International Journal of Computer and Information System (IJCIS)
Peer Reviewed – International Journal
Vol : Vol. 04, Issue 04, December 2023
e-ISSN : 2745-9659
https://ijcis.net/index.php/ijcis/index

Implementation of IDS Using Snort with Barnyard2 Visualization for Network Monitoring in The Informatics Engineering Computer Lab at Muhammadiyah University Surakarta

1st Baihaqi Fatah 2nd Ihsan Cahyo Utomo
Program Studi Informatika
Universitas Muhammadiyah Surakarta
Jl. Gatak 2, Gatak, Pabelan, Kec. Kartasura, Kab. Sukoharjo, Jawa Tengah 57169
1st baihaqifat@gmail.com 2nd Ihsan.cahyo@ums.ac.id

Abstract - The recent surge in cyberattacks should not be taken lightly, especially by large enterprises with sensitive data. Intrusion Detection Systems (IDS) are becoming a critical component for detecting network anomalies. One such network anomaly detection tool is SNORT, with a BASE (Basic Analysis and Security Engine) frontend for efficient data processing. Acting as a bridge between SNORT and BASE, the author uses barnyard2 as a backend to store logs obtained from SNORT into the database. The implementation methodology used in this research is an experimental approach, where the authors conduct experiments through trial and error to achieve the desired results. This IDS system was tested using two types of attacks, namely DDoS and SQL-Injection. The DDoS attack trial uses tools found in Kali Linux, namely Hping3 with 6 scenarios namely FIN, ACK, RST, UDP, SYN, and ICMP with the results detected in the snort database. SQL-Injection attack test using the DVWA vulnerable website with the result detected in the snort database when the attack is carried out. This proves that the accuracy level of the system reaches close to 100% with the rules given and the penetration testing given.

Keywords *: **IDS, SNORT, Barnyard2, Network Monitoring, Security System.***

I. INTRODUCTION

In this era of digitalization, cyber crime attacks are becoming more prevalent. This makes companies and organizations take proactive steps to improve information system security. Network security is very important to pay attention to especially in the era of technology. Many institutions or organizations are not concerned with security issues. However, when the network is attacked and the system fails, the cost of repairing the system will be high. Therefore, more attention should be paid to investing in network security to prevent damage from attack threats, which are increasingly diverse.[1] One way to improve information system security is to use an Intrusion Detection System (IDS). IDS is a security system that aims to detect attacks on computer networks and provide notifications to administrators to take appropriate action.

Informatics Engineering study program of Universitas Muhammadiyah Surakarta (UMS) has a local network that can be accessed by UMS Informatics Engineering students. High activity in accessing the local network results in increased potential for attacks on users of the local network. So an effort is needed to secure the local network in the UMS Informatics Engineering study program.

Guaranteeing and analyzing network data packets to be monitored from things that endanger network connectivity, a system is needed that can detect and prevent attacks, as well as display and send alerts to network admins when an intrusion occurs. Network admins are responsible for all conditions that occur on the network they manage, especially for the network security system. Even though in general a network is equipped with a firewall, this requires the admin to be on standby to monitor log files regularly.[2]

One of the popular IDSs used is Snort. Snort is open source IDS software that is capable of detecting attacks on computer networks in real-time. Snort allows administrators to monitor network traffic and detect attacks based on predefined patterns.[3]

However, Snort is not able to analyze large network traffic efficiently. Therefore, Barnyard2 is used, an open source application that allows Snort to analyze network traffic efficiently and store the analysis results in a database.

The implementation of Snort as an Intrusion Detection System (IDS) represents a groundbreaking advancement in enhancing server security on Ubuntu. Rigorous testing conducted in this research demonstrates the tangible benefits of deploying Snort IDS, showcasing increased resilience against cyber threats within the network infrastructure. This solution's remarkable versatility is a key innovation, proving its efficacy across various operating systems and highlighting its adaptability to diverse IT environments. The research emphasizes Snort's swift detection and response capabilities, positioning it as a reliable and agile tool for proactive threat management.

Furthermore, the study underscores the user-friendly configuration of Snort IDS, making it accessible and manageable for network administrators, irrespective of their expertise levels.

In addition to its rapid response and user-friendly attributes, the open-source nature of Snort represents a significant innovation. This characteristic not only offers transparency, flexibility, and collaborative opportunities for ongoing improvement but also aligns the solution with the dynamic landscape of cybersecurity challenges. By providing an in-depth understanding of Snort and Barnyard2 in IDS implementation, this research not only addresses network security issues but also delivers innovative solutions. The versatility, quick response capabilities, user-friendly configuration, and open-source nature collectively contribute to a comprehensive enhancement in server security on Ubuntu, offering valuable insights into the realm of Intrusion Detection Systems for computer networks.

II. RESEARCH METHODS

Researchers used the experimental method with the aim of implementing IDS using SNORT and Barnyard2 for network monitoring in the Computer LAB of Informatics Engineering, Universitas Muhammadiyah Surakarta. The experimental method facilitates trial and error techniques, allowing researchers to repeat steps that are less precise.[4] In the implementation stage, the author installs and configures SNORT and Barnyard2 on the prepared system, followed by testing the functionality and performance of the system. [5]

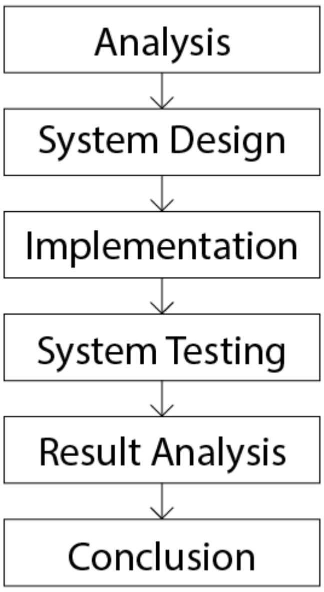

Figure 1. Method workflow

1. Requirement Analysis

there are several hardware and software used for system implementation shown in table 1 and table 2.

Table 1. Hardware Component

No	Device Name	Specification
1	PC Server	• Processor AMD Ryzen 5 2600 • RAM 8 GB • VGA Onboard Ryzen
2	PC Attacker	• Asus A455LF • RAM 8 GB • Processor Intel I3 5005U • VGA Nvidia GeForce 930M

Table 2. Software Component

No	Software Name	Description
1	Virtual Box	used to install Ubuntu Linux
2	Ubuntu Linux	used to install IDS
3	Kali Linux	used for system testing
4	Snort	IDS tools for capture packet
5	Barnyard2	tools for import packer from snort to database postgresql
6	BASE	for visualizing received packets
7	Postgresql	Back-end for snort
8	DVWA	Vilnerable web for sqli attack
9	Mysql	Back-end for DVWA
10	Postfix	to send emails of recorded attacks

2. System Design

Here is the design of some software and hardware.

a. Snort Component

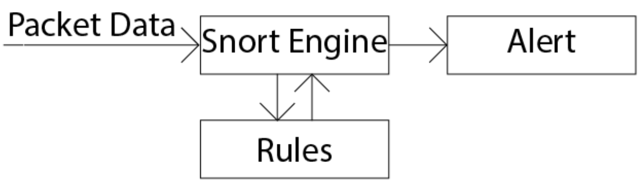

Figure 2. Snort Component

In Figure 2 there are several important components for running Snort, especially in the Snort rules section, in this section it functions to regulate what actions can be captured using Snort. Snort then produces output in the form of a log file and is then processed using the barnyard2 application so that it can be read and easily viewed what types of packets are sent or received by the server.[6]

International Journal of Computer and Information System (IJCIS)
Peer Reviewed – International Journal
Vol : Vol. 04, Issue 04, December 2023
e-ISSN : 2745-9659
https://ijcis.net/index.php/ijcis/index

b. IDS Topology

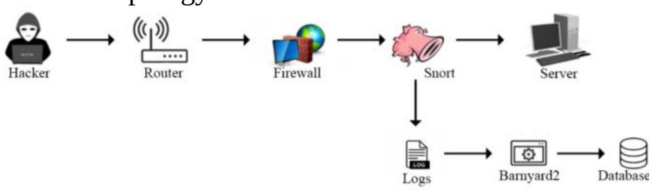

Figure 3. IDS Topology Using Snort

In Figure 3 describes the sequence of detection systems using SNORT which begins with the attacker trying to carry out several attacks on the ubuntu server, before the attack reaches the server, SNORT first detects any unusual attempts that occur on the network. After SNORT successfully detects the attempt, it will then provide output in the form of a log, the log contains anything the attacker does to launch an attack. The form of the log obtained is then visualized using Barnyard2 so that reading the log is easier to understand. Intrusion detection systems with snort are placed in the network to detect intrusions on the monitored system. Therefore, snort must be able to intercept all data from the monitored system, both incoming and outgoing data. IDS snort is connected to the span port of the switch that can capture data traffic from the monitored network.[6]

c. Email Alerting

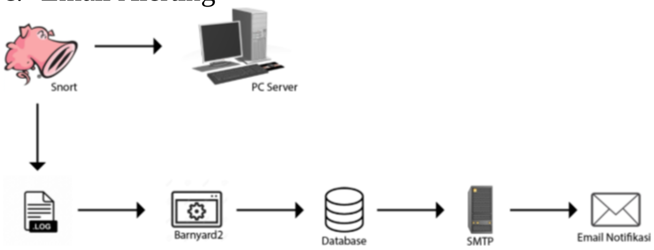

Figure 4. Email Alerting

The author uses email to send notifications when an attack occurs, starting from snort capturing anomalies that occur on the network which are forwarded to the snort database via barnyard2, when the event in the database increases, the python script automatically runs and sends a message to the email that has been prepared.[7]

3. System Testing

System testing is carried out to determine the functionality and performance of the system that has been implemented. The attacks to test the system to be implemented are as follows:

a. DoS or Denial-of-Service

is an attack that aims to make a service or network inaccessible to legitimate users. This attack is usually carried out by sending many unauthorized requests or consuming network resources, so that the service becomes dysfunctional[8].

b. SQL Injection

is an attack technique on web applications that exploits weaknesses in the input data received by the application to execute unwanted SQL commands. In a SQL Injection attack, the attacker inserts SQL code into the input received by the application so that the code is executed by the database without adequate validation or sanitization. The impact of SQL Injection attacks can vary from illegal access to the database, data tampering, to system takeover.[9]

III. RESULT AND ANALYSIS

1. System Implementation

System implementation is done by installing and configuring hardware and software. Based on the device requirements listed in Table 1 and Table 2, the installation process steps can be described as follows.

a. Ubuntu Installation

The author uses Ubuntu 20.04 LTS and installed on a virtualbox on the PC Lab to run the IDS system.

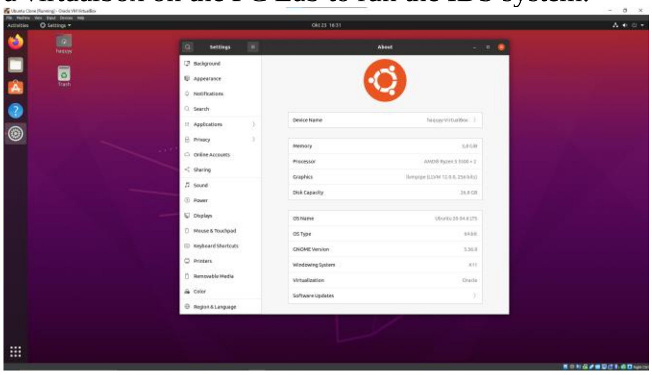

Figure 5. Ubuntu-20.04 LTS Installation on Virtualbox

in Figure 5 explains that the installation of ubuntu on the virtualbox was successful. The selection of the Ubuntu distribution as the IDS installation platform in this study was based on practical considerations including availability, popularity, and ease of use. With its high popularity, Ubuntu offers extensive access to community resources, a comprehensive application repository, and abundant online support. In addition, its intuitive user interface and regular security update support provide stability and security that are important in the context of implementing a security system, such as an IDS, which requires a high level of reliability. In this research, Ubuntu is used as a solid foundation for implementing and running an IDS, making it a suitable choice for achieving the research objectives.[10]

International Journal of Computer and Information System (IJCIS)
Peer Reviewed – International Journal
Vol : Vol. 04, Issue 04, December 2023
e-ISSN : 2745-9659
https://ijcis.net/index.php/ijcis/index

- Snort installation

The author uses snort version 2.9.20 which is obtained from the official snort website, there are several components used for snort installation, among others:

- DAQ 2.0.7 (Data Acquisition)
- libpcap 1.8.1
- libdnet 1.11
- LuaJIT 2.0.5

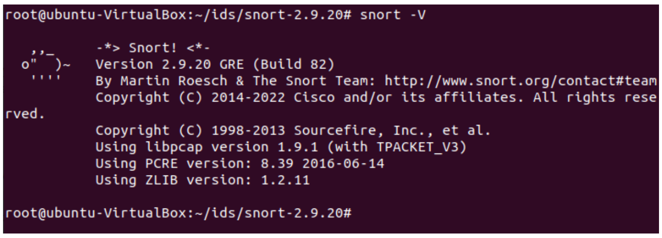

Figure 6. snort-2.9.20 installed successfully

in Figure 6 proves that snort was successfully installed, the next step is to configure the rules so that snort can filter packets entering the server. the rules used are to detect DoS and SQL Injection attacks, the following rules are used by the author.

DoS Rules

```
alert tcp $HOME_NET any -> $HOME_NET any (flags: A; msg:"Possible ACK DoS"; flow: stateless; threshold: type both, track by_dst, count 1000, seconds 3; sid:10001;rev:1;)
alert tcp $HOME_NET any -> $HOME_NET any (flags: S; msg:"Possible SYN DoS"; flow: stateless; threshold: type both, track by_dst, count 1000, seconds 3; sid:10002;rev:1;)
alert tcp $HOME_NET any -> $HOME_NET any (flags: R; msg:"Possible RST DoS"; flow: stateless; threshold: type both, track by_dst, count 1000, seconds 3; sid:10003;rev:1;)
alert tcp $HOME_NET any -> $HOME_NET any (flags: F; msg:"Possible FIN DoS"; flow: stateless; threshold: type both, track by_dst, count 1000, seconds 3; sid:10004;rev:1;)
alert udp $HOME_NET any -> $HOME_NET any (msg:"Possible UDP DoS"; flow: stateless; threshold: type both, track by_dst, count 1000, seconds 3; sid:10005;rev:1;)
alert icmp $HOME_NET any -> $HOME_NET any (msg:"Possible ICMP DoS"; threshold: type both, track by_dst, count 250, seconds 3; sid:10006;rev:1;)
```

SQL Rules

```
alert tcp any any -> any 80 (msg:"SQL Injection Attempt"; content:"%270%27"; sid:10007; rev:1;)
```

- Barnyard2 Installation

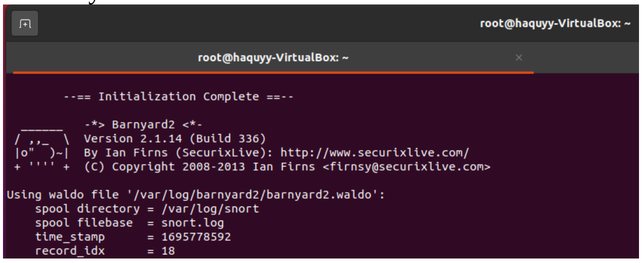

Figure 7. Barnyard2 installation successfully

Figure 7 shows that barnyard2 was successfully installed into the system, using barnyard2 allows log packets received by snort to be sent to the database used to visualize using the BASE front-end.

- BASE Installation

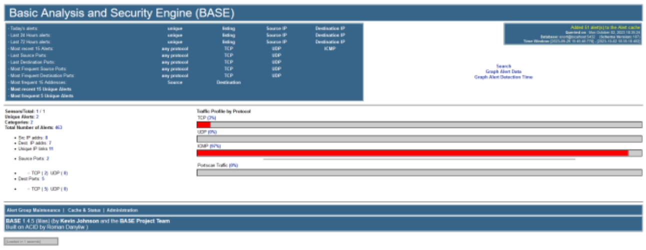

Figure 8. BASE installation successfully

The Basic Analysis and Security Engine (BASE) is a web interface to access stored alerts and provides features to analyze, report, and visualize data from Snort through Barnyard2. Statistics of network activity can be recorded and visualized through graphs as shown below.

- Email Alerting

Email is used to monitor remotely and make it easier to monitor network activity, the author configures email alerting using SMTP configured in the python script.

```
import psycopg2
import smtplib
import time
from email.mime.text import MIMEText
from email.mime.multipart import MIMEMultipart
from email.mime.application import MIMEApplication

db_host = "localhost"
db_name = "snort"
db_user = "snort"
db_password = "snort"

smtp_server = "smtp.gmail.com"
smtp_port = 587
smtp_username = "baihaqifat@gmail.com"
smtp_password = "cfqdabxgfuotnatqurti"
sender_email = "baihaqifat@gmail.com"
recipient_email = "mailalert87@gmail.com"

try:
```

International Journal of Computer and Information System (IJCIS)
Peer Reviewed – International Journal
Vol : Vol. 04, Issue 04, December 2023
e-ISSN : 2745-9659
https://ijcis.net/index.php/ijcis/index

```
    conn = psycopg2.connect(
        host=db_host,
        database=db_name,
        user=db_user,
        password=db_password
    )
except Exception as e:
    print("Error connecting to the database: {e}")
    exit()

cursor = conn.cursor()
query = "SELECT COUNT(*) FROM event LIMIT 10"
jumlah_baris_awal = None
while True:
    cursor.execute(query)
    jumlah_baris = cursor.fetchone()[0]
    rows = cursor.fetchall()
    message = "\n".join([str(row) for row in rows])
    if jumlah_baris_awal is not None and jumlah_baris >
jumlah_baris_awal:
        subject = "SNORT ALERT!!!"
        message = "Adanya Potensi Serangan"
        email_message = MIMEMultipart()
        email_message['From'] = sender_email
        email_message['To'] = recipient_email
        email_message['Subject'] = subject
        email_message.attach(MIMEText(message, 'plain'))
        try:
            server = smtplib.SMTP(smtp_server, smtp_port)
            server.starttls()
            server.login(smtp_username, smtp_password)
            text = email_message.as_string()
            server.sendmail(sender_email, recipient_email,
text)
            server.quit()
            print("Email Notification Send.")
        except Exception as e:
            print("Error sending email: {e}")
    jumlah_baris_awal = jumlah_baris
```

Postfix is one of the core components in the email delivery process. It is responsible for taking emails sent from email clients or other email server applications and delivering them to the intended destination via the SMTP (Simple Mail Transfer Protocol) protocol.[6]

b. Kali Installation

Kali linux is installed on the attacker's pc from the author with specifications as in table 1 section 2.

Figure 9. Kali linux on PC attacker

Figure 9 shows that the installation of kali linux was successful on the attacker's pc, kali linux is used to become an attacker on the system circuit created, kali linux has many tools that are useful for launching attacks and penetration testing, one of the tools used in this experiment is hpin3, where hping3 allows to launch DoS attacks. DoS attacks allow attackers to flood packets to the server in very large quantities so that server performance drops.[11]

2. System Testing

Installation and configuration have been completed at the previous stage, followed by the Snort testing stage whether it can detect intruders or not. In this trial, the modelling used is a client/server network model. Where the server becomes the receiver while the client becomes the sender of packets to the server. From the explanation above, the experiments carried out in this study are simulated in the form of attacks from client computers to server computers. The send email programme will send an email automatically when snort detects an attack.

The first experiment carried out was to detect a DoS attack given from the attacker's pc to the server pc that had IDS installed, the attack was launched using hping3 with the command

a. DoS FIN flag

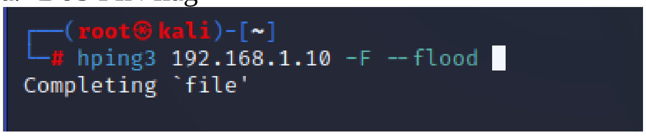

Figure 10. DoS FIN flag using hping3

Figure 11. Email Notification DoS FIN flag

b. DoS ACK & RST flag

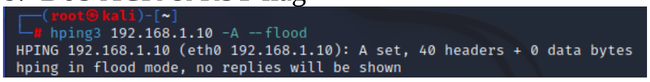

Figure 12. DoS ACK & RST flag using hping3

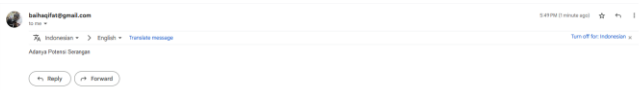

Figure 13. Email Notification DoS ACK & RST flag

c. DoS ICMP flag

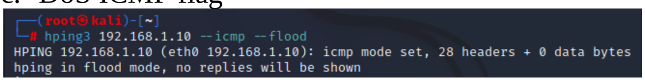

Figure 14. DoS ICMP flag using hping3

Figure 15. Email Notification DoS ICMP flag

International Journal of Computer and Information System (IJCIS)
Peer Reviewed – International Journal
Vol : Vol. 04, Issue 04, December 2023
e-ISSN : 2745-9659
https://ijcis.net/index.php/ijcis/index

d. DoS UDP flag

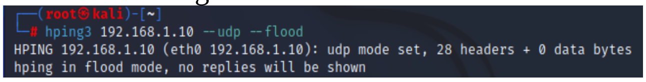

Figure 16. DoS UDP flag using hping3

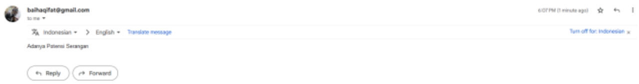

Figure 17. Email Notification DoS UDP flag

e. DoS SYN flag

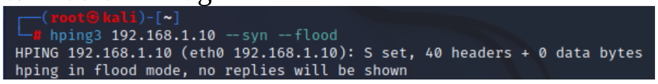

Figure 18. DoS SYN flag using hping3

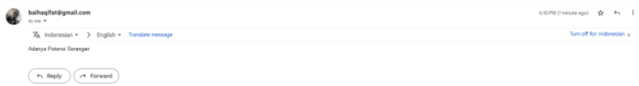

Figure 19. Email Notification DoS SYN flag

DoS attempts are successful with the given rules, email alerting also runs when the attack is launched.

The next experiment is sql injection, the experiment uses a vulnerable web, namely DVWA "Damn Vulnerable Web Application", where the website is provided with many loopholes to be used as a security test on a web server, here is the sql injection used by the author for the attack test.

f. SQL Injection Attack

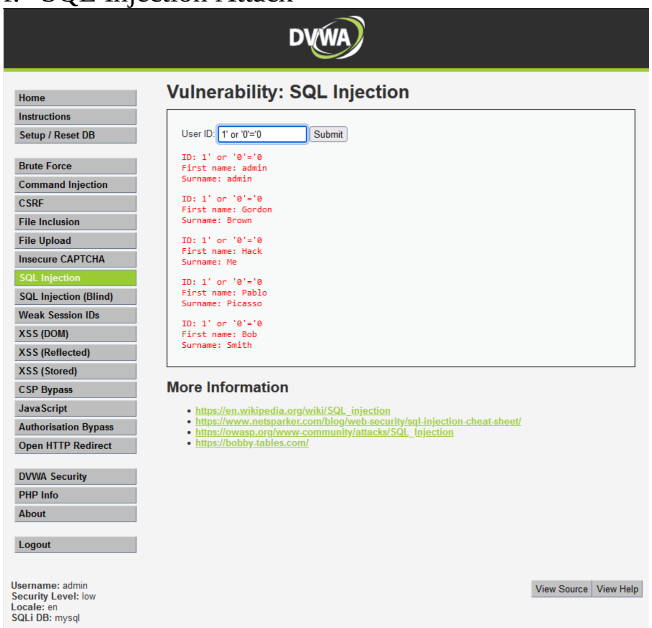

Figure 20. SQL-Injection using DVWA

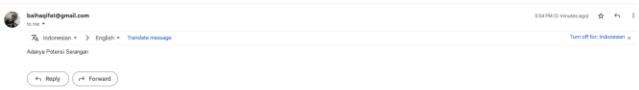

Figure 21. Email Notification SQL-Injection

SQL injection experiment is successfully launched and email alerting runs when sql attack is launched.

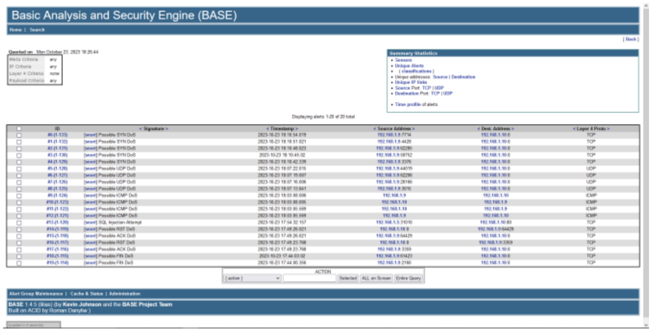

Figure 22. Front-end BASE

The BASE front-end successfully manages the packets recorded by snort according to the given signature.

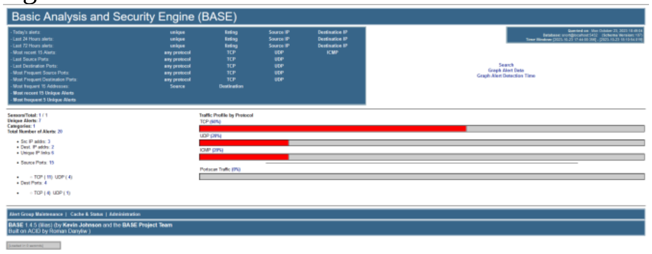

Figure 21. Front-end BASE diagram

In Figure 21, the BASE diagram is used to make it easier to read what types of attacks are coming in, by grouping according to the network protocol of each attack that is launched.

3. Result Analysis

The results of several attacks launched are known that the system test runs as expected, can be grouped with the table below.

Table 3. Result of Test

No	Testing System Scenario	Test Tools	Result	Conclusion
1	DoS FIN flag	Hping3	Detected	Successful
2	DoS ACK flag	Hping3	Detected	Successful
3	DoS RST flag	Hping3	Detected	Successful
4	DoS UDP flag	Hping3	Detected	Successful
5	DoS SYN flag	Hping3	Detected	Successful
6	DoS ICMP flag	Hping3	Detected	Successful
7	SQL Injection	DVWA	Detected	Successful

The system test results in the table above show appropriate performance. The system is able to detect attacks successfully according to predefined rules. The accuracy of the implemented system shows very high success and is almost close to 100%, this shows the effectiveness of using IDS to monitor networks in the campus environment for prevention efforts so that unwanted things do not happen.

THANK-YOU NOTE

Thank you to the IJCIS Team for taking the time to create this template.

REFERENCES

[1] F. Erlacher and F. Dressler, "On High-Speed Flow-Based Intrusion Detection Using Snort-Compatible Signatures," *IEEE Trans. Dependable Secur. Comput.*, vol. 19, no. 1, pp. 495–506, 2022, doi: 10.1109/TDSC.2020.2973992.

[2] W. D. Romadhon, "Implementasi Suricata Idps Untuk Monitoring Jaringan Dengan Visualisasi Elk (Elasticsearch, Logtash, Kibana) Dan Notifikasi Melalui Bot Telegram," 2021, [Online]. Available: http://eprints.ums.ac.id/id/eprint/78171

[3] "Analysis of Digital Forensics in the Implementation of Intrusion Detection using Snort," *FUOYE J. Pure Appl. Sci.*, vol. 7, no. 1, pp. 100–107, 2022, doi: 10.55518/fjpas.ijms6335.

[4] C.-L. Chen and J. L. Lai, "An Experimental Detection of Distributed Denial of Service Attack in CDX 3 Platform Based on Snort," *Sensors*, vol. 23, no. 13, 2023, doi: 10.3390/s23136139.

[5] S. T. Rahman and M. Rabiul Islam, "Experimental Method," in *Principles of Social Research Methodology*, M. R. Islam, N. A. Khan, and R. Baikady, Eds., Singapore: Springer Nature Singapore, 2022, pp. 157–165. doi: 10.1007/978-981-19-5441-2_11.

[6] S. A. Changazi, I. Shafi, K. Saleh, M. H. Islam, S. M. Hussainn, and A. Ali, "Performance enhancement of snort ids through kernel modification," in *2019 8th International Conference on Information and Communication Technologies, ICICT 2019*, 2019, pp. 155–161. doi: 10.1109/ICICT47744.2019.9001286.

[7] A. Goel and A. K. Vasistha, "The Implementation of Rule-Based Early Warning System in Snort Through Email," in *Innovations in Electronics and Communication Engineering*, H. S. Saini, R. K. Singh, V. M. Patel, K. Santhi, and S. V Ranganayakulu, Eds., Singapore: Springer Singapore, 2019, pp. 383–391.

[8] A. Wiranata, N. Karna, A. Irawan, and I. A. Prakoso, "Implementation and Analysis of Network Security in Raspberry Pi against DOS Attack with HIPS Snort," in *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, 2023, pp. 892–896. doi: 10.1109/ICCoSITE57641.2023.10127741.

[9] A. Kapoor, "SQL-Injection Threat Analysis and Evaluation," *SSRN Electron. J.*, 2023, doi: 10.2139/ssrn.4430812.

[10] D. A. P. Putri and A. Rachmawati, "Honeypot cowrie implementation to protect ssh protocol in ubuntu server with visualisation using kippo-graph," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 6, pp. 3200–3207, 2019, doi: 10.30534/ijatcse/2019/86862019.

[11] P. Cisar and R. Pinter, "Some ethical hacking possibilities in Kali Linux environment," *J. Appl. Tech. Educ. Sci. jATES*, vol. 9, no. 4, pp. 129–149, 2019, [Online]. Available: http://doi.org/10.24368/jates.v9i4.139http://jates.org