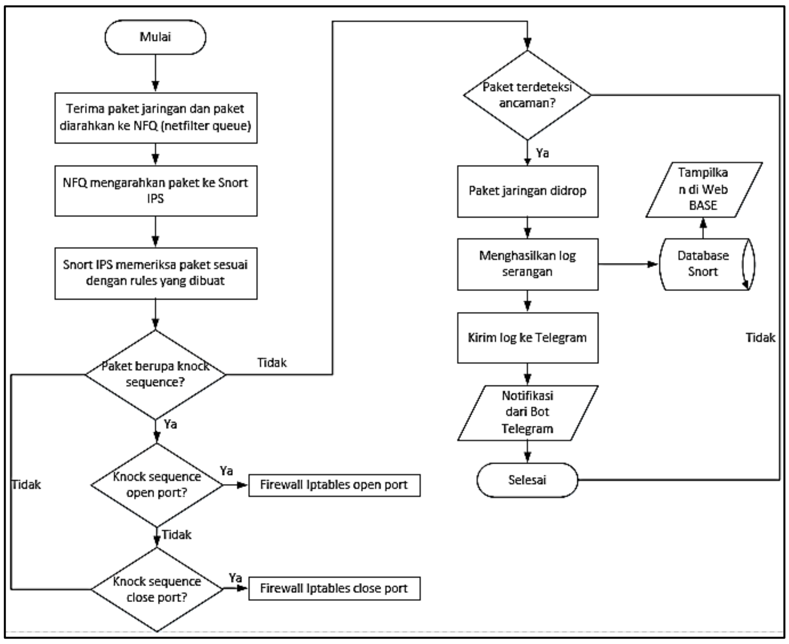

Mulai
Terima paket jaringan dan paket diarahkan ke NFQ (netfilter queue)
NFQ mengarahkan paket ke Snort IPS
Snort IPS memeriksa paket sesuai dengan rules yang dibuat
Paket berupa knock sequence?
Tidak
Ya
Knock sequence open port?
Ya
Firewall Iptables open port
Tidak
Knock sequence close port?
Ya
Firewall Iptables close port
Tidak
Paket terdeteksi ancaman?
Ya
Paket jaringan didrop
Menghasilkan log serangan
Kirim log ke Telegram
Notifikasi dari Bot Telegram
Selesai
Database Snort
Tampilkan di Web BASE
Tidak

Figure 2. System flowchart

Figure 2 shows the flowchart of the system being built. First, the system will receive network packets and the packets are directed to the NFQ (Netfilter queue). After that, NFQ will direct the packet to Snort IPS and Snort IPS then checks the packet according to the rules created. If the packet is in the form of a knock sequence and what is entered is for an open port then the iptables firewall will open the port. If not, port knocking will check whether the knock sequence entered is for a closed port. If it is true then the iptables firewall will carry out a close port mechanism. If not then the data packet is ignored. Furthermore, if a package is detected as a threat or attack, IPS will automatically drop the package and IPS generates a log and stores it in the Snort database to be displayed on the BASE (Basic Analysis and Security Engine) web as well as sending it to Telegram and the administrator will receive a notification from telegram.

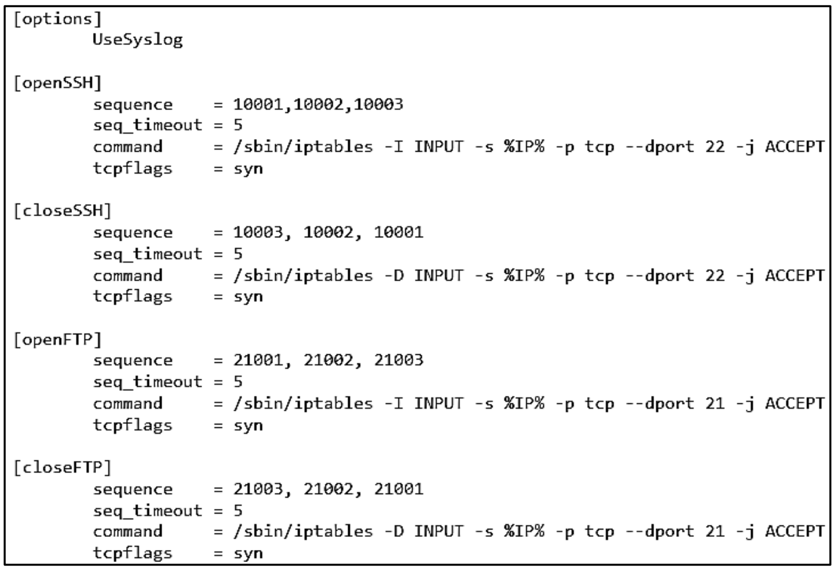
Implementation is the application stage by carrying out installation and configuration according to the design at the design stage. Enforcement is the testing stage of the system that has been built. The testing consists of several tests including port knocking testing, intrusion prevention system testing, and telegram monitoring system testing

Enhancement is the evaluation stage of systems that have been tested and identifying areas that require improvement. Evaluation includes the success of port knocking in opening ports and closing ports, the success of the IPS system in detecting attacks and handling attacks, and the success of the telegram monitoring system. Based on the evaluation results, improvements are then made such as reconfiguring the configuration or adding rules to the IPS.

4. Result and Discussion

4.1 Port knocking

Port knocking in this research is implemented on the main server using knockd. Figure 3 shows the contents of the port knocking configuration file, knockd. conf.



```
[options]
        UseSyslog

[openSSH]
        sequence    = 10001,10002,10003
        seq_timeout = 5
        command     = /sbin/iptables -I INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn

[closeSSH]
        sequence    = 10003, 10002, 10001
        seq_timeout = 5
        command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 22 -j ACCEPT
        tcpflags    = syn

[openFTP]
        sequence    = 21001, 21002, 21003
        seq_timeout = 5
        command     = /sbin/iptables -I INPUT -s %IP% -p tcp --dport 21 -j ACCEPT
        tcpflags    = syn

[closeFTP]
        sequence    = 21003, 21002, 21001
        seq_timeout = 5
        command     = /sbin/iptables -D INPUT -s %IP% -p tcp --dport 21 -j ACCEPT
        tcpflags    = syn
```

Figure 3. Configuration port knocking

4.2 Intrusion prevention system

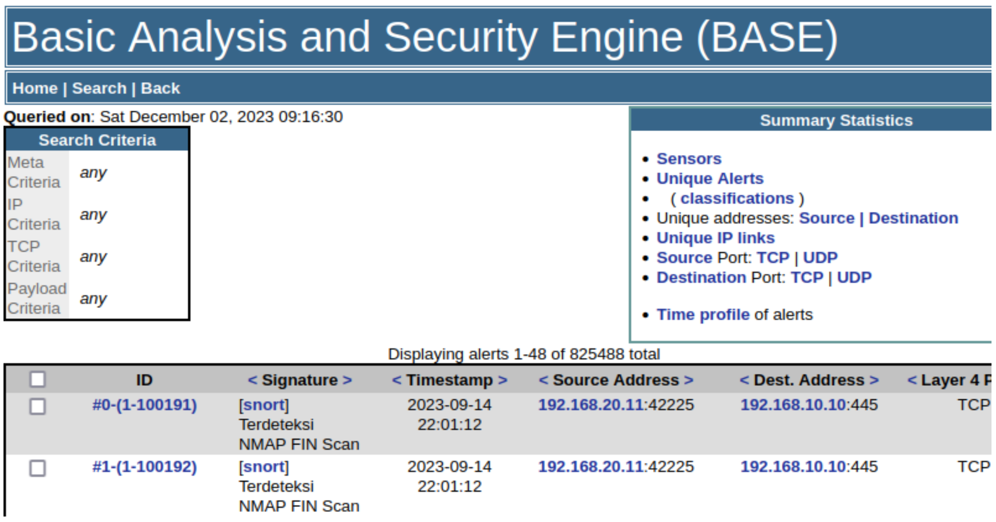
The intrusion prevention system that is used is Snort. Snort can act as an IPS if run in inline mode with data acquisition (DAQ). DAQ is a packet capture scheme and there are several types including NFQ, IPQ, AFPACKET, and IPFW. In this research, what is used is NFQ, where this type uses a queue and configuration rule iptables. In this research, four were applied rules to prevent attacks port scanning, SSH brute force, FTP brute force, and DoS HTTP flood. Figure 4 shows the rules applied.

```
# port scanning
reject tcp any any -> $HOME_NET any(msg:"Terdeteksi NMAP FIN Scan";
flags:F; sid:1000003; rev:1;)

# ssh brute force
reject tcp any any -> $HOME_NET 22 (msg:"Terdeteksi Percobaan SSH Brute
force"; flags:S; flow: to_server; detection_filter: track by_dst, count
3, seconds 10; classtype: attempted-admin; sid:1000004; rev:1;)

# ftp brute force
reject tcp any any -> $HOME_NET 21 (msg:"Terdeteksi Percobaan FTP Brute
force"; flags:S; flow:to_server; detection_filter: track by_dst, count
15, seconds 60; classtype: attempted-admin; sid: 1000005; rev:1;)

# DoS
reject tcp any any -> $HOME_NET 80 (msg:"Terdeteksi DoS HTTP Flood";
flow: to_server, established; flags: PA; detection_filter: track by_dst,
count 1000, seconds 30; classtype:attempted-dos; sid:1000006; rev:1;)
```

Figure 4. Rules snort IPS

4.3 Web Monitoring and Telegram Monitoring System

Web monitoring in this research uses GUI BASE (*Basic Analysis Security Engine*). BASE is a web used to analyze and display attacks detected by Snort. For BASE to work, it requires a tool, namely barnyard2, where this tool functions to store and process binary output from Snort into a MySQL database. Figure 5 shows the display of BASE where there is an attack report containing several information, namely signature, timestamp, source address, destination address, and protocol.


Basic Analysis and Security Engine (BASE)

Home | Search | Back

Queried on: Sat December 02, 2023 09:16:30

Search Criteria	
Meta Criteria	any
IP Criteria	any
TCP Criteria	any
Payload Criteria	any

Summary Statistics
- Sensors
- Unique Alerts
- (classifications)
- Unique addresses: Source | Destination
- Unique IP links
- Source Port: TCP | UDP
- Destination Port: TCP | UDP
- Time profile of alerts

Displaying alerts 1-48 of 825488 total

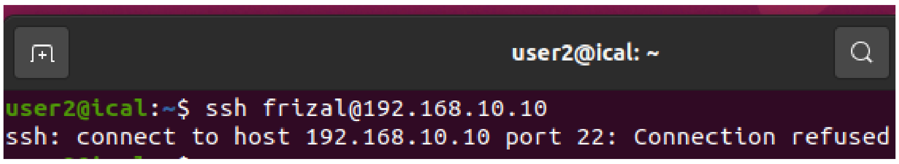
☐	ID	< Signature >	< Timestamp >	< Source Address >	< Dest. Address >	< Layer 4 P
☐	#0-(1-100191)	[snort] Terdeteksi NMAP FIN Scan	2023-09-14 22:01:12	192.168.20.11:42225	192.168.10.10:445	TCP
☐	#1-(1-100192)	[snort] Terdeteksi NMAP FIN Scan	2023-09-14 22:01:12	192.168.20.11:42225	192.168.10.10:445	TCP

Figure 5. Display of BASE

The system coding implemented in Telegram uses the Python programming language. The Telegram monitoring system is a system used to send real-time Snort attack logs to Telegram by sending log data stored in the MySQL database. In this system, there is also a start and stop feature to start and stop Snort and barnyard2 and a report feature to receive attack reports.

4.4 Test Result: Port Knocking testing

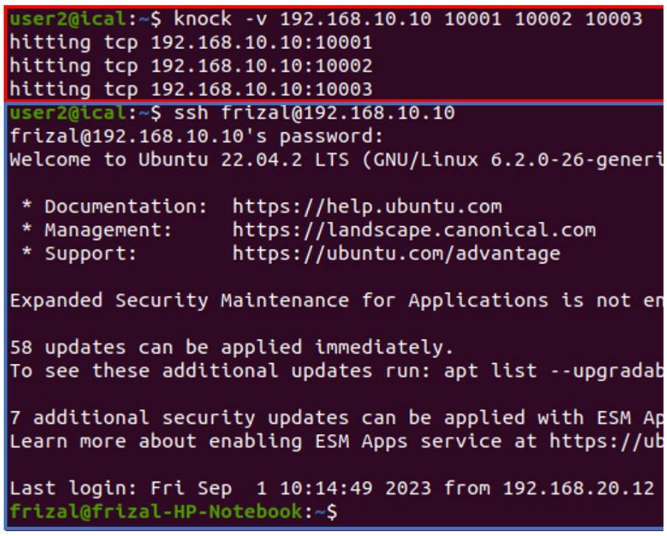
The port knocking test is carried out by the legal user on the main server where the legal user will perform the knocking. Port knocking testing is carried out in three scenarios, namely accessing the port without knocking, accessing the port after knocking, and accessing the port after carrying out the close port mechanism.



```
user2@ical:~$ ssh frizal@192.168.10.10
ssh: connect to host 192.168.10.10 port 22: Connection refused
```

Figure 6. Accessing the port without knocking

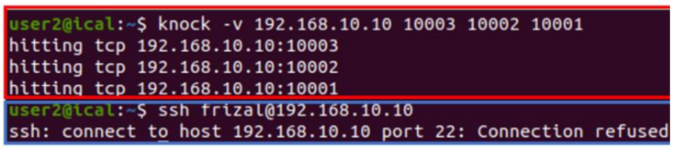
Figure 6 shows the connection to the SSH port was rejected (connection refused) because the port has been closed by iptables.



```
user2@ical:~$ knock -v 192.168.10.10 10001 10002 10003
hitting tcp 192.168.10.10:10001
hitting tcp 192.168.10.10:10002
hitting tcp 192.168.10.10:10003
user2@ical:~$ ssh frizal@192.168.10.10
frizal@192.168.10.10's password:
Welcome to Ubuntu 22.04.2 LTS (GNU/Linux 6.2.0-26-generi

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Expanded Security Maintenance for Applications is not en

58 updates can be applied immediately.
To see these additional updates run: apt list --upgradab

7 additional security updates can be applied with ESM Ap
Learn more about enabling ESM Apps service at https://ub

Last login: Fri Sep  1 10:14:49 2023 from 192.168.20.12
frizal@frizal-HP-Notebook:~$
```

Figure 7. Accessing the port after knocking

Figure 7 shows the results of the port access test after knocking. The red box is the process legal user to knock using the SSH open port knock sequence. The blue box is the legal user connection process to the SSH port where the connection is successful because the knocking process has been carried out so that the port that was previously closed by iptables is reopened.



```
user2@ical:~$ knock -v 192.168.10.10 10003 10002 10001
hitting tcp 192.168.10.10:10003
hitting tcp 192.168.10.10:10002
hitting tcp 192.168.10.10:10001
user2@ical:~$ ssh frizal@192.168.10.10
ssh: connect to host 192.168.10.10 port 22: Connection refused
```

Figure 8. Accessing port after knocking close port

Figure 8 shows the results of the port access test after knocking close the port. The red box is the process for legal users to knock with the knock sequence to close the SSH port. The blue box is the legal user connection process to the SSH port where the connection was refused because the knocking close port process had been carried out so that iptables closed the port again.

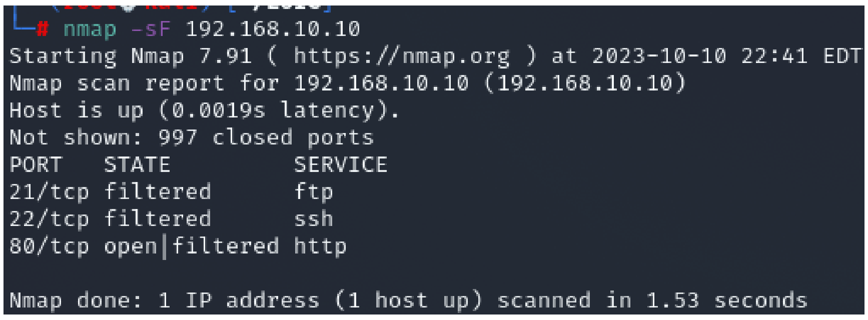
Table 2 is the result of the port-knocking test. The test results show that port knocking has been implemented well because it can differentiate between correct and incorrect knock sequences when the correct knock sequence is used, the port status is open, whereas if the wrong knock sequence is used, the port status remains closed. The results of this test also show that the close port mechanism for closing the port after use is successful in closing the port again.

Table 2. Port Knocking Test Results

Source IP	Knock sequence	Port	Port Status	Information
192.168.20.12	10001 10002 10003	22	Open	Knock sequence open port
192.168.20.12	7000 8000 9000	22	Closed	Wrong Knock sequence
192.168.20.12	10003 10002 10001	22	Closed	Knock sequence close port
192.168.20.12	21001 21002 21003	21	Open	Knock sequence open port
192.168.20.12	20001 20002 20003	21	Closed	Wrong Knock sequence
192.168.20.12	21003 21002 21001	21	Open	Knock sequence close port

4.5 Intrusion Prevention System Testing

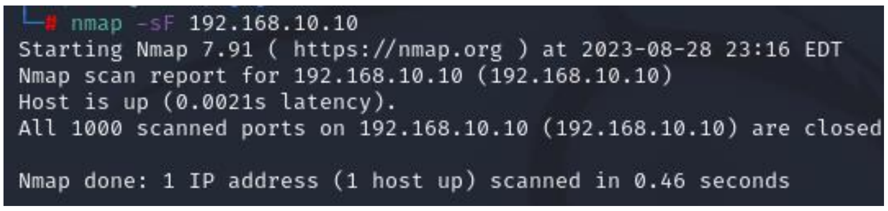
Intrusion prevention system testing is carried out by attackers on the main server. Testing was carried out with 3 types of attacks, namely port scanning, brute force, and DoS. The tool used in port scanning testing is Nmap. Testing was carried out in two conditions, namely before implementing IPS and after implementing IPS.



```
└─# nmap -sF 192.168.10.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-10-10 22:41 EDT
Nmap scan report for 192.168.10.10 (192.168.10.10)
Host is up (0.0019s latency).
Not shown: 997 closed ports
PORT   STATE         SERVICE
21/tcp filtered      ftp
22/tcp filtered      ssh
80/tcp open|filtered http

Nmap done: 1 IP address (1 host up) scanned in 1.53 seconds
```

Figure 9. Port scanning test results before implementing IPS

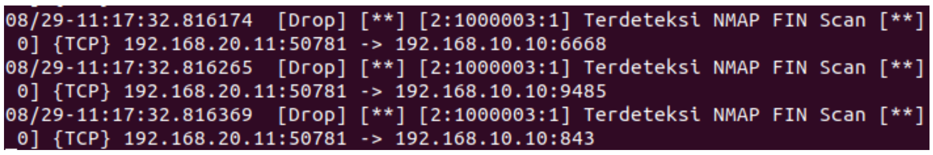
Figure 9 shows the results of a port scanning attack before implementing IPS where the attacker succeeded in obtaining information regarding port 21, port 22, and port 80.



```
└─# nmap -sF 192.168.10.10
Starting Nmap 7.91 ( https://nmap.org ) at 2023-08-28 23:16 EDT
Nmap scan report for 192.168.10.10 (192.168.10.10)
Host is up (0.0021s latency).
All 1000 scanned ports on 192.168.10.10 (192.168.10.10) are closed

Nmap done: 1 IP address (1 host up) scanned in 0.46 seconds
```

Figure 10. Port scanning test results after implementing IPS

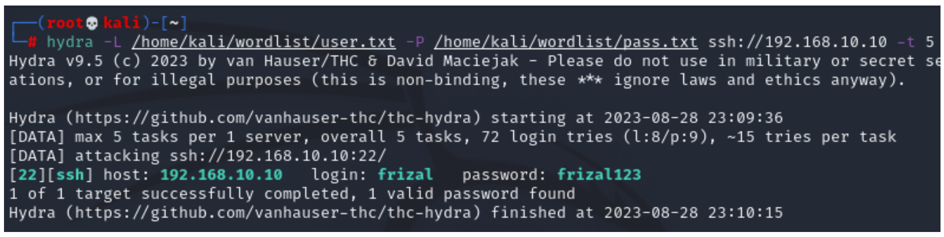
Figure 10 shows the results of a port scanning attack after applying IPS where the attacker does not get any information regarding the port and the attack log appears as seen in Figure 11 which means the attack was successfully detected and there is a Drop description which means the packet was dropped by Snort.



```
08/29-11:17:32.816174  [Drop] [**] [2:1000003:1] Terdeteksi NMAP FIN Scan [**]
 0] {TCP} 192.168.20.11:50781 -> 192.168.10.10:6668
08/29-11:17:32.816265  [Drop] [**] [2:1000003:1] Terdeteksi NMAP FIN Scan [**]
 0] {TCP} 192.168.20.11:50781 -> 192.168.10.10:9485
08/29-11:17:32.816369  [Drop] [**] [2:1000003:1] Terdeteksi NMAP FIN Scan [**]
 0] {TCP} 192.168.20.11:50781 -> 192.168.10.10:843
```

Figure 11. Snort logs after a port scanning attack

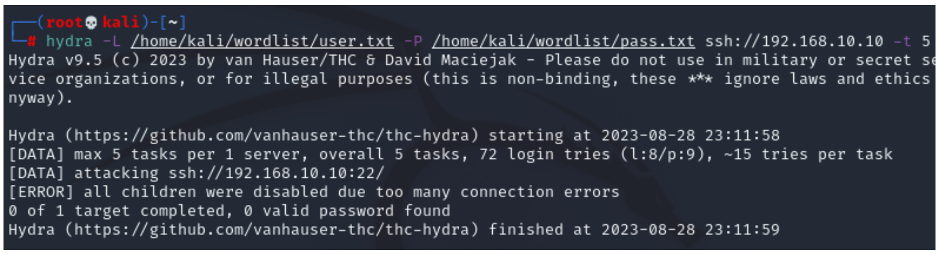
Furthermore, SSH Brute force testing was carried out using the Hydra tool,



```
┌──(root💀kali)-[~]
└─# hydra -L /home/kali/wordlist/user.txt -P /home/kali/wordlist/pass.txt ssh://192.168.10.10 -t 5
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret se
ations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-28 23:09:36
[DATA] max 5 tasks per 1 server, overall 5 tasks, 72 login tries (l:8/p:9), ~15 tries per task
[DATA] attacking ssh://192.168.10.10:22/
[22][ssh] host: 192.168.10.10   login: frizal   password: frizal123
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-08-28 23:10:15
```

Figure 12. SSH brute force test result before implementing port knocking and IPS

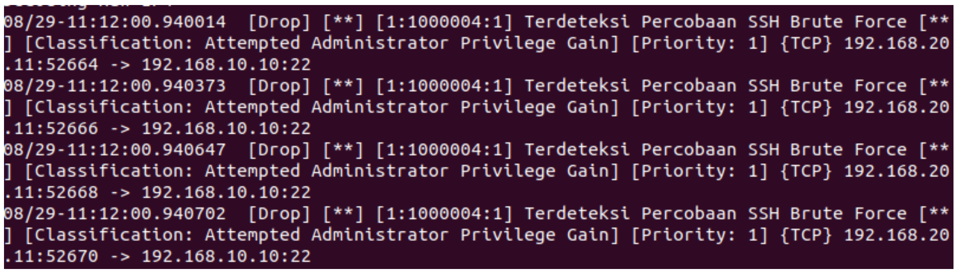
Figure 12 shows the results of a brute force SSH attack with Hydra before implementing port knocking and IPS where the attacker succeeded in getting the username and password to access SSH.



```
┌──(root💀kali)-[~]
└─# hydra -L /home/kali/wordlist/user.txt -P /home/kali/wordlist/pass.txt ssh://192.168.10.10 -t 5
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret se
vice organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics
nyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-28 23:11:58
[DATA] max 5 tasks per 1 server, overall 5 tasks, 72 login tries (l:8/p:9), ~15 tries per task
[DATA] attacking ssh://192.168.10.10:22/
[ERROR] all children were disabled due too many connection errors
0 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-08-28 23:11:59
```

Figure 13. SSH brute force test result after implementing port knocking and IPS

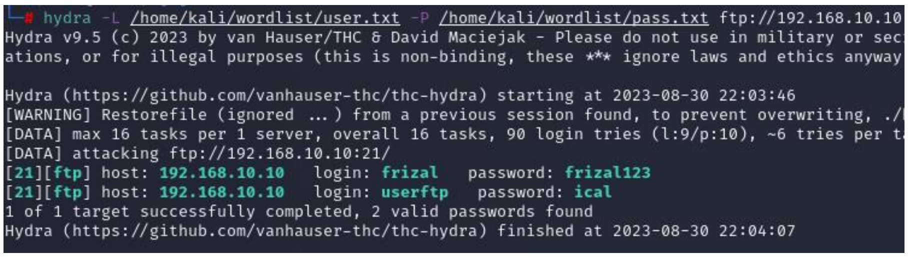
Figure 13 shows the results of a brute force SSH attack with Hydra after applying port knocking and IPS where the attacker failed to carry out the attack and did not succeed in getting any information and a log appears as seen in Figure 14 which shows that the attack was successfully detected and there is a Drop statement which means the packet was dropped by Snort.



```
08/29-11:12:00.940014  [Drop] [**] [1:1000004:1] Terdeteksi Percobaan SSH Brute Force [**
] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.20
.11:52664 -> 192.168.10.10:22
08/29-11:12:00.940373  [Drop] [**] [1:1000004:1] Terdeteksi Percobaan SSH Brute Force [**
] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.20
.11:52666 -> 192.168.10.10:22
08/29-11:12:00.940647  [Drop] [**] [1:1000004:1] Terdeteksi Percobaan SSH Brute Force [**
] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.20
.11:52668 -> 192.168.10.10:22
08/29-11:12:00.940702  [Drop] [**] [1:1000004:1] Terdeteksi Percobaan SSH Brute Force [**
] [Classification: Attempted Administrator Privilege Gain] [Priority: 1] {TCP} 192.168.20
.11:52670 -> 192.168.10.10:22
```

Figure 14. Snort logs after an SSH brute force attack

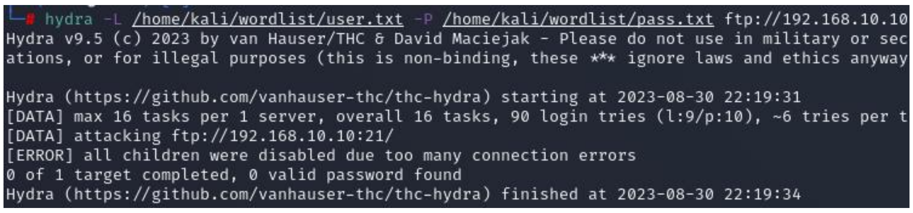
Furthermore, FTP Brute force testing was carried out using the Hydra tool.



```
└─# hydra -L /home/kali/wordlist/user.txt -P /home/kali/wordlist/pass.txt ftp://192.168.10.10
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or sec
ations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-30 22:03:46
[WARNING] Restorefile (ignored ...) from a previous session found, to prevent overwriting, ./
[DATA] max 16 tasks per 1 server, overall 16 tasks, 90 login tries (l:9/p:10), ~6 tries per t
[DATA] attacking ftp://192.168.10.10:21/
[21][ftp] host: 192.168.10.10   login: frizal   password: frizal123
[21][ftp] host: 192.168.10.10   login: userftp   password: ical
1 of 1 target successfully completed, 2 valid passwords found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-08-30 22:04:07
```

Figure 15. FTP brute force test result before implementing port knocking and IPS

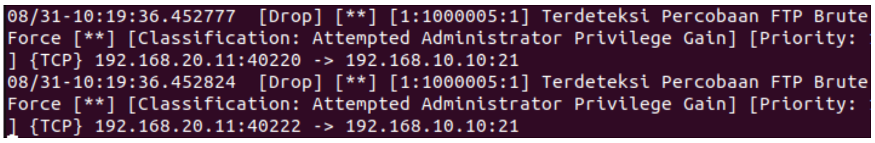
Figure 15 shows the results of a brute force FTP attack with Hydra before implementing port knocking and IPS where the attacker succeeded in getting the username and password to access FTP.



```
└─# hydra -L /home/kali/wordlist/user.txt -P /home/kali/wordlist/pass.txt ftp://192.168.10.10
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or sec
ations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-08-30 22:19:31
[DATA] max 16 tasks per 1 server, overall 16 tasks, 90 login tries (l:9/p:10), ~6 tries per t
[DATA] attacking ftp://192.168.10.10:21/
[ERROR] all children were disabled due too many connection errors
0 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2023-08-30 22:19:34
```

Figure 16. FTP brute force test result after implementing port knocking and IPS

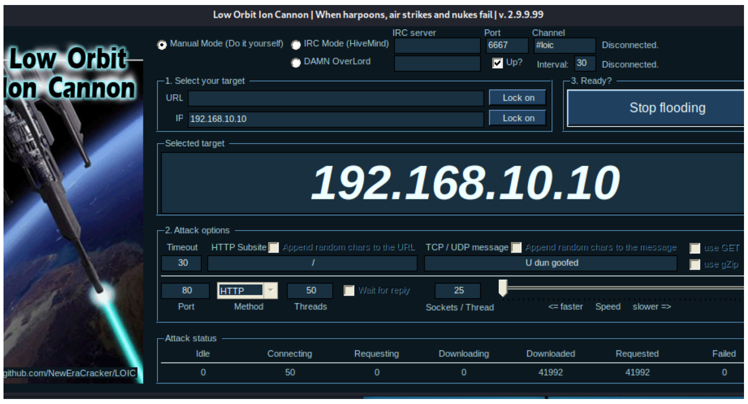
Figure 16 shows the results of a FTP brute force attack with Hydra before port knocking and IPS were applied where the attacker was unable to obtain any information regarding the username and password to access FTP and a log appeared as seen in Figure 17 which shows that the attack was successfully detected and there is a Drop statement which means packet dropped by Snort.



```
08/31-10:19:36.452777  [Drop] [**] [1:1000005:1] Terdeteksi Percobaan FTP Brute
Force [**] [Classification: Attempted Administrator Privilege Gain] [Priority:
] {TCP} 192.168.20.11:40220 -> 192.168.10.10:21
08/31-10:19:36.452824  [Drop] [**] [1:1000005:1] Terdeteksi Percobaan FTP Brute
Force [**] [Classification: Attempted Administrator Privilege Gain] [Priority:
] {TCP} 192.168.20.11:40222 -> 192.168.10.10:21
```

Figure 17. Snort logs after an FTP brute force attack

Moreover, a DoS attack is carried out. The tools used in this DoS testing are LOIC (*Low Orbit Ion Canon*) and slowhttptest. The LOIC attack was carried out using the HTTP method against the target on port 80 with 50 threads as seen in Figure 18.



Low Orbit Ion Cannon | When harpoons, air strikes and nukes fail | v. 2.9.9.99

[✓ Manual Mode (Do it yourself)] IRC Mode (HiveMind) DAMN OverLord
IRC server: Port: 6667 Channel: #loic Disconnected.
Up? ✓ Interval: 30 Disconnected.

1. Select your target
URL:
IP: 192.168.10.10

3. Ready? Stop flooding

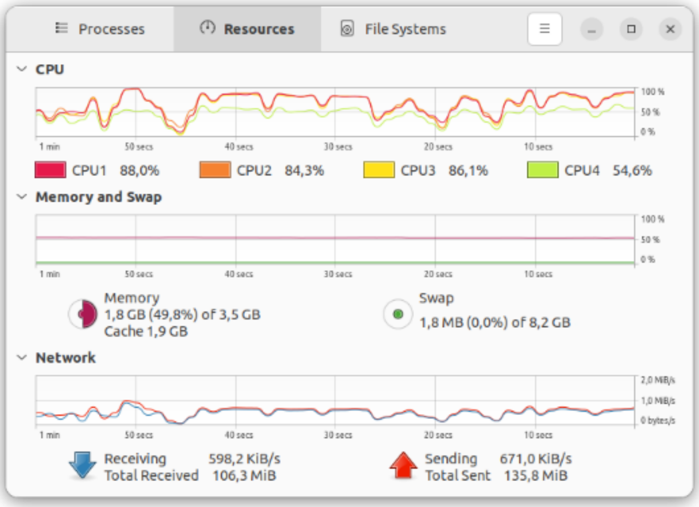
Selected target: 192.168.10.10

2. Attack options
Timeout: 30 HTTP Subsite: / TCP / UDP message: U dun goofed
Port: 80 Method: HTTP Threads: 50 Sockets / Thread: 25

Idle	Connecting	Requesting	Downloading	Downloaded	Requested	Failed
0	50	0	0	41992	41992	0

Low Orbit Ion Cannon
Lock on
Append random chars to the URL
Append random chars to the message
use GET
use gZip
Wait for reply
<= faster Speed slower =>
Attack status
github.com/NewEraCracker/LOIC

Figure 18. LOIC display



CPU1 88,0% CPU2 84,3% CPU3 86,1% CPU4 54,6%

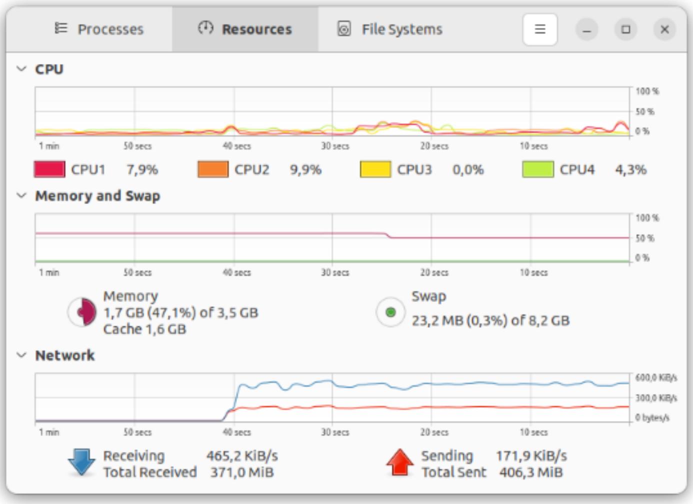
Memory
1,8 GB (49,8%) of 3,5 GB
Cache 1,9 GB

Swap
1,8 MB (0,0%) of 8,2 GB

Receiving 598,2 KiB/s
Total Received 106,3 MiB

Sending 671,0 KiB/s
Total Sent 135,8 MiB

Figure 19. The server monitor system before IPS is applied when a DoS LOIC attack



CPU1 7,9% CPU2 9,9% CPU3 0,0% CPU4 4,3%

Memory
1,7 GB (47,1%) of 3,5 GB
Cache 1,6 GB

Swap
23,2 MB (0,3%) of 8,2 GB

Receiving 465,2 KiB/s
Total Received 371,0 MiB

Sending 171,9 KiB/s
Total Sent 406,3 MiB

Figure 20. The server monitor system after IPS is applied when a DoS LOIC attack

Figure 20 and Figure 21 show the CPU and memory usage monitoring system on the server before and after IPS is applied during a DoS attack with LOIC. When a DoS attack was carried out with LOIC, CPU and memory usage increased, but after IPS was implemented, CPU usage returned to normal. The graph of average CPU and memory usage during DoS with LOIC can be seen in Figure 21.

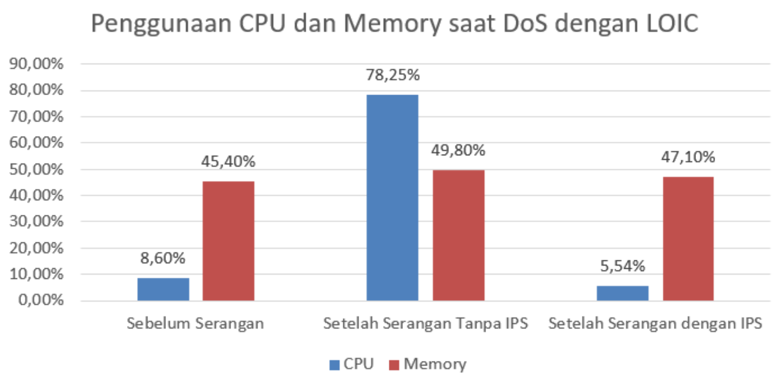
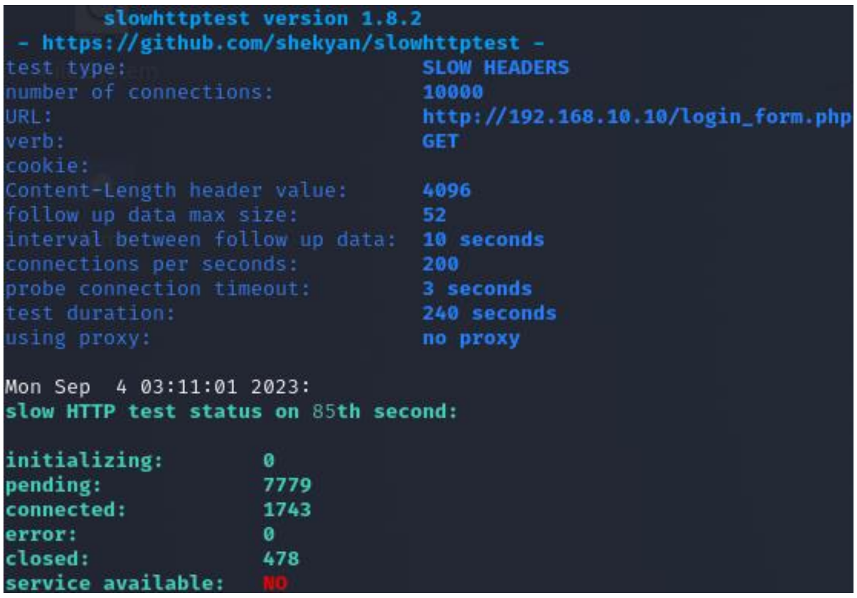


Figure 21. Graph of Server CPU and Memory Usage during DoS LOIC

Figure 21 shows that before the attack, the average CPU usage was 8.6% and memory was 45.4% and there was an increase after the DoS HTTP Flood attack using LOIC with an average CPU usage of 78.25% and memory of 49.8%. However, after IPS was implemented, the average CPU usage remained stable at 5.54% and memory at 47.1%. This shows that the IPS rule applied is effective in preventing DoS attacks with LOIC.



```
        slowhttptest version 1.8.2
 - https://github.com/shekyan/slowhttptest -
test type:                        SLOW HEADERS
number of connections:            10000
URL:                              http://192.168.10.10/login_form.php
verb:                             GET
cookie:
Content-Length header value:      4096
follow up data max size:          52
interval between follow up data:  10 seconds
connections per seconds:          200
probe connection timeout:         3 seconds
test duration:                    240 seconds
using proxy:                      no proxy

Mon Sep  4 03:11:01 2023:
slow HTTP test status on 85th second:

initializing:        0
pending:             7779
connected:           1743
error:               0
closed:              478
service available:   NO
```

Figure 22. Slowhttptest display

Figure 22 is what slowhttptest looks like when the attack is carried out. The attack was carried out using 10,000 connections and there was a statement "service available: no" which indicated that slowhttptest had made the service unusable.

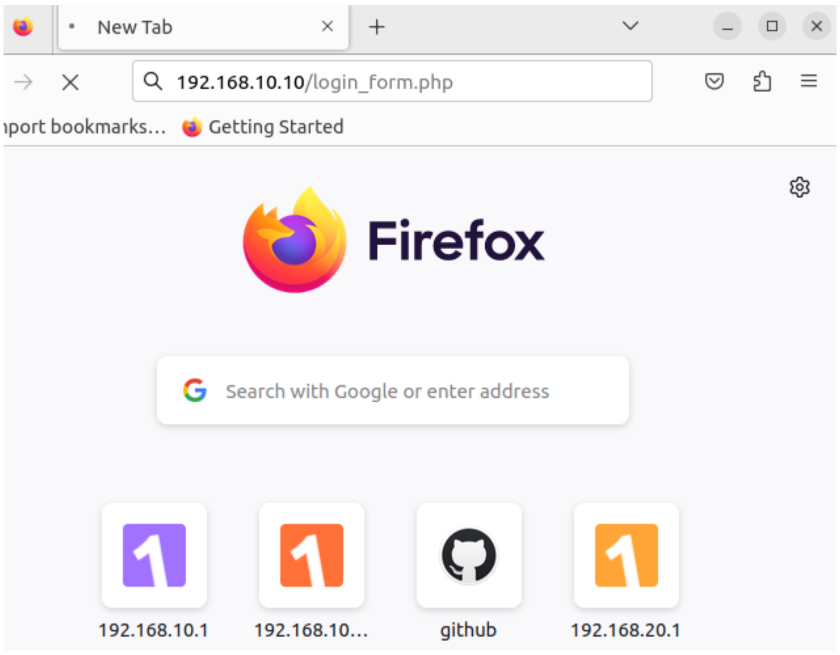


Figure 23. The condition of the web server at the time of the attack before implementing IPS

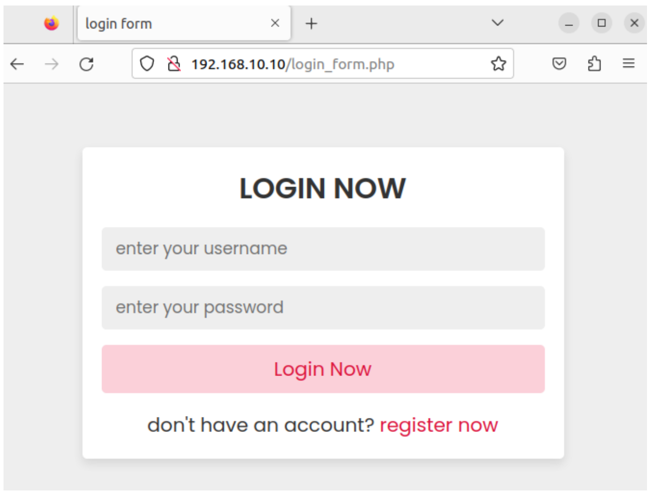


Figure 24. The condition of the web server at the time of the attack after implementing IPS

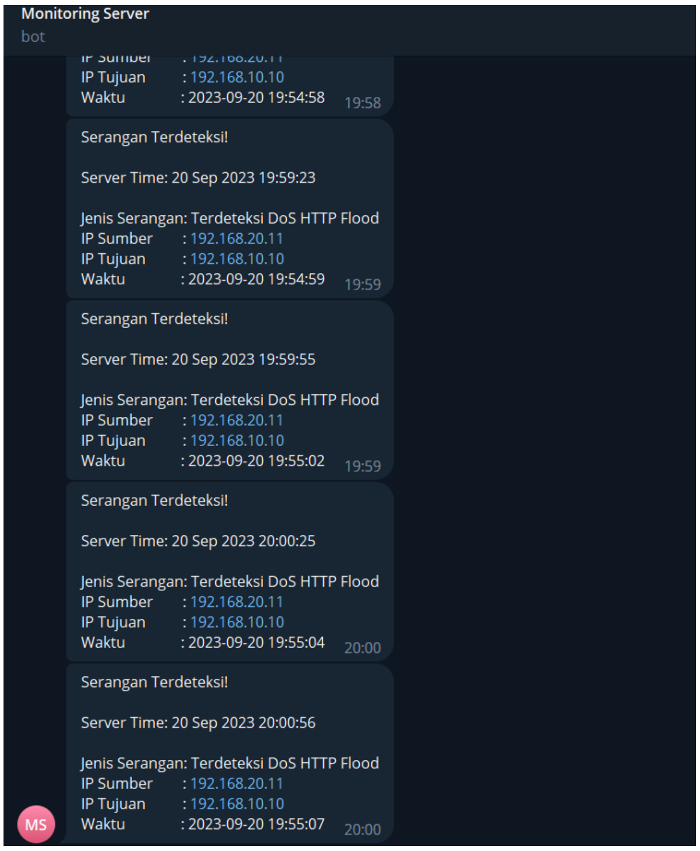
Figure 23 and Figure 24 show the condition of the web during the slowhttptest attack, where this attack causes the web server service to experience buffering or takes time to be accessed and sometimes cannot be accessed. However, when the IPS rule is applied to the attack, the web continues to run in normal conditions and can be accessed by users. This shows that the packet has been dropped by Snort, which means the rules applied have succeeded in preventing DoS attacks with slowhttptest. Moreover, Table 3 is the results of testing the intrusion prevention system where the results of the attacks tested, namely port scanning FIN scan, brute force SSH, brute force FTP, as well as DoS attacks with LOIC and slowhttptest were successfully detected by the system and the action taken by the system was to drop packets related to the attack.

Table 3. Intrusion Prevention System Test Results

Attacker IP	Attack Type	Snort Status	IPS Action	Information
192.168.20.11	Port scanning	Detected	Drop	Attack Package dropped
192.168.20.11	Brute force SSH	Detected	Drop	Attack Package dropped
192.168.20.11	Brute force FTP	Detected	Drop	Attack Package dropped
192.168.20.11	DoS LOIC	Detected	Drop	Attack Package dropped
192.168.20.11	DoS slowhttptest	Detected	Drop	Attack Package dropped

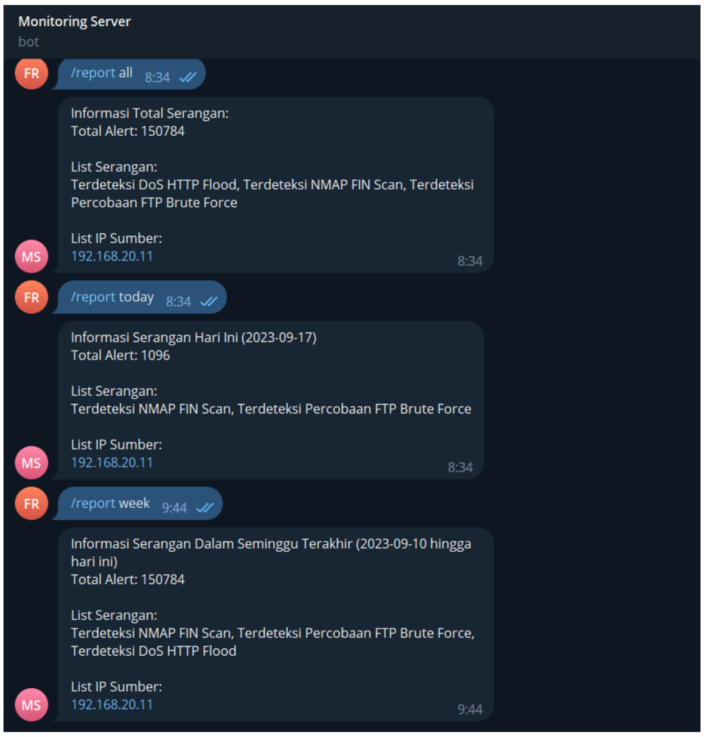
4.6 Telegram Monitoring System Testing

Testing of the Telegram Monitoring System was carried out using two tests. First, Telegram real-time notifications to test whether the attack log was successfully sent or not. Second, black box testing to test whether the features implemented in the Telegram bot that is created function properly and to ensure that the input and output of the features implemented in Telegram are in line with expectations. Moreover, Figure 25 shows the appearance of Telegram when receiving an attack notification. There is some information such as the type of attack, source IP, destination IP, and time of attack.



```
Monitoring Server
bot
IP Sumber    : 192.168.20.11
IP Tujuan    : 192.168.10.10
Waktu        : 2023-09-20 19:54:58   19:58

Serangan Terdeteksi!

Server Time: 20 Sep 2023 19:59:23

Jenis Serangan: Terdeteksi DoS HTTP Flood
IP Sumber    : 192.168.20.11
IP Tujuan    : 192.168.10.10
Waktu        : 2023-09-20 19:54:59   19:59

Serangan Terdeteksi!

Server Time: 20 Sep 2023 19:59:55

Jenis Serangan: Terdeteksi DoS HTTP Flood
IP Sumber    : 192.168.20.11
IP Tujuan    : 192.168.10.10
Waktu        : 2023-09-20 19:55:02   19:59

Serangan Terdeteksi!

Server Time: 20 Sep 2023 20:00:25

Jenis Serangan: Terdeteksi DoS HTTP Flood
IP Sumber    : 192.168.20.11
IP Tujuan    : 192.168.10.10
Waktu        : 2023-09-20 19:55:04   20:00

Serangan Terdeteksi!

Server Time: 20 Sep 2023 20:00:56

Jenis Serangan: Terdeteksi DoS HTTP Flood
IP Sumber    : 192.168.20.11
IP Tujuan    : 192.168.10.10
MS  Waktu    : 2023-09-20 19:55:07   20:00
```

Figure 25. Telegram notification when an attack occurs



```
Monitoring Server
bot
FR  /report all  8:34

Informasi Total Serangan:
Total Alert: 150784

List Serangan:
Terdeteksi DoS HTTP Flood, Terdeteksi NMAP FIN Scan, Terdeteksi
Percobaan FTP Brute Force

List IP Sumber:
MS  192.168.20.11   8:34

FR  /report today  8:34

Informasi Serangan Hari Ini (2023-09-17)
Total Alert: 1096

List Serangan:
Terdeteksi NMAP FIN Scan, Terdeteksi Percobaan FTP Brute Force

List IP Sumber:
MS  192.168.20.11   8:34

FR  /report week  9:44

Informasi Serangan Dalam Seminggu Terakhir (2023-09-10 hingga
hari ini)
Total Alert: 150784

List Serangan:
Terdeteksi NMAP FIN Scan, Terdeteksi Percobaan FTP Brute Force,
Terdeteksi DoS HTTP Flood

List IP Sumber:
MS  192.168.20.11   9:44
```

Figure 26. Report feature on the Telegram monitoring system

Figure 26 shows the appearance of the report feature on the Telegram monitoring system where this feature can be used to receive information regarding total alerts, list of attacks, and list of IP sources of attacks per week, month, per day, and all attacks.

Moreover, Table 4 shows the results of Telegram real-time notification testing where the log results are that all the attacks tested were successfully sent to Telegram and the average time difference between the detection of attacks (detection time) and sending notifications (notification sent time) on Telegram was 2.9 seconds.

Table 4. Telegram Real-time Notification Test Results

Attack Type	IP Attacker	Detection Time	Notification Sent Time	Time Difference
Port scanning FIN Scan	192.168.20.11	14:01:10	14:01:12	2
SSH Brute force	192.168.20.11	14:08:53	14:08:55	2
FTP Brute force	192.168.20.11	14:03:01	14:03:02	1
DoS LOIC	192.168.20.11	14:13:41	14:13:45	4
DoS Slowhttptest	192.168.20.11	14:11:00	14:11:02	2
Port scanning FIN Scan	192.168.20.11	14:52:39	19:52:42	3
SSH Brute force	192.168.20.11	14:53:21	14:53:25	4
FTP Brute force	192.168.20.11	14:53:41	14:53:43	2
DoS LOIC	192.168.20.11	14:56:11	14:56:16	5
DoS Slowhttptest	192.168.20.11	14:54:17	14:54:21	4

Furthermore, Table 5 is a table of black box testing results for the Telegram monitoring system. From this table it can be seen that the Telegram bot has functioned well, this is proven by the correspondence between actual results with expected results in a predetermined test case.

Table 5. Blackbox Testing Results Telegram Monitoring System

Test Case	Test Scenario	Expected Result	Actual Result
Starting the service Snort and barnyard2	Run the /start command in Telegram	Service Snort and barnyard2 run and there is a reply message “Service Snort and Barnyard2 is enabled. Bot ready to monitor logs”	It is successful and the output corresponds expected result
Stopping service Snort and barnyard2	Execute the /stop command in the telegram	Service Snort and barnyard2 are stopped and there is a reply message “Service Snort and Barnyard2 has been turned off. Bot stops monitoring logs”	It is successful and the output corresponds expected result
Report all attacks	Running the command /report all in telegram	The reply message consists of the total number of attack alerts, a list of attacks, and a list of source IPs	It is successful and the output corresponds expected result
Today's attack report	Execute command/report today on telegram	The reply message consists of the number of attack alerts, a list of attacks, and a list of IP sources of attacks on that day	It is successful and the output corresponds expected result

Test Case	Test Scenario	Expected Result	Actual Result
Attack report in the last week	Execute command /report week on telegram	The reply message consists of the number of attack alerts, a list of attacks, and a list of IP sources of attacks in the last week	It is successful and the output corresponds expected result
Attack report in the last 1 month	Execute command/report month on telegram	The reply message consists of the number of attack alerts, a list of attacks, and a list of IP sources of attacks in the last 1 month	It is successful and the output corresponds expected result
Attack reports on specific dates	Execute command/report dd-mm-yy on telegram	The reply message consists of the number of attack alerts, a list of attacks, and a list of attack source IPs on a specific date	It is successful and the output corresponds expected result
List of telegram commands	Execute command/help on telegram	A reply message that displays a list of commands in Telegram	It is successful and the output corresponds expected result

5. Conclusions

A network security system design has been produced with port knocking and an Intrusion prevention system (IPS) on a Linux Server. Port knocking was implemented to prevent unauthorized access on ports 21 (FTP) and 22 (SSH). The IPS used is Snort inline mode with DAQ NFQ that uses queues and configuration iptables. The system test results show port knocking successfully opens and closes the port according to the sequence specified so that the system can prevent unauthorized access to the port. Attack testing results port scanning, SSH brute force, and FTP brute force show that the attacker was unable to obtain port, username, and password information for access rights to the main server. The LOIC DoS attack increased CPU usage on the server to 78.25% and memory to 49.8% but after rule IPS is applied, CPU usage returns to stability, namely 5.4%, and memory to 47.1%. DoS slowhttptest causes the webserver to be difficult or even inaccessible after rule IPS is applied, users can access the web on the server normally even though an attack has been carried out. So it is concluded, that IPS succeeded in blocking all the attacks tested because the attack package was by rules IPS created in Snort so that the packet is detected as a threat and IPS drops the packet. In the Telegram monitoring system, all attack logs tested were successfully sent with an average time difference of 2.9 seconds, and the report, start, and stop features resulted as expected.

Future research is expected to be able to apply dynamic port knocking so that the knock sequence can change so that the attacker has difficulty finding out and adding IPS rules to other attacks such as malware, backdoors, DoS with other tools, and so on so that IPS can prevent many attacks or can change methods. IPS detection becomes anomaly-based. Apart from that, you can also add other features to the Telegram monitoring system.

Acknowledgments: The author would like to thank the Department of Informatics and Computer Engineering Faculty of Engineering Universitas Negeri Makassar, which supports the completion and publication of this research.

Author contributions: All authors are responsible for building Conceptualization, Methodology, analysis, investigation, data curation, writing—original draft preparation, writing—review and editing, visualization, supervision of project administration, funding acquisition, and have read and agreed to the published version of the manuscript.

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

References

1. A. Ramadhani, "Keamanan Informasi," N-JILS, vol. 1, no. 1, pp. 39–51, Jun. 2018, doi: 10.30999/n-jils.v1i1.249. [Crossref]
2. W. W. Purba and R. Efendi, "Perancangan dan analisis sistem keamanan jaringan komputer menggunakan snort," AITI, vol. 17, no. 2, pp. 143–158, Feb. 2021, doi: 10.24246/aiti.v17i2.143-158. [Crossref]
3. I. K. Bayu, M. Yamin, and L. Aksara, "Analisa keamanan jaringan WLAN dengan metode peneration testing (Studi Kasus : Laboratorium sistem informasi dan programming teknik informatika UHO)," SemanTIK : Teknik Informasi, vol. 3, no. 2, pp. 69–78, 2017. [Crossref]
4. A. Saputro, N. Saputro, and H. Wijayanto, "Demilitarized zone and port knocking methods for computer network security," csecurity, vol. 3, no. 2, pp. 22–27, Dec. 2020, doi: 10.14421/csecurity.2020.3.2.2150. [Crossref]
5. R. Suwanto, I. Ruslianto, and M. Diponegoro, "Implementasi intrusion prevention system (IPS) menggunakan snort dan iptable pada monitoring jaringan lokal berbasis website," Jurnal Komputer dan Aplikasi, vol. 07, no. 1, pp. 97–107, 2019. [Crossref]
6. Y. Aulia, "Indonesia Hadapi 700 Juta Serangan Siber Pada 2022," Teknologi. Accessed: Jan. 31, 2023. [Online]. Available: https://purwasuka.hallo.id/politik-hukum/pr-2313800519/indonesia-hadapi-700-juta-serangan-siber-pada-2022
7. H. Muhammad, I. W. A. Arimbawa, and A. H. Jatmika, "Analisis perbandingan sistem autentikasi port knocking dan single packet authorization pada server raspbian," Jurnal Informatika dan Rekayasa Elektronik, vol. 2, no. 1, pp. 28–37, 2019. [Crossref]
8. M. H. Dar and S. Z. Harahap, "Implementasi snort intrusion detection system (IDS) pada sistem jaringan komputer," INFORMATIKA, vol. 6, no. 3, pp. 14–23, Sep. 2018, doi: 10.36987/informatika.v6i3.1619. [Crossref]
9. A. Gupta and L. S. Sharma, "Performance evaluation of snort and Suricata intrusion detection systems on ubuntu server," presented at the Proceedings of ICRIC 2019: Recent Innovations in Computing, Springer International Publishing., 2020, pp. 811–821. [Crossref]
10. J. Al Amien, "Implementasi Keamanan Jaringan Dengan Iptables Sebagai Firewall Menggunakan Metode Port Knocking," Jurnal Fasilkom, vol. 10, no. 2, pp. 159–165, 2020. [Crossref]
11. R. Ernawati, I. Ruslianto, and S. Bahri, "Implementasi metode port knocking pada sistem keamanan server ubuntu virtual berbasis web monitoring," Jurnal Komputer dan Aplikasi, vol. 10, no. 01, pp. 158–169, 2022. [Crossref]
12. D. T. Yuwono, "Analysis performance Intrusion Detection System in detecting cyber-attack on Apache Web Server," IT Journal Research and Development, vol. 6, no. 2, pp. 169–178, 2022. [Crossref]
13. N. Christianto and W. Sulistyo, "Model Pemantauan Keamanan Jaringan Melalui Aplikasi Telegram Dengan Snort," Jurnal Teknik Informatika dan Sistem Informasi, vol. 7, no. 3, pp. 702–714, 2021. [Crossref]
14. A. R. Hakim, J. Rinaldi, and M. Y. B. Setiadji, "Design and Implementation of NIDS Notification System Using WhatsApp and Telegram," presented at the 2020 8th International Conference on Information and Communication Technology (ICoICT), IEEE, 2020, pp. 1–4. [Crossref]
15. R. Agustin, I. Fitri, and N. D. Natashia, "Implementasi metode Intrusion Detection Systems (IDS) dan Intrusion Prevention Systems (IPS) berbasis Snort Server untuk keamanan jaringan LAN," Jurnal Informatika, vol. 18, no. 1, pp. 71–84, 2018. [Crossref]
16. N. A. Santoso, K. B. Affandi, and R. D. Kurniawan, "Implementasi keamanan jaringan menggunakan port knocking," J. Janitra Inform. Sis. Inf., vol. 2, no. 2, pp. 90–95, Oct. 2022, doi: 10.25008/janitra.v2i2.156. [Crossref]
17. R. Muzawi, "Aplikasi pengendalian port dengan utilitas port knocking untuk optimalisasi sistem keamanan jaringan komputer," Sains dan Teknologi Informasi, vol. 2, no. 1, pp. 52–58, 2016. [Crossref]
18. A. S. Fadhlillah, N. B. A. Karna, and A. I. Irawan, "Analisis performansi IDS menggunakan metode deteksi anomaly-based terhadap serangan DoS," eProceedings of Engineering, vol. 6, no. 2, 2019. [Crossref]

19. F. Arsin, M. Yamin, and L. Surimi, “Implementasi security system menggunakan metode IDPS (intrusion detection and prevention system) dengan layanan realtime notification,” vol, vol. 3, no. 2, pp. 39–48, 2017. [Crossref]
20. A. R. Gunawan, N. P. Sastra, and D. M. Wiharta, “Penerapan keamanan jaringan menggunakan sistem snort dan honeypot sebagai pendeteksi dan pencegah malware,” JTE, vol. 20, no. 1, pp. 81–88, Mar. 2021, doi: 10.24843/MITE.2021.v20i01.P09. [https://ojs.unud.ac.id/index.php/jte/article/view/69655Crossref]
21. L. A. Wahsheh and J. Alves-Foss, “Security policy development: Towards a life-cycle and logic-based verification model,” American Journal of Applied Sciences, vol. 5, no. 9, pp. 1117–1126, 2008. [Crossref]