


# The Analysis Of Honeypot Performance Using Grafana Loki And ELK Stack Visualization

Yahya Alexander Djo Njoera<sup>1</sup>, I Nyoman Buda Hartawan<sup>2\*</sup>, Anak Agung Gede Bagus Ariana<sup>3</sup>,  
Evi Dwi Krisna<sup>4</sup>

<sup>1,2\*,3,4</sup>Institut Bisnis dan Teknologi Indonesia, Denpasar, Indonesia

Article Info	ABSTRACT
<b>Keywords:</b> Honeypot, Grafana Loki, ELK Stack, Data Visualization.	Through the development of current technology, many agencies have implemented technology in the form of computers and servers to improve company operations. However, there are several aspects of threats in the form of cyber attacks that lurk someone when using a computer connected to the internet. One way to prevent these attacks is to use a honeypot application. Honeypot is a system used to deceive hackers who carry out cyber attacks on the system. Every attack received by the honeypot will be recorded in a log. However, reading log data from the honeypot is still difficult to do directly. So an application is needed that can visualize log data from the honeypot. In this study, the visualization applications used are Grafana Loki and ELK Stack. The purpose of this study was to determine the performance of Grafana Loki and ELK Stack in using system resources and in visualizing data. The results of this study indicate that Grafana Loki when processing or not processing honeypot log data uses less system resources compared to ELK Stack. ELK Stack uses 37.8% or 2930 MB of memory, while Grafana Loki only uses 9.7% or 769 MB. Although ELK Stack requires more CPU and memory resources, its data visualization is easier to do compared to Grafana Loki.
This is an open access article under the <a href="#">CC BY-NC</a> license 	<b>Corresponding Author:</b> I Nyoman Buda Hartawan Institut Bisnis dan Teknologi Indonesia <a href="mailto:buda.hartawan@instiki.ac.id">buda.hartawan@instiki.ac.id</a>

## INTRODUCTION

In today's technological developments, many companies and agencies have implemented technology to improve the operation or smoothness of the company's business. In a company, at least one computer is needed that can be connected to the internet or intranet network (Widodo, 2015) (Widodo, 2015). If a company has a large business operation, then the company needs a server that can support each of these operations. (Eka et al., 2010) (Willy Andrian & Dedy Prasetya Kristiadi, 2022) (Ariata, 2023). When connected to the internet, users can search for information widely according to their company's needs. However, there are several aspects of threats that lurk when someone searches for information on the internet. Several aspects of threats such as corporate identity theft, network scanning attacks, corporate database authentication that is easily attacked by hackers without user knowledge, and attacks on servers that cause the company's system services to go down (Annovazzi, 2022) (Kaspersky, 2022) (Sulaksono & Suharyanto, 2020). Based on data from the Head of

the National Cyber and Crypto Agency (BSSN), the total traffic anomalies throughout 2022 reached 976,429,996 times. The types of attacks include malware 56.84 percent, data leaks 14.75 percent, and trojan activity 10.90 percent (CNN Indonesia, 2023). This aspect of the threat will also have a detrimental impact on the company because every company wants their confidential data not to be lost and can only be accessed by certain people (Gunawan et al., 2021).

One way to secure a computer network is to install or configure a network security device, namely a firewall (Sulaksono & Suharyanto, 2020). A firewall is a network security tool that functions to check all data traffic entering or leaving a computer network. So the main purpose of a firewall is to limit access to data that is considered dangerous or not permitted (Klusaitė, 2022). However, sometimes to get better features in securing the network, a firewall requires a license with a subscription system that needs to be paid periodically (Sulaksono & Suharyanto, 2020). Therefore, an additional security system is needed that is able to work with the operating system's built-in firewall. This system is Honeypot. Honeypot is a system that is built very similar to the company's original system so that it can trick hackers as if they were attacking the company's original system. The main purpose of the honeypot system is to trick hackers with a fake system and get information about attacks carried out by the perpetrators and obtain hacker information (Wastumirad & Darmawan, 2021) (Nurrahman, 2019). Information obtained from the honeypot system is stored in the form of log data. The information stored in the log data is in the form of the perpetrator's IP address, the port of the attacked system, the service attacked by the perpetrator, and the time of the attack from the perpetrator (Wibawa et al., 2020).

The implementation of the honeypot system also requires several supporting systems. In this study, the supporting systems that will be used are Cowrie and Dionaea. Cowrie is a type of honeypot designed to imitate the SSH (Secure Shell) service. Cowrie also has weak account information so that hackers can be trapped on the fake server that has been created (Tati Ernawati & Fikri Faiz Fadhlur Rachmat, 2021). Meanwhile, Dionaea is a type of honeypot used to imitate several system services, namely: FTP (File Transfer Protocol), TFTP (Trivial File Transfer Protocol), SMB (Server Message Block), and other file sharing services (Wastumirad & Darmawan, 2021). Dionaea can also detect malware attacks (Rachman et al., 2019). Every attack log data that has been received by the honeypot also needs to be analyzed by an administrator. However, every information in the log data is very difficult to read directly (Syaifuddin et al., 2022). Therefore, to make it easier to read log data information, an application is needed that is useful for visualizing honeypot log data in graphical form or clearly in writing. There are several applications that can be used to read log data information, such as Modern Honey Network, ELK Stack, and Grafana Loki.

The applications that will be used in this study are Grafana Loki and ELK Stack. Grafana Loki and ELK Stack are popular applications used to analyze log data. Both of these applications are popular because they have features for customizing dashboards that allow users to create data displays according to their needs (Yigal, 2018) (Anand, 2024). Both of these applications are open source, allowing users to see how these two applications work (Yigal, 2018) (Tilwani, 2023). Through the open source nature, application users can ensure

that their data is not stored and used for unwanted purposes (Cemazar, 2022). In addition, statistics from Github show that Kibana (part of the ELK Stack) has more than 17,000 code commits, while Grafana has around 14,000 code commits. The number of code commits can reflect the level of activity and contribution of the user community of both applications (Yigal, 2018). This makes it easier for users to exchange ideas when facing several technical problems. The final result of this study is a comparison of the performance between the visualization methods of Grafana Loki and ELK Stack on honeypots.

## METHODS

In this research, the experimental method is used to test honeypot performance using visualization tools like Grafana Loki and the ELK Stack. The steps include designing an experiment by creating cyber attack scenarios on the honeypot, implementing the honeypot and visualization tools, collecting data on attack activities and honeypot responses, and analyzing the data using Grafana Loki and the ELK Stack to compare honeypot performance.

A virtual server will be used to run the operating system along with the honeypot and visualization applications Grafana Loki and the ELK Stack. A laptop will be used to access and simulate cyber attacks on the virtual server over the intranet. Below are the specifications of the virtual server and the laptop that will be used to design and access the honeypot and its visualization applications.

**Table 1.** Virtual server specification

Component	Spesification			
	Honeypot Server	Monitoring Server	Client	Attacker
CPU	2 Core	4 Core	AMD Ryzen 7-4800H	Intel Core i3-3217U
RAM	4 GB RAM	8 GB RAM	16 GB	4 GB RAM
Penyimpanan	128 GB HDD	128 GB HDD	512 GB	512 GB HDD

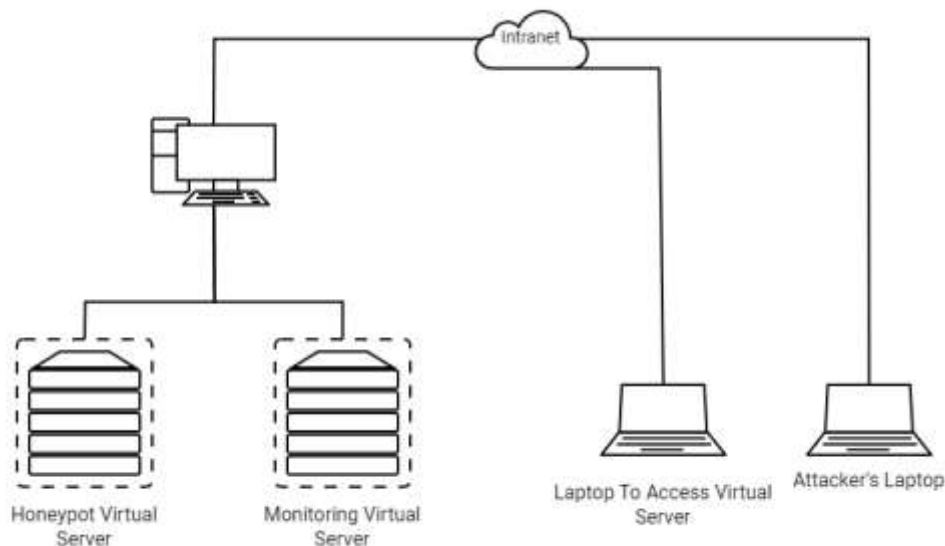
The following are the software used in this research along with their functions. Grafana Loki is used to collect and manage logs of attack activities and honeypot responses. Additionally, the ELK Stack is used to visualize and analyze the collected data to compare honeypot performance.

**Table 2.** List of software

Software	Function
<i>Honeypot Cowrie</i>	Detecting cyber attacks in the SSH (Secure Shell) area
<i>Honeypot Dionaea</i>	Detecting cyber attacks such as DoS (Denial of Service) and port scanning
<i>ELK Stack</i>	Visualizing log data from Cowrie and Dionaea
<i>Grafana Loki</i>	Visualizing log data from Cowrie and Dionaea
<i>Glance</i>	Displaying CPU and memory performance information from the system
<i>Putty</i>	Remote access via SSH (Secure Shell) to the virtual server

Software	Function
<i>Filebeat</i>	Used to send each attack data to the ELK Stack
<i>Promtail</i>	Used to send each attack data to Grafana Loki
<i>Virtual Box</i>	Used to create a virtual server
<i>Ubuntu</i>	Operating system for honeypots and visualization applications
<i>Nmap</i>	Performing port scanning attacks on the system
<i>Hydra</i>	Performing brute force attacks on the system
<i>LOIC</i>	Performing DoS (Denial of Service) attacks on the system

The network topology used in this research can be seen in Figure 1. This study utilizes one computer, one virtual honeypot server, one virtual monitoring server, one laptop as an access medium to the virtual honeypot and monitoring servers, and one laptop used to conduct cyber attacks on the virtual honeypot server. All devices used in this research are connected to the intranet. The virtual honeypot server will integrate with the virtual monitoring server so that the attack data received by the honeypot can be immediately sent to the virtual monitoring server for visualization.

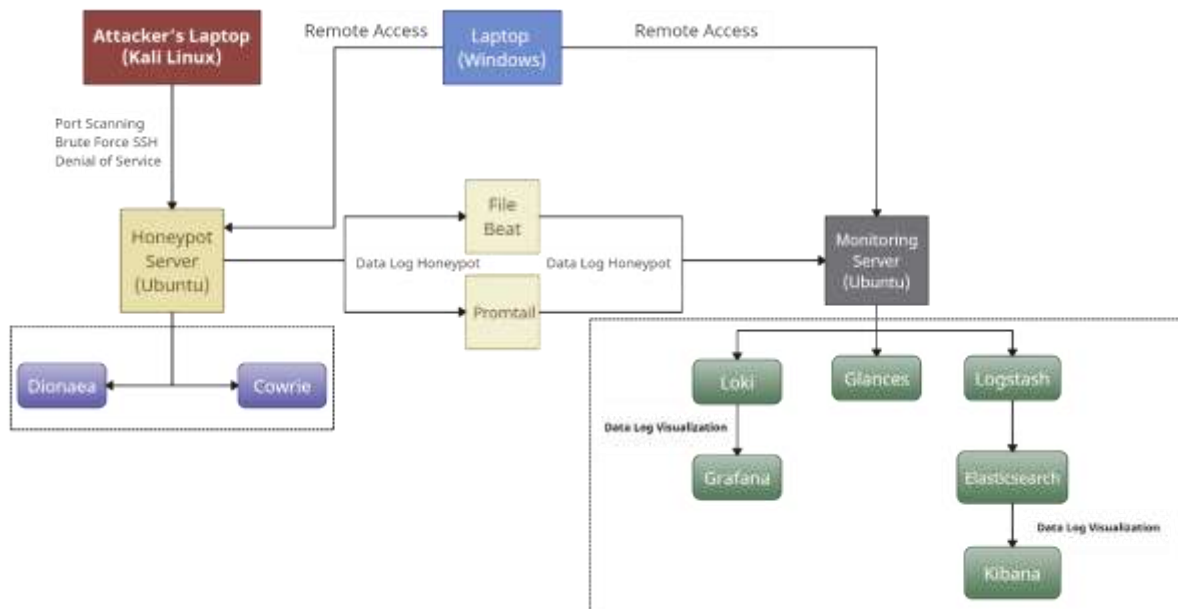


**Figure 1.** Network Topology

1. The block diagram used as the foundation for the system design in this research can be seen in Figure 2. Below is an explanation of the block diagram used in this study:
2. Honeypot Cowrie and Dionaea are designed on the same virtual server. This virtual server uses the Ubuntu operating system. Each attack data received by the honeypot server will be sent via the Promtail and Filebeat applications to the monitoring server for visualization.
3. The visualization applications Grafana Loki and ELK Stack will be designed on a different virtual server from the honeypot server and will operate as the monitoring server. This virtual server will also use Ubuntu as its operating system.
4. This research will use a laptop as a medium for configuring the honeypot server and the monitoring server. This laptop can also be used to access the dashboard to view the

visualization results from the Grafana Loki and ELK Stack applications through a web browser. Additionally, the laptop can be used to conduct attacks on the honeypot server. Three types of attacks will be conducted: port scanning, brute force SSH, and DoS (Denial of Service).

5. After the attack data is successfully obtained by the honeypot server, integration is carried out between the honeypot server and the monitoring server using Promtail agents for Grafana Loki and Filebeat for ELK Stack. Through these agents, the collected attack data is directed to the Grafana Loki and ELK Stack platforms on the monitoring server. Once the data reaches the monitoring server, Grafana Loki and ELK Stack will process this data. The processed data is visualized and displayed through the respective monitoring platform dashboards. Administrators can easily monitor and analyze detected attacks through a web browser interface. During the data processing of honeypot attack logs from each monitoring platform, the performance of each monitoring platform will be recorded through the Glances application.



**Figure 2.** Diagram Block

In testing attacks on the honeypot server, a laptop with Kali Linux is used. Three types of attacks will be tested: port scanning, brute force SSH, and Denial of Service (DoS).

- 1) Port Scanning Testing Scenario: This scenario aims to find open ports on the honeypot server. The focus is on testing the Dionaea honeypot system. The application used for port scanning testing is Nmap. The goal is to ensure that the Dionaea honeypot system can detect port scanning attacks and store the attack data in log, SQLite, or JSON formats. The port scanning test scenario involves:
  - a. Running the Dionaea honeypot on the server.
  - b. Simulating a port scanning attack using the Nmap application from the attacking laptop. The test can be run with the command `sudo nmap -sS -p 1-1000 -sC`

- 10.101.104.146 in Nmap. The command components include sudo for running the command with admin authority, nmap for initiating the port scan, -sS for SYN scan protocol, -p 1-1000 for scanning ports 1 to 1000 on the target, -sC for detecting services and vulnerabilities, and 10.101.104.146 as the target IP address.
- c. If the Dionaea honeypot operates correctly, attack data should be stored in log, SQLite, or JSON formats.
  - d. The port scanning attack testing scenario is complete.
- 2) Brute Force SSH Testing Scenario: This scenario aims to gain unauthorized access to a computer or server by trying various username and password combinations until access is granted. The focus is on testing the Cowrie honeypot system. The application used for brute force SSH testing is Hydra. The goal is to ensure that the Cowrie honeypot system can detect brute force SSH attacks and store the attack data in log and JSON formats. The brute force SSH test scenario involves:
- a. Running the Cowrie honeypot on the server.
  - b. Simulating a brute force SSH attack using the Hydra application from the attacking laptop. The test can be run with the command `hydra -L /home/pot/Desktop/user32.txt -P /home/pot/Desktop/pass32.txt 10.101.104.146 ssh` in Hydra.
  - c. If the Cowrie honeypot operates correctly, attack data should be stored in log and JSON formats.
  - d. The brute force SSH attack testing scenario is complete.
- 3) Denial of Service (DoS) Testing Scenario: This scenario aims to disable a computer or server by overwhelming it with a large volume of data traffic continuously, causing the computer or server to work harder to handle the incoming data traffic. The focus is on testing the Dionaea honeypot system. The application used for DoS attack testing is LOIC. The goal is to ensure that the Dionaea honeypot system can detect DoS attacks and store the attack data in log, SQLite, or JSON formats. The DoS test scenario involves:
- a. Running the Dionaea honeypot on the server.
  - b. Simulating a DoS attack using the LOIC application from the attacking laptop. The test can be configured as shown in Figure 3. The configuration includes setting the target IP address in the "IP" field (10.101.104.146), "Lock on" to lock the target IP, setting the port to "42", "Method" to "TCP", and "Threads" to "150". After completing the configuration, the DoS attack can be executed by pressing the "IMMA CHARGIN MAH LAZER" button.
  - c. If the Dionaea honeypot operates correctly, attack data should be stored in log, SQLite, or JSON formats.
  - d. The DoS attack testing scenario is complete.



Figure 3. DoS attack configuration

## RESULTS AND DISCUSSION

### Results of Testing Monitoring Server Resources - Cowrie

1. Based on the brute force SSH attack experiments on the Cowrie honeypot, the following are the results of CPU% and Memory% usage from Grafana Loki and ELK Stack when receiving “json” log data from the Cowrie honeypot. CPU% and Memory% usage were recorded using the Glances application. The recorded values represent the highest resource usage between CPU% and Memory% for each experiment. According to Table 3, the summary of the testing is as follows:
2. The highest CPU usage for Grafana Loki occurred in the tenth experiment, at 29.0%. For Grafana Loki’s memory usage, the highest value was in the third experiment, at 10.3% or 817 MB.
3. The highest CPU usage for ELK Stack occurred in the first experiment, at 94.2%. For ELK Stack’s memory usage, the highest value was in the ninth experiment, at 36.9% or 2.86 GB.
4. From the results in Table 3, it is evident that the ELK Stack components use more server resources compared to Grafana Loki when processing brute force SSH attack data. Across several experiments, the average CPU% usage of Grafana Loki was relatively lower compared to ELK Stack. In terms of memory usage, Grafana Loki used less memory compared to ELK Stack. The highest CPU usage was observed in the ELK Stack component, with a value of 94.2% in the first experiment, and the highest memory usage was also observed in the ELK Stack component, with a value of 36.9% or 2.86 GB.

**Table 3.** Resource Usage During Brute Force SSH Attack

Testing	Grafana Loki		ELK Stack	
	CPU%	MEM%	CPU%	MEM%
1	12.1	9.0 (717 MB)	94.2	34.5 (2.68 GB)
2	21.3	9.9 (787 MB)	31.3	35.6 (2.76 GB)
3	23.4	10.3 (817 MB)	26.4	36.0 (2.79 GB)

Testing	Grafana Loki		ELK Stack	
	CPU%	MEM%	CPU%	MEM%
4	11.3	9.8 (783 MB)	22.7	36.3 (2.82 GB)
5	14.3	9.6 (765 MB)	21.9	36.4 (2.83 GB)
6	17.3	9.8 (778 MB)	13.5	36.5 (2.83 GB)
7	21.3	9.8 (777 MB)	39.0	36.8 (2.86 GB)
8	24.2	9.8 (775 MB)	15.8	36.8 (2.86 GB)
9	26.9	10.1 (804 MB)	24.3	36.9 (2.86 GB)
10	29.0	10.2 (807 MB)	20.0	36.9 (2.86 GB)

### Results of Testing Monitoring Server Resources - *Dionaea*

Based on the port scanning attack experiments on the Dionaea honeypot, the following are the results of CPU% and Memory% usage from Grafana Loki and ELK Stack when receiving ".json" log data from the Dionaea honeypot. CPU% and Memory% usage were recorded using the Glances application. The recorded values represent the highest resource usage between CPU% and Memory% for each experiment. According to Table 4, the summary of the testing is as follows:

1. The highest CPU usage for Grafana Loki occurred in the eighth experiment, at 6.4%. For Grafana Loki's memory usage, the highest value was in the fifth experiment, at 9.8% or 778 MB.
2. The highest CPU usage for ELK Stack occurred in the second experiment, at 21.1%. For ELK Stack's memory usage, the highest value was in the ninth experiment, at 37.8% or 2.93 GB.
3. From the results in Table 4, it is evident that the ELK Stack components use more server resources compared to Grafana Loki when processing port scanning attack data. Across all experiments, the average CPU% and memory usage of Grafana Loki were relatively lower compared to ELK Stack. The highest CPU usage was observed in the ELK Stack component, with a value of 21.1% in the second experiment, and the highest memory usage was also observed in the ELK Stack component, with a value of 37.8% or 2.93 GB.

**Table 4.** Resource Usage During Port Scanning Attack

Testing	Grafana Loki		ELK Stack	
	CPU%	MEM%	CPU%	MEM%
1	4.5	9.0 (712 MB)	15.7	37.3 (2.89 GB)
2	4.2	9.3 (742 MB)	21.1	37.4 (2.90 GB)
3	5.0	9.4 (743 MB)	12.5	37.5 (2.91 GB)
4	4.8	9.6 (759 MB)	12.2	37.5 (2.91 GB)
5	4.5	9.8 (778 MB)	12.3	37.6 (2.92 GB)
6	4.4	9.8 (777 MB)	11.2	37.6 (2.92 GB)
7	5.7	9.5 (753 MB)	11.0	37.7 (2.93 GB)
8	6.4	9.8 (776 MB)	15.2	37.7 (2.93 GB)
9	6.4	9.6 (763 MB)	11.3	37.8 (2.93 GB)
10	5.7	9.7 (769 MB)	15.5	37.8 (2.93 GB)

### Resource Comparison of Grafana Loki and ELK Stack

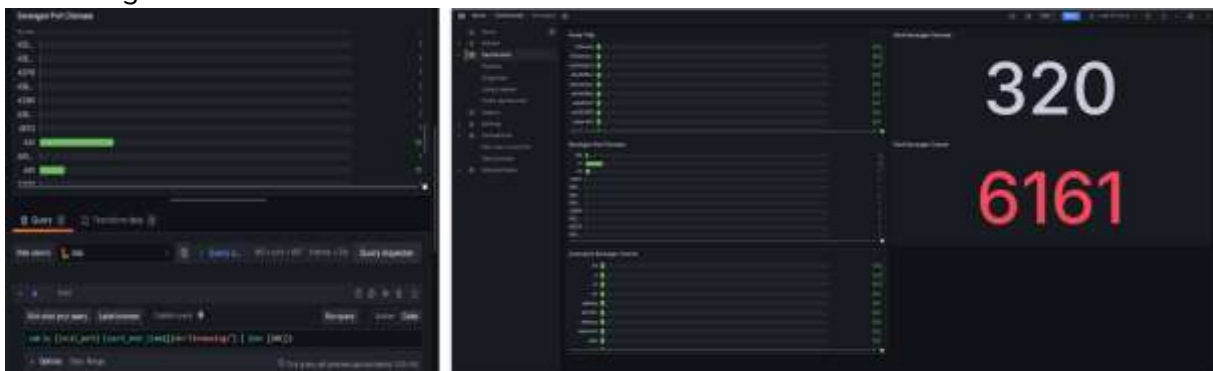
Below is a table showing the total usage of CPU% and Memory% after all the tests have been completed. The specifications used for the monitoring server are a 4-core CPU and 8GB of RAM. Based on the information in Table 5, the total CPU% usage of the system while running Grafana Loki is 2.0%, and its memory usage is 9.7% or approximately 769 MB. In contrast, the total CPU% usage of the system while running ELK Stack is 3.2%, and its memory usage is 37.8% or approximately 2.93 GB. This makes Grafana Loki a lighter visualization application, using 1.2% less CPU and 28.1% less memory or 2161 MB compared to ELK Stack.

**Table 5.** Resources Comparison of Grafana Loki and ELK Stack

<i>Grafana Loki</i>		<i>ELK Stack</i>	
CPU%	MEM%	CPU%	MEM%
2.0 %	9.7 % (769 MB)	3.2 %	37.8 % (2.93 GB)

### Data Visualization of Grafana Loki

Data log visualization from Grafana Loki is performed on a web browser of the host by entering the IP address and port of Grafana from the monitoring server. In this study, Google Chrome is the web browser used. Visualization on Grafana can be accessed through the dashboard menu. Grafana uses LogQL (Log Query Language) for visualization. According to Figure 4, the query used is sum by (local\_port) (count\_over\_time({job="dionaealogs"} | json [24h])). The query sum by (local\_port) calculates the total number of occurrences grouped by local\_port. This means the query will count the number of occurrences for each unique value of local\_port. The part (count\_over\_time({job="dionaealogs"} | json [24h])) is the query that counts occurrences within a specific time range. Within the curly braces {}, the filter {job="dionaealogs"} restricts occurrences to jobs with the label job equal to dionaealogs. The operator | json [24h] is used to retrieve logs in JSON format and obtain the necessary fields for visualization over the last 24 hours. The final result of the data visualization design on Grafana Loki is shown in Figure 4. The data displayed from Dionaea includes information on ports experiencing attacks from port scanning and Denial of Service (DoS) as well as the total number of attacks against Dionaea. Meanwhile, the data visualization from Cowrie shows the usage of usernames and passwords during brute force SSH attacks and the total number of attacks against Cowrie.



**Figure 4.** Data visualization of *Grafana Loki*

### Data Visualization of ELK Stack

Data log visualization from ELK Stack is performed on a web browser by entering the IP address and port of Kibana from the monitoring server. In this study, Google Chrome is the web browser used. Visualization in Kibana can be accessed through the dashboard menu. Kibana uses fields mapped in Elasticsearch for visualization. The result of the data visualization design in ELK Stack is shown in Figure 5. The data displayed from Dionaea includes information on ports experiencing attacks from port scanning and Denial of Service (DoS), as well as the total number of attacks against Dionaea. Meanwhile, the data visualization from Cowrie shows the usage of usernames and passwords during brute force SSH attacks and the total number of attacks against Cowrie.

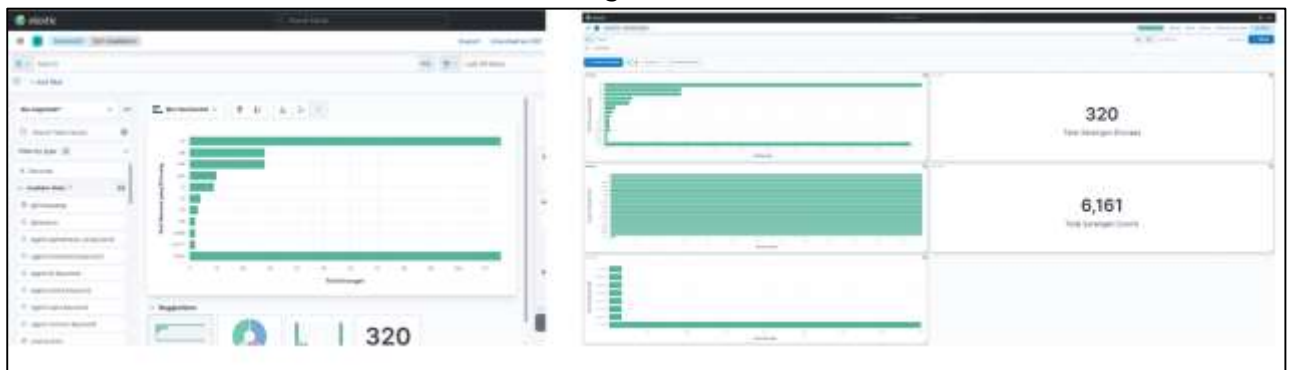


Figure 5. Hasil Visualisasi Data *Grafana Loki*

### Feature Comparison of Grafana Loki and ELK Stack

Here is a table comparing the features of Grafana Loki and ELK Stack:

**Table 6.** Feature Comparison of Grafana Loki and ELK Stack

Feature	<i>Grafana Loki</i>	<i>ELK Stack</i>
Data Visualization	With LogQL queries	With fields mapped in Elasticsearch
<i>Auto Refresh Interval</i>	Limited options available	Can be set manually in seconds, minutes, and hours
<i>Persistent Data Storage</i>	Not available	Available
<i>Log Line Display Capacity</i>	Cannot display entire JSON log entries beyond 6000 lines	Capable of displaying all lines of JSON log
Data log Storage Efficiency	Lighter compared to ELK Stack	Higher compared to Grafana Loki
<i>Reporting</i>	Exports visualizations to CSV, with data displayed by time	Exports visualizations to CSV, with data showing the most recent visualization.

Based on Table 6, here is an explanation of the feature comparison between Grafana Loki and ELK Stack:

1. Auto Refresh Interval: Grafana Loki offers limited auto-refresh options with predefined intervals. This means visualizations can only be updated at intervals such as every 5 seconds, 10 seconds, or other preset options available in the system. In contrast, ELK Stack provides a more flexible and customizable auto-refresh feature. Users can set the auto-refresh interval in seconds, minutes, or hours according to their needs.
2. Persistent Data Storage: Persistent data storage refers to the application's ability to retain data even when the system is turned off. Grafana Loki has limitations in storing log data when the server is down. If the server shuts down or restarts, the log data collected by Grafana Loki will not be retained. Consequently, after the server is back online, the log data needs to be re-sent through Promtail to ensure all missing data is collected again. On the other hand, ELK Stack provides persistent data storage capabilities. Even if the server is turned off, the log data collected by ELK Stack remains available and can be accessed once the server is restarted. This makes ELK Stack better at maintaining log data integrity and minimizing data loss due to server interruptions.
3. Log Line Display Capacity: When querying log data from the Cowrie honeypot, Grafana Loki has limitations in displaying JSON log data that exceeds 6000 lines. This limitation can be a hindrance when analyzing large and complex log data, as not all information can be displayed and analyzed directly through the Grafana Loki interface. In contrast, ELK Stack demonstrates better capability in this regard, where it can display the full detail of JSON log data. Therefore, for analysis requiring full visibility into every detail of the log, ELK Stack offers a more effective solution compared to Grafana Loki.
4. Data Log Storage Efficiency: Based on the conducted tests, Grafana Loki shows higher efficiency in managing and storing log data from the honeypot server compared to ELK Stack. In Grafana Loki, the storage for Dionaea log data is 30.9 KB and for Cowrie log data is 2.77 MB. In comparison, ELK Stack requires 201.9 KB for Dionaea log data and 3.4 MB for Cowrie log data. This difference indicates that Grafana Loki reduces log data storage requirements compared to ELK Stack, demonstrating more efficient storage usage.
5. Reporting: Both Grafana Loki and ELK Stack can export visualization results to CSV format, but they differ in how the data is exported. Grafana Loki allows for CSV export showing incremental updates in visualization data, providing a dynamic view of changes over time. In contrast, ELK Stack exports only the most recent visualization data in CSV format. This difference highlights Grafana Loki's advantage in providing more dynamic and continuously updated information for users.

## CONCLUSION

Based on the research findings, the CPU usage of the ELK Stack application is higher compared to Grafana Loki when processing attack data. Regarding memory usage, ELK Stack uses 37.8% or 2930 MB of memory, whereas Grafana Loki uses 9.7% or 769 MB. Despite ELK Stack requiring more CPU and memory resources, its data visualization is more straightforward as it relies on fields pre-mapped by Elasticsearch. In contrast, Grafana Loki requires knowledge of LogQL query code to visualize data, making it more complex. In terms

of feature offerings, Grafana Loki excels in log data storage and provides data export in CSV format every minute. Meanwhile, ELK Stack is superior in flexibility with refresh intervals, handling detailed JSON log data, and maintaining log data availability when the server is down.

## REFERENCE

- Annovazzi, D. (2022). *How Google Cloud can help stop credential stuffing attacks*. Google Cloud. <https://cloud.google.com/blog/products/identity-security/how-google-cloud-can-help-stop-credential-stuffing-attacks>
- Cemazar, S. A. (2022). *10 biggest advantages of open-source software*. <https://www.rocket.chat/blog/open-source-software-advantages>
- Eka, R., Rachman, A., & Wahyu, T. (2010). Virtual Private Server ( VPS ) Sebagai Alternatif Pengganti Dedicated Server. *Seminar on Intelligent Technology and Its Applications, SITIA*, 2–7.
- Gunawan, A. R., Sastra, N. P., & Wiharta, D. M. (2021). Penerapan Keamanan Jaringan Menggunakan Sistem Snort dan Honeypot Sebagai Pendeteksi dan Pencegah Malware. *Majalah Ilmiah Teknologi Elektro*, 20(1), 81. <https://doi.org/10.24843/mite.2021.v20i01.p09>
- Kaspersky. (2022). *Hacktivists step back giving way to professionals: a look at DDoS in Q3 2022*. Kaspersky. [https://www.kaspersky.com/about/press-releases/2022\\_hacktivists-step-back-giving-way-to-professionals-a-look-at-ddos-in-q3-2022](https://www.kaspersky.com/about/press-releases/2022_hacktivists-step-back-giving-way-to-professionals-a-look-at-ddos-in-q3-2022)
- Nurrahman, A. F. (2019). Low-Interaction Honeypot Dengan Dionaea Untuk Mendukung Keamanan Jaringan. *Journal of Informatics and Technology*, 2(4), 28–37.
- Rachman, P., Yugitama, R., & Sulistyo. (2019). Efisiensi Monitoring Honeypot Dengan Menggunakan Visualisasi Dan Otomatisasi Laporan Log Serangan. *Jurnal IT*, 10(3), 245–252.
- Sulaksono, W. A., & Suharyanto, C. E. (2020). Implementasi Honeypot Sebagai Sistem Keamanan Jaringan Pada Virtual Private Server. *InfoTeklar : Jurnal Nasional Informatika Dan Teknologi Jaringan*, 5(1), 90–95.
- Syaifuddin, Ahadin, W. B., & Sari, Z. (2022). Visualisasi Data Attacker Activity Log Portable Modern Honey Network. *Jurnal Repositor*, 4(1), 95–102. <https://doi.org/10.22219/repositor.v4i1.1446>
- Tati Ernawati, & Fikri Faiz Fadhlur Rachmat. (2021). Keamanan Jaringan dengan Cowrie Honeypot dan Snort Inline-Mode sebagai Intrusion Prevention System. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, 5(1), 180–186. <https://doi.org/10.29207/resti.v5i1.2825>
- Tilwani, R. (2023). *Grafana vs Kibana - Which One to Choose & Why?* <https://humalect.com/blog/kibana-vs-grafana-differences>
- Wastumirad, A. W., & Darmawan, M. I. (2021). Implementasi Honeypot Menggunakan Dionaea Dan Kippo Sebagai Penunjang Keamanan Jaringan Komunikasi Komputer. *Jurnal Teknologi*, 9(1), 80–91. <https://doi.org/10.31479/jtek.v9i1.119>
- Wibawa, G. H. P., Sasmita, I. G. M. A., & Raharja, I. M. S. (2020). Analisis Data Log Honeypot

- Menggunakan Metode K-Means Clustering. *Jurnal Ilmiah Merpati (Menara Penelitian Akademika Teknologi Informasi)*, 8(1), 13. <https://doi.org/10.24843/jim.2020.v08.i01.p02>
- Widodo, A. (2015). Implementasi Monitoring Jaringan Komputer Menggunakan Dude. *Jurnal Teknologi Informasi*, 11(1), 1–10. <https://journal.ubm.ac.id/index.php/teknologi-informasi/article/view/255>
- Willy Andrian, & Dedy Prasetya Kristiadi. (2022). Pengembangan Manajemen Keamanan Informasi Database Dan Aplikasi Dengan Optimasi Keamanan Website. *Jurnal Sistem Informasi Dan Teknologi (SINTEK)*, 2(2), 63–68. <https://doi.org/10.56995/sintek.v2i2.48>