Arab American University
Faculty of Graduate Studies

Multi-Channel Fusion Model for Data Logs Analysis and Anomaly Detection in Data Centers

By

Ibrahim Ahmad Naser Abubaker

Supervisor
Dr. Majdi Owda

Co-Supervisor
Dr. Amani Yousef Owda

This thesis was submitted in partial fulfillment of the requirements for the Master's degree in Cybercrimes and Digital Evidence Analysis

June/2024

I

Thesis Approval

Multi-Channel Fusion Model for Data Logs Analysis and Anomaly Detection in Data Centers

By

Ibrahim Ahmad Naser Abubakr

This thesis was defended successfully on 29/06/2024 and approved by:

Committee members	Signature
1. Dr. Majdi Owda: Supervisor	Majdi Owda .....
2. Dr. Amani Owda: Co- Supervisor	Amani Owda .....
3. Dr. Muath Sabha: Internal examiner	Muath Sabha .....
4. Dr. Mohammed Hussein: External examiner	M.HUSSEIN .....

Declaration

I certify that this thesis titled “Multi-Channel Fusion Model for Data Logs Analysis and Anomaly Detection in Data Centers” is my work, has been completely done by myself, and does not contain work from other related works or research done in the same domain, and only submitted especially to complete a master’s degree in Cybercrimes and Digital Evidence Analysis.

The Name of The Student: Ibrahim Ahmad Naser Abubaker

ID: 201920296

Signature: Ibrahim Abubaker

Date: 29/10/2024

Dedication

I want to express my gratitude to my mother, my wife, and my children Zain, the twins (Ahmad and Sara), and Farah for their unwavering support. I dedicate this work to my father Soul, and I wish him eternal peace. I am grateful to the Arab American University for this opportunity and to all those who believe that nothing is impossible if you work hard.

Acknowledgment

I want to take this opportunity to express my profound gratitude and appreciation to my supervisor Dr. Majdi Owda and Co-supervisor Dr. Amani Owda, for their unwavering support, guidance, and invaluable instructions that have helped me achieve this work. They have not only directed me but also shared their wealth of experience, for which I am truly grateful.

Abstract

In a computer data center environment, the operation of numerous systems is highly sensitive due to the critical nature of the services they provide. Maintaining uninterrupted operations and upholding the triad pyramid of security, which encompasses confidentiality, integrity, and availability (CIA), is imperative. Currently, the security event logs sourced from various channels, including SIEM agents, IDS, IPS, antivirus, and other security devices, are monitored individually and separately. This leads to incidents being displayed on each device without any correlation with other security incidents. Additionally, the actions taken by security appliances are locally applied to the respective devices without coordination with neighbouring devices, creating a disjointed security framework. This study is dedicated to enhancing the security posture of systems and applications operating within data centers, focusing on the comprehensive logging of security events on all devices and applications to achieve information security purposes. This study presents a novel security procedural model to bridge the gap in the aforementioned security framework. The model integrates the playbook decision into a harmonious orchestration concept of data center peripheral devices by being implemented on the devices and applications within the data center. The new approach aims to enhance the security maturity by applying the action of decision taken by the model either creating or updating existing controls on all nodes. Following the implementation of this new model, notable improvements were observed. Specifically, there was a clear increase in the effectiveness of the IPS based on the actions reflected in the security devices chain. Also, there was a significant reduction in false positives compared to the previous IDS/IPS framework. This highlights the effectiveness of the new model in enhancing the security stance and operational integrity of the data center environment.

Table of Contents

List of Tables

List of Figures

List of Appendices

List of Abbreviations

AI	Artificial Intelligence
API	Application Programming Interface
AXFR	Asynchronous Transfer Full Range
CC / C2	Command And Control
CNN	Convolutional Neural Network
CTC	Computational Time Consumed
DBMS	Database Management System
DDoS	Distributed Denial of Service
DGA	Domain Generation Algorithm
DNS	Domain Name System
DoS	Denial Of Service
EDR	Endpoint Detection and Response
EPS	Event Per Second
FSM	Finite State Machines
FW	Fire Wall
G2B	Government To Businesses
GCC	Government Computer Center
GUI	Graphical User Interface

GWO	Grey Wolf Optimization
G2C	Government To Citizen
G2G	Government To Government
HDFS	Hadoop Distributed File System
HIS	Health Information System
IaaS	Infrastructure as a Service
IDS	Intrusion Detection System
IIS	Internet Information Services
IoT	Internet of Things
IP	Internet Protocol
IPS	Intrusion Prevention System
ISP	Internet Service Provider
JDBC	Java Database Connectivity
JMS	Java Message Service
LSTM	Long Short-Term Memory Network
MA-SIEM	Mobile Agent-Based Security Information and Event Management
mDNS	Multicast Domain Name System
ML	Machine Learning

MLP	Multi-Layer Perceptron
MTIT	Ministry Of Telcom and IT
MX	Mail Exchange
NIC	Network Interface Card
NLP	Natural Language Processing
OS	Operating System
PA	Palo Alto
PaaS	Platform as a Service
PCA	Principal Component Analysis
PDU	Power Distribution Units
RegEx	Regular Expression
REST API	Representational State Transfer Application Programming Interface
RNN	Recurrent Neural Network
SaaS	Software as a Service
SFTP	Secure File Transfer Protocol
SIEM	Security Information and Event Management
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol

SNMP	Simple Network Management Protocol
SSH	Secure Shell
UPS	Uninterruptible Power Supplies
URI	Uniform Resource Identifier
UTM	Unified Threat Management
VIP	Virtual Ip
WAF	Web Application Firewall
WMI	Windows Management Instrumentation
XDR	Extended Detection and Response
XML	Extensible Markup Language
TCP	Transmission Control Protocol
ImCNN	Improved Multichannel Convolutional Neural Network

1. Chapter One Introduction

Multi-Fusion Logs Analyses Introduction

Most devices, applications, and systems write logs about all transactions that occur through or originate from them [1] and it varies according to their multiple sources and types. Indeed, unusual movements may sometimes be exposed to or emanate from them. Occasionally, a particular transaction is reported from multi-channels of logs reporters, and these logs are fused into a single stream. Commonly, these suspicious transactions are called security events. As [2] defines data fusion "*Data fusion techniques combine data from multiple sensors, and related information from associated databases, to achieve improved accuracies and more specific inferences that could be achieved by using a single sensor alone*". Manipulating logs fused from multi-channel sensors to be considered one stream of logs is complex. Therefore, collecting data logs shall go through multiple stages of analysis called the data log fusion life cycle, the following stages illustrating data logs fusion life cycle as shown in Figure 1.1 to produce consistent results stored in database files that are ready to be analyzed and to detect anomalies as follows: (1) Data gathering, (2) Data cleaning, (3) Data parsing, (4) Data normalization, and (5) Data correlation [3] [4].

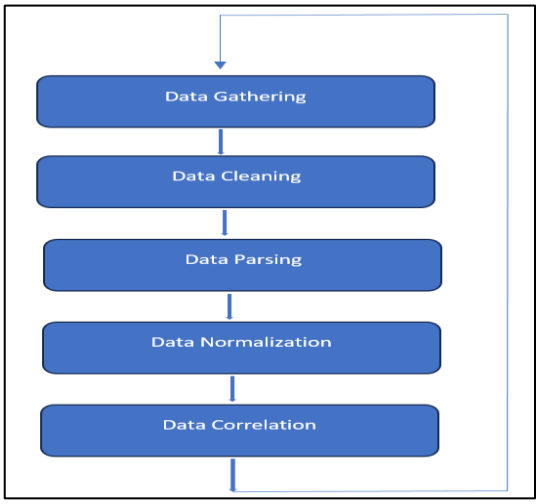


Figure 1.1 Data Logs Fusion Life Cycle.

Extracting security events from data logs or raw data can prove to be quite a complex and challenging process. Due to the most common obstacles:

1. The variety in technology has made the attack techniques different, so no universal technique is used for general anomaly detection.
2. The raw data of most devices and applications is not clean and unstructured, leading to inaccurate decisions.
3. Lack of labeled datasets covering all necessary features.
4. Most platforms only analyze security events from a single point of view, whereas handling log streaming from multiple channels is rare, and there is a lack in this research field.

It has been observed that several Security Information Event and Management (SIEM) Systems have been developed to tackle the issue of consolidating logs into a single repository (Fusion Databases [4]). SIEM manages security incidents from Intrusion Detection System (IDS) reporters of devices that have special software with sensors

around the data center, whether it is an appliance, software, firewall, or environmental sensors called reports. Once security sensors detect an event, they send it to the SIEM system. Then it performs a series of operations to generate alerts sent to the dashboard or administrators via email, SMS, or nudge to prompt them to take necessary actions to ensure the safety and security of the system [5]. Furthermore, IDS detects intrusion based on predefined rules and signature databases, so if the attack wasn't in the IDS database, then the rule didn't apply; consequently, there was no attack and no alert, although the system was being compromised [6][7].

The security controls applied to the devices regarding the events received from them will increase the effectiveness of the IDS. Moreover, using Machine Learning (ML) in the IDS increases the efficiency of the system and networks in detecting abnormalities [8]. This study will use ML to optimize the time needed to make the correct decisions in line with security controls. The commercial SIEM used in this study runs on high-specification hardware resources so that it can manage incidents and reduce Computational Time Consumed (CTC), which may be affected if resources are insufficient to meet concurrent transaction work qualifications and the number of unprocessed records. Even though this research used a high-performance machine, there is a time lag that sometimes reaches over 60 seconds in a single query. This is another challenge facing analysts. So, most SIEM vendors are trying to enhance the performance of the CTC by making better ML, especially in real-time monitoring [10].

The problem of manipulating the multi-channel logs that have been fused for anomaly detection in the data center is the focus of this thesis, in which the following questions will be addressed:

1. What is the most effective technical model for multi-channel data log fusion for anomaly detection in data centers, and how can the proposed model impact the performance of IPS systems?
2. How can the proposed multi-channel fusion model be integrated with a data center's security Logs management system to ensure coordinated threat response across all reporting nodes?

Data Center devices and applications serve as reporters, sending security events to the SIEM using various protocols. The agents of the same brand of SIEM are also called reports and allow integration capabilities to parse, normalize, and correlate logs, as well as security event logs from Windows and Linux servers (with Fabric Agent integration). Third-party applications that do not have the same technology as SIEM will send their logs using various protocols, including Simple Network Management Protocol (SNMP) v2/v3, Syslog, Secure Shell (SSH), and others. This study will provide a detailed description of these protocols later in the following chapters. All logs shall undergo into five steps, defined as the life cycle of the log journey: data gathering, cleaning, parsing, normalization, and correlations, as mentioned in the aforementioned figure of this chapter. After the transaction log has been passed through the five stages of the data logs life cycle, the fused database receives security events logs from multiple channels in live monitoring mode, which are then correlated based on the data received from SIEM agents or other sources. This helps to improve the accuracy and effectiveness of the security monitoring process. This study proposed a new model of security procedures applied to the chain of reporters (channels) based on the playbook concept, which guarantees that all systems are orchestrated to play harmonic rules together. After conducting the experiment, several positive outcomes were observed. **First,** the IDS device demonstrated the ability to share

its knowledge with other devices, resulting in mutual learning. **Second**, the Intrusion Prevention System (IPS) performance showed a significant improvement. **Lastly**, applying our model led to a substantial enhancement in IDS performance. In conclusion, the empirical results highlight a noteworthy improvement in the effectiveness of the IDS. Furthermore, the new model effectively reduced the occurrence of false positives previously detected by the IDS/IPS.

Research Objectives

The main goal of this study is to achieve the highest level of security in data center environments with minimal time and cost; this will be accomplished by:

1. Identifying patterns, threats, and anomalies.
2. Applying the proposed model by analyzing fused logs instead of analyzing them separately and performing the required action.

Contribution

To the best of the author's knowledge, there are no studies in the open literature that have utilized data fusion techniques with the same diversity of data the author sourced from different feeders to predict anomalies related to all fused streaming data. Nor is there a solution for preventing anomalies occurrence on all nodes of IDS. This study is to fill this gap by analyzing the logs of the data center environment and investigating the behavior of anomalies within each log that contains data about malicious activity, whether it is reported as an anomaly or not.

The approach involves using the fused data feeds into the SIEM as well as creating innovative techniques in the data processing stage within the manual data loading model.

Furthermore, each suspicious log is individually studied to extract patterns related to abnormalities. Ultimately, decisions made by the supposed model will be reflected across all other IDSs of the fused channels. The study conducted exploratory data processing on the logs obtained from various devices and applications. Each data set was presented separately and reviewed to extract semantics. This enabled authors to comprehend the behavior of actions taken on a specific transaction, as well as the relationship between two or more actions taken on the same transaction in the chain.

Thesis structure

This chapter presents an introduction to the thesis, followed by an overview of how the rest of the work is organized, as shown in Figure 1.2

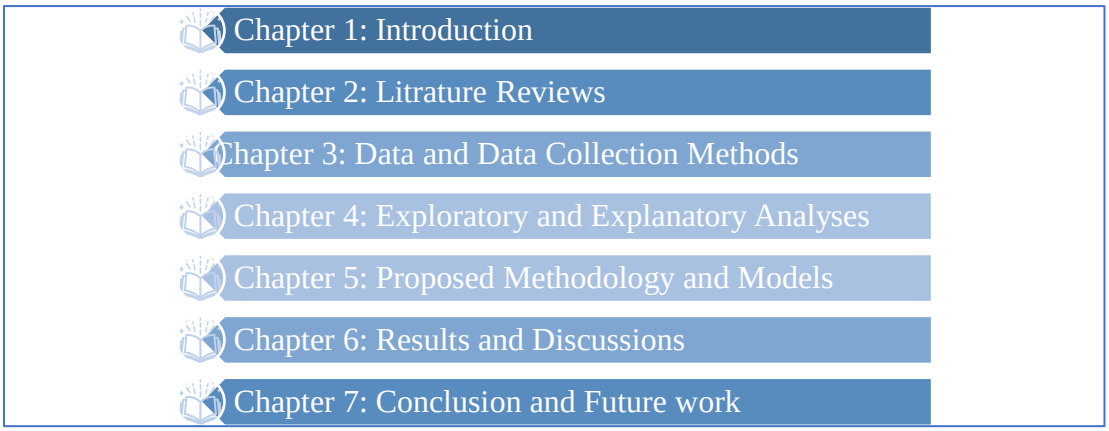


Figure 1.2 Diagram of Thesis Structure.

Chapter 2: Provides a comprehensive review of the literature and background related to log fusion and anomaly detection.

Chapter 3: The data and data collection methods elucidate the data sources and collection techniques.

Chapter 4: Offers an in-depth exploration of the data through graphical representations, and data explanations shedding light on the dataset and the interrelationships among its elements.

Chapter 5: Outlines the proposed methodology for developing a preventive approach model.

Chapter 6: Presents and deliberates on the findings of fusion models utilizing SIEM and Manual Loading data.

Chapter 7: Encapsulates the conclusion and succinctly represents the outcomes.

2. Chapter Two Background and Literature Review

This chapter presents a summary and critical analysis of relevant publications on multi-channel and data fusion log analysis of security anomaly detection and prediction of data centers' traffic flow.

Introduction

Through previous studies in the field of analyzing log records related to information security, there is a high interest in the field of security logs, especially addressing the events of the incident, specifically the logs streamed from multi-channel sources in data centers, besides the importance of the data about each anomaly that may occur while the systems and applications are running. Previous research has indicated a significant interest in analyzing logs related to information security events, commonly called incident events on each device separately. The significance of analyzing records encompassing information security data is rooted in service availability assurance.

In this chapter, the study of the Thesis aims to offer a detailed overview of the general concepts related to the life cycle of log records, from their inception to the end of the transaction journey. This analysis intends to provide a comprehensive understanding of the fundamental processes and mechanisms involved in log records. All logs recorded in the journal are vital for maintaining service functionality live by reporting abnormalities at the hardware or software level. It enables prompt issue resolution by tracing journals to remediate bugs and maintain the system's Confidentiality, Integrity, and Availability (CIA).

Nowadays, A multitude of programs, devices, and sensors operate within data centers generating logs [9]. However, collecting and analyzing these logs could be hard for analysts due to the varied log types and data resources [10]. In the previous study, the

researchers proposed a model that combined data as logs and monitoring data to compare the results based on Artificial Intelligence (AI) on the running services. In addition, ML was applied to the sensors on the machine and the application. They constructed a dictionary based on Natural Language Processing (NLP) and topic modeling from the data captured. The researcher reported correspondence of anomalies in the monitored data and log files in their results, but they had limitations in implementing the solution to the wide logs dataset, which wasn't remediated. Recently, most applications and services have run 24/7, and all of them generate logs to ensure quality-of-service delivery [9][11]. This leads the developer to build the trigger of alerts based on the service and differently, so log gathering involves different ways of preserving data about events on systems and applications; this research will describe the mechanisms and how the logs work. Regarding the previous methods [12] [13], logs are usually generated from applications or sensors based on two mechanisms: the **first** is when the alert has occurred, then the developed library starts writing this event, and the **second** is generated in every specific time interval. Each of the previous methods has pros and cons that will be discussed later.

Data Logs

Logs are crucial for keeping services functioning and tracking faults, errors, or abnormal behavior at the hardware or service level. Nowadays, most programs and devices, even all the sensors used everywhere, create logs [10]. Here, a brief description of the logs that study experiment will collect in gathering stage:

1. **System logs** are files containing stored data and logs generated by the operating system and applications running on the data center servers and devices. They

contain information about system startup and shutdown, software updates, and system errors [14].

2. **Security logs** are generated by security appliances and tools such as IPS, IDS, Firewall, WAFs, and Endpoint Detection and Response (EDR) / Extended Detection and Response (XDR) software [15]. Security application logs are generated by specific applications running on the data center servers [16].
3. **Environmental logs** are generated by environmental sensors, and they contain information about temperature, humidity, and other environmental factors that might affect the data center functions [17].

In computer science, logging means keeping information about processes, problems, errors, or any event triggered in the systems or applications running on the computational device. This data is stored in a textual format, and it provides information about the system and, or application [12] [18]. The logs' importance comes from the service's importance, again applications and devices in the Data Center generate logs to maintain the quality of the service delivery in a good way; the service is not only for the public; for example, the services of the Government to Businesses (G2B), Government to the Citizen (G2C), or Government to Government (G2G), but may be internally and between the systems themselves [9] [11]. The high demand and interest in log analyses, especially from an information security perspective make the industrial sector invest in developing applications for log security solutions; the sub-section 2.12 will explain some of them, showing advantages and disadvantages [19].

Logs Gathering

There are different ways to preserve the data about events that occurred on systems and applications that describe the mechanism and the ways the logs work regarding the following two methods [12][13]. The first one is based on the native logging system, which requires an application or software to write event data on a file with a specific format about abnormal behaviors or regular for auditing purposes. The second one depends on a third party known as libraries or Application Programming Interface (API); those shall work under the same privileges of the system to gain access to processes run on the application of the machine, then will write the events in a textual file, this task known as monitoring services [20]. The platforms of many companies generate large numbers of logs consistently. Usually, streaming data, such as queries, access logs, availability, performance, etc., require a specific log system software with common characteristics [21]:

1. Build a linkage tunnel between the application and analysis systems and deduce their relationship.
2. Real-time and online analysis systems and offline analysis systems like Hadoop.
3. Provide high scalability; when the amount of data increases, it can be expanded horizontally by adding nodes.

Anomaly detection is very crucial in many sectors, especially in the field of human life. In Health Information System (HIS), predicting medical events from multiple resources is very complex due to unexpected results from sensors, and this may depend on other outputs [22]. To solve this problem, they enhanced an extended-long Short-Term Memory Network (LSTM) algorithm based on the fusion of medical events. The

researchers correlated all the events from those sensors to choose the best diagnosis. However, they applied their experiment to a limited dataset, and missing values did not improve. On the other hand, ML limited the researcher's improvement due to the critical domain they are in, but this could be improved in the future.

Nowadays, open-source logging systems in terms of, design architecture, load balancing, scalability, and fault tolerance, such as Facebook's Scribe, LinkedIn's Kafka, and Cloudera's Flume, have become one of the utilities for logs management in big companies dealing with enormous amounts of data [23]. Facebook developed a Facebook scribe [24] to transport a vast amount of data in petabytes in an abbreviated time. A distributed system transports data from A to B with a minimum period and failure even during downtime [25]. In addition, scribe allows its users to define and run custom rules based on categories; the Figure 2.1 illustrates the high level of scribe architecture. The distributed data center of Facebook around the world is an advantage to managing logs separately according to the dataset of each data center, particularly in learning phase.

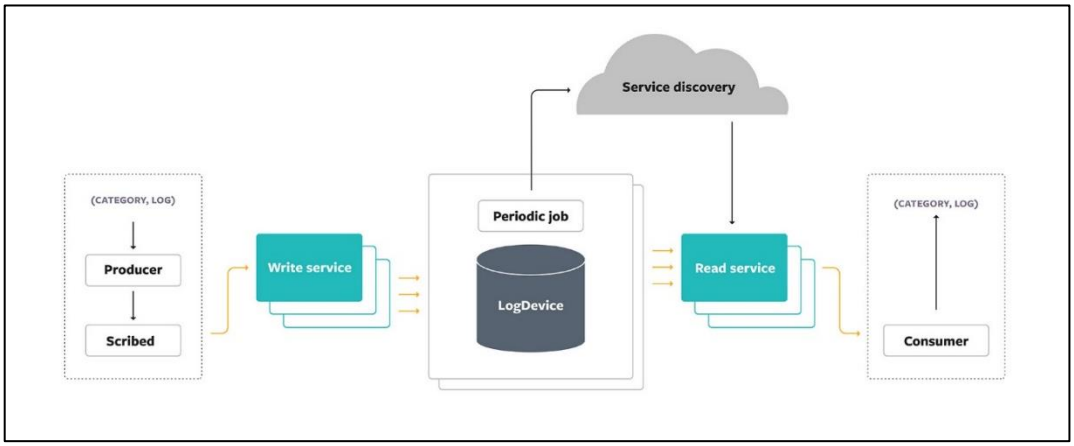


Figure 2.1 Facebook Scribe Architecture [24]

Another tool for gathering logs is Apache Kafka is an open-source system that stores large streams of data of events by many companies for monitoring and analyzing

events [26]. Kafka runs on the client and server-side using Transmission Control Protocol (TCP) and may be deployed on the physical machine or VMware server. Kafka is used in a large and broad scope of work, i.e., LinkedIn uses Kafka for streaming data. Also, Apache Samza [24] uses Kafka in transferring data because of its feature of storage and handling applications in fast memory. Cloudera's Flume is another tool for streaming logs and events into Apache Hadoop. It allows efficient data streaming, and the scale is dependable and built for the Hadoop scale [27]. Flume is a wonderful tool for detecting fraud, aggregating data center sensors and Internet of Things (IoT) and doing alerts. Notably that all previously mentioned tools were used as enterprise logs streaming

Data Center

A data center is a fully equipped and manufactured place where computers, servers, and network devices occur. This place has a well-prepared infrastructure, such as an anti-flooding system, a fire detection system, and an anti-corruption building. Also, it has an air conditioning system, redundant power supplies, and high-availability network links [28]. In addition, security controls are applied either physically, electronically, or as software. This environment allows data centers to run services on primes and may also run cloud services such as Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [29]. Recently, the data centers had the same architecture everywhere, but they were slightly different in size and the services they provided. A data center is like a human, a baby, or an adult. Both usually have the same body parts but different behaviors, and the services are also varied. So, the data center is a facility divided into physical and logical parts. The physical parts contain the building itself and infrastructure equipment, such as the previously mentioned parts, and hardware for

computing devices, including network routers and switches, in addition to other appliances for servers, storage, and others. Moreover, data centers include the management systems and software related to those components and services.

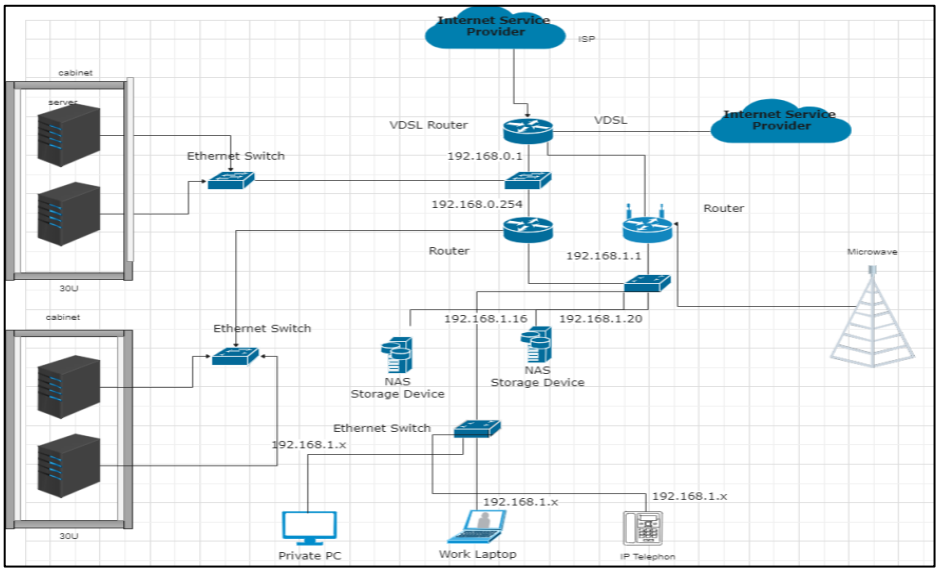


Figure 2.2 Data Center Diagram.

A typical data center architecture includes several key components, as shown in Figure 2.2 having the main common elements as:

1. Racks mountain cabinet.
2. Servers and Storage systems.
3. Network switches and routers.
4. Power distribution units (PDUs) and uninterruptible power supplies (UPSs).
5. Cooling systems.
6. Fire suppression systems.
7. Physical security systems, such as cameras and access controls.

The next subsection describes the data center logs in detail.

Data Center Logs

As mentioned previously, data center components typically generate logs of various types and sizes. This section outlines the most commonly used in the data centers and exposed to be used in the experiment of this study.

1. System logs: files store events and messages generated by the operating system and applications running on the data center servers and devices. It has information about system startup and shutdown, software updates, and system errors [14]. It also contains hardware logs generated by the hardware itself, such as servers, storage, power supply, hardware usage, failures, and other events related to the hardware.
2. Security logs, Security appliances and tools such as IPS, IDS, Firewall, WAFs, and EDR/XDR software generate logs describing security abnormalities, attempted hacks, and other information security events, as well as access logs containing logs about data center access control systems [15].
3. Network logs generated by network devices such as routers, switches, and load balancers contain information about network traffic, including Internet Protocols (IP) addresses, ports, and protocols used [30].
4. Application logs are generated by specific applications running on the data center servers [16]. It has information about application usage, errors, and performance.
5. Environmental logs come from environmental sensors, and they contain information about temperature, humidity, and other environmental factors that might affect the data center. The study of [17] proposed a solution to detect

failure or damage events in the AC, UPS, and other environment components using the “Raspberry Pi” kit as a tool to analyze the data coming from sensors around the data center to analyze the log form anomaly detection.

6. Audit logs, these types of logs are generated by the data center management systems and contain information about changes made to the data center administration, such as adding or removing hardware, security settings, and other configuration changes [31]The previous study surveyed different approaches to protecting and securing logs for later audit management.
7. User activity logs refer to the recorded interactions between a user and a software application or website, encompassing various actions such as clicks, page views, searches, and other events. These logs serve several purposes, including improving user experience, software debugging, and analyzing user behavior. Notably, user activity logs are valuable in web analytics, enabling web developers to gain insights into user interactions. Developers can identify frequently visited pages, popular features, and actions that lead to desired outcomes through analysis. In addition, it is used to search for anomalies in the activities of the users [32].
8. Transaction logs, a journal within a database management system (DBMS), that documents all the activities connected with a transaction. They are necessary to preserve the accuracy and consistency of the data in a database [33]. Transaction logs track all database changes, such as additions, updates, and removals, which are generally kept on disk and continually updated when transactions occur. Each transaction is assigned a unique identification number to link all the modifications made inside it.

In addition, Transaction logs are essential for assuring data consistency and durability. In DBMS transaction logs can be used to restore the database to a consistent state in the case of a failure, such as a hardware malfunction or power loss. The transaction log keeps track of all database modifications made since the last backup and can be played back to update the database.

Data Center Threats

In [34] they introduced threat modeling for cloud data center infrastructures as a systematic approach for identifying and evaluating potential security threats in cloud infrastructure. Their approach aims to identify the most critical security risks to the infrastructure and prioritize security measures to mitigate them. The researcher defined the process of threat modeling as typically involving the following steps [35]:

1. Identify assets: This involves identifying the assets that are most critical to the infrastructure, such as data, systems, and networks.
2. Determine threat actors: Identify potential threats, such as hackers, insiders, and malware, and who may pose a risk to the infrastructure.
3. Identify threats: Identify the specific security threats that could impact the infrastructure, such as unauthorized access, data leaks, and denial of service attacks.
4. Evaluate risks: Identify each threat's likelihood and the impact and determine which poses the most significant danger to the asset.
5. Prioritize mitigation: Based on risk evaluation, security measures are prioritized to address the most critical threats; the above process is occasionally identified as risk assessment.

Threat modeling for cloud data center infrastructure aims to provide a comprehensive understanding of potential security risks so that efforts are directed to developing adequate security measures to reduce those risks [36]. This will help organizations to maintain the confidentiality, integrity, and availability of their data and systems in the cloud.

In the [35] they conducted a threat modeling and assessment of cloud data center infrastructures by simulating two fictitious cloud data center infrastructures. They applied three different threat models, including the attack surface, attack tree, and attack graph, to assess potential security threats from various perspectives and at multiple levels of abstraction. The study lacked experience in physical data centers. In addition, they reported that the algorithm they used in the experiment had weakness.

Finally, Cybersecurity threats pose significant risks to data centers here [37] the survey talked about all threats of data centers causing risk in the field of information technology. Computer viruses, which are a type of malicious program, can alter computer operations without user consent by embedding themselves in other files to spread. Contrary to popular belief, viruses require activation through user action or system vulnerabilities, not just by inserting infected media. Over time, file types vulnerable to viruses have expanded beyond traditional executable formats like .exe to include documents (.doc) and images (.gif). Worms, more complex than viruses, spread independently through networks, consuming bandwidth and potentially crashing systems. Trojan horses disguise themselves as useful software but enable unauthorized access once executed, often leading to data theft or system compromise. Rootkits enable covert access to systems, allowing cybercriminals to control devices undetected, posing significant risks to personal and financial information.

Logs Compression

Generally, the logs generated from the application and system are massive, and many systems need over 100 gigabytes compressed per 24 hours. Here, data compression is required to reduce the dynamic range of data by compressing the higher amplitude values of the same event while leaving the lower frequency of data values relatively unaffected. In [38] they addressed the challenge faced by log compression research by achieving a high compression rate while maintaining efficient and lossless compression. To accurately and effectively evaluate log compression methods, obtaining a vast quantity of logs from a wide range of systems is necessary, like the assessment process of log parsers, which next subsection talks about the parsing process background.

Logs Parsing

Parsing involves reading and interpreting the log files to extract meaningful information, such as the number of requests made to a web server, the frequency of errors, the IP addresses of users, and the type of requests being made. This information can be used to diagnose issues, track performance, and optimize the system, and it can be done manually or automatically using specialized tools and software regrading to the authors study in [38] and as shown in Figure 3.2 [39] Illustrated the automated log parsing to identify patterns and anomalies in the Hadoop log files, which can help identify issues and optimize the system more efficiently. Zuh Jieming, in the previous study, found that Log-based features can be categorized into numerical and graphical.

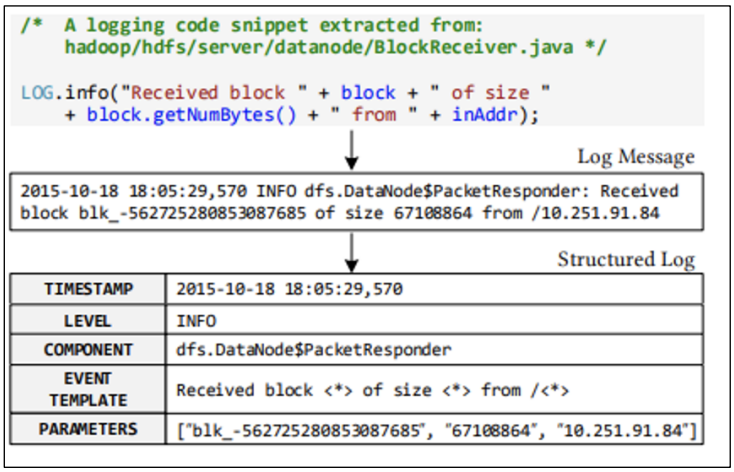


Figure 2.3 Automated Log Parsing

.

Logs Analysis

Log analysis examines and interprets log data generated by computer systems, applications, or networks. It aims to identify patterns, anomalies, and insights in the data that can improve the system's performance, security, and reliability [40]. To start analyzing the logs after the process of parsing has been done, the mining process shall take place here; the mining data of the logs consists of the following steps [41]:

1- Partitioning

2- Syntax extraction

3- Monitoring anomaly

Next is the explanation of each step partitioning, extraction, and monitoring consequently.

Log Partition

There are several types of log portioning, depending on the need to use the logs and how the analyst could reuse the data [42]. Here are the main criteria of data partition process:

1- Time-based partitioning: dividing log data based on time intervals, such as daily, weekly, or monthly partitions. This can make it easier to manage and analyze log data over time.
2- Size-based partitioning: This is based on file size, such as creating a new log file once the current file reaches a specific size limit. It can help prevent large log files from becoming difficult to manage.
3- Event-based partitioning: This is based on specific events or activities, such as separating application logs from system logs or security logs from operational logs. It can make it easier to analyze specific types of log data and identify potential issues.
4- Geographical partitioning: classifying the logs based on their geographic location, such as separating logs from different regions or data centers.
5- Custom: Log data management and analysis can be customized to meet specific needs based on specific criteria or requirements, such as separating logs for different departments or projects within an organization.

Syntax Extractions

Textual logs in a log partition must be converted into formats suitable for machine learning algorithms to analyze logs automatically. Zuh Jieming found [39] that Log-based features can be categorized into numerical and graphical. Numerical features, the most

used type in the log analysis community, represent the statistical properties of logs and include numerical and categorical fields that can be directly extracted from logs. They enable the representation of information from a log partition as a numerical vector. Meanwhile, graphical feature extraction makes identifying patterns and anomalies in the data more accessible. This can help troubleshoot and resolve issues quickly, improving system performance and efficiency. Graphs and charts can visually represent the log data, helping to identify patterns and trends and making it easier to analyze and understand.

The graphical method uses two standard algorithms for feature extraction [42]. The first one is identifying the objects or set of objects in the logs, such as process number, session ID, Source IP, Destination IP, and other objects needed to construct the graph. The second one used the statical distribution of patterns and the frequency of terms that occurred as what they developed in their research of two methods, 2KDiff and nKDiff, for constructing Finite State Machines (FSM) in log differencing. Specifically, 2KDiff identifies and emphasizes distinctions between two log files that contain partitioned logs, while nKDiff can compare multiple log files simultaneously. While the study presents all variances, specific differences may hold greater importance than others. To accurately differentiate between significant and insignificant variances, engineers may benefit from a quantitative extension that accounts for frequencies and applies statistical methods.

In [43] talked about the technique used in anomaly detection based on the extraction of logs method, which they proposed as a solution to reduce the computational cost. They developed a new feature named “Logevent2vec” to find the relevance between the log event in wrord2vec and the vectorized data. They transformed the output to a sequence of the data vectors using Bray as one of the transformation methods and a type of supervised model such as a neural network. This experiment reduces the computational

cost by 30 times the previous one, which is word2vec, and enhances the accuracy of anomaly detection. The research does not provide sufficient details about implementing the LogEvent2vec method, making it difficult for other researchers to regenerate the results.

Monitoring Anomaly

According to [44], there are two primary subcategories that are generally used in log monitoring and anomaly detection. Batch log-based anomaly detection and streaming log-based anomaly detection. The problem with this approach, from the researcher's point of view, is the supervision framework. Due to that issue, they proposed an offline unsupervised framework with real-time anomaly detection capabilities. Researchers used two techniques: knowledge-based construction using a new log parser and extraction method. The second one is streaming anomaly detecting anomalies after parsing the log data.

Anomaly Detection

Today, the traffic generated by the wide range of applications is massive and diverse. According to the Smart Insight website, Facebook reached 2.934 billion users in October 2022 [45]. So, the need for a solution to find anomalies and prevent the impact of infection around all the networks has become mandatory to make the services trusted and reliable without any disruption.

An anomaly detection technique is a way to detect abnormal traffic pass-through devices or applications, such as fraud, overflow, and other excellent and regular traffic. This makes it difficult to parse and identify bad or good behavior, especially in the data center system logging server, which is very big in volume [46]. Here, the solution proposed

by the researcher compares two techniques; the first one is integrating NLP features from logs with TD-IDF to generate combined characters and the second is training various models, such as logistic regression and Support Victor Machine (SVM), to obtain the outcome. Hicham [47] analyzed the impacts of data breaches and shed light on hacking breaches as the most devastating breach of 1.5 million financial records on 9000 data breaches made public from 2005 to late 2018. The study examines the organizations most targeted by hackers and how their interests have changed. Also, discuss how the frequency of breaches caused by human error has decreased due to increased employee awareness and the application of security standards. In his study, he aims to improve understanding of hacking breaches and help organizations better secure their data by prioritizing the most attacked sectors. This research has facts from 2015 to 2018 that make the dependency on the actual state of attackers ancient. The types and tactics of hacking and data leaks have been developed multiple times over the last five years.

Again, anomaly detection is identifying unusual or abnormal patterns in the system using ML algorithms or statistical models to analyze log data and identify patterns that deviate from normal behavior. This can include unusual spikes or dips in activity, unusual access patterns or attempts, errors or exceptions that occur more frequently than expected, or any other unexpected behavior that may indicate a security breach or system malfunction.

Researcher in the published paper [48] discusses the limitations of single-stage deep learning-based log anomaly detection methods, which focus on specific aspects of log information and may compromise detection performance and robustness. To address this issue, the researchers introduce a multi-stage log anomaly detection method called LogMS, which utilizes a fusion of multiple sources of information and probability label

estimation to comprehensively analyze log data from various perspectives. Experimental results show that Log of Multi-Source (LogMS) outperforms baseline models in robustness testing. The paper also suggested future research directions, including integrating additional sources of information such as system metrics, network traffic, and user behavior patterns to enhance the accuracy and robustness of log anomaly detection models. The primary limitation of the study was the experiment focused on a single aspect of log information neglecting the whole log aspects.

Learning and Training

Monitoring and analyzing logs are essential for ensuring the smooth operation of data centers. However, with the increasing complexity of modern data infrastructures, the volume of logs generated can be overwhelming, making it challenging to detect anomalies that could indicate potential risk. This is where effective anomaly detection training models come into play. By leveraging ML algorithms and statistical analysis, the following models can help identify abnormal patterns in log data, alerting IT teams to potential problems before they become major headaches. During the literature review, two training approaches were found in this domain: classical ML algorithms and deep learning models.

Traditional ML algorithms typically operate based on features explicitly provided by analysts, such as log event count vectors. Anomaly detection can be formulated and addressed with different algorithms, including clustering, classification, and regression.

One widely used algorithm for this purpose is Principal Component Analysis (PCA). By projecting data points onto the (K) principal components, anomalies can be detected when the projected distance exceeds a certain threshold. Wei Xu and his

colleagues first introduced the application of PCA for anomaly detection. Other algorithms, such as classification and support vector machines, are used for anomaly detection. All are categorized as traditional machine learning algorithms [12] where all of them have concentrations in expanding and scalability in addition to the most cons manual engineering.

Deep Learning Models is an advanced field of ML that involves training artificial neural networks with multiple layers to learn and extract hierarchical features from data [49].

Through an examination of literature reviews, it was noticed that many studies utilize the Recurrent Neural Network (RNN) model and deep learning model to analyze logs for anomaly detection. The paper [50] introduced the Deep Log system, which utilizes a Long Short-Term Memory (LSTM) model to learn a system's regular execution patterns by predicting the following log event based on preceding log events. However, certain anomalies may not necessarily deviate from the standard execution path instead of presenting them as irregular parameter values. To address this issue, the deep log also incorporates the LSTM model to verify the validity of parameter value vectors.

The researchers in [51] introduce a hybrid data processing model that combines Grey Wolf Optimization (GWO) and Convolutional Neural Network (CNN) to detect network anomalies effectively. The model operates in two phases, starting with ImGWO, which uses feature selection to achieve an optimal trade-off between two objectives. The second phase involves ImCNN, which classifies network anomalies. The experiment was worked on the hybrid model and was run on benchmark and synthetic datasets. Still, malware detection, specifically in cloud environments, does not exist, as it has the

complexity and diversity of incoming traffic and underlying hardware, making identifying anomalies more challenging.

Another study [52] Proposed a new model for detecting malicious internet traffic using deep learning techniques. This model combines multiple features to improve detection accuracy and reduce false positives. Even though they faced challenges in the high variability and complexity of internet traffic data, it was difficult to detect malicious traffic and distinguish them from legitimate traffic accurately.

The following background shows the researchers analyzing multiple channels of network traffic data, selecting the most relevant features for detection, and using a multi-scale analysis to detect anomalies at different levels of granularity [53]. The research didn't benefit from considering complex scenarios on datasets thus they conducted additional comparisons with other detection algorithms to improve their results. Furthermore, the proposed scheme may not be practical for most of datasets rather than it does not involve AI algorithms.

The researchers in [54] proposed a new method for detecting anomalies in industrial environment using multiple data sources. The technique involves fusing data from different sources, such as machine sensors and cameras to improve anomaly detection accuracy. The research describes the design and implementation of the method and its evaluation using real-world datasets. The results were not comprehensive due to several challenges they faced while developing the anomaly detection system, such as imbalanced data which is a common problem in industrial anomaly detection. The second is the integration of multiple sensor data sources requires a robust fusion approach for analyses and anomaly detections.

Another team of researchers [55] tried to develop a framework for analyzing large amounts of system and network logs for security purposes. They mentioned in their research that the proposed framework is lightweight, distributed, and parallel, allowing faster and more efficient analysis compared to others using a filtering method to prioritize analysis based on the severity of potential security threats. The framework was implemented in the Amazon cloud environment, demonstrating its ease of use and efficiency of the prototype but without considering more complex log analysis applications at other layers. In contrast, the [56] proposed solution to the real-time anomaly login detection problem involves the utilization of an "Off-line Learning + online detection" algorithm model for asynchronous anomaly login detection by identifying and extracting user operating features based on analyzing multi-source logs; this allows differentiation between malicious and legitimate users who are logging-in for the first time, this function reduces the running time by tens of times. This approach was efficient for offline and online, but it took longer than expected in the offline experiment.

Detecting anomalies using multiple data sources involves firstly, integrating data from different domains and periods to improve the accuracy of anomaly detection, and secondly focusing on the structure of the proposed approach, implementation and performance evaluation [57]. The proposed solution in the previous study is designed to handle the challenges posed by large datasets and diverse data sources. Moreover [53] relied on the use of multiple data channels and the selection of important features during data analysis at several levels, namely, feature selection and reduction of calculations. The researchers also thoroughly explored the internal frequency-time correlations within multiple scales, they faced a challenge that the model was not applicable to running AI on complex scenarios and would not make the accuracy of the results more precise.

In [58] researcher talks about two methods for detecting anomalies in multi-view of data, the first one is the deep support vector data description algorithm, and the second addresses multi-perspective anomaly detection using three different fusion techniques. While the work done by [59] is based on analyzing five state-of-the-art deep learning models designed to detect anomalies in system behavior using four publicly available log datasets. The two experiments [58] and [59] were centered around evaluating various aspects of the models, such as training and grouping data, class distribution, presence of data noise, and the ability of the early anomaly detection.

The study [60] talked about a new approach applied to multi-source logs emphasizing the model for detecting attacks that cannot be generated dynamically using data from the original records, for example, user activity behavior anomaly detection method that utilizes PCA practical to implement and demonstrate the approach of the experiment used in [61] is adequate for practical applications although the current process is tailored for event-based anomaly detection, and it hadn't been applied to other attacks, such as access-based anomaly detection, network documents, or cloud applications. Another approach for multi-channel anomaly detection is presented in [62], which utilizes signal detection theory and multi-scale decomposition. The study proposed method combines Ensemble Empirical Mode Decomposition (EEMD) and multi-channel Generalized Likelihood Tatio Test (GLRT) to perform better than traditional multi-scale detection methods. This is due to its ability to consider the internal frequency-time correlations within multiple scales of traffic data. However, the results compared with AI results are relatively poor, and it was done on simple scenarios.

Authors in [63] discuss the Cyber Kill Chain, which was developed by Lockheed Martin and has since been widely adopted as a standard for organizations to understand,

detect, and defend against cyber threats. It is a new anomaly detection technique using CNNs, which can learn patterns and features. complex images, making them ideal for detecting anomalies in visual data. Then, during testing or deployment, if the CNN detects an image that does not fit the learned standard patterns, it is flagged as an anomaly. So, detecting anomalies in system logs using a CNN uses different filters to convolve with embedded log vectors, and multiple convolution layers are employed to compute the results. Therefore, a fully connected layer is added to produce probability distribution results. The CNN-based approach achieves higher and faster detection accuracy than Multi-layer Perceptron (MLP) and LSTM on big data system logs such as Hadoop Distributed File System (HDFS). It does not require any system or application-specific knowledge. Instead, the CNN model is a general method that can parse logs directly. Researchers didn't consider complex system logs for further training, testing, and designing an automatic log analyzer using deep learning to detect anomalies and classify root causes. Those results were corresponding to the [64] study results in the same experiment but in a different environment.

Security Information and Management System (SIEM)

SIEM is a widely deployed system for event security management that collects, aggregates, stores, retrieves data, and visualizes logs from different resources.

The Mobile Agent-Based Security Information and Event Management (MA-SIEM) framework leverages mobile agents to collect and normalize events in near real-time directly on the source device [65]. The framework proposes a dedicated SIEM server primarily focused on correlation and analysis by externalizing the normalization process, which multiple distributed mobile agents carry out across interconnected computers and

devices. Nevertheless, this approach is still not mature enough to give all predictions due to its function of feeding single threats. One more study is literature on commercial SIEM in [66] study developed a visual approach to SIEM by analyzing the alerts triggered based on a predefined rule for monitoring traffic in real-time using the Qradar solution, but it is characterized by its complexity of mange and use. FortiSIEM solution is a comprehensive system that reduces complexity and expands with no limitation on Event Per Second (EPS) if the HW specifications are available and meet the requirement. In addition, it provides an extensive range of features to navigate events that come from different protocols or agents, so it's a powerful solution to collect logs from other types of devices with variant protocols [67].

Table 2.1 shows the comparison between the top common SIEM used in log management.

Table 2.1 Multicriteria Analysis of SIEM Systems

System name	Search for vulnerabilities	Report generation	Cloud support	Support for working with Big Data platforms	Possibilities of integration with new systems tomorrow	Advanced search and data visualization	User friendly interface	Supported operating systems	Main sources of logs	System cost
IBM QRadar	+	+	+	+	+	+	+	linux	Lots of	$ 63000 +
LOGRHYTHM	+	+	+	+	+	+	+	linux/windows	Lots of	$ 28000 +
HPEArcSight	+	+	+	+	+	+	+	linux	Lots of	500 thousand rubles +
Splunk	+	+	+	+	+	+	+	Unix/Windows	Lots of	Free 500 mb $ 5.000 for 1 GB day
McAfee (ESM)	+	+	+	+	+	+	+	Windows	Lots of	$ 261000 +
AlienVault USM	+	+	+	+	+	+	+	Linux/Windows	Lots of	$ 1075/mo.
Alien Vault OS SIM	+	+	–	–	–	+	+	Linux/Windows	Lots of	free
FoniSIEM	+	+	+	+	+	+	+	Linux/Windows	Lots of	$ 900 +
Ixia ThreatARMOR	+	+	–	–	+	+	+	Unix/Windows/ other	Lots of	£ 3158/year
MozDef	–	+	+	+	+	+	+	Centos 7	Json	free
Wazuh	+	+	+	+	+	+	+	Linux	Windows/ Linux logs	free
Prelude OSS	+	+	–	–	–	+	+	Linux	Lots of	free
Prelude SIEM	+	+	–	–	–	+	+	Linux	Lots of	9
Sasan	–	+	–	–	–	+	–	Linux	Lots of	free
Maxpatrol	+	+	+	+	+	+	+	–	Lots of	RUB 1,840,000 +
SolarWinds	+	+	+	+	+	+	+	Linux/Windows agents	Lots of	2.055 € +
ManateEnaine	+	+	–	–	–	+	+	Windows	Windows/ Linux logs	$ 1000 +
EventTracker	+	+	+	+	–	+	+	Windows	Windows/ Linux logs	$8995
Micro Focus ArcSight	+	+	+	+	+	+	+	Linux	Lots of	500 thousand rubles +
Trustwave SIEM Enterprise	+	+	+	+	–	+	+	Centos 7	Lots of	$1000/year
BlackStratus SIEMStonn	+	+	–	–	–	+	+	–	not indicated	–
SearchInfonn SIEM	+	+	+	+	+	+	+	Linux/Windows	Lots of	Negotiable
RuSIEM	+	+	–	–	+	+	+	Ubuntu 16	Lots of	Paid/no fee

The researchers in [68] leverage ontological methods for detecting, recovering, and responding to DDoS attacks. The study integrates three essential securities of IDS/IPS with the SIEM to enhance the network's security posture. The research primarily employs signature-based detection to identify DDoS attacks to bolster the network's resilience against threats. The previous solution was well-equipped for implementation in both IT organizations and industrial environments, offering a robust defense against DDoS attacks.

The study of [69] aimed to merge the IDS and SIEM as one system including a database storing all threats already found by IDS and then analyzing them. The main challenge is not to support all protocols because the system used here is an open-source system platform The main challenge in this experiment was the lack of support for all protocols that collect logs from sensors because the system used here is an open-source system platform which poses limitations in implementing the experiment and thus obtaining better results. On the other side, the research in [70] used another approach that leverages a commercial SIEM (Splunk) to collect all incidents in one set instant and it was focused on the IOT sensors, but. Splunk is a very expensive SIEM and requires experience to work with it [71].

In this thesis, the experiment will be conducted partially on FortiSIEM and the other part using manual processing due to the working conditions in the organization where the experiment will be conducted.

Data Fusion

Data fusion can facilitate the extraction of information from a diverse array of data sources, particularly in the context of data center logs [72]. It has the potential to enhance the robustness of the proposed model by compensating for the limitations of another model and optimizing the search space for the development of effective solutions. Furthermore, fusion techniques can be used to examine the reliance on specific pattern relationship concepts. Therefore, this thesis presents a focused investigation of recent research on fusion-based anomaly detection logs, highlighting the importance of fusion in threat prediction.

Another study was conducted on data fusion [73] which presented an approach to classification in network security and anomaly detection that combines regular binary attacks and multiple attacks to detect anomalies and classify attacks. The study showed a significant challenge in highly unbalanced traffic which made the results weak. The lack of use of ML in the ZYELL dataset also made the results weak.

Conclusion

This chapter reviewed the most recent studies related to the thesis work. In conclusion, the comprehensive analysis of data center logs, including the various types of logs, threats, logs compression, parsing, analysis, and anomaly detection methods, demonstrates the complexity and importance of maintaining secure and reliable data center services and operations un-interpretable. It found that integrating security information events with data logs fusion model to detect the anomaly is a challenge for researchers to tackle. Also, existing SIEM solutions do not offer real-time monitoring for both SIEM and real-time IPS together. This study proposes using SIEM as a predictor of incidents and acting based on learned behavior as the proposed model will do in this work.

The next chapter presents the data logs and data collection methods that are used to collect the data from different sensors, which is the first step in the research development journey.

3. Chapter Three The Data Logs and Data Collection Methods

Introduction

This chapter thoroughly explains the data description in the thesis, the data collection techniques employed, and the advantages of utilizing these techniques over others. Indeed, multiple methods are needed because various data sources are being used. In this work, four data resources will be used in the experiment of this thesis, as shown in Figure 3.1, the sources are categorized according to data that come from Routers, Firewalls, OS (windows, Unix-based), and ESXi virtual machines. The Table 3.1 listed all the devices in the previous four categories. Later, the following chapters will describe each of the above-mentioned resources separately with related functions in accordance with the methodology followed in the thesis.

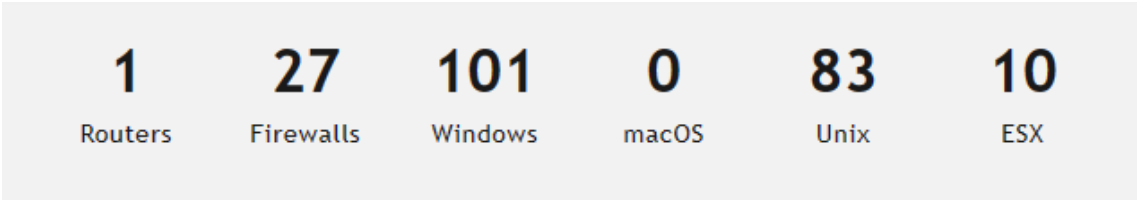


Figure 3.1 Data Sources

With the rapid development of systems and applications log files become larger and more significant [74]. So, the developer who creates the logging mechanism must know where to save the logs, when to write them in the journal, and what to write.

After having the logs in the previous conditions, a compression technique is applied to reduce the volume of logs in order to maintain the logs ready for the next cleaning phase, then to the parsing technique. After that, specific algorithms are needed

to be applied to those data after parsing and compression methods were mentioned above to analyze the logs data for anomaly detection [52].

The life cycle of log analysis starts with gathering and collecting the data from different applications and systems called reports or sensors occasionally, then log compression, parsing, and finally analysis, these steps are illustrated in Figure 3.2 [38]. This process particularly works on a single log streaming or one type of log; however, this process doesn't apply to the fusion data logs. There is a gap between the practical and theoretical approaches because many developers do their research on their data, and many companies are afraid of providing accurate data to be analyzed and applied to scientific studies. In this approach, the investigation of accurate data from the vast number of logs gathered from many different applications, systems, and sensors in the Data Center from a strategic point of view will take place in the later sections. This will lead the proposed model to reduce this gap and get more accurate results for anomaly detection. This is the strength and the advantage of this study because it investigates the real data in MTIT.

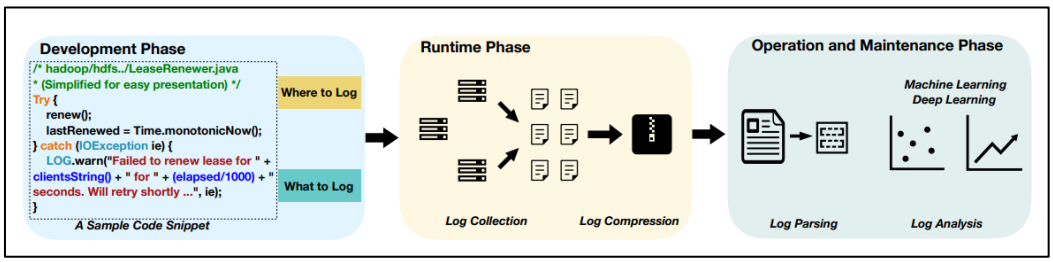


Figure 3.2 Framework of Logs Analytics

Below is the subsection that describes in detail the access protocols used to dump the data from multi-channel resources in the real data center environment.

Access Protocols

Access protocols allow the SIEM systems and analysts to obtain data from various sources, enabling more thorough analysis and better security monitoring. Figure 3.3 shows the impressive array of devices and sensors connected to the fusion log system. It contains the Microsoft Windows servers having the most traffic, followed by Forti OS devices such as FW, WAF, antivirus, and agent.

Table 3.1 List of Devices Feeding Logs

Device Type	**Count**
Microsoft Windows Server	57
Generic Linux	16
Fortinet FortiWeb	3
Ubuntu Linux	1
Fortinet FortiOS	13
Redhat Linux	2
Cisco IOS	1
Generic Unix	2
F5 Big-IPOS	2
CentOS Linux	2
WatchGuard Firebox	3
Fortinet FortiMail	1
Fortinet FortiDDoS	1
Ubuntu Linux 18.04.1 LTS	1
Sophos XG Firewall	3

Forcepoint NG Firewall	1
VMware ESXi Server	10
Palo Alto PAN-OS	2

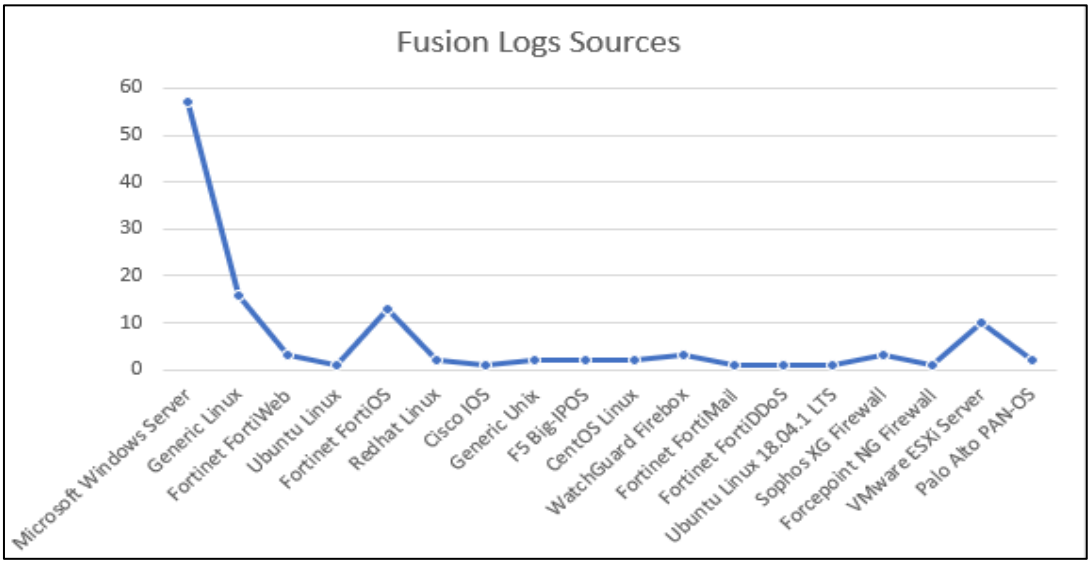


Figure 3.3 Fusion Logs Sources

All devices and sensors that are connected to the log repository may use one or more protocols to provide data logs to a single-point repository. For example, windows OS may send NetFlow and use the SIEM agent. The chart in Figure 3.4 describes which protocols are used to deliver data logs and how frequently each is used. Next, all protocols used to provide logs are described in detail.

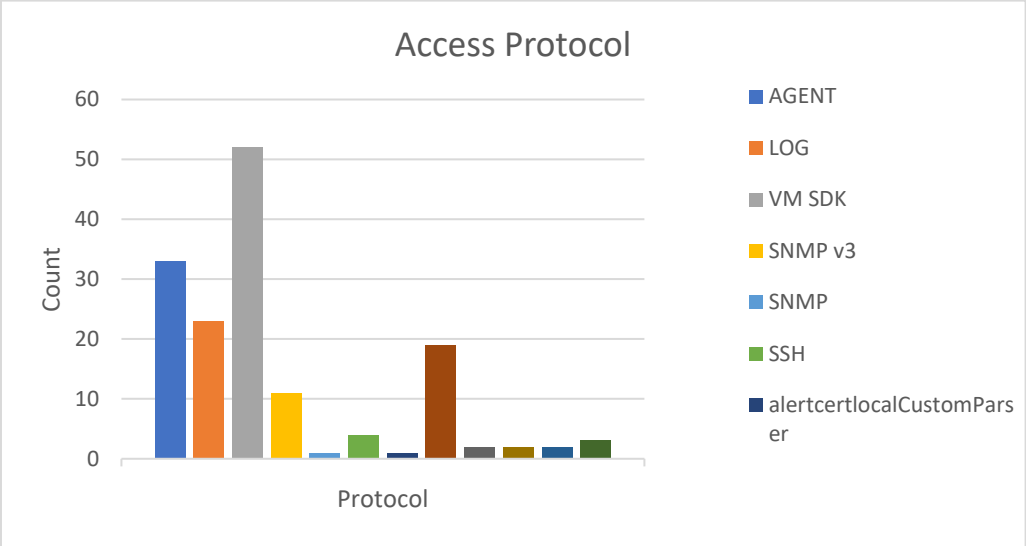


Figure 3.4 Access Protocol.

Syslog

The Syslog protocol is used to transmit event messages over IP networks and is commonly used UDP. The RFC:5424 [75] consists of a Syslog format that utilizes three layers as syslog content, Syslog application, and Syslog transport, and all of these layers have certain types of functions, the following diagram shown in Figure 3.5 describes the consistent layers as follow:

- Originator: generate the logs messages
- Collector: gathering syslog content
- Relay: forward the logs to another hub
- Transport sender and receiver: send and receive the logs from forwarders and others.

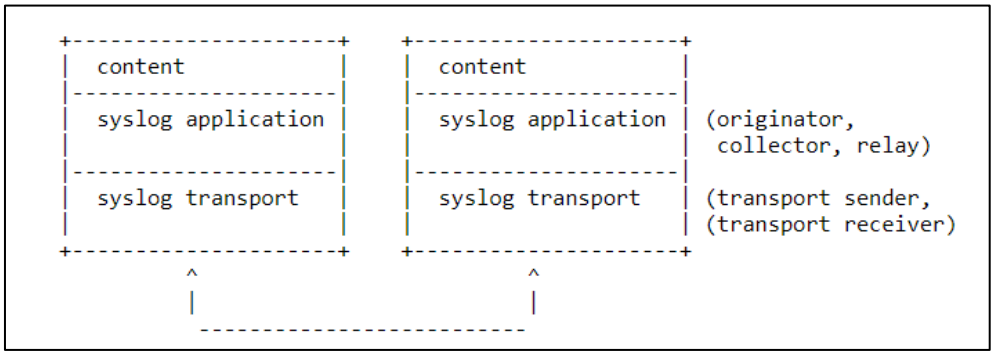


Figure 3.5: Syslog Layers

Syslog is widely used in SIEM systems and extensively adopted in log management applications. The researchers in [76] assessed the effectiveness of syslog protocol based on IDS, they assured that they can accurately identify various attacks from raw data logs.

Simple Network Management Protocol (SNMP)

SNMP controls and keeps track of networked devices, and it is frequently utilized in many different network management systems. In [77] used SNMP-based system for monitoring network devices, and it could efficiently track network performance and quickly identify network issues. This protocol is used for collecting and analyzing network traffic data based on SNMP version of v1, v2, and v3 which have more security and authentication features [78].

NetFlow

A network protocol was developed by Cisco company, and it was designed to capture IP and network traffic. It offers comprehensive insights about network usage and helps administrators comprehend and optimize network performance, anomaly detection, and maintaining of network security [79].

Java Message Service (JMS)

A messaging protocol called JMS exchanges messages between applications. A study by [80] developed a JMS-based framework offering high-performance data integration and processing capabilities. Enterprise service buses and message-oriented middleware are only two examples of distributed systems where JMS has been extensively deployed.

Secure File Transfer Protocol (SFTP)

Referred to safe file sharing protocol, as many applications that call for secure data transfer over networks have extensively used it. According to a [81] study, a secure data-sharing system built on SFTP can offer efficient and safe data transfer with little overhead.

Java Database Connectivity (JDBC)

JDBC is used to connect to databases store and retrieve data. Many Java-based applications that need database connectivity have to use it. A JDBC-based framework also used for accessing massive data as proposed in [82] who discovered that it could offer high-performance data access capabilities without overhead.

Representational State Transfer Application (REST) API

It is a web application architecture that uses stateless communication between the client and server. It has been widely used in log management systems to search and retrieve log data. A REST API-based log management system was suggested in a study by [83] and it was discovered that it can offer effective log data retrieval and analysis capabilities.

Log Forwarding

In general, log forwarding occurs between fabric solutions from the same brand. In the case of this experiment, the SIEM solution is Forti brand which enables all Forti products which support this protocol to send logs to the SIEM through log forwarding protocol [84].

Windows Management Instrumentation (WMI)

Windows infrastructure has been widely utilized in log management systems for gathering and analyzing Windows management messages logs. A WMI-based log management system was suggested in a study by [85].

Secure Shell (SSH)

The SSH protocol is employed to secure remote access to systems. It is a safe log data transfer and maintenance. SSH-based log management system was introduced in a study by [86], and offers low-cost, secure, and effective log data transport securely.

Simple Mail Transfer Protocol (SMTP)

Email messages are sent using the SMTP, and the logs of the SMTP server have been used to notify and inform system administrators about data that goes in and out of the email server. It offers effective log delivery and alerting capabilities. In general, the Figure 3.6 summarize main protocol features and dependencies; all listed protocols had been used in the log management system where the study is examined [87].

Aspect	Description	Protocol	Key Features
Syslog	Logging system events and messages from various devices and applications.	UDP (port 514), TCP, TLS.	Broad view of system operations, wide range of event types, standardized format (RFC 5424).
NetFlow	Monitoring and analyzing network traffic flows.	UDP or TCP; various versions (e.g., NetFlow v5, v9).	Granular flow-level data, insights into network usage, helps in detecting anomalies and security threats.
SNMP	Monitoring and managing network devices such as routers, switches, servers, and other networked devices.	UDP (ports 161 for queries and 162 for traps), TCP; SNMPv1, SNMPv2c, SNMPv3.	Comprehensive device status and performance information, remote configuration.
JMS	Enabling communication between different components of a distributed application.	JMS API over various transport protocols (e.g., TCP, HTTP).	Asynchronous message delivery, supports different message types.
SFTP	Secure file transfer over a secure shell (SSH) connection.	SSH, typically using port 22.	Secure file transfer, strong encryption, supports file and directory operations.
WMI	Managing and monitoring Windows-based systems and applications.	DCOM (Distributed Component Object Model), WinRM (Windows Remote Management); ports 135 (DCOM), 5985/5986 (WinRM).	Extensive management capabilities for Windows environments.
SSH	Securely accessing and managing network devices and systems remotely.	SSH protocol over port 22.	Strong encryption for secure communication, versatile for remote management.
SMTP	Sending and receiving email messages.	SMTP protocol over port 25, 465 (SMTPS), or 587 (submission).	Reliable email delivery, supports authentication, and encryption.
JDBC	Connecting and executing queries with relational databases.	JDBC API over TCP/IP to connect to databases.	Facilitates interaction with relational databases, supports SQL queries and updates.
REST	Accessing and manipulating resources via HTTP methods.	HTTP(S).	Simple and stateless communication, widely used in web and mobile applications.

Figure 3.6 List of Protocols

Data Descriptions

As is well-known, there are plenty of logs generated from an extensive range of applications and devices. In this regard, the following section will focus on the most common and essential reporting devices or sometimes called feeders, especially the devices that record security events. Furthermore, the description of those logs will be illustrated in real-time of services launched in the Palestine Government Computer Center (GCC) in the Ministry of Telecom and Information Technology (MTIT).

Later, the subsections below describe each platform used for logging in this study, for example, operating systems (OS) machines, network devices, agents, and applications.

Operating System Machine

This research has different OS types that is used in the experiment, as shown in Figure 3.7, mainly to have the nearly complete log type needed for the experiment. These included Windows-based OS, Linux-based OS, Firewall OS, WAF, Routers, VMware,

DDOS appliances, and others. The following is a detailed description of all of OS types consecutively.

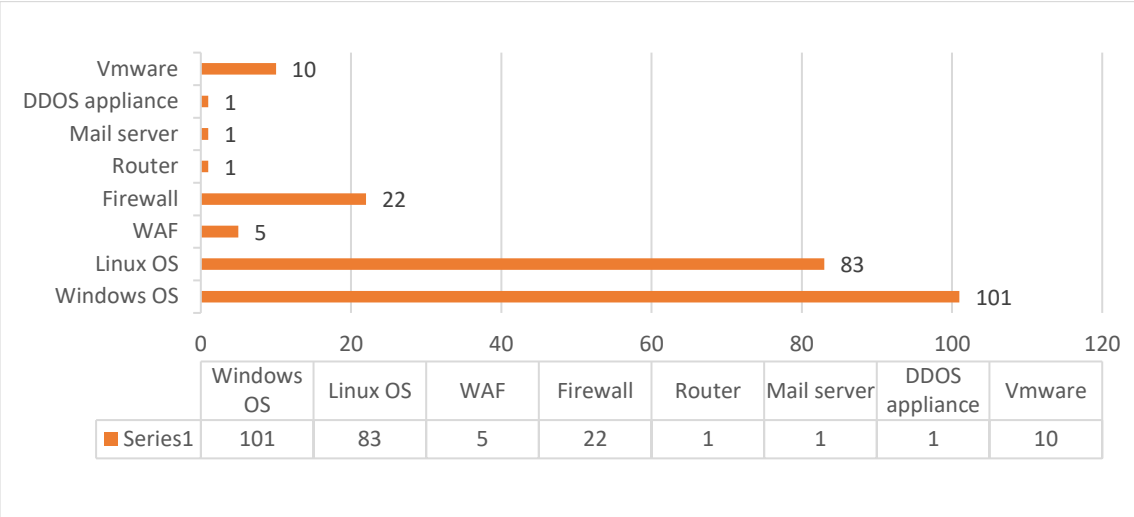


Figure 3.7 Operating System Machine

The adjacent Figure 3.7 provides statistics on most devices connected to the SIEM, with the largest share being Windows machines, then Linux-based machines, and next to them is FW. These numbers do not mean that the high number is sending a lot of logs, but the number of Windows servers is the most machines reporting logs, however, the DDOS machine has the most logs ever in the data center.

1.1.1.1 Windows-based OS

From previous data in the aforementioned figure, it is obvious that the Windows machine has the largest portion of logs, furthermore, multiple applications on a Windows server produce logs recording data about the activities of OS and applications with all events that have occurred. Usually, log files are stored under the directory

(C:\Windows\System32\Winevt\Logs) as event files with (evtx) format [88] as shown in Figure 3.8.

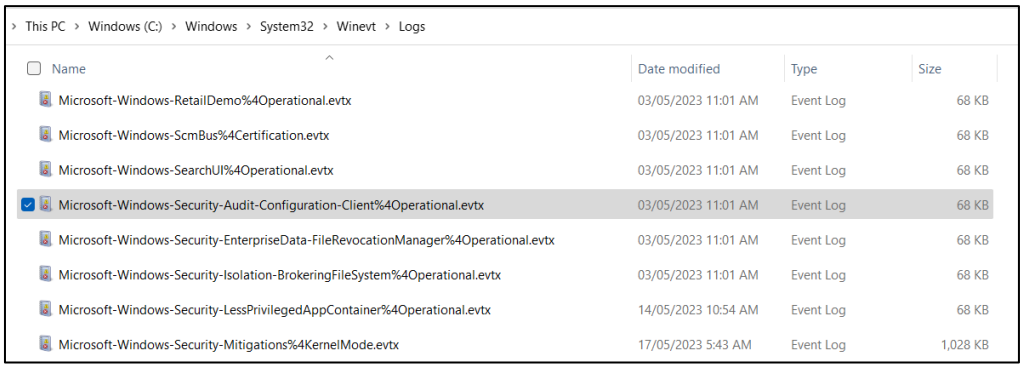


Figure 3.8 Windows Logs Directory

The following is a description of Windows machine OS logs generated from various applications running on Windows:

1. Event Viewer Logs

The Event Viewer application has been included with the windows operating system since July 1993 with Windows NT 3.1 [89]. Since then, Event Viewer has been included in every succeeding version of Windows, including Windows 95 to Windows 11, as well as Windows server version also. As shown in Figure 3.9 Event Viewer is a Graphic User Interface (GUI) for viewing, filtering, and analyzing event logs created by different Windows components and programs. It provides many log types, such as application, security, and system, to assist administrators in monitoring and troubleshooting system and application events.

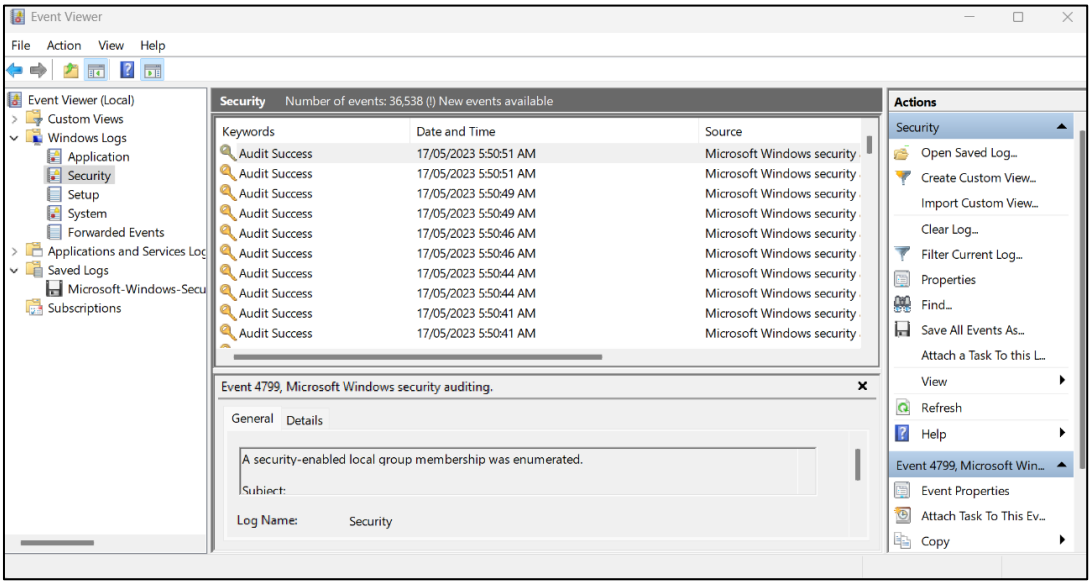


Figure 3.9 Windows Event Viewer

Later, the OS logs will be discussed in terms of the functions of the system itself.

2. Operating System Logs

Applications logs are significant to identify and resolve problems in Windows-based OS, such as data about the health and performance of installed applications, system services, and drivers. Commonly, OS generates logs automatically about the applications including a very wide range of information, error messages, warnings, and more. And helping administrators address issues affecting their applications, including crashes, failures, and performance issues. For example, if an application crashes frequently, the application logs might give significant information regarding the reason for the problem. Similarly, if an application takes a long time to start or react, the logs might assist in identifying potential bottlenecks or resource limitations. Another type of loges comes from an active directory which is described below.

3. Active Directory Logs

Due to the sensitivity of active directory logs especially in tracking anomaly events. The daily work of IT administrators, network engineers, and security specialists is looking for active directory logs. The logs include an array of information about the activity of people and devices on a network. In addition, active directory logs have the capability of gathering events from many network devices, including servers, workstations, routers, switches, and firewalls [90]. The illustrated image in Figure 3.10 provides an overview of the login log attempts with relevant details such as timestamp, username, login status, and other important data.

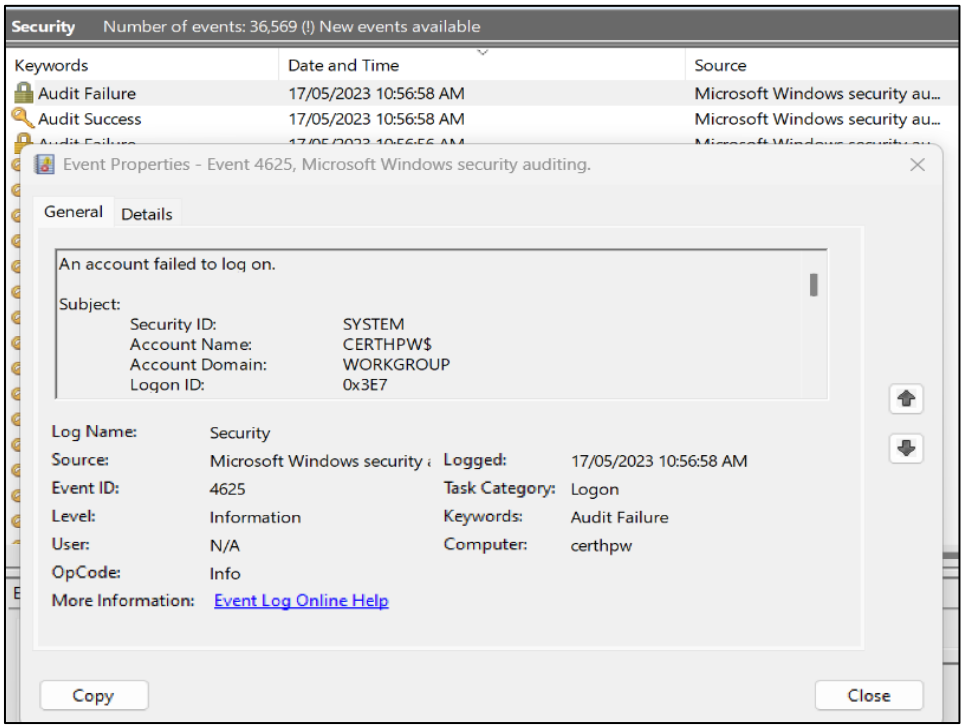


Figure 3.10 Active Directory Event

4. SQL Server Logs

Investigating SQL server logs is an effective way to audit queries and database health and integrity to ensure the security and consistency of the database. Logs have a lot

of information about database activity, engine problems, and security incidents. Continuous monitoring of these logs to identify vulnerabilities enables analysts to avoid potential risks and minimize the likelihood of data tampering. So, the remediation is to take appropriate action to harden the engine and close the vulnerability or build trust measures to deny the next threat that exhibits the same behavior. Figure 3.11 shows the log types of the SQL server on Windows. Another type of log that will be described below is an exchange server.

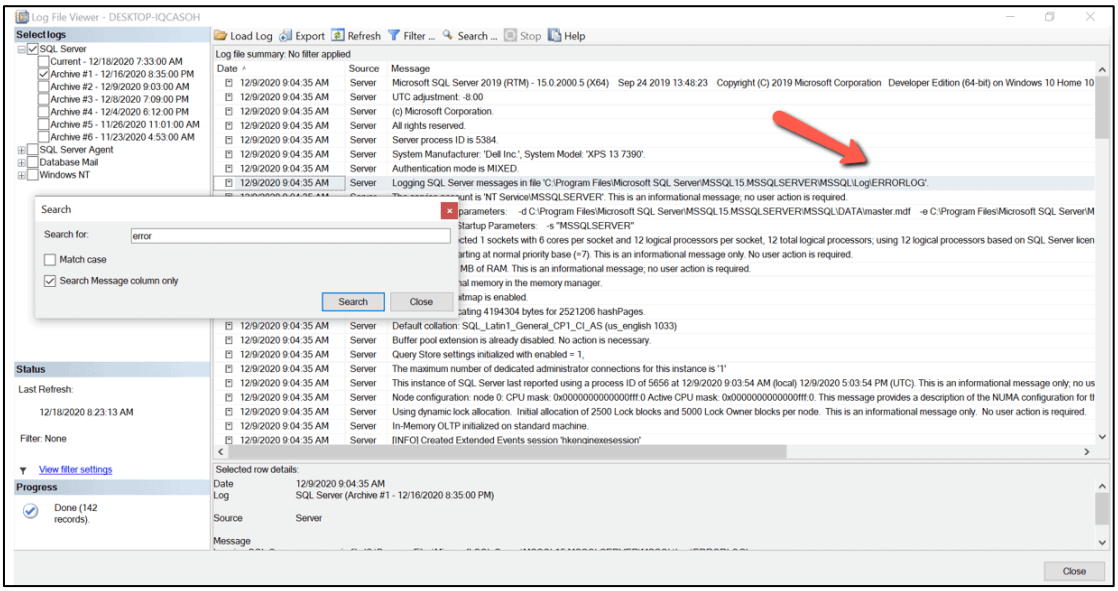


Figure 3.11 SQL Server Log

5. Exchange Server Logs

Microsoft Exchange Server provides email messaging services, the server creates logs to record events such as email delivery, mailbox access, client connections, and other activities. Exchange Server logs help administrators monitor, troubleshoot, and address email service problems. Commonly logs are saved under the following path

(<ExchangeInstallPath>\TransportRoles\Logs\) in Windows OS, a sample of logs is displayed in Figure 3.12 which shows the date, time, and session status.

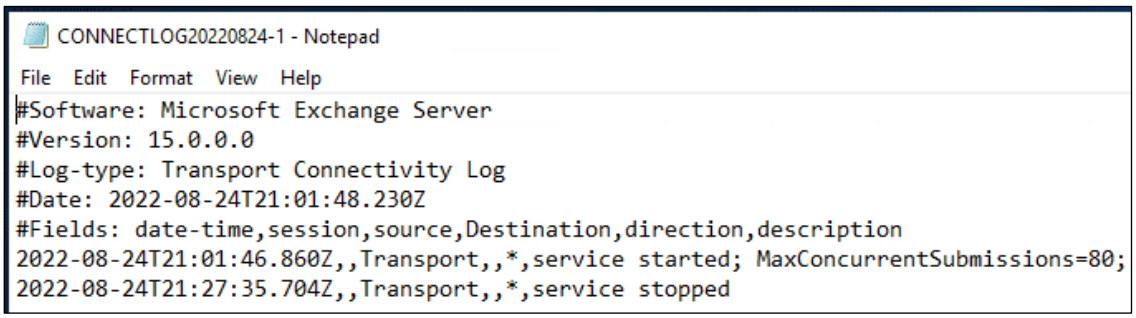


Figure 3.12: Exchange Server Log

6. Internet Information Services (IIS)

Web server logs are among the most critical logs on the global internet. Most web applications that run on a web server create logs of every web transaction and back-end services (database, FTP, operating algorithms, etc.). Those servers generally have big journals of logs. One of them is IIS which is running on the Windows OS, Figure 3.13 shows a sample of raw data of the logs of IIS.

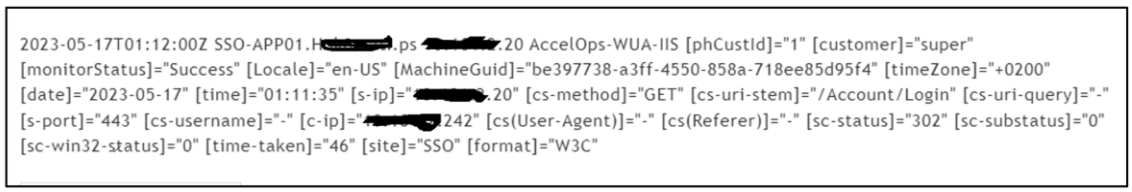


Figure 3.13: IIS Log

Due to the importance of the IIS log, the following Table 3.2 describes the extracted data of IIS log messages.

Table 3.2 IIS Log Messages Description

Field Description	Field Value
Date	2023-05-03T07:36
Server Name(Reporting Device)	SSO-APP01.HO****.PS
Server IP(Reporting IP)	X.X.X.20
Event Parser Name	AccelOps-WUA-IIS
PhCustId: ID of the Organization	1
customer	super
Monitor Status	Success

Locale	en-US
MachineGuid (UUID)	be397738-a3ff-4550-858a-718ee85d95f4
timeZone	200
date	03/05/2023
time	7:35:51
c-ip: Source IP	X.X.X.243
Source Host Name	HOST-10.101.2.243
cs-method (HTTP Method)	GET
cs-URI-stem (Uniform Resource Identifier)	/
cs-URI-query (the query being performed by the client)	=
s-port (port number)	443
cs-username	-
cs (User-Agent)	-
cs (Referer)	-
sc-status (HTTP Status)	500
sc-sub status (HTTP Sub-status)	0
sc-win32-status (HTTP Win32 Status)	0
Time-taken (the time it takes from when IIS receives the first byte in the request until it sends out the last byte in that request).	21
Site(Web Server Instance)	SSO
format	W3C

Event Type	AO-WUA-IIS-Web-Server-Error

The objective of extracting data is to understand each field of information in the incident analysis stage. Most of the fields contain data related to some type of attack. For instance, the (HTTPSstatus) code is about the status of the request, and if the response is 404 it means that the server cannot find the requested resources; repeated responses like this led to the conclusion that it is suspicious behavior from the client side.

7. FTP log

Different types of FTP log formats depend on the engine running this service. For example, IIS server generates FTP logs as shown in Figure 3.14. The following Table 3.3 describes each field in the log that appeared in the transaction.

Figure 3.14 FTP Log – IIS

The data in the above figure contains critical information related to FTP activity, including timestamps, client IP addresses, associated usernames, unique session identifiers, executed FTP commands (e.g., "GET" for /file/document.txt retrieval), and requested file paths. This data holds valuable information, particularly in the domain of network security. It can be utilized to analyze FTP usage patterns, identify potential security vulnerabilities, and enhance security measures to safeguard against unauthorized access or data breaches [91]. The following table describes each field in the logs in the previous figure.

Table 3.3 FTP Logon IIS Server Description

Field	**Description**
Date and Time	represents the date and time when the FTP activity occurred.
IP Address	"192.168.1.50" is the IP address of the client that initiated the FTP connection.
Hyphen (-)	Indicates that the log entry does not include information about the user accessing the FTP server.

Username	"USERNAME" represents the username associated with the FTP activity. In this example, the placeholder "USERNAME" is used.
[12345]	Represents the unique session ID or identifier associated with the FTP session.
FTP Command	"GET" indicates the FTP command that was executed. In this case, it retrieves a file from the server.
Requested File	"/files/document.txt" represents the file or directory path requested by the client.

In the next section, the study will review logs related to another operating system, specifically the UNIX-based platform.

1.1.1.2 Unix Based

Unix-based OS is widely used in data centers because of its reliability, adaptability, and security. The Unix OS was initially founded in 1991 by Linus Torvalds, and It has been developed into a potent platform that supports a variety of hardware, including personal computers, servers, and embedded systems [92]. Unix OS has exhibited better performance and stability than other operating systems, making it a preferable option for mission-critical applications. [93] emphasize Linux's scalability and capacity for effectively managing heavy workloads, making it appropriate for high-performance computing settings.

According to [94], the collaborative nature of Unix development, including a global community of developers, has led to constant innovation and rapid fixing of bugs. Unix systems have also been widely adopted in the business and academic sectors because they support open standards and are available in various architectures. Overall, Unix is a

stable and reliable operating system due to its strong foundation, community-driven development methodology, and ability to meet the diverse requirements of many domains.

As per the [94] survey, Unix is an open-source system enabling prompt patching of vulnerabilities and subjecting them to regular review. Linux's resilience to security risks is reinforced by its stringent access controls, privilege separation, and comprehensive auditing systems. The directory journaling at Unix logs goes under the path (/var/log) as illustrated in Figure 3.15

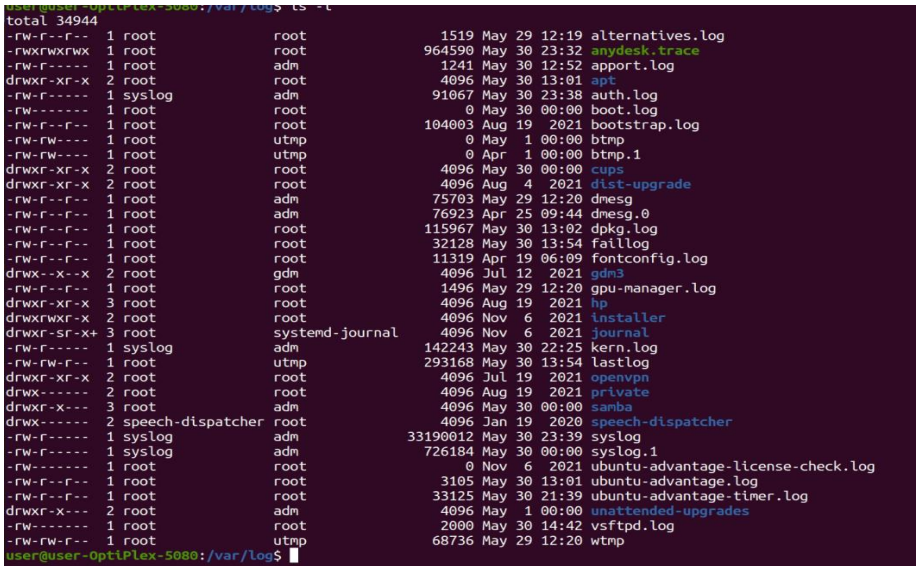


Figure 3.15 Linux Logs

The most critical logs were viewed frequently by the investigators to gain insight into the anomaly events that occurred on the system or application are described below:

1- Auth.log

Auth.log is a vital log file that includes information about authentication-related events. It is usually found in the (/var/log) directory and is owned by the root user which keeps track of all system access attempts, including successful and failed logins, password

changes, and user authentication failures [95], as shown in Figure 3.16. Analyzing this log is valuable for tracking user behavior and identifying security breaches.

Figure 3.16 Auth Log

2- Syslog

System logs often provide the most information about OS. It may be found at (/var/log/syslog), and it contains information not available in other logs. Figure 3.17 shows a sample of Syslog.

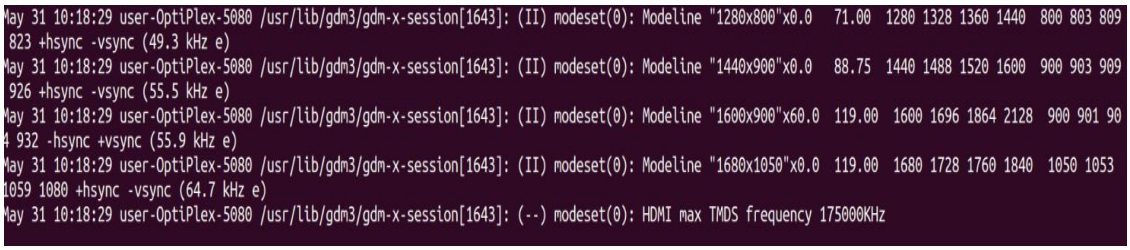


Figure 3.17 Syslog

Application Logs

On Unix-based machines, most applications generate logs, such as Apache, samba, vsftpd, and others. This section will provide an overview of the most common applications running on the OS and have logs. For instance, Apache server logs are stored in the default path (/var/log/apache2). These logs typically consist of two main types:

1. Access.log is stored under (/var/log/apache2) as shown in Figure 3.18 which has all records accessing web page URLs.

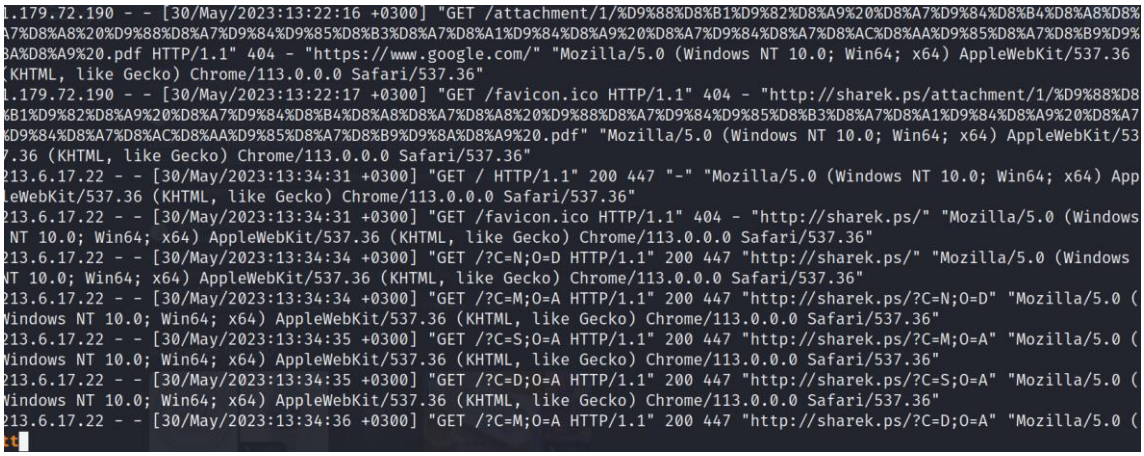


Figure 3.18 Access Log

2. Error.log path is (/var/log/apache2) as shown in Figure 3.19 and records all errors that occurred on the webserver.

Figure 3.19 Error.log

Firewall (FW) Log Message Fields

The FW device operates as the first defending perimeter and is based on the specified and predefined policies created on it. In addition, the Unified Threat Management (UTM) firewall makes decisions based on the definition of the security profile. Nowadays, most FW vendors are competing on AI features in anomaly detection as well as pre-defined policies. Figure 3.20 shows a sample security event of a blocking threat that was dropped during a transaction.

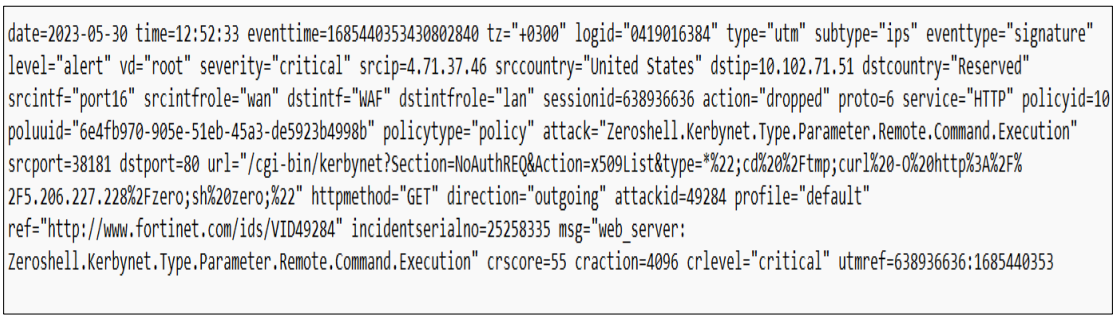


Figure 3.20 UTM Log

Here is the description of the parameters of the UTM log patterns as described in Table 3.4. The data log comprises a comprehensive record of network traffic, encompassing temporal details, device nomenclature, IP addresses, ports, and security ratings. Furthermore, it provides information about source and destination interfaces, geographical locations, protocols, session identification, and application details. Additionally, it includes metrics related to data transmission, such as bytes and packets sent and received, and records security actions taken. This data offers extensive information that can be leveraged for in-depth analysis of network traffic patterns and security incidents.

Table 3.4 UTM log field description

Field	Description
Date	Day, month, and year when the log messages were recorded.
time	The hour clock is when the log messages are recorded.
devname	device name for the traffic's origin.
David	The device's serial number is used to determine the traffic's origin.
eventtime	Epoch time FortiGate triggered the log.
tz	Time zone

logid	Log ID traffic
type	Log type (Traffic, Event, Anomaly, App, Antivirus, DLP, DNS, Email, GTP, IPS, VoIP, WAF, Web Filtering).
subtype	Subtype of the traffic.
level	Security level rating.
vd	Name of the virtual domain in which the log messages were recorded.
srcip	IP address of the traffic's origin.
srcname	Name of the source.
srcport	Port number of the traffic's origin.
srcintf	Interface name of the traffic's origin.
srcintfrole	Name of the source interface.
dstip	The destination IP address for the web.
dstname	Name of the destination.
dstport	Port number of the traffic's destination.
dstintf	Interface of the traffic's destination.
dstintfrole	Name of the destination interface.
srcuuid	UUID of the Source IP Address.
dstuuid	UUID of the Destination IP address.
srccountry	Name of the source country.
dstcountry	Name of the destination country.
sessionid	ID for the session.
proto	The protocol used by web traffic (tcp by default).
action	Status of the session. Uses following definitions

Deny	blocked by firewall policy
Start	session start log (a particular option to enable logging at the start of a session). This means a firewall is allowed.
All Others	allowed by Firewall Policy, and the status shows how it was closed.
policyid	Name of the firewall policy governing the traffic that caused the log message.
policytype	Type of the policy.
poluuid	UUID for the firewall policy.
policyname	Policy name in the firewall.
service	Name of the service.
trandisp	NAT translation type.
appid	ID of the application.
app	Name of the application.
appcat	Category of the application.
apprisk	Risk level of the application.
applist	Application Control profile name.
duration	Duration of the session.
sentbyte	Number of bytes sent.
rcvdbyte	Number of bytes received.
sentpkt	Number of packets sent.
rcvdpkt	Number of packets received.
utmaction	Security action performed by UTM.
dstdevtype	Device type of the destination.

dstosname	OS of the destination.

Conclusion

Understanding data logs and incidents related to devices and applications within the data center is mandatory for building the promised model of the research problem. Also, introducing the environment of logs including the type of incidents, protocols, and data management methods will facilitate the mission of the research. This chapter has an introduction and brief knowledge about the main protocols and types of logs that the model will investigate.

The collected data will be explored in the next chapter using data visualization tools to obtain an overview of the data and to analyze the relationships between the different variables.

4. Chapter Four Exploratory and Explanatory Data Analysis

Introduction

This chapter will employ data visualization techniques, utilizing a range of color-coded charts generated through SIEM solutions and the Plotly open-source library in Python. The objective of this chapter is to provide a comprehensive overview of the data and elucidate the interrelationships between variables. Data visualization tools have significantly streamlined this process by generating and disseminating insightful color-coded charts that succinctly encapsulate the underlying narrative. This section will go deeply into analyzing the data collected, as referenced in Chapter 3, which forms the basis for our proposed model.

First, this section will introduce the devices reporting logs to the FortiSIEM; these devices are integral to our model as considered feeders. As discussed in Chapter Two, they continuously transmit logs either based on time intervals or trigger-based events. The second section will talk about the manual extraction of logs.

FortiSIEM Reporting Devices

In general, SIEM systems are classified according to three main factors: Event Per Second (EPS), traffic volume, and processing incidents per second. Therefore, the more one of them, the higher SIEM rating. In this study, FortiSIEM is used for several reasons. Firstly, it is a device that works in the real environment in the data center. It relies on a virtual machine which allows the system administrator to assign resources as many as possible, also, most of the security devices are of the same type as SIEM.

The reporter devices are characterized by their heightened EPS metrics and serve as fundamental data sources, contributing significantly to the generation of a substantial volume of security events and logs [96]. By investigating the foremost reporting devices based on EPS parameters, analysts can acquire insights into the most active and influential components within their network infrastructure. This knowledge empowers them to concentrate on optimizing performance and security. Here, the illustrations on Figure 4.1 provide a comprehensive understanding of the EPS ratings of reporting devices that enables analysts to prioritize resource allocation, optimize event processing workflows, and proactively address potential security vulnerabilities associated with high-impact devices.

Reporting Device	MAX(Event Rate)	AVG(Event Rate)	MIN(Event Rate)
PAloAlto 2	11522.83	62.08	0.01
PAloAlto 1	10453.01	108.11	0.01
Cloud-Firewall-Primary	3740.83	480.52	0.01
MOH_FG-200F-Right	2451.71	9.25	0
ADMIN-APP03	995.74	3.2	0
PG-APP02-WEB	971.72	5.46	0
PG-APP01-WEB	878.41	5.57	0
SSO-APP02	856.78	1.73	0
PG-APP04-WEB	850.65	5.4	0
ADMIN-APP01	841.44	3.31	0
ADMIN-APP02	817.47	5.59	0
EH-APP02-WEB	787.44	3.64	0
SSO-APP01	784.1	9.91	0
PG-APP03-WEB	778.51	5.22	0
EH-APP01-WEB	739.27	5.46	0
EH-APP03-WEB	738.55	3.44	0
gw	706.31	32.28	0
Epayment-left	689.16	47.69	0.01
PalcertFW	610.16	14.52	3.38

Figure 4.1 Top EPS Devices

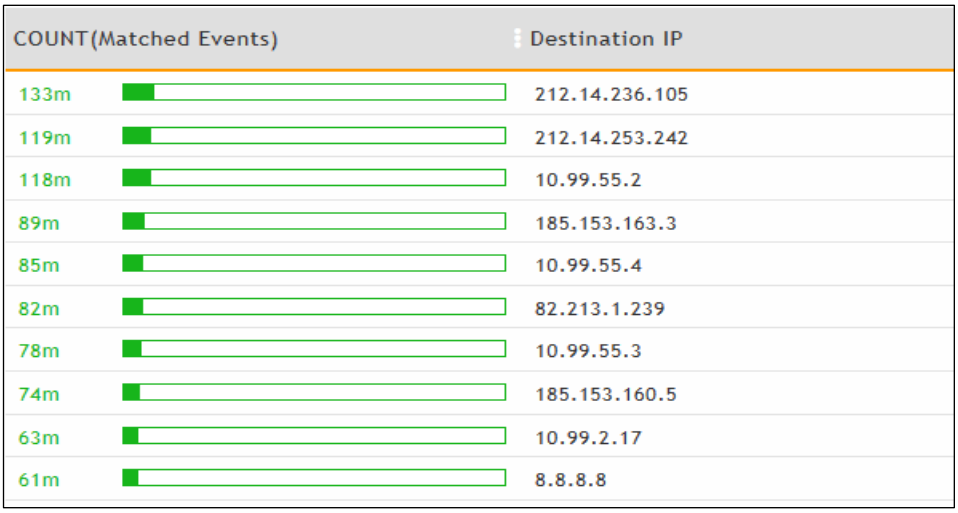
The following Table 4.1 describes the reporting device in SIEM repository database including 18 devices varied from FW, WAF, SIEM agents and servers. All of them send logs to the SIEM and these logs have incidents.

Table 4.1 Reporting Device Definition

Reporting Device	Device Name
PaloAlto 2	Edge Firewall (MTIT)
PaloAlto 1	Edge Firewall (MTIT)
Cloud-Firewall-Primary	UTM Firewall (MTIT)
MOH_FG-200F-Right	MOH firewall
ADMIN-APP03	E-Payment-Collector
PG-APP02-WEB	E-Payment-Collector
PG-APP01-WEB	E-Payment-Collector
SSO-APP02	E-Payment-Collector
PG-APP04-WEB	E-Payment-Collector
ADMIN-APP01	E-Payment-Collector
ADMIN-APP02	E-Payment-Collector
SSO-APP01	E-Payment-Collector
PG-APP03-WEB	E-Payment-Collector
EH-APP01-WEB	E-Payment-Collector
EH-APP03-WEB	E-Payment-Collector
GW	Forti Analyzer for MOFA
E-payment-left	E-payment Firewall
PalcertFW	PALCERT Firewall

The data in Table 4.1 shows the prominent IP addresses that were recipients of many incident events. The frequency of these events serves as a multifaceted indicator of

the incident, with particular emphasis on the significance of a DDOS attack. The graphical representation effectively delineates the primary IPs that have garnered over 100 million hits within a condensed Domain Name System (DNS) and were subsequently employed to reroute DNS queries from Google servers (8.8.8.8) to the National servers located in Palestine.



COUNT(Matched Events)	Destination IP
133m	212.14.236.105
119m	212.14.253.242
118m	10.99.55.2
89m	185.153.163.3
85m	10.99.55.4
82m	82.213.1.239
78m	10.99.55.3
74m	185.153.160.5
63m	10.99.2.17
61m	8.8.8.8

Figure 4.2 Top Destination IP

The general overview of Figure 4.2 provides insight into the top destination IP reported in 8 hours, revealing that both external and internal IP addresses were being targeted. For instance, public IP addresses ending with (105, 242) saw the highest incoming traffic, and these IP addresses are known to be assigned to MTIT DNS and routed to GCC, indicating that the traffic is targeting data center IP addresses. While it is considered normal behavior if the traffic volume remains under ML control, if it exceeds the 133M threshold, it could indicate a potential DDoS attack.

Event Name	COUNT(Matched Events)	Event Severity
FortiGate appctrl ips pass	884k	1
FortiGate event dns ftgd cat allow	191k	3
Unknown event type	103k	1
FortiGate event dns url filter allow	50k	1
FortiWeb-20000017	38k	7
Unknown event type	35k	3
Generic FortiGate Event	32k	5
Server certificate is resigned as untrusted	22k	7
Firewall invalid traffic denied	22k	1
Generic FortiGate Event	13k	3

Figure 4.3 Top FortiSIEM External Events by Severity Count

The visual representation is shown in the above Figure 4.3 showcases the foremost external security events extracted by the SIEM during comprehensive monitoring of logs emanating from all assets in the data center. The aforementioned figure illustrates the cumulative peak of events generated by these devices over an 8-hour interval. Based on the chart examination, event frequency analysis regardless of severity reveals that FORTINET's IPS successfully passed 884K events with severity 1, indicating low severity. Conversely, an event categorized as critical with a severity rating of 7, about the server certificate being deemed untrusted, of 22K instances. It is imperative to note that the event with a severity rating of 1 holds greater significance than the event with a severity rating of 7, thereby warranting attention to the potential occurrence of a DDOS notification.

Event Name	Event Severity	Source IP	COUNT(Matched Events)
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.2.110	13k
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.1.250	7213
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.2.160	6362
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.13.11	6202
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.13.41	6104
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.254.9.53	5860
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.99.13.47	5472
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.98.68.150	3122
FortiSIEM detected host names created via Domain Generation Algorithm	5	10.98.67.61	3079

Figure 4.4 Hosts Create DGA

The above data in Figure 4.4, was taken during 8 hours of monitoring showing the events under the Domain Generation Algorithm (DGA) detection. Attackers generate new domain names with IP to make it difficult for threat hunters to detect the attacks. This is one of the most dangerous functions for executing Command and Control (CC / C2) malware.

The detection of DGAs has been a major area of interest in business solutions in the field of information security. Technicians have focused on developing and refining detection methods to identify and mitigate the threats posed by DGAs. The research [97] investigated techniques, such as ML algorithms, statistical analysis, and behavioral modeling, to accurately identify DGAs within network traffic and malware. Additionally, the previous academic study has emphasized the importance of real-time detection and response mechanisms to counter the dynamic nature of DGA-generated domains. Overall, it should play a crucial role in enhancing the understanding of DGAs and devising robust detection strategies to safeguard against their malicious activities.

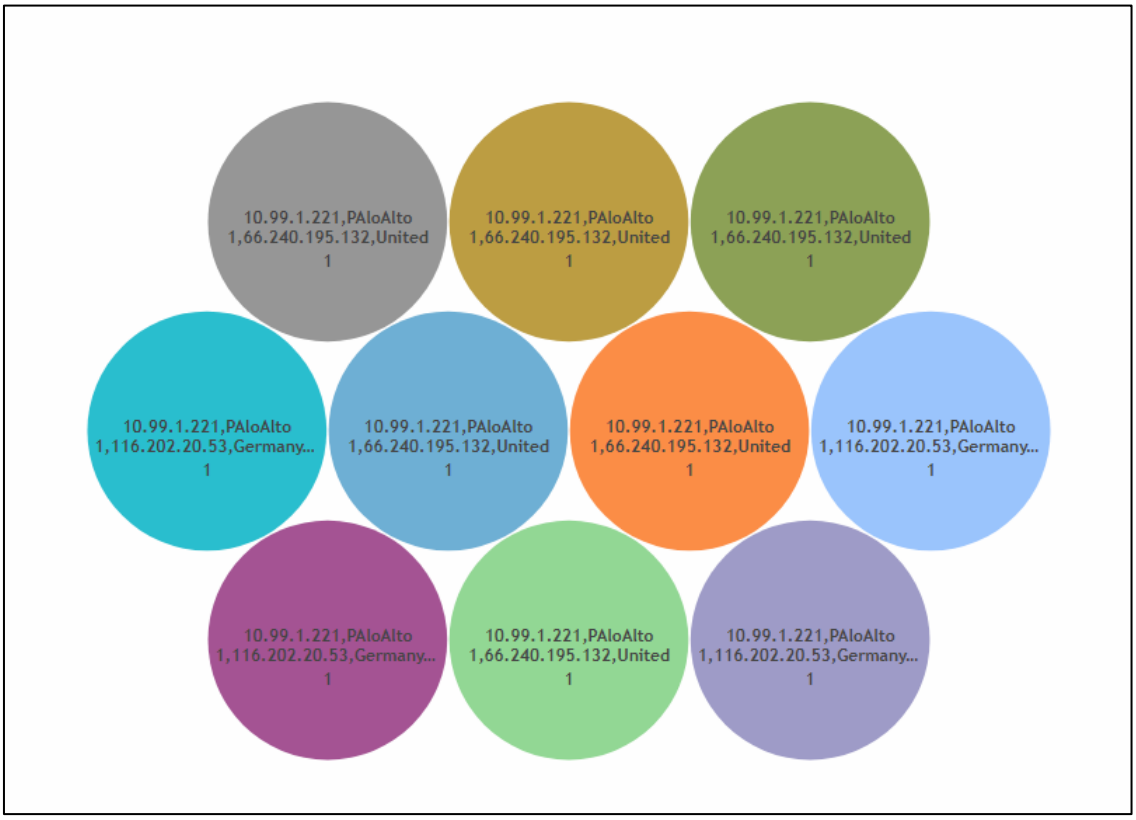


Figure 4.5 AXFR Protocol –DNS Zone Transfer

Misconfigurations in DNS settings can render DNS servers vulnerable to various types of attacks and potential data leaks. For instance, one such vulnerability involves the exposure of internal network structures when responding to Asynchronous Transfer Full Range (AXFR) requests. This form of attack, known as DNS zone transfer, can lead to unauthorized access to sensitive information, including local DNS and network IPs, resulting in significant data leakage [98]. It is essential to promptly address and rectify any misconfigurations to mitigate the risk of such security breaches and prevent unauthorized access to sensitive network information. Figure 4.5 emphasizes the trying of zone transfer in which the Palo Alto device reported those incidents, it is found that the same logs were reported from agents on DNS servers with more details.

Source IP	Source Host Name	Destination IP	Destination TCP / UDP Port	COUNT(Matched Events)
[redacted].11	EXT-DNS01	212.14.236.105	53 (DOMAIN)	8916
[redacted]5.12	EXT-DNS02	82.213.1.239	53 (DOMAIN)	7284
[redacted]5.11	EXT-DNS01	82.213.1.239	53 (DOMAIN)	7075
[redacted]6.5	HOST-10.101.16.5	8.8.8.8	53	6048
[redacted]5.12	EXT-DNS02	212.14.236.105	53 (DOMAIN)	6015
[redacted]6.5	HOST-10.101.16.5	8.8.8.8	53 (DOMAIN)	5328
[redacted]0.40	[redacted]K	142.250.180.174	443 (HTTPS)	2438
[redacted]8.139	[redacted]EN	142.251.37.238	443 (HTTPS)	2170
[redacted]3.2	HOST-10.101.13.2	8.8.8.8	53 (DOMAIN)	2016
[redacted]53	HOST-10.254.9.53	213.244.72.26	53 (DOMAIN)	1731

Figure 4.6 Firewall Deny – Outbound

The Figure 4.6 presents a table view detailing the top denied traffic events caused by implicit or custom policies from internal IPs to outbound WAN. This view allowed us to identify any potentially suspicious traffic originating from servers, botnets, infected systems, or human intervention. Upon analysis, it has been discovered that the primary denied traffic originated from a local DNS with the IP address ending with (x.x.x5.11) attempting to communicate with the national autonomous system 212.14.236.105. The UTM blocked this connection because it did not permit a nonauthoritative transfer between the local DNS and the public one. This concern shall be raised for the SOC team regarding the suspicious nature of the incident.

Source IP	Event Name	COUNT(Matched Events)	Event Severity
10.99[redacted]	FortiGate appctrl ips block	627	10
10.99[redacted]	FortiGate appctrl ips block	626	10
213.[redacted]	FortiGate appctrl ips block	363	10
10.99[redacted]	FortiGate appctrl ips block	186	10
10.99[redacted]	FortiGate appctrl ips block	182	10
213.6[redacted]	FortiGate appctrl ips block	158	10
213.24[redacted]	FortiGate appctrl ips block	148	10
83.244[redacted]	FortiGate appctrl ips block	132	10
10.9[redacted]	FortiGate appctrl ips block	56	10
172.16[redacted]	FortiGate appctrl ips block	32	10

Figure 4.7 Top Source IP, Events by Severity

During the 60-minute interval within working hours in Figure 4.7, the listed IPs were observed as sources attempting unauthorized requests. This determination was based on the FortiGate application control IPS, which signified that these IPs encompassed both private and public addresses endeavoring to access prohibited applications, violating internal use policies. When private IPs attempt to access applications within legal parameters, it is expected that such attempts occur minimally. However, if there are multiple repeated attempts, it necessitates further investigation by analysts to determine whether the device is compromised or exhibiting other suspicious behaviors. Furthermore, in accordance with best practices, it is advisable not to assign public IPs to machines directly. Instead, utilizing a mapping of public IPs to private IPs through Virtual IP (VIP) or other mapping techniques in firewalls serves to obfuscate the private network from the public domain, thereby making the discovery of the internal network architecture is difficult.

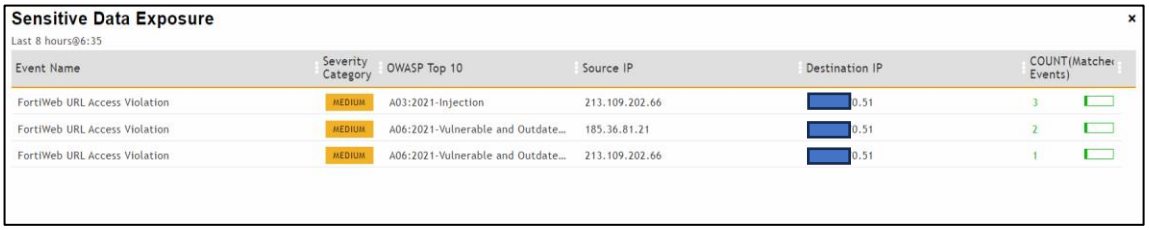
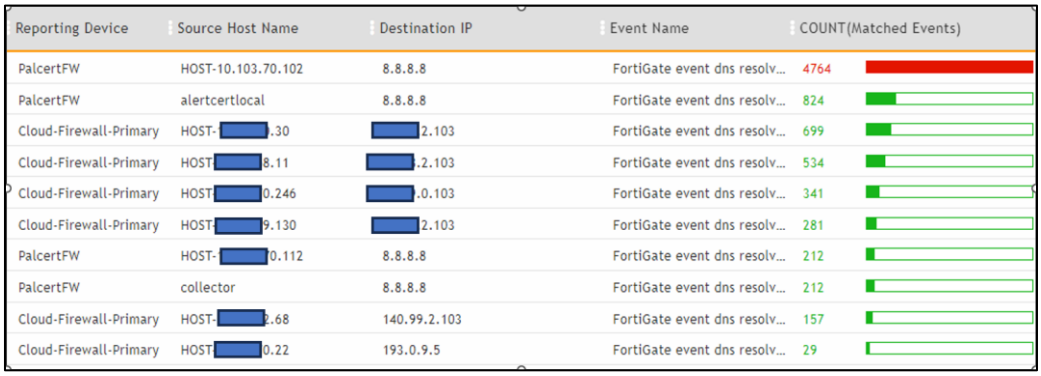


Figure 4.8 Sensitive Eata Exposure

WAF is designed to mitigate a wide range of cyber threats that target web applications positioned at layer seven of the OSI model, WAFs are ideally deployed in front of servers and below UTM Firewalls for comprehensive protection. By leveraging a well-configured policy, WAFs can effectively detect and prevent SQL injection attacks, for example, a common method used by attackers is to manipulate databases through web applications as show on the traffic above in Figure 4.8. Additionally, WAFs are adept at identifying security weaknesses stemming from the use of outdated libraries and other application vulnerabilities, providing an added layer of defense against potential exploits [99].

DNS and mDNS

Keeping careful monitoring and securing of DNS traffic is vital to prevent potential abuse through DNS amplification attacks or other DNS-related vulnerabilities. Proactive monitoring of network traffic is essential for identifying unusual patterns and ensuring that all exposed services are fortified against potential exploits. By comprehending the distribution of source IPs across various ports, network administrators can prioritize securing the most targeted services and bolster their network's overall security posture.



Reporting Device	Source Host Name	Destination IP	Event Name	COUNT(Matched Events)
PalcertFW	HOST-10.103.70.102	8.8.8.8	FortiGate event dns resolv...	4764
PalcertFW	alertcertlocal	8.8.8.8	FortiGate event dns resolv...	824
Cloud-Firewall-Primary	HOST-[redacted].30	[redacted]2.103	FortiGate event dns resolv...	699
Cloud-Firewall-Primary	HOST[redacted]8.11	[redacted].2.103	FortiGate event dns resolv...	534
Cloud-Firewall-Primary	HOST[redacted]0.246	[redacted].0.103	FortiGate event dns resolv...	341
Cloud-Firewall-Primary	HOST[redacted]9.130	[redacted]2.103	FortiGate event dns resolv...	281
PalcertFW	HOST-[redacted]0.112	8.8.8.8	FortiGate event dns resolv...	212
PalcertFW	collector	8.8.8.8	FortiGate event dns resolv...	212
Cloud-Firewall-Primary	HOST-[redacted]2.68	140.99.2.103	FortiGate event dns resolv...	157
Cloud-Firewall-Primary	HOST[redacted]0.22	193.0.9.5	FortiGate event dns resolv...	29

Figure 4.9 DNS Resolving Error

Another type of DNS attack is flooding DNS with a substantial influx of spurious traffic is a deliberate method aimed at overwhelming the DNS server, effectively impeding its ability to resolve legitimate domains. Consequently, the DNS becomes temporarily or even permanently unavailable. Monitoring such incidents gives valuable insights into the characteristics of the attack, offering valuable information regarding potential misconfigurations and vulnerabilities exploited by attackers [100]. Figure 4.9 shows the real of the study dataset where the DNS resolves illegal domains, sometimes trying to resolve local IPs, this indicator shows a most there are malware trying to make reconnaissance on networks.

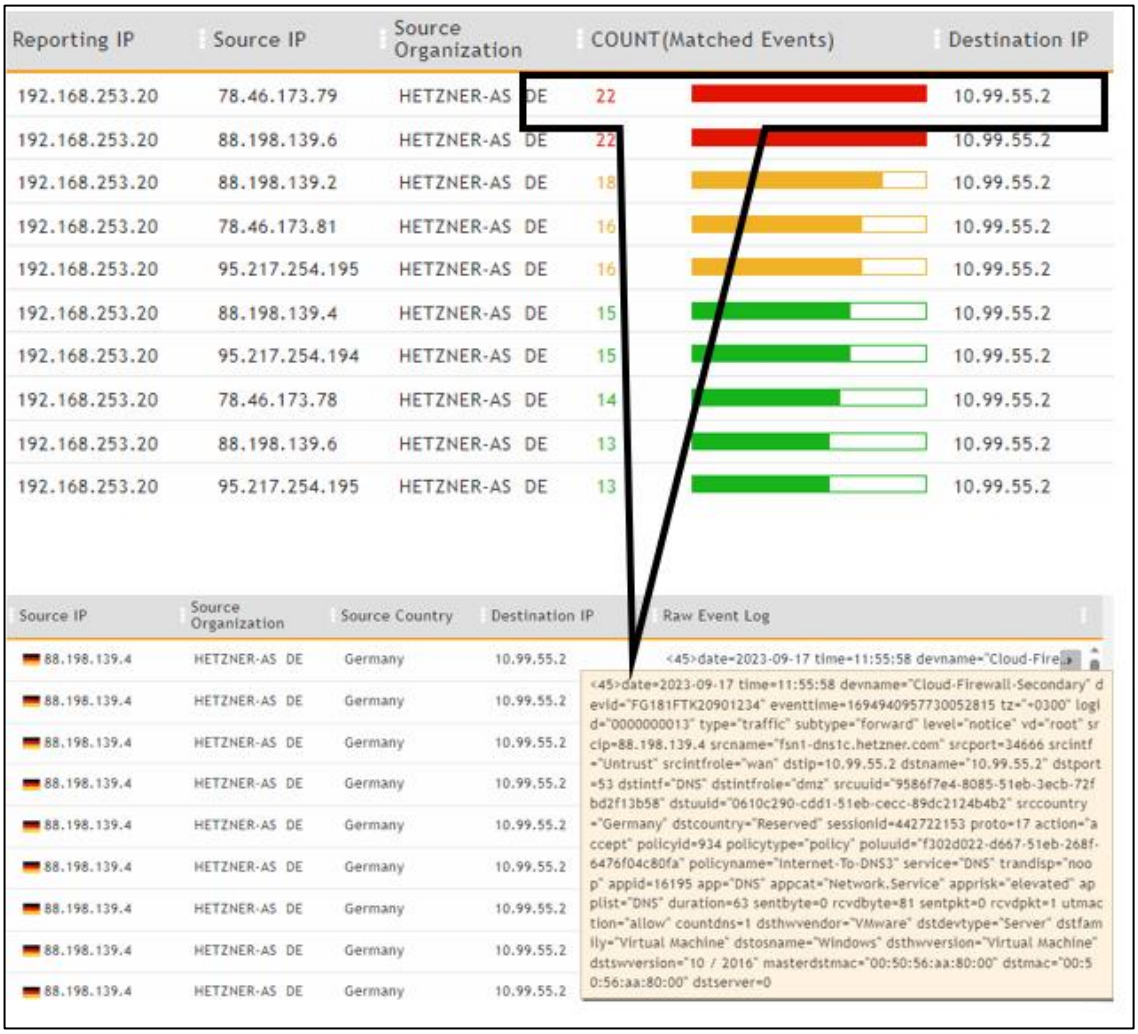


Figure 4.10 Attacks Against DNS

The provided Figure 4.10 presents a conspicuous spike in DNS traffic originating from specific internet service providers (ISP). In particular, HETZNER, an ISP located in Germany, has been identified as a prominent source of this heightened DNS traffic. This flow in traffic is attributed to the genuine IP range associated with HETZNER. In response, comprehensive measures have been established to proactively block all traffic emanating from Hetzner. Subsequently, a rigorous monitoring framework has been established to scrutinize the genuine services originating from this source, ensuring the facilitation of authorized and legitimate transactions.

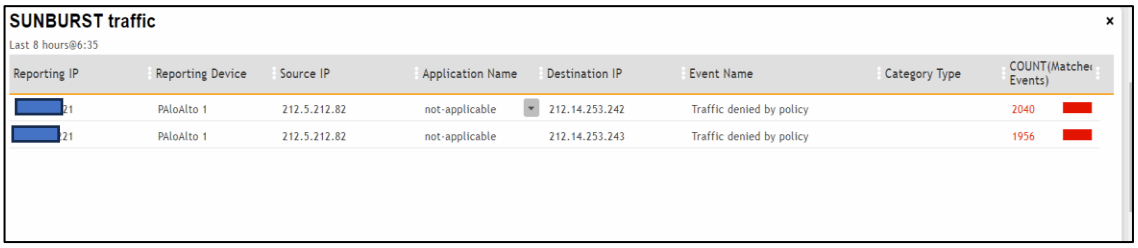
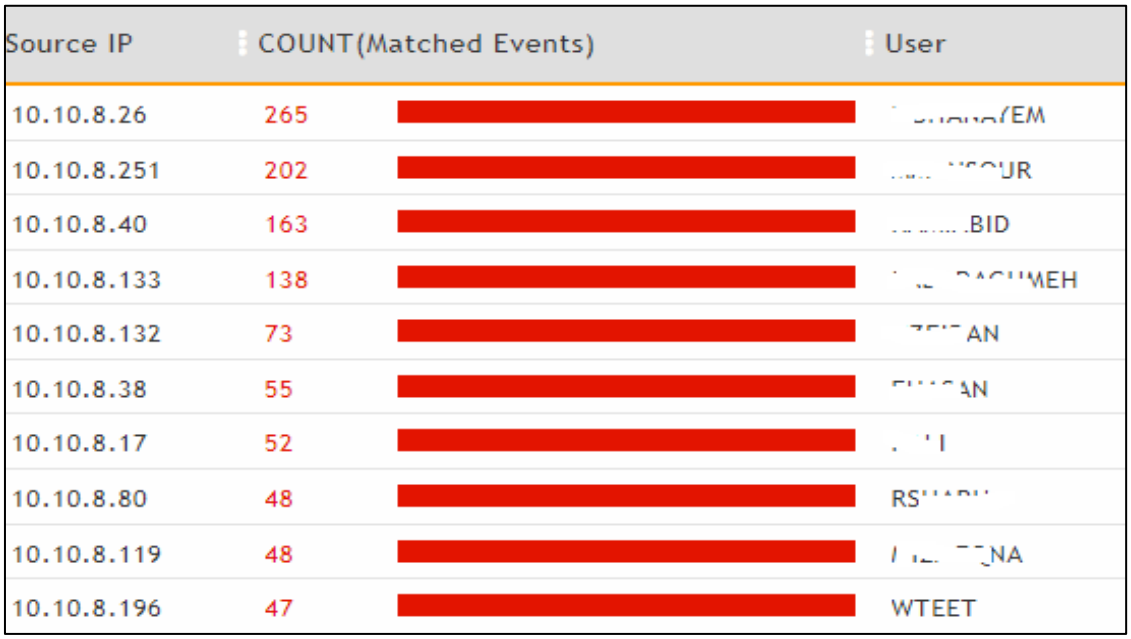


Figure 4.11 Sunburst Traffic

Most data centers run different tools for monitoring the network performance; the attackers definitely will exploit the vulnerabilities of known products running on some in data centers. One of these attacks is the Sunburst attack, also known as the SolarWinds cyberattack, was a highly sophisticated supply chain attack discovered in late 2020 [101]. This attack involved the insertion of a malicious code into the SolarWinds Orion software. The compromised software was then distributed to numerous organizations, including government agencies and major corporations, allowing the attackers to gain unauthorized access to sensitive data and systems. The Sunburst attack is a significant threat highlighting the far-reaching impact of supply chain vulnerabilities in the realm of cybersecurity. This attack underscored the need for enhanced security measures and vigilance in the face of evolving cyber threats, particularly those that exploit trusted relationships within supply chains [102].

The network has experienced a notable surge in malicious attacks resembling the Sunburst malware, despite the organization's cessation of SolarWinds software usage, as shown in Figure 4.11. Although the software was previously installed and subsequently uninstalled, the persistent presence of the malware in the network remains unremedied. While the PaloAlto device detected the malware, IDS agents did not. Consequently, a

function of the proposed model is taking measures to advise the remaining agents and IDS within the framework about the unauthorized nature of this code. Then, will execute a "playbook" concept to implement appropriate detection actions.



Source IP	COUNT(Matched Events)	User
10.10.8.26	265	[illegible]/EM
10.10.8.251	202	[illegible]UR
10.10.8.40	163	[illegible].BID
10.10.8.133	138	[illegible]MEH
10.10.8.132	73	[illegible]AN
10.10.8.38	55	[illegible]AN
10.10.8.17	52	[illegible]
10.10.8.80	48	RS[illegible]
10.10.8.119	48	[illegible]NA
10.10.8.196	47	WTEET

Figure 4.12 Successful Network Login

Data shown in the illustration Figure 4.12 shows successful login counts. multiple successful attempts to log in through network devices. This could signify legitimate logins or successful brute-forcing. Upon closer examination of the figure, it is evident that a username ending “EM" has successfully logged into the device with the IP address (10.10.8.26), a total of 256 times within an 8-hour past. It's important to note that these IPs are assigned to personal computers, and they are not for applications or servers. This behavior of the logged information raises concerns due to its abnormal pattern. Therefore, it is crucial for investigators and information security specialists in the organization to conduct a comprehensive inspection of all the devices mentioned above to ascertain the nature of these logins and address any potential security risks.

Source IP	User	COUNT(Matched Events)	Reporting Device
10.98.70.30	[redacted]	66	Cloud-Firewall-Primary
10.98.70.26	[redacted]	25	Cloud-Firewall-Primary
10.255.0.2	[redacted]	22	Cloud-Firewall-Primary
10.99.1.1	[redacted]	14	PAloAlto 1
10.99.1.1	[redacted]	13	PAloAlto 2
10.99.1.1	[redacted]	10	PAloAlto 1
10.10.8.40	admin	9	MOFA
10.98.70.9	[redacted]	8	Cloud-Firewall-Primary
10.99.1.1	[redacted]	8	Epayment-left
10.212.134.201	[redacted]	6	Cloud-Firewall-Primary
10.98.70.30	Admin	5	Cloud-Firewall-Primary
10.98.70.30	unknown	5	Cloud-Firewall-Primary
192.168.6.230	admin	4	MOH_FG-200F-Left
10.98.70.5	[redacted]	3	Cloud-Firewall-Primary
10.98.70.10	[redacted]	3	Cloud-Firewall-Primary
10.98.70.10	[redacted]	3	Cloud-Firewall-Primary
10.98.70.11	[redacted]	3	Cloud-Firewall-Primary
10.98.70.26	[redacted]	3	Epayment-WAF-Secondary
10.98.70.27	[redacted]	3	Cloud-Firewall-Primary
10.98.70.30	[redacted]	3	Epayment-WAF-Primary

Figure 4.13 Failed Network Login

On the other hand, the analysis of failed login attempts within the 60-day data presented in the Figure 4.13 above reveals patterns indicative of potential brute-forcing activities. These results were compiled over a 60-day period and all the attempts originated from the internal LAN. Notably, the users behind these attempts were deemed legitimate based on the available logs. In the realm of security, it is crucial to consider whether the intended user was indeed behind these failed attempts or if their devices may have been compromised by malware. To gain further clarity, a thorough examination of

corresponding logs related to the detection of malware vulnerabilities within the IDS logs will be essential in making an informed decision.

Reporting IP	User	COUNT(Matched Events)
10.2.10.2	adam	1
10.2.10.2	audit	1
10.2.10.2	tech2	1
10.2.10.2	temp	1
10.103.70.1	'ittmtgt	1
10.103.70.1	aaadmin	1
10.103.70.1	abs	1
10.103.70.1	absadmin	1
10.103.70.1	absssl	1

Figure 4.14 Failed VPN Login

When trying to match related logs to each other, a significant indicator arises from VPN connection attempts. The observational data spanned approximately one week, as shown in Figure 4.14. The frequency of these attempts serves as a crucial metric in assessing the vulnerability of my perimeter IPs to potential attackers. A high number of intrusions attempts within a specific short timeframe triggers heightened risk. Consequently, this is indicative of low severity and warrants appropriate assessment within the broader context of network security.

Reporting IP	User
10.2.10.2	adam
10.2.10.2	admin
10.2.10.2	audit
10.2.10.2	temp
10.2.10.2	test
10.103.70.1	accountant
10.103.70.1	accounting
10.103.70.1	adam
10.103.70.1	adminbujl
10.103.70.1	administrator

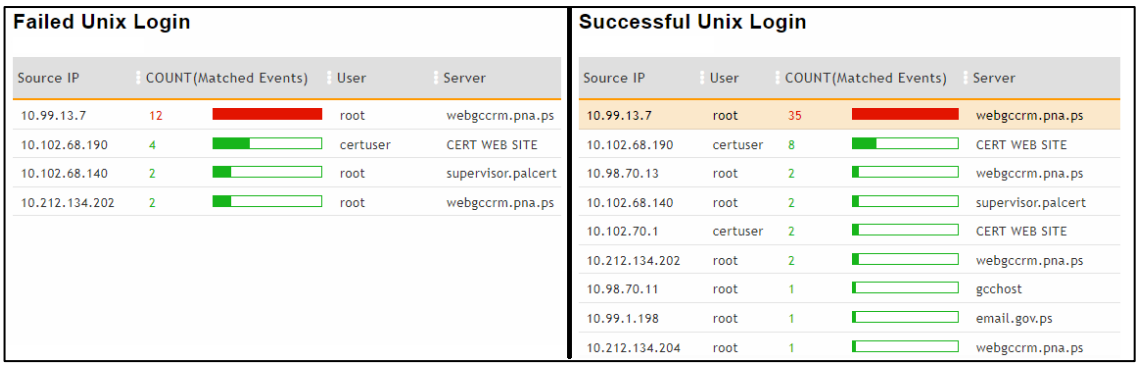
Figure 4.15 Failed Login - VPN

Besides the above analyses, another examination will take place here in the username used to login; the preceding figure illustrates attempted VPN logins. Upon analyzing the incoming traffic to the edge devices via public IPs, it was observed that users were attempting to log in using unusual usernames. While minor typographical errors in usernames are conceivable, the entry of entirely unfamiliar usernames such as "adminbujl," is not typical and inconsistent with the department's authorized usernames of the VPN users on FW (10.103.70.1). Consequently, it is imperative for the analyst to conduct vigilant monitoring of successful logins on the specific VPN.

Server	User	COUNT(Matched Events)
Cloud-Firewall-Primary		102

Figure 4.16 Account Lockout

Over a 30-day, the cloud firewall shows the number of users has been blocked, as illustrated in Figure 4.16. The system was configured to trigger an account lock after three consecutive failed login attempts within a 300-second window. Consequently, if the number of attempts surpassed this threshold, the respective account would be locked. The data presented on the above screen indicates that over the 30-day period, a total of 102 accounts were locked. This suggests that multiple accounts and various resources were encompassed within this count. Another crucial issue within the fusion model pertains to the persistence of user connections through alternative routes, even when the account has been locked. Thus, a review of the existing policy is needed and the recommendation here to activate ML for a couple of weeks to take the appropriate actions next time.



Failed Unix Login

Source IP	COUNT(Matched Events)	User	Server
10.99.13.7	12	root	webgccrm.pna.ps
10.102.68.190	4	certuser	CERT WEB SITE
10.102.68.140	2	root	supervisor.palcert
10.212.134.202	2	root	webgccrm.pna.ps

Successful Unix Login

Source IP	User	COUNT(Matched Events)	Server
10.99.13.7	root	35	webgccrm.pna.ps
10.102.68.190	certuser	8	CERT WEB SITE
10.98.70.13	root	2	webgccrm.pna.ps
10.102.68.140	root	2	supervisor.palcert
10.102.70.1	certuser	2	CERT WEB SITE
10.212.134.202	root	2	webgccrm.pna.ps
10.98.70.11	root	1	gcchost
10.99.1.198	root	1	email.gov.ps
10.212.134.204	root	1	webgccrm.pna.ps

Figure 4.17 Success / Failed Unix Login

In the above Figure 4.17 its obviously observed that two distinct critical values shown in both images, one for successful logins and the other for failed logins. The significance of these critical values in contrasting scenarios is noteworthy. A high number of failed login attempts indicates potential credential guessing, while an exceptionally high frequency of root user logins to a specific server is considered critical, as it may signify automated access, given that human logins typically involve a time delay for password entry and access.

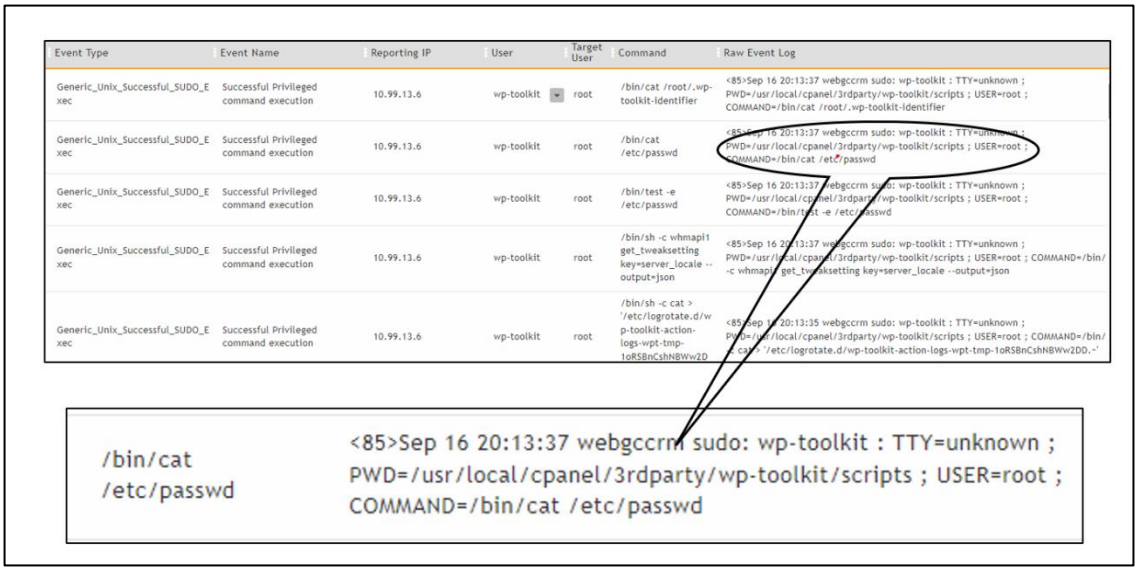


Figure 4.18 Unix Server Privileges Command Execution

Upon investigation data in Figure 4.18 , it was discovered that malicious commands were executed by root user account, resulting in a successful password change. Despite multiple commands aimed at acquiring the user list, the command (/bin/test—e/etc/passwd) manages to update the root user's password. It is strongly recommended to promptly alert the administrator about this incident, as they were unaware of these activities. Subsequent remedial action was initiated following a thorough analysis of the situation.

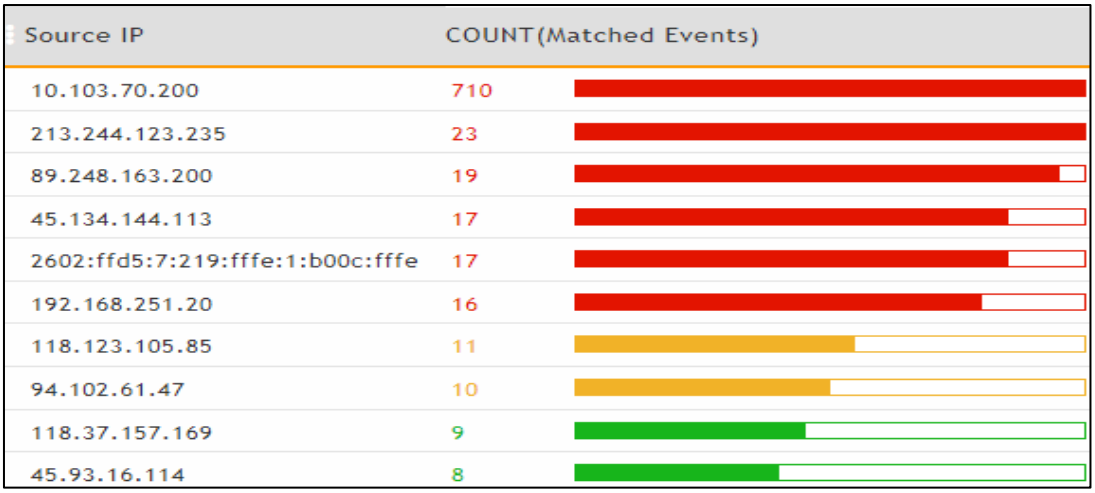


Figure 4.19 Top Scanners

Over the 24 hours, the identified IPs have been actively scanning the network terminals for discovery purposes, with the high frequency of scans indicating an extensive data gathering effort. In this context, "scans" refer to the periodic checks performed by devices to assess the connectivity of network terminals parallel to a heartbeat function. Instances of low scan frequencies may suggest infrequent reporting by devices or potential malfunctions within the network. It is imperative for the administrator to address these abnormalities and ensure the proper functioning of the affected devices.

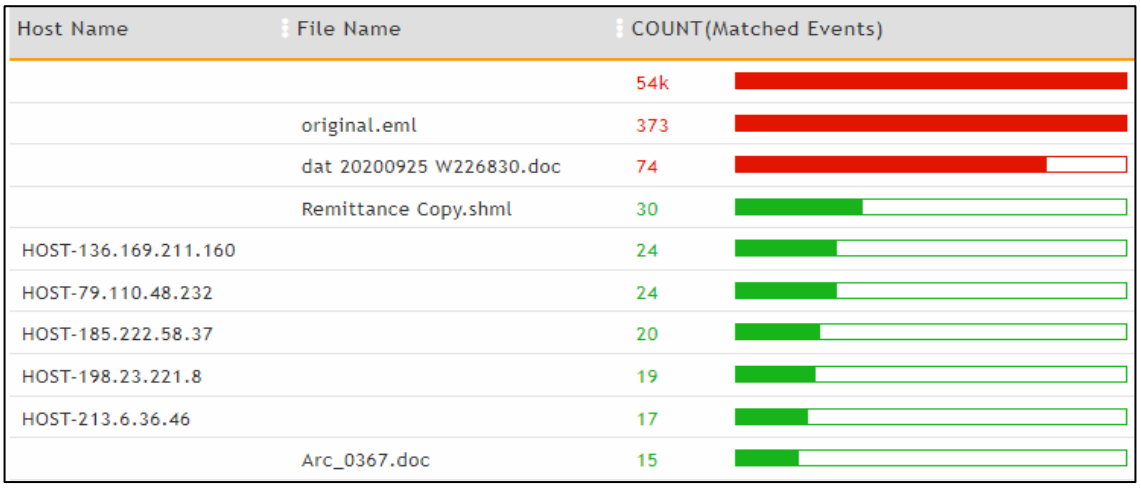
The high number shown in the above Figure 4.19 signifies the maximum frequency of scans conducted by the specific IP. If this activity is associated with a known function on a recognized device, it is considered normal. However, if it pertains to an unknown device, it may indicate a potential security incident. To address this, the function should be defined during the ML time to detect abnormalities, and subsequently, appropriate actions should be relayed to other nodes within the security infrastructure.

Reporting IP	Source IP	COUNT(Matched Events)	Event Severity
192.168.253.20	fe80::b3a8:8cbe:2fff:7655	194k	3
192.168.253.20	fe80::96dc:60ad:b0b4:d0b5	96k	3
192.168.253.20	10.98.69.18	28k	3
192.168.253.20	fe80::f88c:4c98:697f:b424	19k	3
192.168.253.20	fe80::2a1f:9982:30d9:9060	18k	3
172.16.16.1	10.70.99.238	18k	3
192.168.254.20	10.102.20.20	18k	3
192.168.254.20	10.101.24.12	17k	3
192.168.253.20	fe80::22e3:42be:62f1:c250	16k	3
192.168.253.20	10.98.70.14	13k	3

Figure 4.20 Blocked Attacks

During regular working hours, as illustrated in Figure 4.20 , the predominant portion of denied traffic originates from two IPv6 sources, indicating that these servers are

configured with IPv6 or the administrator by mistake leave IPv6 enabled. As the organization does not currently adhere to an IPv6 policy within its network, the denied traffic is attributed to the UTM and WAF. Additionally, other devices engaging in illegal traffic are denied based on implicit policy or due to the submission of erroneous requests. To address this, it is imperative to ensure the proper configuration of the Network Interface Card (NIC) on all servers to align with organizational standards and thoroughly monitor the overall denied traffic.



Host Name	File Name	COUNT(Matched Events)
		54k
	original.eml	373
	dat 20200925 W226830.doc	74
	Remittance Copy.shml	30
HOST-136.169.211.160		24
HOST-79.110.48.232		24
HOST-185.222.58.37		20
HOST-198.23.221.8		19
HOST-213.6.36.46		17
	Arc_0367.doc	15

Figure 4.21 Malware Found by Endpoint Protection

Another log from endpoint protection software the events without host names in Figure 4.21 are generated by the Email server's Endpoint protection. These events are indicative of another stream of logs from the Endpoint protection, categorized as malware. Consequently, the reported traffic pertains to detected malware. Upon delving into the traffic originating from the email server, which has been classified as malware, an investigation revealed that the alerts are 100% genuine. This investigation also unveiled the presence of suspicious file activities during this transaction. Another type of malware

detection comes from network devices; for 24 hours, SIEM received the above incidents from network protection software as shown in the Figure 4.22.

Source IP	COUNT(Matched Events)
213.6.146.188	306
213.6.17.210	40
10.255.3.29	18
185.38.142.138	13
10.102.68.180	11
45.137.22.171	11
213.6.248.146	11
165.21.15.154	8
198.199.69.40	7
45.137.22.123	4

Figure 4.22 Malware Found by Network Protection

Further analysis of a discovered malware is shown in Figure 4.23 . The SIEM logs reveal attempts to launch variant attacks on addresses registered within the organization. Subsequent analysis of sample logs substantiated the occurrence of the attack. The attacker attempted to execute a Linux command on the shell running on the targeted machine. Following a comprehensive investigation, the presence of the attacker's shell was confirmed and remains active up to date.

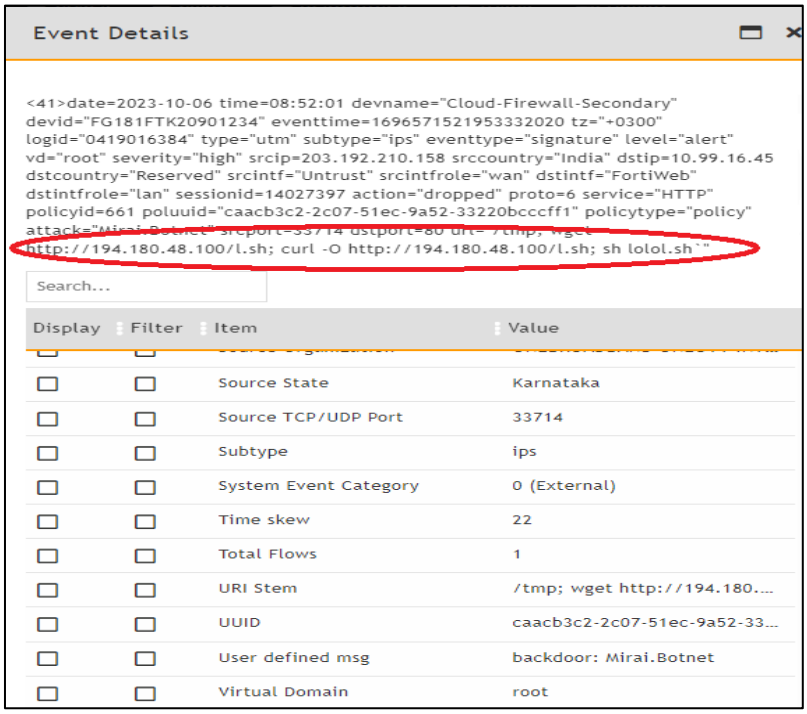


Figure 4.23 Command and Control on Malware Found

The provided event details in Figure 4.23 indicates an alert triggered by the eCloud-Firewall-Secondary device, specifically the FG181FTK. The event, categorized as a high-severity signature, is associated with an attempted intrusion from the source IP address 203.192.210.158 originating from India. The event pertains to a dropped packet related to the HTTP protocol and is associated with policy ID 661. The event is classified under the "IPS" subtype and is indicative of a backdoor attack attributed to the Mirai Botnet. The action taken in response to this event was to drop the packet. Furthermore, the event is associated with a session ID 14027397 and is linked to the "FortiWeb" destination interface. The payload of the event includes a URI stem and a user-defined message denoting the presence of the Mirai Botnet backdoor.

IP Protocol	COUNT(DISTINCT Source IP)	Destination TCP / UDP Port
6 (TCP)	483	23 (TELNET)
6 (TCP)	193	443 (HTTPS)
6 (TCP)	105	80 (HTTP)
17 (UDP)	82	547 (DHCPV6-SERVER)
6 (TCP)	51	22 (SSH)
6 (TCP)	49	445 (MICROSOFT-DS)
17 (UDP)	48	5353 (ZEROCONF)
6 (TCP)	44	8080 (HTTP-APP-SERVER-0)
6 (TCP)	39	8443 (HTTPS-ALT)
17 (UDP)	37	53 (DOMAIN)

Figure 4.24 Top Inbound Scanned Ports

Breaking down the data in the Figure 4.24 appears to offer a comprehensive summary of network traffic, featuring the count of unique source IP addresses COUNT (DISTINCT Source IP) and their respective destination TCP/UDP ports. Here's a detailed analysis and breakdown of the crucial elements:

Protocols Overview

Notable TCP ports such as HTTP (80), HTTPS (443), SSH (22), and others are highlighted, signifying the targeting or usage of various services. The figures represent the count of unique source IPs observed for each destination port, providing insights into the volume of distinct IP addresses attempting to communicate with specific ports. Here, breaking down by main Destination Ports:

- 443 HTTPS: 483 unique source Ips indicates substantial access or attack attempts on secure web traffic utilizing HTTPS services.
- 80 HTTP 105 unique source IP implies access or attack attempts on standard web traffic over HTTP.
- 23 TELNET 5 unique source IPs telnet traffic is noteworthy due to its susceptibility to exploits stemming from the lack of encryption

- 22 SSH 49 unique source IPs represents secure shell traffic, encompassing legitimate administration activities and potential brute-force attack attempts
- 445 MICROSOFT-DS 44 unique source IP frequently targeted for vulnerabilities related to Server Message Block (SMB).
- 8080 HTTP-APP-SERVER-0 39 unique source Ips commonly associated with proxy servers and web application servers.
- 8443 HTTPS-ALT 37 unique source IPs serve as an alternative port for HTTPS, reflecting similar usage patterns to port 443.
- High-Traffic UDP Ports 53 DNS 37 unique source IPs indicate domain name resolution traffic, crucial for network operations and susceptible to DNS amplification attacks.
- 5353 Multicast Domain Name System (mDNS), 48 unique source Ips utilized for multicast DNS, facilitating service discovery in local networks.

There are high volume ports like 443 (HTTPS) and 80 (HTTP) are prime targets, necessitating robust security measures such as up-to-date SSL/TLS configurations and web application firewalls.

Security Measures

- Critical ports (e.g., 22 for SSH, 445 for Microsoft-DS) should be secured with strong authentication mechanisms and monitored for irregular activity.

Telnet Considerations:

- If feasible, disabling Telnet (port 23) in favor of more secure protocols like SSH is advisable.

Open ports on the servers shown in Figure 4.25 are a gateway to communicating with preferable network devices. Therefore, network administrators should scan all network peripherals occasionally to ensure all ports are authorized and well-known. Therefore, there is communication between those opened ports on the devices and other internal or external peripherals.

Device IP	Device Name	App Port/Protocol
[redacted]0.13	ePayment switch ...	SNMP(UDP/161)
[redacted]0.14	ePayment switch ...	SNMP(UDP/161)
[redacted]0.51	CERT WEB SITE	DOMAIN(UDP/53)
[redacted]0.51	CERT WEB SITE	FTP(TCP/21)
[redacted]0.51	CERT WEB SITE	HTTP(TCP/80);HT...
[redacted]0.51	CERT WEB SITE	SNMP(UDP/161)
[redacted]0.51	CERT WEB SITE	SSH(TCP/22)
[redacted]159	MTIT Switch Fujit...	DHCPV6-CLIENT(U...
[redacted]159	MTIT Switch Fujit...	DHCPV6-SERVER(U...

Figure 4.25 Servers Open Ports

The proposed framework will eliminate these communications based on the action that will be triggered on all devices. In Figure 4.25 it's clearly observed that external source IPs are connecting through open ports.

Source IP	Source Host Name	SUM(Total Bytes64)	Destination TCP / UDP Port	Destination IP
[redacted].102	DCGCC.MTIT.DC	22.94 MB	53 (DOMAIN)	212.14.236.105
[redacted]5.11	EXT-DNS01	20.62 MB	53 (DOMAIN)	212.14.236.105
[redacted]5.11	EXT-DNS01	17.57 MB	53 (DOMAIN)	82.213.1.239
[redacted].11	HOST-10.10.0.11	31.01 MB	53 (DOMAIN)	8.8.4.4
[redacted]5.12	EXT-DNS02	14.65 MB	53 (DOMAIN)	82.213.1.239
[redacted]71.4	CERT WAF	8.57 MB	53 (DOMAIN)	8.8.8.8
[redacted]3.6	10.99.13.6	8.71 MB	53 (DOMAIN)	8.8.8.8
[redacted].102	DCGCC	22.06 MB	53 (DOMAIN)	212.14.236.105
192.168.1.100	HOST-192.168.1.100	239.47 MB	443 (HTTPS)	20.113.57.33
192.168.253.40	HOST-192.168.253.40	19.06 MB	53 (DOMAIN)	8.8.8.8

Figure 4.26 Top Conversation to Outside Country

Figure 4.26 shown above provides a comprehensive overview of various website categories and the corresponding counts of denied web traffic for each category. This data was carefully monitored and observed over one week within a specific organizational context. The extensive nature of the data is attributed to its temporal scope of one week and its focus on a single organization, offering valuable insights into the web traffic landscape within this defined framework.

Source IP	Web Site Category	COUNT(Matched Events)
10.99.55.11	Newly Observed Domain	19k
10.99.55.12	Newly Observed Domain	15k
10.255.3.86	Proxy Avoidance	7547
10.255.3.60	Games	6238
10.99.55.11	Unrated	5840
10.99.55.12	Unrated	5627
10.255.3.42	Advertising	4911
10.255.3.146	Games	3160
10.255.3.49	Advertising	2565
10.99.55.11	Dynamic DNS	2202

Figure 4.27 Blocked Outbound Web Traffic

Of note is the significant blocking activity reported as new domains which registered the blocking of nearly 20,000 websites in response to triggers from the WAF. Additionally, another reporter which the second DNS recorded blocked approximately 15,000 attempts for the "Newly Observed Domain" category, indicating instances where the WAF encountered previously unknown domains for the first time. This underscores the dynamic nature of web traffic and the proactive measures taken to manage potential threats.

Furthermore, the various categories outlined in the figure, including Game, Unrated, Advertising, and others, are defined by the internal policy of the organization. These categories play a pivotal role in shaping the organization's approach to managing and regulating web traffic, reflecting the strategic alignment with organizational policies and security protocols.

Figure 4.28 Entities by Risk

In the above Figure 4.28, showing the total number of incidents per category, 5007 devices, 37 users. 18 high risks, 593 medium risks, and 4433 low risks have affected both devices and users' nodes in the last 20 days. To provide a specific example, a random sample was taken from one of these devices to verify an incident, revealing a clear text transfer of a password as shown in Figure 4.29.

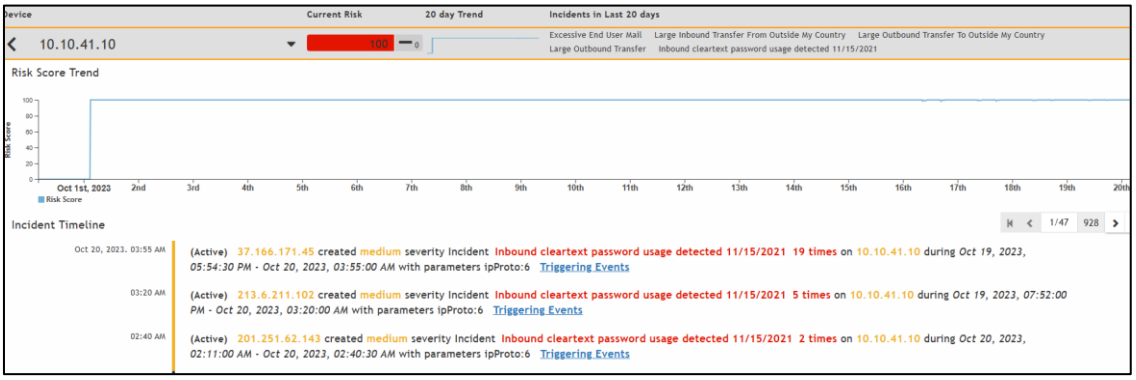


Figure 4.29 Device Incident Details

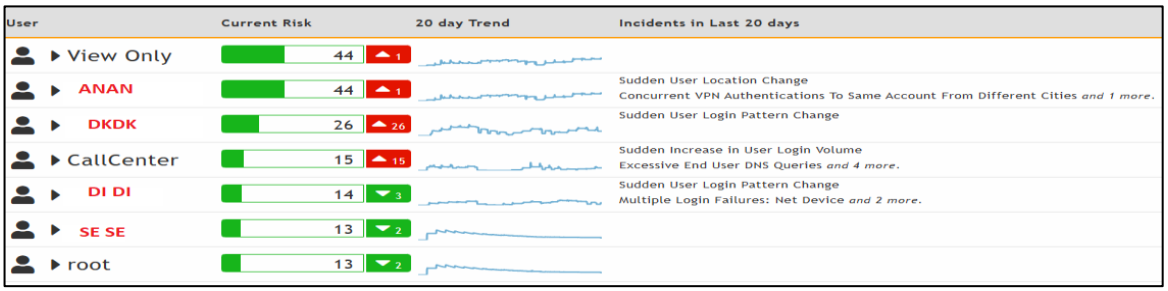


Figure 4.30 User Entity Risk

On the other hand, user risks are also crucial entities that warrant consideration in the analysis. For instance, let go and analyze the incident in Figure 4.30 , if a user (ANAN) experiences a sudden location change that is physically impossible, such as logging in from two different geographic locations simultaneously, or if the user (DIDI) demonstrates a pattern of frequent and abrupt changes in login behavior coupled with multiple failed attempts, these scenarios raise red flags. Additionally, an incident may arise when a user (call center) exhibits a sharp and unexpected surge in login volume, surpassing the expected amount and occurring simultaneously, indicating potential unauthorized access.

Conclusion

Explanations of visual data are critical processes that provide an overview of the data. This process simplifies and presents many incidents as a simple graph that can be easily understood by humans. This process not only simplifies the data but also directs attention to important information. SIEM visualization tools facilitate the analysis of logs containing incidents from multiple sources, combining them into a single stream. Analyzing the data logs is crucial for building a model that aligns with the objectives of the research problems.

The next chapter will introduce and discuss the proposed methodology and models in detail. This chapter will delve into the data logs processing life cycle, discussing each step corresponding to its function.

5. Chapter Five The Proposed Methodology and Models

Introduction

This chapter aims to propose the methodology employed in this study and introduces the proposed model for analyzing the fused traffic from multiple channels. The data flow from diverse sources goes in several stages as shown in Figure 5.1 which illustrate the model stages of the fused traffic of the experiment. Utilizing data fusion methodologies to analyze the behavior of attacks and anomalies within the data center environment involves aggregating and integrating data from various sources to anticipate anomalies. This complicated process encompasses refining the ML model and utilizing the threat database to facilitate well-informed and precise decisions.

The methodology used in this study involves traffic analysis using a Security Information and Event Management (SIEM) system, in addition, manual extraction is performed to validate the proposed model by aligning manual detection with automatic detection in SIEM. In both approaches, the established procedures for handling traffic are implemented as follows: The process begins with data collection, which requires collecting data from all available data log sources provided by applications, systems, and IDS/IPS devices in all their forms. Then, logs are transferred to the SIEM or Syslog server in the form in which the data was issued, knowing that this data constitutes secure incident logs based only on the initial classification set on the devices and applications that provide the logs. This is considered a one stage of data classification. After that, the cleaning stage enters; this is because the data often contains redundant data and impurities, and normally it is non-structured data. The logs undergo a cleaning process to ensure that they are consistent, sound, and in an organized structure. The third and pre-analysis stage is

normalization, which includes arranging, organizing, and formatting the data in a standard format. In the fourth stage, the data is subjected to normalization and correlation. After that, log analysis is performed, which includes multiple steps such as segmentation and learning. All of the above steps are visualized in Figure 5.1. Upon reception of security incident logs from the aforementioned feeders, the reported incidents will undergo processing in the ML mode, utilizing the predefined algorithms within the SIEM system, which will be elaborated upon in subsequent chapters. Subsequently, based on the decisions made, the most suitable action will be implemented across all other nodes concurrently, aiming to minimize the impact on network effectiveness while striving to enhance operational performance and efficiency.

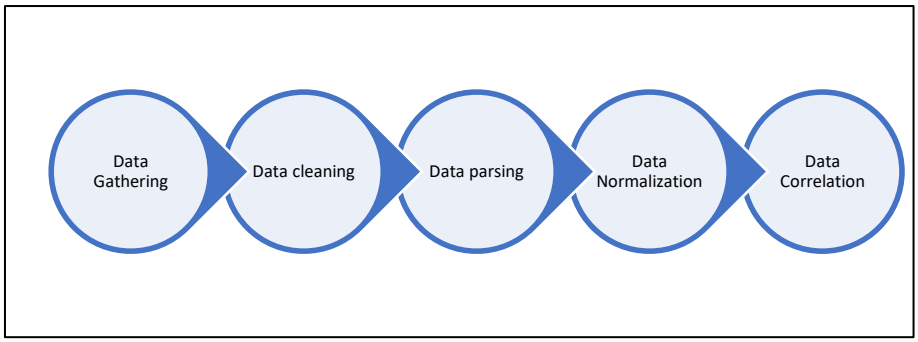


Figure 5.1 Data Logs Processing

Once again, the log data is collected from various sources, including routers, firewalls, switches, servers, and applications, using different protocols as outlined in Chapter 4. Most logs flow through protocols such as Syslog, SNMP, and agents, which send the logs to a central repository. The following section will describe the five stages of data log processing.

Data Logs Processing

This section will provide a detailed explanation of each stage of processing data logs flowing from log reporters in the form of information security incidents, with an explanation of the methodology in each stage as follows:

Data Gathering

The log collection process concerns acquiring raw data and pre-processing it before proceeding to the next stages. Specifically, this process involves gathering logs from sensors, routers, applications, and other sources. These logs contain vital information about system status, timestamps, network traffic, firewall activity, and behavioural data.

This process will utilize two methods. The first method focuses on collecting log data from a single type of transaction perspective and tracing its journey across all nodes, email transactions will be the case in this experiment. The second method involves obtaining logs from all sources within a defined time frame. It could be expressed for the data logs gathered into the repository as follows:

$$\text{Datalog} = \{\log_1, \log_2, \log_3, \ldots, \log_n\}$$

Where $\{\log_1\}$ refers to the source where the data comes from. The Figure 5.2 illustrates the data collection model from multiple sources in the data center that streams all the logs to the SIEM application that acts as a log repository for the purposes of this study. Data flows from various sensors in the data center and fused to form a flow of big data that contains important information that may help in detecting anomalies and taking appropriate preventive measures on all decision-making nodes of sensors which reduces the occurrence of breaches in the future.

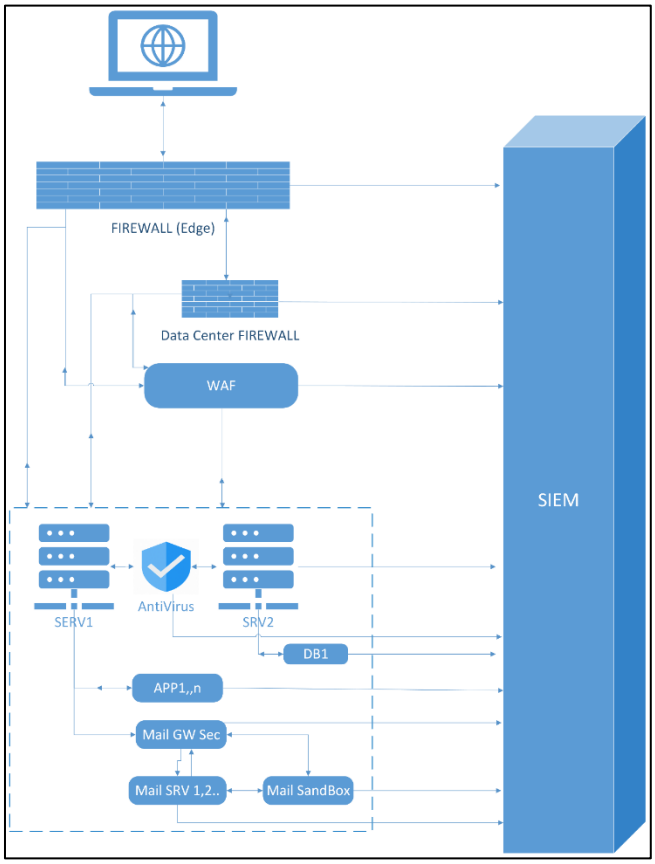


Figure 5.2 Data Logs Model

Before proceeding to the cleaning stage, the classification of data logs originating from multiple channels represents a crucial step in the data processing workflow. This complicated process entails organizing incoming data logs based on their importance, which portrays the nature of the transaction. Therefore, based on the experience of this research, the data will be classified according to its risk. The study is grounded in a theoretical and practical analytical exploration aimed at discovering anomalies in transactions within data centers by leveraging data that comes from data center sensors. In this context, the classification process involves the identification of logs to be analyzed from diverse channels, ensuring their appropriate segregation to facilitate the exclusion of unnecessary data with minimal priority in terms of information security. In this

experiment, the data will be classified to save time and increase the efficiency in real-time data processing.

This study was based on a rigorous methodology for data collection, where approximately five TB of information security incident logs were collected after conducting a comprehensive data classification. These logs were extracted from a raw dataset of up to 60 TB. To ensure the effectiveness of the experiments, the focus was on the diversity of the collected logs, including email logs and logs related to suspicious activities. These types of logs were considered essential for studying incidents, as they were analyzed according to two main models: acquisition within a specific time period and tracking the transaction journey through all stages of the incident. This study contributes to enhancing the understanding of security incident mechanisms and how to deal with them to enhance anomaly detection techniques.

The logs collected for security purposes are classified according to their types, the most important of which are logs containing suspicious information security events such as login failure attempts, direct access logs, Command and Control (CC) logs, and general events like system reboots, service failures, network logs, database logs, and device logs. Many logs are excluded because they are not necessary in the thesis study for security purposes, despite the importance of these logs in other domains.

In this study, a multi-stage classification approach was adopted to process the security event data. In the first stage, a source-based data filtering mechanism was applied, so that only relevant security events related to information security were retained. In the second stage, a precise classification of security events was developed based on their severity level. This classification included four levels: critical, medium, low severity, and “informational.” More details about this classification, and its impact on the results of the

study, will be provided in the following sections. After data collection and initial classification and analysis, the second stage will involve data cleaning to ensure data quality and accuracy.

Data Cleaning

The raw data of the logs must be cleaned before analysis to enhance its efficiency and accuracy during the analysis process. Basically, data cleaning involves removing unusable and irrelevant data from the log file, thus preparing it to be clean and ready for analysis [103]. Obtaining clean and reliable data is one of the most important challenges in data analysis. This ensures the accuracy of the analyses and the reliability of the resulting decisions [104]. The initial phase of the data cleaning process involved identifying and processing duplicate records with identical attributes. This phase was of utmost importance. Duplication was observed in both individual records file and aggregated records file. Eliminating them was of utmost importance to improve and facilitate subsequent processing and analysis.

Removing duplicate data from logs improves consistency and reduces the size of log files. Upon detailed examination of the logs, it was observed that the duplicate inclusion of FW device names and serial numbers significantly impacts processing time, especially when dealing with large volumes of logs. Therefore, this data, which is not useful in this study methodology, was removed. Upon closer inspection of the file, it was observed that additional features, such as IP geolocation, network subnet mask, and ISP-related data, are no longer imperative for each entry and can be omitted.

Data standardization: During this phase, the data was carefully checked and standardized. It was observed that all records containing IP addresses were captured in

IPv4 format. However, differences in time representation were observed across different files, indicating that the time was recorded according to the system in which it was created.

For instance, time was represented in various systems such as Unix Time (Epoch) and ISO 8601 format. To unify the representation, all time formats were standardized to ISO 8601 format, as illustrated in Figure 5.3.

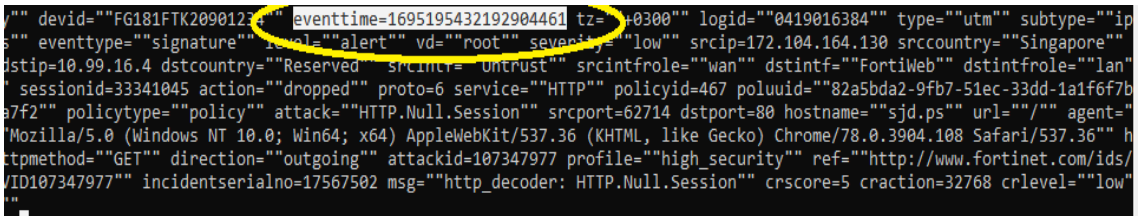


Figure 5.3 Epoch Time Format

After the data cleaning process is complete, the data logs are highly accurate and consistent, free of duplicate and illogical values, consistent units and formats are used for all values in each field, and uniform naming is used for all variables. Missing values are handled appropriately (such as deletion or replacement). The final data is complete, and all records contain all the essential information required for analysis and it is ready for the next phase to be parsed.

Data Parsing

Logs are commonly raw data that contain heterogeneous data from the system or devices that generate the logs; extracting anomalies from those logs must go into the analytics steps to verify whether this action is an anomaly or not. Data must be collected to achieve this stage so the analytical program or analyst can detect the anomaly. Depending on the different data log sources, there are also different log formats, for

example, plain text, comma-separated values (CSV), or a specialized format such as JSON or XML. Here the experiment required two types of parsing methods:

- Manual Parser: For small logs or very specific needs where manual analysis of specific and specific but large files is required, the Python platform was used here and was powerful enough to generate a very large amount of output, which took a long time but allowed us to freely make the required improvements.
- Automatic Parser: In this method, specialized SIEM software was used, which has built-in functions and an easy-to-use interface to simplify the analysis.

Furthermore, the extraction of structured data and meaningful information from unstructured raw data is also needed to allow the SIEM to easily read the file and quick analysis takes place that yields the appropriate and prompt action. Traditional and automated parsing is based on the technique of parsing logs, which can be used to analyze large or small data. Again, this study used two methods of log parsing, the automated one is used for SIEM purposes although it is a semantics parser that uses regular Expression (RegEx) or combined expression. And traditional parsing, custom code is developed, as mentioned in the case above. During the experiment, all log data files which were not encoded by the SIEM, a custom procedure was used to parse the raw data into a meaningful log from the aspect of SIEM.

In SIEM, most vendors have predefined parsers, such as Windows server logs, routers, firewalls, etc. Sometimes, specific events need to build custom parsing Extensible Markup Language (XML) to extract particular features, as in the Cisco parser in FortiSIEM, as shown in Figure 5.4.

```
<eventParser name="CiscoIOSParser">
   <deviceType>
      <Vendor>Cisco</Vendor>
      <Model>IOS</Model>
      <Version>ANY</Version>
   </deviceType>
</eventParser>
```

Figure 5.4 Cisco Custom Parser in FORTISIEM

Parsing logs in SIEM solutions such as Splunk, FortiSIEM, and LogRhythm involves the extraction and transformation of raw log data into a structured format for analysis and correlation. For example, Splunk utilizes its proprietary search processing language to parse and index log data for real-time monitoring and analysis, while FortiSIEM employs built-in parsers to normalize and enrich log data from diverse sources, enabling comprehensive threat detection and response. Similarly, LogRhythm's AI Engine uses advanced parsing and normalization techniques to convert raw logs into actionable intelligence for security monitoring and compliance [105].

The study employs the FortiSIEM system for automated parsing in addition to the manual parsing techniques. Figure 5.5 displays logs of an internally developed application for an alert system, capturing logs for every action during runtime. It's crucial for administrators to stay alert to instances where users are blocked due to their activities within the application. Therefore, a custom parser has been developed, as attached in the following figure.

```
<156>Jun 28 05:33:24 alertcertlocal Application-authintication <05:33:19>
date=2022-06-28 time=05:33:19 devname='alertswebsite' devid='000' type='e
vent' subtype='system' level='alert' vd='root' logdesc='Admin login disabled' e
venttype:'PH_AUDIT_ACCOUNT_LOCKED' srcip=10.102.68.170 dstip=10.102.7
0.100 action='login' status='failed' reason='exceed_limit' sign:'Application-au
thintication' reportingIP='10.102.70.100' msg='Login disabled from user ibra
him for 60 seconds because of 3 bad attempts'
```

Figure 5.5 Alert System Logs Data

A custom parser was constructed using XML tags and intricate terminology specific to the functions within the SIEM platform. This suggested parser serves the purpose of transforming unstructured and unreadable data into a structured and readable format regarding the SIEM function. The successful outcome of this code parser is illustrated in Figure 5.7 , while the visual representation of the utilized code is presented in Figure 5.7.

```
<patternDefinitions>
 <pattern name="patYear"><![CDATA[\d{4}]]></pattern>
 <!-- the strict version of host name pattern (RFC952) -->
 <pattern name="patHostName"><![CDATA[[A-Za-z](?:[0-9A-Za-z.-]*[0-9A-Za-z])?]]></pattern>
</patternDefinitions>
<eventFormatRecognizer><![CDATA[^\s*<:gPatSyslogPRI>?(?:<:gPatWeekday>\s+)?<:gPatMon>\s+
<:gPatDay>\s+(?:<:gPatTime>\s+alertcertlocal\s+<:patYear>|<:patYear>\s+<:gPatTime>|<:gPatTime>)\s+
alertcertlocal\s+(?:<:patHostName>\s+<:gPatIpAddr>|\[<:gPatIpAddr>\]|<:gPatIpAddr>|<:patHostName>)\s]]>
</eventFormatRecognizer>
<parsingInstructions>
<switch>
<case>
<collectFieldsByRegex src="$_rawmsg">
<regex><![CDATA[<:gPatMon>\s+<:gPatDay>\s+(?:<:gPatTime>\s
+<:patYear>|<:patYear>\s+<:gPatTime>|<:gPatTime>)\s+
(?:<reptDevName:patHostName>\s+
<_reptIp:gPatIpAddr>|\[<_reptIp:gPatIpAddr>\]|<_reptIp:gPatIpAddr>|<reptDevName:patHostName>)\b]]>
</regex>
</collectFieldsByRegex>
<when test="exist reptDevName">
<when test="matches($reptDevName, 'alertcertlocal')">
 <setEventAttribute attr="_body">replaceStringByRegex($_rawmsg, "'", "")</setEventAttribute>
<collectFieldsByKeyValuePair kvsep="=" sep=" " src="$_body">
 <attrKeyMap attr="subtype" key="subtype"/>
 <attrKeyMap attr="logLevel" key="level"/>
 <attrKeyMap attr="eventDesc" key="logdesc"/>
 <attrKeyMap attr="eventtype" key="eventtype"/>
 <attrKeyMap attr="srcIpAddr" key="srcip"/>
 <attrKeyMap attr="destIpAddr" key="dstip"/>
 <attrKeyMap attr="action" key="action"/>
 <attrKeyMap attr="newStatus" key="status"/>
 <attrKeyMap attr="reason" key="reason"/>
 <attrKeyMap attr="msg" key="msg"/>
</collectFieldsByKeyValuePair>
</when>
</when>
 <setEventAttribute attr="_syslogNGHeader">yes</setEventAttribute>
</case>
 <default/>
</switch>
<when test="exist _reptIp">
<when test="$_reptIp != '127.0.0.1'">
<when test="$_reptIp != '0.0.0.0'">
<when test="$_reptIp != '255.255.255.255'">
 <setEventAttribute attr="reptDevIpAddr">$_reptIp</setEventAttribute>
</when>
</when>
</when>
</when>
</parsingInstructions>
```

Figure 5.6 XML Parser

An inherent challenge in analyzing the aforementioned log lies in comprehending the log file context, which is tailored to the organization's unique requirements and the custom nature of the logs associated with this application. In essence, these logs deviate from conventional, widely recognized application logs. It is imperative for the SIEM team, in collaboration with whom the analytics solutions were developed, to gain a comprehensive understanding of this distinct data. The next step for data logs is normalization, the following subsection will describe the process of the normalizations.

Event Attribute	Value
Action	login
Collector ID	1
Count	1
Destination IP	10.102.70.100
Event Description	Admin login disabled eventtype:PH_AUDIT_ACCOUNT_LOCKED
Event ID	0
Event Severity	1
Event Severity Category	LOW
Log Level	alert
Message	Login disabled from user ibrahim for 60 seconds because of 3 bad attempts
New Status	Failed
Event Parser Name	alertcertlocalCustomParser
Organization ID	1
System Event Category	0
Event Receive Time	Jan 20, 2024, 01:30:47 AM
Reason	exceed_limit sign:Application-authintication
Relaying IP	127.0.0.1
Reporting IP	127.0.0.1
Reporting Device	alertcertlocal
Reporting Model	Linux
Reporting Vendor	Generic
Source IP	10.102.68.170
Subtype	system

Figure 5.7 Custom Parser Result

Log Normalization

Normalization is a pivotal process in the anomaly detection workflow, particularly when dealing with the fusion of streaming logs. In the specific context of this case study, the fusion of data center logs encompasses logs sourced from multiple channels. Normalization is obligatory as a preliminary measure to facilitate comprehensive log analysis and it ensures that the logs correspond to standardized attributes, including timestamp, source IP, destination IP, security level, and actions [106]. This uniformity in log structure is fundamental for establishing homogeneity and consistency in the log data laying the groundwork for effective and uniform analysis[107]. Furthermore, the process of data normalization proceeds to the subsequent stage only upon the fulfillment of specific

criteria. To enter the normalization process, the dataset must undergo comparison with a recognized standard dataset. Subsequently, the normalized logs are adapted to integrate with the SIEM system. In the experimental framework of this study, the normalization process preceded the integration with the SIEM. As a result, the logs that were normalized prior to integration exhibit a highly customized format, distinct from the widely recognized log format. In contrast, the logs that underwent normalization within the SIEM environment were sourced from well-established devices and applications, such as Cisco, Fortinet devices, Forti Agent, Juniper, and others as shown in Figure 5.8 which is refer to the SIEM normalized data log.

```
<185>date=2024-09-21 time=23:47:17 devname="FG-MNE-B" devid="FG6H1E5819902206"
eventtime=1705873637593647215 tz="+0200" logid="0419016384" type="utm" subtype="ips"
eventtype="signature" level="alert" vd="root" severity="critical" srcip=172.23.21.59
srccountry="Reserved" dstip=216.218.206.69 dstcountry="United States" srcintf="Flr-1"
srcintfrole="lan" dstintf="port2" dstintfrole="undefined" sessionid=10685874
action="dropped" proto=17 service="udp/6828" policyid=77 poluuid="59d5eef2-27cc-51ea-9505-fd6ca69f7951"
 policytype="policy" attack="Sality.Botnet" srcport=63794 dstport=6828 direction="incoming"
 attackid=36018 profile="default" ref="http://www.fortinet.com/ids/VID36018"
 incidentserialno=44004669 msg="backdoor: Sality.Botnet" crscore=50 craction=4096 crlevel="critical"
```

Figure 5.8 Data Normalization

The SIEM engine has the capability to do advanced normalization, which enables the administrators to create relationships between tables of incident logs from different devices. Normally, the SIEM dashboard does 1NF, and not so much 2NF, but administrators can do what they want. In this study, it was conducted advanced normalization on the diverse data present in the incident log. After that logs must go under correlation procedures as described later.

Logs Correlations

The correlation process applied to the logs originates from multiple sources, establishing connections between these sources and their respective parameters to enable

more accurate decisions and reduce reliance on a single reporter. This approach serves to reduce the likelihood of false positives [108]. Moreover, the correlation of events is imperative, particularly as events of the same traffic traverse diverse nodes. It is noteworthy that not all devices report identical events and actions. While certain nodes may exhibit a severity level of "information" others may block transactions due to policy violations, as classified in a severity level of "high" This process is recognized as Security Orchestration, Automation, and Response (SOAR). This study's contribution lies in identifying and refining the decision process within a chain to ensure the appropriate action is taken. The actions executed within the network perimeter, and security FWs are pivotal and warrant accurate consideration.

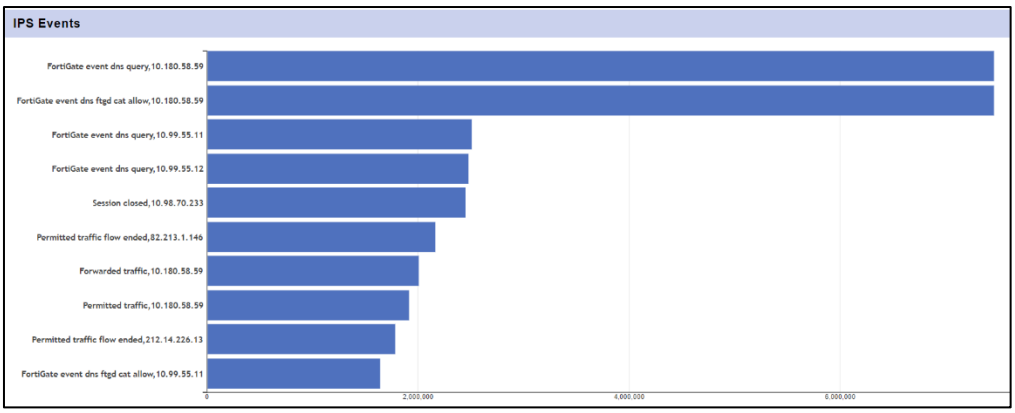


Figure 5.9 Traffic Flow Perimeter

All the sources shown in the above Figure 5.9 contribute to reporting events to the SIEM system. The traffic flow originates from the internet and traverses through the chain within the data center, for instance, email traffic passes through multi-level starting at the edge firewall, then through the UTM, Email Gateway security, and finally to the mailbox server, and if the email has an attachment will undergo to further security level which is

the sandbox. Each of these nodes reports events to the SIEM in accordance with their respective log settings.

As previously mentioned, only security incident events were reported while the Syslog protocol was enabled, while other devices utilize NetFlow to forward all traffic to the SIEM. Subsequently, the SIEM initiates anomaly detection alerts based on the events reported from each node. That’s why SIEM vendors don't like NetFlow steaming: Netflow exhausts SIEM resources, especially in processing and storage, and the same concern in manual loading. In this study, only security incident logs were used due to the lack of resources available on the personal computer used for the experiment. An additional example, some devices don’t report an incident for brute forcing attack, but others do. On the devices reporting brute forcing attempts, they will typically respond to account access actions such as blocking the user who attempts to get access or suspending his account for a while of time. However, when this type of traffic is cross-referenced with data from FW equipped with Intrusion Detection Systems (IDS), an alert may be triggered, such as "an intruder attempting to gain access to the target destination", In this scenario, the correlation of these two logs enables the identification of a brute-forcing attack. Consequently, the correlation process enhances the effectiveness of the IDS system across the entire environment and facilitates the analytics team in tracking attacks from source to destination.

The analysis of data logs is based on two primary factors:

1. Extracting logs related to a specific transaction from all devices through which this traffic passes in the data center, sometimes called transaction journey.

2. Another factor is extracting logs based on a specific time slot. This means extracting and analyzing all incoming and outgoing logs from all devices operating in the data center within a specific period.

Fusing data logs and implementing the Playbook procedure is a new approach proposed in this thesis, which aims to improve forecasting detections by collecting data from multi-channel sources instead of relying on one. The proposed model addresses the development of a platform for collecting logs from multiple sensors and applying the five procedures model that were adopted in the work methodology (collection, cleaning, parsing, normalization, and correlation) in order to obtain a fused and consolidated record that contains all the data required for the next stage, which is analyses of the logs. The experiment was designed and implemented based on simulating two scenarios, one involves studying a single transaction during its journey across all devices and comparing the records in each one. In contrast, the others consist in studying the entire file over a specific period.

Meanwhile, each scenario is analyzed separately, where the first scenario is dealt with by collecting all the logs from the devices that passed through the transaction. In the case of this study, the email transaction was taken, where all the logs related to this transaction were collected starting from Perimeter FW, then Datacenter FW, then WAF, then Email Gateway Security, then Sandbox, then the email server, and studying these logs at all stages to view the decisions of each device regarding this transaction journey and comparing those decisions with each other and entering the ML in the SIEM system in order to make the appropriate decision and reflect this decision and learn to the rest of the devices by applying the proposed model through the Application Programming Interface (API) to reflect the action taken on one node and apply it to all other nodes in a

synchronous and harmonious manner which is called later orchestration way, and enhancing the intrusion prevention system on all devices by updating their security profiles.

The second scenario is to view all logs regardless of their type in a specific period, and they can be investigated on one or two devices. Here we find a big challenge and difficulty in tracking those logs in this scenario if they are collected from more than two devices due to their large size and multiplicity. In the subject of this study, the experiment was prepared on the security perimeter devices, which are PA and DDOS, and the results of a DDOS-type hack were studied. The results of the experiments for both previous scenarios will be presented in the next chapter.

Conclusion

This chapter discusses the implementation of multiple models from various channels. While the result for a single log may be good, it is essential to study all the influences and factors that affect incident logs from a broader perspective. To achieve this goal, the study proposes data fusion techniques to analyze all factors that impact the security of services. The results have been optimized for all targeted log feeders, the best results have been achieved for SIEM and manual loading. The next chapter will discuss the results for each scenario taken as an experimental channel and compare the results of the fusion model with the legacy one.

6. Chapter Six Results and Discussion

Introduction

This chapter will thoroughly examine the empirical data obtained from the logs collected from node devices, including security devices and agents. The processing of incident logs has evolved through five distinct stages within the log incident process, as detailed in Chapter 5. The analyses have been conducted based on the trigger alerts of the incidents, which allowed each incident to be scrutinized individually and enhanced the model and machine learning capabilities within the SIEM system. Because some analyses are based on time series, this study focuses on analyzing the logs received from the fusion model of several and different sources and making the appropriate decision based on the decisions taken by the other devices. This reinforces the principle of working on the rules of the playbook concept through API push notifications or updating the control on remote devices.

Raw data is extracted from different sources based on a specific period allowing the organization to collect data from various sources and create a comprehensive data set for analysis and processing. By organizing the raw data according to particular time slots, it becomes easier to manage and can extract valuable information, contributing to more efficient and effective data analysis and decision-making processes.

Data parsing involves the process of analyzing and interpreting data in various formats, including data format, encoding, CSV format, and data arrangements. It encompasses the extraction and transformation of data into a structured format that can be easily manipulated and analyzed. So, understanding the data format and encoding is crucial

for accurately interpreting the information while organizing data into CSV format, allowing ease of use, exchange, and manipulation.

In this experiment, it was shown that there is a link between data arrangements and the process of organizing and structuring data systematically to improve its efficiency and ease of analysis and use this what the methodology covers it in the parsing section in the previous chapter. Proper organization of data is also a crucial factor in ensuring the accuracy and speed of analysis, which allows for extracting valuable insights and a deeper understanding of the available information. The success of analysis processes depends largely on the quality of the prior arrangement and coordination of data, as chaos or inappropriate organization can negatively affect the ability to adopt accurate action. Therefore, effective data analysis is the basis for making informed decisions based on a deep understanding of the available data, which contributes to improving performance and achieving the study goals.

While dealing with data processing additional challenges raised, one of which is how to fix encoding issues, such as the "utf-8" error which indicates invalid byte decoding. When handling encoding errors such as "*byte 0xd9 in position 7515: invalid continuation byte*". Resolving encoding problems enables the successful parsing and processing of logs and allows analysts to extract valuable information to detect potential anomalies or security incidents within the data logs. Addressing encoding issues could improve the reliability and accuracy of the log analysis, ultimately contributing to more effective decision-making and proactive security measures.

In addition, the size of the dumped data logs was very large, reaching 4 TB per month at the time of dumping. The reason for taking per month, not per day or week is that some devices rarely send logs while others do so frequently. It is found that the minimum

duration for keeping logs for analysis is at least one month to match the recent data with the nearest ones from other devices. Further obstacles were encountered while analyzing the data logs in this study, leading to challenges in dealing with retrieving and manipulating historical data. For example, some queries took 3 minutes to finish despite being executed on the SIEM, which has high-performance resources (128-core CPU, 256 GB RAM, and 20 TB SSD local storage), along with SAN storage of 500 TB.

The following section presents the results of the empirical study conducted to examine the research hypotheses, as explained in the previous chapters. An experiment and two different scenarios will be presented, with a detailed analysis of the results obtained for each scenario, and an explanation of their impact on the study as a whole. This analysis will discuss the extent to which the results are consistent with the thesis hypotheses and the significance of these results in the context of the research.

Scenario A: Analyzing a Transaction, Then Compare The Actions Throughout Its Journey

Attackers deceive victims with a good title, but the inside is malicious. Analyzing the complete action of the chain involving all devices comprehensively while a specific transaction passes through all the nodes and this will be the case in this scenario. A random sample of suspicious emails was selected from a pool of similar emails to investigate the case and run the experiment as presented in Figure 6.1 the raw data of suspicious emails.

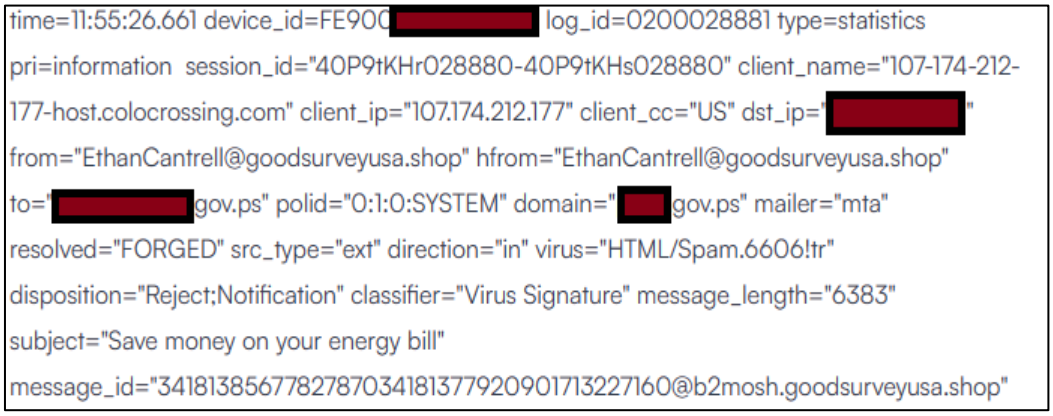


Figure 6.1 Email Raw Data Log

Following the parsing of the logs, as shown in Figure 6.1, the extracted data reveals that the sender possesses a "poor reputation" This determination is further corroborated by the results obtained from a free online tool, as illustrated in Figure 6.2. Subsequently, an additional test was conducted after the initial examination of the sender client IP (107.174.212.177). This involved retrieving a Mail Exchange (MX) record from the email domain server (goodsurveyusa.shop). Notably, the investigation revealed that the MX record shares the same IP address as the client, indicating that the suspicious emails originate from the same server.

Table 6.1 Email Extraction Data

Data	**Value**
Client Information	
Client Name	107-174-212-177-host.colocrossing.com
Client IP	107.174.212.177
Client Country Code	US with IP 107.174.212.177
Email Information	

From	EthanCantrell@goodsurveyusa.shop
To	***@***.gov.ps
Subject	Save money on your energy bill.
Email gateway security devices detect this email and give the following information.	
Resolved	FORGED
Virus	HTML/Spam.6606!tr
Disposition	Reject; Notification
Classifier	Virus Signature

Further validation of the sender's reputation was conducted through a blocklist check, revealing that the IP was blacklisted on numerous servers worldwide, as shown in Figure 6.3.

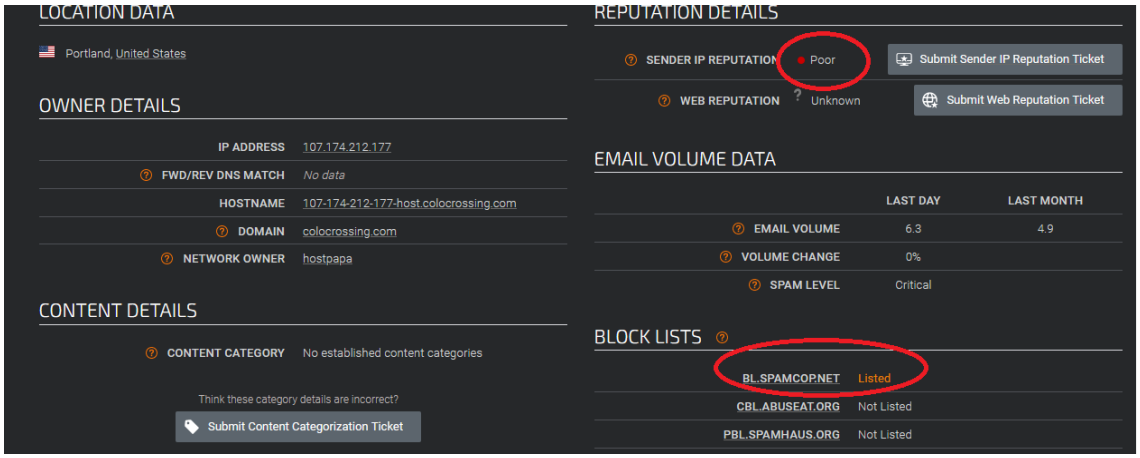


Figure 6.2 Email Address Reputation Check

The collective findings from these investigations unequivocally indicate the sender's suspicious activities, which are deemed highly irregular and not conducive to legitimate operations. It is noteworthy that the traffic traverses through the Edge FW before

progressing to the subsequent layer, the UTM FW, and ultimately reaching the email gateway security.

	Blacklist	Reason
LISTED	Abusix Mail Intelligence Blacklist	107.174.212.177 was listed Detail
LISTED	Hostkarma Black	107.174.212.177 was listed Detail
LISTED	ivmSIP	107.174.212.177 was listed Detail
LISTED	ivmSIP24	107.174.212.177 was listed Detail
LISTED	PSBL	107.174.212.177 was listed Detail
LISTED	RATS Spam	107.174.212.177 was listed Detail
LISTED	s5h.net	107.174.212.177 was listed Detail
LISTED	SORBS NEW	107.174.212.177 was listed Detail
LISTED	SORBS SPAM	107.174.212.177 was listed Detail
LISTED	SPAMCOP	107.174.212.177 was listed Detail
LISTED	Spamhaus ZEN	107.174.212.177 was listed Detail
LISTED	UCEPROTECTL1	107.174.212.177 was listed Detail
LISTED	UCEPROTECTL2	107.174.212.177 was listed Detail
LISTED	UCEPROTECTL3	107.174.212.177 was listed Detail
OK	0SPAM	
OK	0SPAM RBL	
OK	Abuse.ro	
OK	Abusix Mail Intelligence Domain Blacklist	
OK	Abusix Mail Intelligence Exploit list	
OK	Anonmails DNSBL	
OK	BACKSCATTERER	
OK	BARRACUDA	

Figure 6.3 Blacklist Check

Additionally, other devices provided information regarding the same email transaction as shown in Figure 6.3 which illustrates the flow of the email transaction, wherein the sender, located in Texas Dallas, USA, sends an email with an attachment over the Internet to a specific email address within the government organization. Upon reaching the first node within the data center network, namely the edge FW (x.x.x.221) the email proceeded to the subsequent security level without any block monitor actions. The next security level was managed by the UTM FW (x.x.x.20), which first checked the IPS database signature. Finding it clear, the firewall permitted the email and then proceeded to conduct a second check using the File Antivirus.

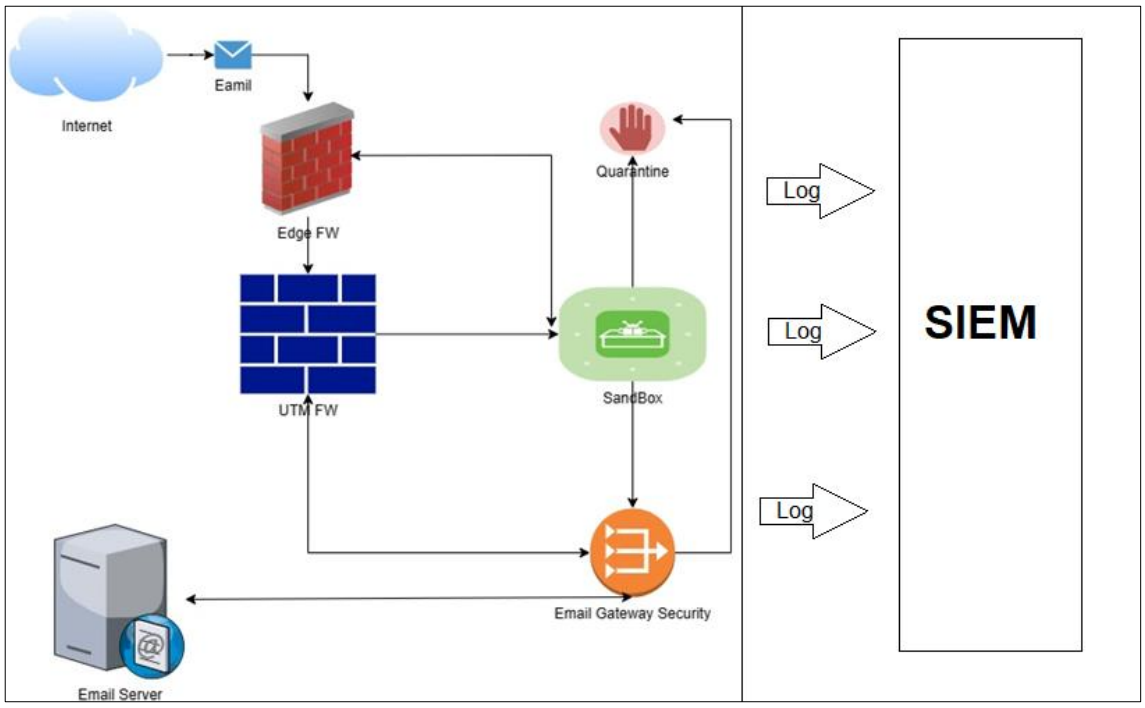


Figure 6.4 Email transaction flow

The UTM FW cannot detect whether the email body and its content are safe or dangerous because it has an attachment, the built-in role of the UTM FW is to send it to the Sandbox (x.x.x.10) running outside the environment. The Sandbox reported that there was a virus inside the file (HTML/Spam.6606!tr), and the action taken was to reject the email and quarantine it. Figure 6.5 illustrates the correlations of logs related to the email case.

Time	Reporting IP	Event Type	Event Name
11:56:05 AM	[redacted]221	PAN-OS-TRAFFIC-end-allow	Permitted traffic flow ended
11:55:56 AM	[redacted]20	FortiGate-traffic-allowed	Permitted traffic
11:55:48 AM	[redacted]10	FortiMail-Antivirus-virus-infected	FortiMail detected virus
11:55:48 AM	[redacted]10	FortiMail-Antispam-spam-found	FortiMail Antispam spam found
11:55:48 AM	[redacted]10	FortiMail-Antispam-spam-found	FortiMail Antispam spam found
11:55:48 AM	[redacted]10	FortiMail-History-Reject	Mail Rejected
11:55:48 AM	[redacted]10	FortiMail-Antispam-spam-found	FortiMail Antispam spam found
11:55:48 AM	[redacted]20	FortiGate-antivirus-file-submitted	File submitted to Sandbox
11:55:44 AM	[redacted]20	FortiGate-appctrl-ips-pass	FortiGate appctrl ips pass

Figure 6.5 Email Log's Correlations

Supervised Machine Learning algorithms supported by SIEM used in this experiment such as Random Forest and SVM classify data based on learned patterns that are predefined in the SIEM. In the context of email anomaly detection, this means training these algorithms on a large dataset of emails that are classified as "normal" or “anomalous.” such as emails that a user receives, newsletters, or marketing emails, which are classified as "normal" and examples of phishing attempts such as spam, or emails containing malware, which are classified as "anomalous" During training, the algorithms identify key features and patterns that distinguish between the two classes. Random Forest achieves this by creating multiple decision trees and merging their outputs, while SVM creates a sublevel that perfectly separates the classes. Once trained, these models can analyze incoming metadata of emails, and extract relevant attributes such as sender, receiver subject, and attachments. Then classify them based on learned patterns, identifying potentially malicious emails that deviate from the established "normal" profile.

This will increase the efficiency of the email anomaly detection mechanism by providing a more accurate and adaptive approach than traditional rule-based systems. The

results of the learned model will be used as input to the proposed model developed to detect the anomaly and apply the concept of orchestrations model.

Note that this experiment relied on Machine Learning used in the SIEM system with determinations of the parameters widely available in the system itself, and reliance was placed on what is available to save effort and time in developing learning algorithms that are supposed to give the same results, so the researcher preferred to jump for this step, the aim of which is to benefit from the functions of the algorithm instead of writing a source code for its operation.

Results

The email came from the USA to Palestine and was controlled by government personnel. It passed through three security devices, but only one device detected that it was a fishing email, and the two devices passed it without any action. The transaction will pass next time also if there are no interventions on the transparent devices to update their security profile or control.

Applying the proposed model to the case above by making the system adopt the decision of the sandbox which defines the file as a virus, the model pushes its notifications to the PaloAlto and Fortinet UTM to create or update existing policy on PaloAlto to block public IP where the email comes from, and to Fortinet UTM to update its security profile of to monitor state to learn the machine about these IPs range. Here will see how our model affects the mail server gateway.

By applying the proposed model to the aforementioned case, the model will adopt the decision of the sandbox, which classifies the file as a virus. Subsequently, the model initiates the push notifications to PaloAlto FW and Fortinet UTM to create a policy on

PaloAlto to block the public IP from which the email originates and updates the security profile on Fortinet UTM to set the IPs range under monitor state and enable ML mode.

This study aims to assess the impact of the proposed model on the mail server gateway by comparing the performance of the email server before and after the application of the proposed model. The subsequent figures clearly illustrate the disparities between the two cases. The data was collected on the first working day (Sunday). illustration on the left in Figure 6.6 shows the volume of emails over 24 hours, revealing a high volume during peak hours prior to the implementation of the proposed model. In contrast, the right illustration in the same figure demonstrates a notable decrease in the volume of emails under similar conditions following the implementation of the model.

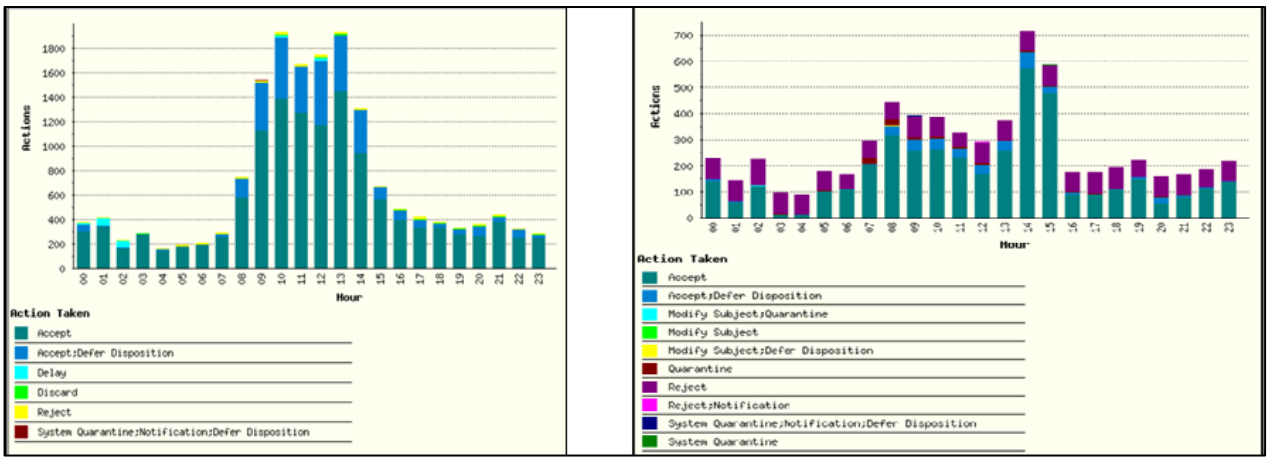


Figure 6.6 Email Performance Comparison: Pre- vs. Post-Model

The comparison of the results between the two cases, focusing on the common feature of (Accept) email only, reveals a notable 32% improvement in efficiency following the implementation of the experiment, as demonstrated in Table 6.2. This confirms that the proposed model has successfully achieved the objectives outlined in the mission of this study.

Table 6.2 Hourly email stats

Time	# of Accepted email (Before)	# of Accepted email (After)	Difference
0	295	139	0.47
1	337	55	0.16
2	161	117	0.73
3	272	10	0.04
4	138	9	0.07
5	173	99	0.57
6	187	108	0.58
7	253	201	0.79
8	571	312	0.55
9	1121	256	0.23
10	1378	258	0.19
11	1265	227	0.18
12	1164	166	0.14
13	1443	256	0.18
14	932	571	0.61
15	560	476	0.85
16	386	86	0.22
17	326	86	0.26

18	325	105	0.32
19	275	145	0.53
20	259	52	0.20
21	372	74	0.20
22	245	112	0.46
23	241	136	0.56
AVG	12679	4056	0.32

Challenges

Our main goal is to show that log correlation can improve intrusion detection systems; the concern is how different attacks are recorded in different logs and how correlating these logs can help identify them.

Our main goal is to show that correlating logs to each other will improve IDS. The concern lies in how to combine the logs of these attacks stored in different places and how to link them in order to identify anomalies.

In addition, the results showed that the overall impact on network efficiency was not negatively affected, but rather witnessed a significant improvement compared to the performance before applying the proposed model. This improvement is attributed to several factors. First, the model relies on an external server that communicates with the other devices via an API using the HTTP protocol. This mechanism is characterized by its one-way communication and is small in size, which reduces the load on the network. Second, the model's ability to block suspicious transactions from the first end device (perimeters FW) in the organization, prevents distribution within the internal network. This

contributes to reducing the consumption of resources at the hardware level (such as memory and processor) and improves the overall performance and efficiency of the network. These results indicate that the proposed model is capable of enhancing network security without negatively affecting its performance and may even contribute to improving it. This enabled the researcher to apply the proposed model in real-time with a time delay of up to milliseconds without any impact on the network or application performance.

Our primary objective is to demonstrate how fused log correlation can enhance intrusion detection systems. And how the correlation of these logs could help identify anomalies. Due to the large amount of data and the challenges associated with retrieving logs from previous archives, as mentioned in the previous discussion, certain queries occasionally took more than 3 minutes to finish their job. In addition, some applications also took time to deliver their logs. This factor significantly impacts the model's performance, potentially leading to delays but this delay does not affect the network performance but affects the delivery of the trigger actions. The next section will show the second case of log analyses which has a different scenario.

Scenario B: Study The Whole Log File in A Specific Period

In this specific investigation, the primary focus is directed toward the analysis of WAF logs to identify any potential indicators of malicious activity within the file. The file (WAFMTIT.CSV) was opened and loaded into the proposed model database using Python code on the PyCharm platform. As explained in Chapter 3, the file must undergo 5 stages of the data log lifecycle. Subsequently, the data from the WAFMTIT file was crossed match with a second file (DDOSMTIT.CSV) dumped from the DDOS appliance. The transaction

dialogue, described in Figure 6.7, involves various stages through which the transaction progresses and can follow specific paths. The path that a transaction takes depends on the query type. For example, some queries travel from the Internet to WAF, then UTM, while others take a different path, bypassing WAF and going through DDOS, Palo Alto, UTM, and then to servers, and so on. The behavior at each possible query path will impact the other security devices, aligning with the proposed model's concept.

In this scenario experiment, the first device that receives the transactions from the internet is a DDOS appliance, which operates as a volumetric measure, determining whether incoming traffic establishes an attack based on predefined thresholds related to internet downlinks or uplinks. Legitimate traffic volume is permitted based on the appliance's real-time database entries, while suspicious activity is flagged for further audit.

The subsequent stage involves perimeter FW, often called the Edge FW, which predominantly operates as a proxy. Some iterations of this device also integrate threat management capabilities, primarily relying on database signature analysis to ascertain whether an incoming query aligns with known signatures. Queries that match existing signatures are promptly blocked. Importantly, incident logs are forwarded to the SIEM for comprehensive tracking. The proposed model ensures that such events are broadcast to all subsequent devices for database updates, including the preceding device.

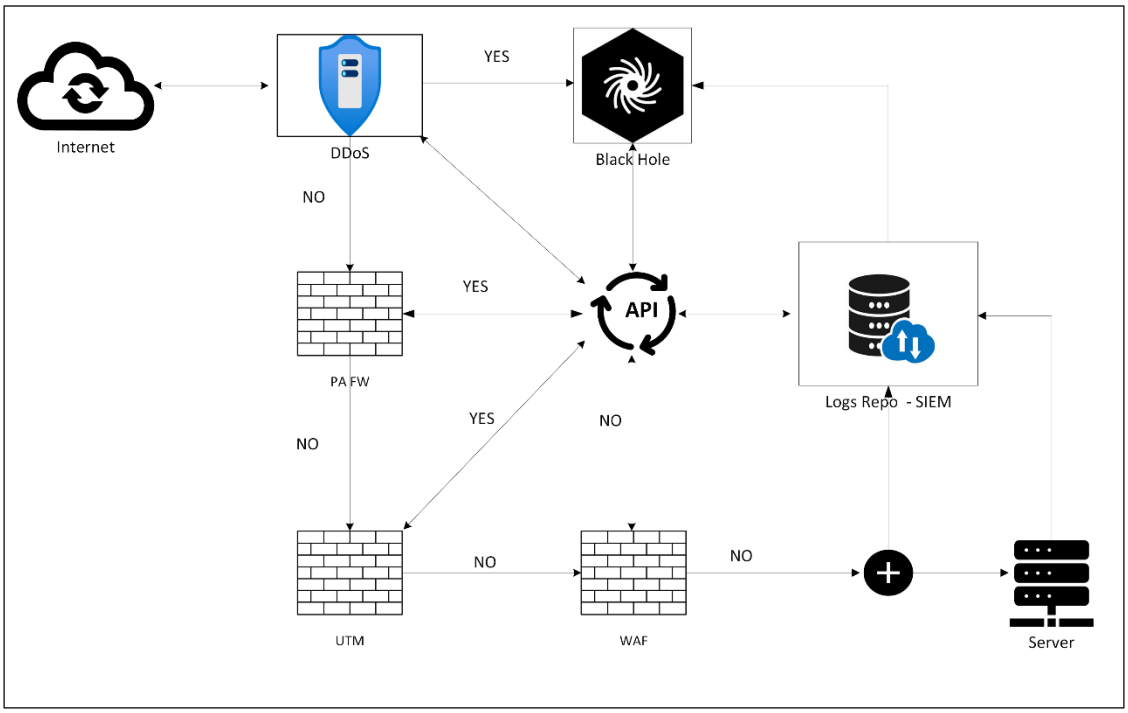


Figure 6.7 Transaction Dialogue

This comprehensive approach minimizes the processing overhead and computational power consumption on downstream devices, such as WAF, by streamlining the transaction flow directly from DDOS to WAF before subsequently routing to the Edge FW and beyond. Furthermore, the DDOS appliance contributes to this process by enhancing its functionality to identify malicious activity originating from specific IP addresses proactively. Subsequent stages involving UTM and antivirus solutions, notably CrowdStrike, followed a similar procedure, thereby establishing a cohesive and systematic approach to transaction analysis and security enforcement.

In this context, there is prior work done on the files undergoing the experiment applying the methodology adopted in this experiment; first, the data had been dumped from two devices, WAF, and DDOS machines, respectively, and those two devices as an option of many compound scenarios exist. The number of possible transactions could be calculated based on below Equation. For example, if there are 2 devices in the chain to

reach the application so the number of logs for this transition is (2+1) the following formula can express the number of transactions possibility in the chain.

$$T(n)=\frac{n(n+1)}{2}-1 \qquad \text{……Equation 1 transactions probability}$$

Due to the varying number of transaction possibilities based on the concept mentioned above, one of them was taken to address this problem.

The reason for choosing the above two devices as the experiment scenario is the difficulty of cleaning data and parsing the logs. The MTIT WAF logs are data that requires significant effort to clean, and it has redundant data, not useful patterns, and overall unstructured data. This operation required reading and investigating the file carefully, then building scripts to do this job, and finally, it has been shown that not all applicable filters can be covered. Furthermore, some minor challenges were encountered and required more effort than initially expected. One of these challenges involved dealing with time and date in various formats, such as system epoch time, system time, and less commonly known formats.

Custom parsing was written to parse the file and enable the data to be read after cleaning. Appendix 1 shows the parser built to make the log file readable. The implementation originally used Python code to read, search patterns, draw charts, and build the API advising other neighbor devices. Appendix 2 and Appendix 3 show the Python code for pattern search and API, respectively, while the data cleaning uses Excel functions. The appendices provide only the main function.

Figure 6.8 Shows the sample of one of the millions of records in the WAF log. This log corresponds to the DOS log in the DDOS appliance. After investigation, it was found that some DDOS attacks are prohibited due to their geo-location, as shown in the log WAF

but the queries are still received into the entire network even though WAF has a predefined policy to block these location IPs, this will lead to full exploitation on downlinks and uplinks as the DDOS attack uses the TCP protocol, which is known to open three-way handshaking sessions and keep those sessions opened until a response message is received this makes the line full of frames and then the network falls.

This study addresses the challenge faced by inexperienced administrators in effectively configuring data center devices to prevent DDoS attacks. The proposed solution involves proactively pushing advisories to DDoS appliances, enabling them to block malicious traffic and redirect it to black holes before an attack can fully materialize.

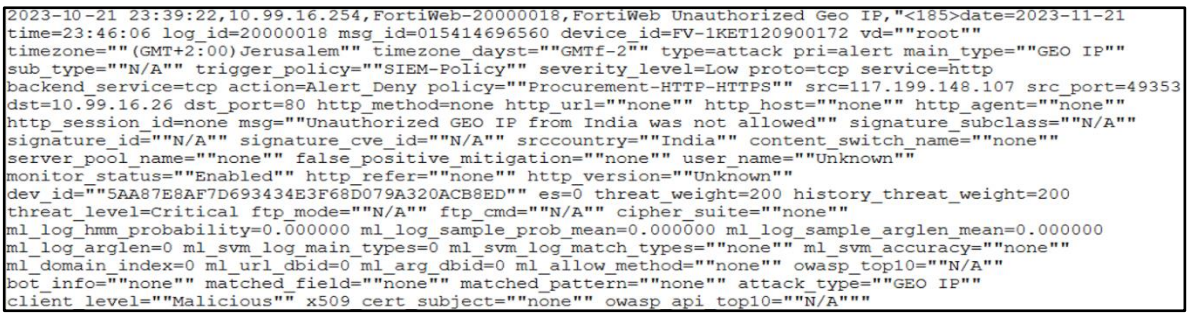

2023-10-21 23:39:22,10.99.16.254,FortiWeb-20000018,FortiWeb Unauthorized Geo IP,"<185>date=2023-11-21
time=23:46:06 log_id=20000018 msg_id=015414696560 device_id=FV-1KET120900172 vd=""root""
timezone=""(GMT+2:00)Jerusalem"" timezone_dayst=""GMTf-2"" type=attack pri=alert main_type=""GEO IP""
sub_type=""N/A"" trigger_policy=""SIEM-Policy"" severity_level=Low proto=tcp service=http
backend_service=tcp action=Alert_Deny policy=""Procurement-HTTP-HTTPS"" src=117.199.148.107 src_port=49353
dst=10.99.16.26 dst_port=80 http_method=none http_url=""none"" http_host=""none"" http_agent=""none""
http_session_id=none msg=""Unauthorized GEO IP from India was not allowed"" signature_subclass=""N/A""
signature_id=""N/A"" signature_cve_id=""N/A"" srccountry=""India"" content_switch_name=""none""
server_pool_name=""none"" false_positive_mitigation=""none"" user_name=""Unknown""
monitor_status=""Enabled"" http_refer=""none"" http_version=""Unknown""
dev_id=""5AA87E8AF7D693434E3F68D079A320ACB8ED"" es=0 threat_weight=200 history_threat_weight=200
threat_level=Critical ftp_mode=""N/A"" ftp_cmd=""N/A"" cipher_suite=""none""
ml_log_hmm_probability=0.000000 ml_log_sample_prob_mean=0.000000 ml_log_sample_arglen_mean=0.000000
ml_log_arglen=0 ml_svm_log_main_types=0 ml_svm_log_match_types=""none"" ml_svm_accuracy=""none""
ml_domain_index=0 ml_url_dbid=0 ml_arg_dbid=0 ml_allow_method=""none"" owasp_top10=""N/A""
bot_info=""none"" matched_field=""none"" matched_pattern=""none"" attack_type=""GEO IP""
client_level=""Malicious"" x509_cert_subject=""none"" owasp_api_top10=""N/A""""

Figure 6.8 WAF Log

The action of our model successfully takes place in DDOS. The accurate result will be visible when flooding occurs, even though we notice very few transactions are dropped at DDOS under the policy created by the study model.

Conclusion

Data log fusion analyses for anomaly detection is a novel approach proposed in this thesis, aiming to optimize prediction results by collecting data from multiple sources rather than relying on a single source. This approach has been applied using two types of logs and scenarios. One of them involves studying a single transaction and comparing the

actions throughout its journey, while the other involves studying the entire file within a specific period. In both scenarios, the anomaly detection results indicate actions related to security behaviors. Currently, each device takes responsibility individually without providing advice to others. The proposed model implements an API to consider the action taken on one node and apply it to all other nodes that act as orchestrators, which enhances prevention on all devices by updating their security profiles.

The next chapter will address the conclusion of the thesis, where it summarizes the results of the study and what improvements there are in future work.

7. Chapter Seven Conclusion and Future Work

In conclusion, the research introduced a novel multi-channel log data fusion model for data log analyses and anomaly detection using the one fusion database of security events. It comprehensively addressed the complexities of multi-channel log analytics and proposed a robust methodology to enhance anomaly detection mechanisms. The experiment involved the meticulous analysis of a substantial 4 TB sample of logs over 30 days, which underwent rigorous processing stages, including data cleaning, semantic extraction and parsing, normalization, correlation, and event analysis. The results revealed a significant threefold increase in the performance of IPS following the application and implementation of the proposed model, which relies on the concept of playbook rules across all devices and software. This illustrates the significant impact of decisions made by one and advising the others.

The experiment significantly improved the efficiency of the IPS. In contrast to previous findings, it discovered that the email sandbox detections besides the notifications of Edge FW (Palo Alto) are directly proportional to the study model setup. Additionally, at the same time as WAF detections, the action applies to the DDOS machine policy, aligning with the thesis goals of establishing a robust security ecosystem in the data center. This has resulted in a reduction of false positives and an increase in IPS efficiency based on the IDS function.

During the experiment, a significant reduction of forged emails on the mail server decreased by approximately 32% of the previous volume. In addition, the effect of the proposed model in the second scenario is observed also. Previously, SIEM received 130 GB of logs per day, of which WAF reported 8 GB per day and DDOS reported 14 GB. After implementing the new model, DDOS still sends 14 GB, but WAF now reports 5.44

GB. Notably, the volume in the DDOS logs remained unaffected due to its role as the primary node for transactions from the internet. In addition, the results of the proposed model do not appear on the DDOS device because the model does not face a real simulation, and if intends to do a fake simulation to prove the concept of the model it will be costly. It remains to prove the results of the model when a DDOS attack occurs, noting that only there is a blocking of some traffic through the policy that was implemented by the model. Also, another indicator was observed after the proposal was implemented, analysts made significantly fewer notifications, decreasing by over 37%.

Our future work will focus on improving the performance of the running device and the fusion database, reducing the CTC of the advanced model, and improving the use of ML in both models instead of relying solely on the algorithms supported by SIEM, which enables the researcher to use them freely and without restriction, as well as extending the system with additional intruder detection systems to improve the detection accuracy.

References

A.A. Salih and M. B. Abdulrazzaq, ‘Cyber security: performance analysis and challenges for cyber attacks detection’, *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 31, no. 3, pp. 1763–1775, 2023.

D. L. Hall and J. Llinas, ‘An introduction to multisensor data fusion’, *Proceedings of the IEEE*, vol. 85, no. 1, pp. 6–23, 1997.

P. Ristoski and H. Paulheim, ‘Semantic Web in data mining and knowledge discovery: A comprehensive survey’, *Journal of Web Semantics*, vol. 36, pp. 1–22, 2016.

L. Evans, M. Owda, K. Crockett, and A. F. Vilas, ‘Big data fusion model for heterogeneous financial market data (findf), in *Intelligent Systems and Applications: Proceedings of the 2018 Intelligent Systems Conference (IntelliSys) Volume 1*, Springer, 2019, pp. 1085–1101.

M. Cinque, D. Cotroneo, and A. Pecchia, ‘Challenges and directions in security information and event management (SIEM)’, in *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, IEEE, 2018, pp. 95–99.

N. Hubballi and V. Suryanarayanan, ‘False alarm minimization techniques in signature-based intrusion detection systems: A survey’, *Comput Commun*, vol. 49, pp. 1–17, 2014.

H. S. Milan and K. Singh, ‘Reducing false alarms in intrusion detection systems–a survey’, *International Research Journal of Engineering and Technology (IRJET) e-ISSN*, vol. 2395, p. 0056, 2018.

S. Aljawarneh, M. Aldwairi, and M. B. Yassein, ‘Anomaly-based intrusion detection system through feature selection analysis and building hybrid efficient model’, *J Comput Sci*, vol. 25, pp. 152–160, 2018.

A.Pecchia, M. Cinque, G. Carrozza, and D. Cotroneo, ‘Industry practices and event logging: Assessment of a critical software development process’, in 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, IEEE, 2015, pp. 169–178.

L. Viola, E. Ronchieri, and C. Cavallaro, ‘Combining Log Files and Monitoring Data to Detect Anomaly Patterns in a Data Center’, *Computers*, vol. 11, no. 8, p. 117, 2022.

N. A. Panayiotou and V. P. Stavrou, ‘Government to business e-services–A systematic literature review’, *Gov Inf Q*, vol. 38, no. 2, p. 101576, 2021.

S. He, P. He, Z. Chen, T. Yang, Y. Su, and M. R. Lyu, ‘A Survey on Automated Log Analysis for Reliability Engineering’, Jul. 01, 2021, *Association for Computing Machinery*. doi: 10.1145/3460345.

L. Van Hoye, P.-J. Maenhaut, T. Wauters, B. Volckaert, and F. De Turck, ‘Logging mechanism for cross-organizational collaborations using

Hyperledger Fabric', in *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, IEEE, 2019, pp. 352–359.

'Sun Fire TM V490 Server Administration Guide Please Recycle', 2005. [Online]. Available: www.sun.comhttp://www.sun.com/hwdocs/feedback

R. Ávila, R. Khoury, R. Khoury, and F. Petrillo, 'Use of security logs for data leak detection: a systematic literature review', *Security and communication networks*, vol. 2021, pp. 1–29, 2021.

R. Copstein, E. Karlsen, J. Schwartzentruber, N. Zincir-Heywood, and M. Heywood, 'Exploring syntactical features for anomaly detection in application logs', *it-Information Technology*, vol. 64, no. 1–2, pp. 15–27, 2022.

D. E. Kurniawan, M. Iqbal, J. Friadi, R. I. Borman, and R. Rinaldi, 'Smart Monitoring Temperature and Humidity of the Room Server Using

Raspberry Pi and Whatsapp Notifications', *J Phys Conf Ser*, vol. 1351, no. 1, p. 012006, 2019, doi: 10.1088/1742-6596/1351/1/012006.

B. Chen, J. Song, P. Xu, X. Hu, and Z. M. Jiang, 'An automated approach to estimating code coverage measures via execution logs', in *Proceedings of the 33rd ACM/IEEE International Conference on Automated Software Engineering*, 2018, pp. 305–316.

Y. Zeng, J. Chen, W. Shang, and T.-H. Chen, 'Studying the characteristics of logging practices in mobile apps: a case study on f-droid', *Empir Softw Eng*, vol. 24, pp. 3394–3434, 2019.

S. Achar, 'Cloud Computing Forensics', *International Journal of Computer Engineering and Technology*, vol. 13, no. 3, 2022.

S. Allam, 'An Exploratory Survey of Hadoop Log Analysis Tools', *Sudhir Allam," AN EXPLORATORY SURVEY OF HADOOP LOG ANALYSIS*

TOOLS", International Journal of Creative Research Thoughts (IJCRT), ISSN, pp. 2320–2882, 2018.

S. Liu, X. Wang, Y. Xiang, H. Xu, H. Wang, and B. Tang, 'Multi-channel fusion LSTM for medical event prediction using EHRs', *J Biomed Inform*, vol. 127, p. 104011, 2022.

T. R. Rao, P. Mitra, R. Bhatt, and A. Goswami, 'The big data system, components, tools, and technologies: a survey', *Knowl Inf Syst*, vol. 60, pp. 1165–1245, 2019.

R. Pandey, A. Singh, A. Kashyap, and A. Anand, 'Comparative Study on Realtime Data Processing System', in *2019 4th International Conference on Internet of Things: Smart Innovation and Usages (IoT-SIU)*, IEEE, 2019, pp. 1–7.

D. W. M. S. Manolis Karpathiotakis, ‘Scribe: Transporting petabytes per hour via a distributed, buffered queueing system’, https://engineering.fb.com/2019/10/07/data-infrastructure/scribe/.

G. Wang *et al.*, ‘Consistency and completeness: Rethinking distributed stream processing in apache kafka’, in *Proceedings of the 2021 international conference on management of data*, 2021, pp. 2602–2613.

V. C. Puttini and J. Schimiguel, *Cloudera: uma abordagem para análise de logs do proxy Squid-Cache usando Hadoop, Flume, MapReduce e Impala.* Revista de Ubiquidade, vol. 2, no. 1, pp. 45–62, 2019.

J. Gao, H. Wang, and H. Shen, ‘Smartly handling renewable energy instability in supporting a cloud datacenter’, in *2020 IEEE international parallel and distributed processing symposium (IPDPS)*, IEEE, 2020, pp. 769–778.

M. Saraswat and R. C. Tripathi, ‘Cloud computing: Analysis of top 5 CSPs in SaaS, PaaS and IaaS platforms’, in *2020 9th International Conference System Modeling and Advancement in Research Trends (SMART)*, IEEE, 2020, pp. 300–305.

A. Anilkumar, A. Shibu, M. A. Varghese, P. P. Sajan, and A. L. Sreedeep, ‘Detecting and Analysing Network Logs Using Machine Learning Techniques’, *REVISTA GEINTEC-GESTAO INOVACAO E TECNOLOGIAS*, vol. 11, no. 3, pp. 271–286, 2021.

A. Ali, M. Ahmed, and A. Khan, ‘Audit Logs Management and Security - A Survey’, *Kuwait Journal of Science*, vol. 48, no. 3, 2021, doi: 10.48129/kjs.v48i3.10624.

S. Zou, H. Sun, G. Xu, and R. Quan, ‘Ensemble strategy for insider threat detection from user activity logs’, *Computers, Materials and Continua*, vol. 65, no. 2, 2020, doi: 10.32604/cmc.2020.09649.

M. R. Anwar, R. Panjaitan, and R. Supriati, ‘Implementation Of Database Auditing By Synchronization DBMS’, *International Journal of Cyber and IT Service Management*, vol. 1, no. 2, 2021, doi: 10.34306/ijcitsm.v1i2.53.

A. Villalón-Huerta, I. Ripoll-Ripoll, and H. Marco-Gisbert, ‘A Taxonomy for Threat Actors’ Delivery Techniques’, *Applied Sciences*, vol. 12, no. 8, p. 3929, 2022.

N. Alhebaishi, L. Wang, S. Jajodia, and A. Singhal, ‘Threat modeling for cloud data center infrastructures’, in *Foundations and Practice of Security: 9th International Symposium, FPS 2016, Québec City, QC, Canada, October 24-25, 2016, Revised Selected Papers 9*, Springer, 2017, pp. 302–319.

W. Xiong and R. Lagerström, ‘Threat modeling – A systematic literature review’, *Comput Secur*, vol. 84, pp. 53–69, Jul. 2019, doi: 10.1016/J.COSE.2019.03.010.

Ö. Aslan, S. S. Aktuğ, M. Ozkan-Okay, A. A. Yilmaz, and E. Akin, ‘A Comprehensive Review of Cyber Security Vulnerabilities, Threats, Attacks, and Solutions’, 2023. doi: 10.3390/electronics12061333.

S. He, J. Zhu, P. He, and M. R. Lyu, ‘Loghub: a large collection of system log datasets towards automated log analytics’, *arXiv preprint arXiv:2008.06448*, 2020.

J. Zhu *et al.*, ‘Tools and benchmarks for automated log parsing’, in *2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP)*, IEEE, 2019, pp. 121–130.

A. Oliner, A. Ganapathi, and W. Xu, ‘Advances and challenges in log analysis’, *Commun ACM*, vol. 55, no. 2, 2012, doi: 10.1145/2076450.2076466.

L. Zhang and B. Ashuri, ‘BIM log mining: Discovering social networks’, *Autom Constr*, vol. 91, 2018, doi: 10.1016/j.autcon.2018.03.009.

H. Amar, L. Bao, N. Busany, D. Lo, and S. Maoz, ‘Using finite-state models for log differencing’, in *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2018, pp. 49–59.

J. Wang *et al.*, ‘LogEvent2vec: LogEvent-to-vector based anomaly detection for large-scale logs in internet of things’, *Sensors*, vol. 20, no. 9, p. 2451, 2020.

V. Zeufack, D. Kim, D. Seo, and A. Lee, ‘An unsupervised anomaly detection framework for detecting anomalies in real time through network system’s log files analysis’, *High-Confidence Computing*, vol. 1, no. 2, p. 100030, 2021.

‘FACEBOOK STATISTICS AND TRENDS’, https://datareportal.com/essential-facebook-stats.

A. Sandhu and S. Mohammed, ‘Detecting Anomalies in Logs by Combining NLP features with Embedding or TF-IDF’, 2022.

H. Hammouchi, O. Cherqi, G. Mezzour, M. Ghogho, and M. El Koutbi, ‘Digging deeper into data breaches: An exploratory data analysis of hacking breaches over time’, in *Procedia Computer Science*, 2019. doi: 10.1016/j.procs.2019.04.141.

Z. Yu, S. Yang, Z. Li, L. Li, H. Luo, and F. Yang, ‘LogMS: a multi-stage log anomaly detection method based on multi-source information fusion and probability label estimation’, *Front Phys*, vol. 12, p. 1401857, 2024.

S. Dong, P. Wang, and K. Abbas, ‘A survey on deep learning and its applications’, 2021. doi: 10.1016/j.cosrev.2021.100379.

M. Du, F. Li, G. Zheng, and V. Srikumar, ‘DeepLog: Anomaly detection and diagnosis from system logs through deep learning’, in *Proceedings of the ACM Conference on Computer and Communications Security*, 2017. doi: 10.1145/3133956.3134015.

S. Garg, K. Kaur, N. Kumar, G. Kaddoum, A. Y. Zomaya, and R. Ranjan, ‘A hybrid deep learning-based model for anomaly detection in cloud datacenter networks’, *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, 2019, doi: 10.1109/TNSM.2019.2927886.

Q. Lin, H. Zhang, J. G. Lou, Y. Zhang, and X. Chen, ‘Log clustering based problem identification for online service systems’, in *Proceedings - International Conference on Software Engineering*, 2016. doi: 10.1145/2889160.2889232.

L. Huang, J. Ran, W. Wang, T. Yang, and Y. Xiang, ‘A multi-channel anomaly detection method with feature selection and multi-scale analysis’, *Computer Networks*, vol. 185, p. 107645, 2021, doi: https://doi.org/10.1016/j.comnet.2020.107645.

Y. Wang, J. Peng, J. Zhang, R. Yi, Y. Wang, and C. Wang, ‘Multimodal Industrial Anomaly Detection via Hybrid Fusion’, *arXiv preprint arXiv:2303.00601*, 2023.

X. Shu, J. Smiy, D. Yao, and H. Lin, ‘Massive Distributed and Parallel Log Analysis For Organizational Security’.

J. Tao, W. Wang, N. Zheng, T. Han, Y. Chang, and X. Zhan, ‘An Abnormal Login Detection Method Based on Multi-source Log Fusion Analysis’, in *2019 IEEE International Conference on Big Knowledge (ICBK)*, 2019, pp. 229–235. doi: 10.1109/ICBK.2019.00038.

H. Eldardiry *et al.*, ‘Multi-source fusion for anomaly detection: using across-domain and across-time peer-group consistency checks.’, *J. Wirel. Mob. Networks Ubiquitous Comput. Dependable Appl.*, vol. 5, no. 2, pp. 39–58, 2014.

P. Jakob, M. Madan, T. Schmid-Schirling, and A. Valada, ‘Multi-perspective anomaly detection’, *Sensors*, vol. 21, no. 16, p. 5311, 2021.

V.-H. Le and H. Zhang, ‘Log-based anomaly detection with deep learning: How far are we?’, in *Proceedings of the 44th International Conference on Software Engineering*, 2022, pp. 1356–1367.

Z. Jia, C. Shen, X. Yi, Y. Chen, T. Yu, and X. Guan, ‘Big-data analysis of multi-source logs for anomaly detection on network-based system’, in *IEEE International Conference on Automation Science and Engineering*, 2018. doi: 10.1109/COASE.2017.8256257.

Q. Hu, B. Tang, and D. Lin, ‘Anomalous user activity detection in enterprise multi-source logs’, in *IEEE International Conference on Data Mining Workshops, ICDMW*, 2017. doi: 10.1109/ICDMW.2017.110.

L. Huang, J. Ran, W. Wang, T. Yang, and Y. Xiang, ‘A multi-channel anomaly detection method with feature selection and multi-scale

analysis', *Computer Networks*, vol. 185, p. 107645, 2021, doi: https://doi.org/10.1016/j.comnet.2020.107645.

R. Al-Shaer, J. M. Spring, and E. Christou, 'Learning the associations of mitre att & ck adversarial techniques', in *2020 IEEE Conference on Communications and Network Security (CNS)*, IEEE, 2020, pp. 1–9.

S. Lu, X. Wei, Y. Li, and L. Wang, 'Detecting Anomaly in Big Data System Logs Using Convolutional Neural Network', in *2018 IEEE 16th Intl Conf on Dependable, Autonomic and Secure Computing, 16th Intl Conf on Pervasive Intelligence and Computing, 4th Intl Conf on Big Data Intelligence and Computing and Cyber Science and Technology Congress(DASC/PiCom/DataCom/CyberSciTech)*, IEEE, Aug. 2018, pp. 151–158. doi: 10.1109/DASC/PiCom/DataCom/CyberSciTec.2018.00037.

N. Moukafih, G. Orhanou, and S. Elhajji, 'Mobile agent-based SIEM for event collection and normalization externalization', *Information & Computer Security*, vol. 28, no. 1, pp. 15–34, 2020.

A. Majeed, R. ur Rasool, F. Ahmad, M. Alam, and N. Javaid, ‘Near-miss situation based visual analysis of SIEM rules for real time network security monitoring’, *J Ambient Intell Humaniz Comput*, vol. 10, pp. 1509–1526, 2019.

S. Gnatyuk, R. Berdibayev, A. Fesenko, O. Kyryliuk, and A. Bessalov, ‘Modern SIEM Analysis and Critical Requirements Definition in the Context of Information Warfare’, *Cybersecurity Providing in Information and Telecommunication Systems II 2021*, vol. 3188, no. 2, pp. 149–166, 2021.

S. D. Çakmakçı, H. Hutschenreuter, C. Maeder, and T. Kemmerich, ‘A framework for intelligent DDoS attack detection and response using SIEM and ontology’, in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, IEEE, 2021, pp. 1–6.

T. Laue, C. Kleiner, K.-O. Detken, and T. Klecker, ‘A SIEM architecture for multidimensional anomaly detection’, in *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, IEEE, 2021, pp. 136–142.

E. T. Anumol, ‘Use of machine learning algorithms with SIEM for attack prediction’, in *Intelligent Computing, Communication and Devices: Proceedings of ICCD 2014, Volume 1*, Springer, 2015, pp. 231–235.

M. Hristov, M. Nenova, G. Iliev, and D. Avresky, ‘Integration of Splunk Enterprise SIEM for DDoS Attack Detection in IoT’, in *2021 IEEE 20th International Symposium on Network Computing and Applications (NCA)*, IEEE, 2021, pp. 1–5.

R. C. King, E. Villeneuve, R. J. White, R. S. Sherratt, W. Holderbaum, and W. S. Harwin, ‘Application of data fusion techniques and technologies for wearable health monitoring’, 2017. doi: 10.1016/j.medengphy.2016.12.011.

N. AlDahoul, H. Abdul Karim, and A. S. Ba Wazir, ‘Model fusion of deep neural networks for anomaly detection’, *J Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00496-w.

H. Li, G. Zhou, S. Zhou, S. Chen, S. Mao, and T. Jin, ‘Multi-source Heterogeneous Log Fusion Technology of Power Information System Based on Big Data and Imprecise Reasoning Theory’, in *International Conference on Communication Technology Proceedings, ICCT*, 2020. doi: 10.1109/ICCT50939.2020.9295848.

R. Gerhards, ‘The syslog protocol’, 2009.

T. B. T. Nguyen, T. L. Liao, and T. A. Vu, ‘Anomaly detection using one-class SVM for logs of juniper router devices’, in *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering, LNICST*, 2019. doi: 10.1007/978-3-030-30149-1_24.

A. Hwoij, M. Al-Kasassbeh, and M. Al-Fayoumi, ‘Detecting Network Anomalies using Rule-based machine learning within SNMP-MIB dataset’, *arXiv preprint arXiv:2002.02368*, 2020.

E. Gamess and S. Hernandez, ‘Performance Evaluation of SNMPv1/2c/3 using Different Security Models on Raspberry Pi’, 2021.

B. Claise, ‘Rfc 3954: Cisco systems netflow services export version 9’, 2004, *RFC Editor*.

A. Banerjee, R. A. K. Saravanaguru, and A. Thangavelu, ‘Integrating Distributed Traffic Monitoring Systems using Java Messaging Service Framework’.

J. Tao, N. Zheng, W. Wang, T. Han, X. Zhan, and Q. Luan, ‘A behavior sequence clustering-based enterprise network anomaly host recognition method’, in *Proceedings - 10th IEEE International Conference on Big Knowledge, ICBK 2019*, 2019. doi: 10.1109/ICBK.2019.00039.

B. Glasbergen, M. Abebe, K. Daudjee, and A. Levi, ‘Sentinel: universal analysis and insight for data systems’, *Proceedings of the VLDB Endowment*, vol. 13, no. 12, pp. 2720–2733, 2020.

C.-I. Fan, H.-W. Hsiao, C.-H. Chou, and Y.-F. Tseng, ‘Malware detection systems based on API log data mining’, in *2015 IEEE 39th annual computer software and applications conference*, IEEE, 2015, pp. 255–260.

F. Larghi, ‘LLAMA. A system for log management and analysis on a complex distributed environment’, 2019.

F. Sönmez, M. Zontul, O. Kaynar, and H. Tutar, ‘Anomaly detection using data mining methods in it systems: a decision support application’, *Sakarya University Journal of Science*, vol. 22, no. 4, pp. 1109–1123, 2018.

P. Dymora and M. Mazurek, ‘An innovative approach to anomaly detection in communication networks using multifractal analysis’, *Applied Sciences*, vol. 10, no. 9, p. 3277, 2020.

S. Gómez-Meire, C. G. Márquez, E. P. Aray-Cappello, and J. R. Méndez, ‘Using Live Spam Beater (LiSB) Framework for Spam Filtering during SMTP Transactions’, *Applied Sciences*, vol. 12, no. 20, p. 10491, 2022.

H. Studiawan, F. Sohel, and C. Payne, ‘A survey on forensic investigation of operating system logs’, *Digit Investig*, vol. 29, pp. 1–20, 2019.

‘Event Viewer’, https://learn.microsoft.com/en-us/shows/inside/event-viewer. Accessed: May 17, 2023. [Online]. Available: https://learn.microsoft.com/en-us/shows/inside/event-viewer.

B. I. Mokhtar, A. D. Jurcut, M. S. ElSayed, and M. A. Azer, ‘Active Directory Attacks—Steps, Types, And Signatures’, *Electronics (Basel)*, vol. 11, no. 16, p. 2629, 2022.

A. A. Hamza and J. s urayh Al-Janabi, ‘Detecting Brute Force Attacks on SSH and FTP Protocol Using Machine Learning: A Survey’, *Journal of Al-Qadisiyah for Computer Science and Mathematics*, vol. 16, no. 1, pp. 21–31, 2024.

A. Adekotujo, A. Odumabo, A. Adedokun, and O. Aiyeniko, ‘A comparative study of operating systems: Case of windows, unix, linux, mac, android and ios’, Int J Comput Appl, vol. 176, no. 39, pp. 16–23, 2020.

B. Ward, *How Linux works: What every superuser should know*. no starch press, 2021.

J. Corbet, A. Rubini, and G. Kroah-Hartman, *Linux device drivers*. ‘ O’Reilly Media, Inc.’, 2005.

C. Doskas, ‘Log Analysis Series’, 2021.

A. Tariq, J. Manzoor, M. A. Aziz, Z. U. A. Tariq, and A. Masood, ‘Open source SIEM solutions for an enterprise’, *Information & Computer Security*, vol. 31, no. 1, pp. 88–107, 2022.

M. Baruch and G. David, ‘Domain generation algorithm detection using machine learning methods’, *Cyber security: power and technology*, pp. 133–161, 2018.

M. Skwarek, M. Korczynski, W. Mazurczyk, and A. Duda, ‘Characterizing vulnerability of DNS AXFR transfers with global-scale scanning’, in *2019 IEEE Security and Privacy Workshops (SPW)*, IEEE, 2019, pp. 193–198.

S. TOPRAK and A. G. YAVUZ, ‘Web application firewall based on anomaly detection using deep learning’, *Acta Infologica*, vol. 6, no. 2, pp. 219–244, 2022.

L. Fang, H. Wu, K. Qian, W. Wang, and L. Han, ‘A Comprehensive Analysis of DDoS attacks based on DNS’, in *Journal of Physics: Conference Series*, IOP Publishing, 2021, p. 012027.

E. AlMasri, M. Alkasassbeh, and A. Aldweesh, ‘Towards Generating a Practical SUNBURST Attack Dataset for Network Attack Detection.’, *Computer Systems Science & Engineering*, vol. 47, no. 2, 2023.

A. Jenkinson, *Stuxnet to Sunburst: 20 Years of Digital Exploitation and Cyber Warfare*. CRC Press, 2021.

V. Chitraa and D. A. S. Thanamani, 'Web log data cleaning for enhancing mining process', *vol*, vol. 1, p. 7, 2012.

X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, 'Data cleaning: Overview and emerging challenges', in *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 2016. doi: 10.1145/2882903.2912574.

F. Skopik, M. Wurzenberger, and M. Landauer, *Smart Log Data Analytics*. Springer, 2021.

C. C. Aggarwal, *Data Mining: The Textbook*. 2015. doi: 10.1007/978-3-319-14142-8_10.

J. S. P. Fong, K. Wong Ting Yan, J. S. P. Fong, and K. Wong Ting Yan, ‘Data Normalization’, *Information Systems Reengineering, Integration and Normalization: Heterogeneous Database Connectivity*, pp. 287–316, 2021.

M. Egger, F. Sulu-Gambari, and L. Lebreton, ‘First evidence of plastic fallout from the North Pacific Garbage Patch’, *Sci Rep*, vol. 10, no. 1, p. 7495, 2020.

Appendices

Appendix 1: XML Parser Code

```
<patternDefinitions>
<pattern name="patYear"><![CDATA[\d{4}]]></pattern>
<!-- the strict version of host name pattern (RFC952) -->
<pattern name="patHostName"><![CDATA[[A-Za-z](?:[0-9A-Za-z.-]*[0-9A-Za-
z])?]]></pattern>
</patternDefinitions>
<eventFormatRecognizer><![CDATA[^\s*<:gPatSyslogPRI>?(?:<:gPatWeekday>\s+)
?<:gPatMon>\s+
<:gPatDay>\s+(?:<:gPatTime>\s+alertcertlocal\s+<:patYear>|<:patYear>\s+<:gPatTim
e>|<:gPatTime>)\s+
alertcertlocal\s+(?:<:patHostName>\s+<:gPatIpAddr>|\[<:gPatIpAddr>\]|<:gPatIpAdd
r>|<:patHostName>)\s]]>
</eventFormatRecognizer>
<parsingInstructions>
<switch>
<case>
<collectFieldsByRegex src="$_rawmsg">
<regex><![CDATA[<:gPatMon>\s+<:gPatDay>\s+(?:<:gPatTime>\s
+<:patYear>|<:patYear>\s+<:gPatTime>|<:gPatTime>)\s+
(?:<reptDevName:patHostName>\s+
```

<_reptIp:gPatIpAddr>|\[<_reptIp:gPatIpAddr>\]|<_reptIp:gPatIpAddr>|<reptDevNam

e:patHostName>)\b]]>

</regex>

</collectFieldsByRegex>

<when test="exist reptDevName">

<when test="matches($reptDevName, 'alertcertlocal')">

<setEventAttribute attr="_body">replaceStringByRegex($_rawmsg, """,

"")</setEventAttribute>

<collectFieldsByKeyValuePair kvsep="=" sep=" " src="$_body">

<attrKeyMap attr="subtype" key="subtype"/>

<attrKeyMap attr="logLevel" key="level"/>

<attrKeyMap attr="eventDesc" key="logdesc"/>

<attrKeyMap attr="eventtype" key="eventtype"/>

<attrKeyMap attr="srcIpAddr" key="srcip"/>

<attrKeyMap attr="destIpAddr" key="dstip"/>

<attrKeyMap attr="action" key="action"/>

<attrKeyMap attr="newStatus" key="status"/>

<attrKeyMap attr="reason" key="reason"/>

<attrKeyMap attr="msg" key="msg"/>

</collectFieldsByKeyValuePair>

</when>

</when>

<setEventAttribute attr="_syslogNGHeader">yes</setEventAttribute>

</case>

```
<default/>
</switch>
<when test="exist _reptIp">
<when test="$_reptIp != '127.0.0.1'">
<when test="$_reptIp != '0.0.0.0'">
<when test="$_reptIp != '255.255.255.255'">
<setEventAttribute attr="reptDevIpAddr">$_reptIp</setEventAttribute>
</when>
</when>
</when>
</when>
</parsingInstructions>
```

Appendix 2: Code

```
import csv
import matplotlib.pyplot as plt
import plotly.express as px
def search_in_two_files(file_path1, file_path2, file_path3, file_path4,  keyword):
    match_found_in_file1 = False
    match_found_in_file2 = False
    match_found_in_file3 = False
    match_found_in_file4 = False
    count_file1 = 0
    count_file2 = 0
    count_file3 = 0
    count_file4 = 0
    print(f"Searching for '{keyword}' in {file_path1} and {file_path2} and {count_file3}
and {count_file4}:")
    with open(file_path1, 'r', newline='', encoding='utf-8') as csvfile:
        reader = csv.reader(csvfile)
        header = next(reader)  // readimg
        for row in reader:
            if any(keyword.lower() in column.lower() for column in row):
                match_found_in_file1 = True
                count_file1 += 1
                print(f"Found '{keyword}' in {file_path1}, row: {row}")
```

```
with open(file_path2, 'r', newline='', encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile)
    header = next(reader)
    for row in reader:
        if any(keyword.lower() in column.lower() fr column in row):
            match_found_in_file2 = True
            count_file2 += 1
            print(f"Found '{keyword}' in {file_path2}, row: {row}")
with open(file_path3, 'r', newline='', encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile)
    header = next(reader)
    for row in reader:
        if any(keyword.lower() in column.lower() for column in row):
            match_found_in_file3 = True
            count_file3 += 1
            print(f"Found '{keyword}' in {file_path3}, row: {row}")
with open(file_path4, 'r', newline='', encoding='utf-8') as csvfile:
    reader = csv.reader(csvfile)
    header = next(reader)
    for row in reader:
        if any(keyword.lower() in column.lower() for column in row):
            match_found_in_file4 = True
            count_file4 += 1
            print(f"Found '{keyword}' in {file_path4}, row: {row}")
```

```
    if match_found_in_file1 and match_found_in_file2 and match_found_in_file3 and
match_found_in_file4:
        print(f"Matches found for '{keyword}' in both files.")
        print(f"Total occurrences of '{keyword}' in {file_path1}: {count_file1}")
        print(f"Total occurrences of '{keyword}' in {file_path2}: {count_file2}")
        print(f"Total occurrences of '{keyword}' in {file_path3}: {count_file3}")
        print(f"Total occurrences of '{keyword}' in {file_path4}: {count_file4}")
        print(f"Total occurrences of '{keyword}' in both files: {count_file1 + count_file2 +
count_file3 + count_file4}")
        # Plotting
        labels = [f'{file_path1} ({count_file1} occurrences)', f'{file_path2} ({count_file2}
occurrences)',f'{file_path3} ({count_file3} occurrences)',f'{file_path4} ({count_file4}
occurrences)']
        values = [count_file1, count_file2, count_file3, count_file4]
        fig = px.bar(x=labels, y=values, labels={'x': 'Files', 'y': 'Occurrences'},
                     title=f'Occurrences of "{keyword}" in Files')
        fig.show()
    else:
        print(f"No matches found for '{keyword}' in both files.")
        print(f"Total occurrences of '{keyword}' in {file_path1}: {count_file1}")
        print(f"Total occurrences of '{keyword}' in {file_path2}: {count_file2}")
        print(f"Total occurrences of '{keyword}' in {file_path3}: {count_file3}")
        print(f"Total occurrences of '{keyword}' in {file_path4}: {count_file4}")
# Example usage
```

```
file_path1 = 'file2_original.csv'

file_path2 = 'file2_original.csv'

file_path3 = 'file1_original.csv'

file_path4 = 'file2_original.csv'

keyword_to_search = input("Enter the keyword to search for: ")

search_in_two_files(file_path1, file_path2, file_path3, file_path4, keyword_to_search)
```

Appendix 3: API

```
import requests
import json
import os
requests.packages.urllib3.disable_warnings()
BASE_URL = "https://192.168.2.100"
API_LOGIN = "/logincheck"
API_LOGOUT = "/logout"
API_FIREWALL_POLICY = "/api/v2/cmdb/firewall/policy/"
API_FIREWALL_ADDRESS = "/api/v2/cmdb/firewall/address/"
USERNAME = "testuser"
PASSWORD = "Test@2023"
BLOCKED_IP_FILE = "blocked_ip.json"  # ibrhahim breeak
def login():
    login_url = f"{BASE_URL}{API_LOGIN}"
    payload = {
        'username': USERNAME,
        'secretkey': PASSWORD
    }
    response = requests.post(login_url, data=payload, verify=False)
    if response.status_code == 200 and 'apsid' in response.cookies:
        return response.cookies
    else:
        raise Exception(f"Failed to log in to Fortinet firewall: {response.text}")
```

```
# Function to get firewall policies
def get_firewall_policies(cookies):
    policy_url = f"{BASE_URL}{API_FIREWALL_POLICY}"
    response = requests.get(policy_url, cookies=cookies, verify=False)
    if response.status_code == 200:
        return response.json()
    else:
        raise Exception(f"Failed to fetch firewall policies: {response.text}")
def create_address_object(cookies, address_name, ip):
    address_url = f"{BASE_URL}{API_FIREWALL_ADDRESS}"
    headers = {
        'Content-Type': 'application/json'
    }
    payload = {
        "name": address_name,
        "subnet": ip + "/32"
    }
    response = requests.post(address_url, headers=headers, cookies=cookies,
data=json.dumps(payload), verify=False)

    if response.status_code == 200:
        return response.json()
```

```
    else:
        raise Exception(f"Failed to create address object: {response.text}")


// start second
def create_firewall_policy(cookies, policy_data):
    policy_url = f"{BASE_URL}{API_FIREWALL_POLICY}"
    headers = {
        'Content-Type': 'application/json'
    }
    response = requests.post(policy_url, headers=headers, cookies=cookies,
data=json.dumps(policy_data), verify=False)
    if response.status_code == 200:
        return response.json()
    else:
        raise Exception(f"Failed to create firewall policy: {response.text}")
# Function to logout
def logout(cookies):
    logout_url = f"{BASE_URL}{API_LOGOUT}"
    response = requests.post(logout_url, cookies=cookies, verify=False)
    if response.status_code != 200:
        raise Exception(f"Failed to log out from Fortinet firewall: {response.text}")
file_path = / waseem / project / siem / out.json
def read_blocked_ip(file_path):
    if not os.path.exists(file_path):
```

```
        raise Exception(f"Blocked IP file not found: {file_path}")
    with open(file_path, 'r') as file:
        data = json.load(file)
        if 'blocked_ip' not in data:
            raise Exception("Blocked IP not found in the file")
        return data['blocked_ip']
def main():
    try:
        # Read the blocked IP from the JSON file
        blocked_ip = read_blocked_ip(BLOCKED_IP_FILE)
        address_name = f"Block_IP_{blocked_ip.replace('.', '_')}"
        print(f"Read blocked IP: {blocked_ip}")
        print("Logging in...")
        cookies = login()
        print("Login successful.")
        print(f"Creating address object for IP: {blocked_ip}")
        address_object = create_address_object(cookies, address_name, blocked_ip)
        print(f"Address object created successfully: {json.dumps(address_object,
indent=4)}")
        new_policy = {
            "json": {
                "name": f"Block_{blocked_ip}",
                "srcintf": [
                    {
```

```
        "name": "any"
    }
],
"dstintf": [
    {
        "name": "any"
    }
],
"srcaddr": [
    {
        "name": address_name
    }
],
"dstaddr": [
    {
        "name": "all"
    }
],
"action": "deny",
"status": "enable",
"schedule": "always",
"service": [
    {
        "name": "ALL"
```

```
                }
            ],
            "logtraffic": "all"
        }
    }
    print("Creating new firewall policy to block the IP...")
    created_policy = create_firewall_policy(cookies, new_policy)
    print("New policy created successfully:")
    print(json.dumps(created_policy, indent=4))
    print("Logging out...")
    logout(cookies)
    print("Logout successful.")
except Exception as e:
    print(f "An error occurred: {e}")
if __name__ == "__main__":
    main()
```

المُلخص

في بيئة مراكز بيانات الحاسوب، يعتبر تشغيل الأنظمة وبقاءها متوفرة حساساً للغاية نظرًا للطبيعة الحرجة للخدمات التي تُقدمها. كما ويعد الحفاظ على العمليات التي تقوم بها تلك الانظمة دون انقطاع وتحقيق مستويات الامن الثلاث (السرية والنزاهة والتوافرية) أمرًا ضروريًا. في الوقت الحالي، تتم مراقبة سجلات الأحداث الأمنية بشكل فردي ومنفصل والتي يتم الحصول عليها من قنوات مختلفة بما فيها جدران الحماية والمُنقحات ومكافح الفيروسات وأجهزة الأمان الأخرى. ويحدث ذلك بإدارة الاحداث الأمنية على كل جهاز منفرداً دون أي ارتباطات بالحوادث الأمنية الأخرى داخل مركز البيانات. بالإضافة إلى ذلك، يتم تطبيق الإجراءات التي تتخذها أجهزة أمن المعلومات محلياً على الأجهزة المعنية دون التنسيق مع الأجهزة المجاورة مما يؤدي إلى إنشاء إطار أمني مفكك. تهدف هذه الدراسة لتعزيز وتحسين الوضع الأمني للأنظمة والبرامج التي تعمل داخل مراكز البيانات، مع التركيز على التسجيل الشامل للأحداث الأمنية من أجل ضمان سلامة وتوافرية وسرية الخدمات التي تُقدم للأفراد وبين الأنظمة بذاتها.

تقدم هذه الدراسة نموذجاً إجرائياً أمنياً جديداً لسد الفجوة في الإطار الأمني المذكور أعلاه. يدمج هذا النموذج قرار قواعد اللعبة (Playbook) في مفهوم التنسيق المتناغم للأجهزة الطرفية لمركز البيانات من خلال تنفيذ هذا النموذج الجديد على الأجهزة والتطبيقات داخل مركز البيانات بشكل متزامن ومتناغم. ولمعالجة هذه الفجوة في البنية الأمنية الحالية، تم العمل على تطوير النموذج الإجرائي لمحاكاة التفاهمات بين سجلات الاحداث الأمنية المنصهرة والواردة الى قاعدة السجلات المدمجة. يعمل هذا النموذج في إطار عمل الدليل التشغيلي بين نقاط السلسلة في الشبكة والتي تمثل الأجهزة المُرسلة للسجلات الأمنية وتنفيذ الإجراءات على كتل السلسلة الأمنية بناء على أفضل القرارات المُتخذة. ويتمحور مفهوم الدليل التشغيلي حول إنشاء سير عمل تنسيقي يعمل على مبدأ تنفيذ قواعد اللعب عبر جميع الأنظمة.

بعد تنفيذ هذا النموذج الجديد، لوحظت تحسينات كبيرة على وجه الخصوص في زيادة فعالية نظام الحماية (IPS) استنادًا إلى الإجراءات المنعكسة على كافة أجزاء السلسلة، بالإضافة إلى انخفاض كبير في الإيجابيات الكاذبة مقارنة بإطار عمل IDS/IPS السابق. هذا يبرز فعالية النموذج الإجرائي الأمني الجديد في تعزيز الوضع الأمني وسلامة العمليات في بيئة مركز البيانات.