Scientific Research Publishing

Journal of Information Security, 2024, 15, 378-409
https://www.scirp.org/journal/jis
ISSN Online: 2153-1242
ISSN Print: 2153-1234

Incident Detection Based on Differential Analysis

Mohammed Ali Elseddig[1], Mohamed Mejri[2]

[1]Computer Science and Information Technology, Sudan University of Science and Technology, Khartoum, Sudan
[2]Computer Science Department, Laval University, Quebec, Canada
Email: moho.school@gmail.com, momej@iulaval.ca

How to cite this paper: Ali Elsiddig, M. and Mejri, M. (2024) Incident Detection Based on Differential Analysis. *Journal of Information Security*, **15**, 378-409.
https://doi.org/10.4236/jis.2024.153022

Received: May 29, 2024
Accepted: July 26, 2024
Published: July 29, 2024

Abstract

Internet services and web-based applications play pivotal roles in various sensitive domains, encompassing e-commerce, e-learning, e-healthcare, and e-payment. However, safeguarding these services poses a significant challenge, as the need for robust security measures becomes increasingly imperative. This paper presented an innovative method based on differential analyses to detect abrupt changes in network traffic characteristics. The core concept revolves around identifying abrupt alterations in certain characteristics such as input/output volume, the number of TCP connections, or DNS queries—within the analyzed traffic. Initially, the traffic is segmented into distinct sequences of slices, followed by quantifying specific characteristics for each slice. Subsequently, the distance between successive values of these measured characteristics is computed and clustered to detect sudden changes. To accomplish its objectives, the approach combined several techniques, including propositional logic, distance metrics (e.g., Kullback-Leibler Divergence), and clustering algorithms (e.g., K-means). When applied to two distinct datasets, the proposed approach demonstrates exceptional performance, achieving detection rates of up to 100%.

Keywords

IDS, SOC, SIEM, KL-Divergence, K-Mean, Clustering Algorithms, Elbow Method

1. Introduction

The 21st century has witnessed the profound impact of the Internet, emerging as one of the most transformative inventions in our lives. Presently, the Internet transcends numerous boundaries, revolutionizing the way we communicate, engage in recreational activities, conduct work, shop, socialize, enjoy music and

movies, order food, manage finances, extend birthday wishes to friends, and more. The indispensability of these service applications is paramount for modern organizations, demanding uninterrupted availability and global accessibility around the clock.

The exponential growth of sensitive services and web-based applications has become a magnet for hackers seeking lucrative gains, technological secrets, including vaccine-related information, or any competitive edge. This surge in valuable data has not only enticed criminal organizations globally but has also led certain governmental entities to recruit exceptionally skilled security experts for cyberattack operations.

The continuous expansion of both lawful and unlawful activities has led to an exponential increase in the complexity and volume of Internet traffic. As a result, network security administrators grapple with ever-evolving and intricate challenges, striving to swiftly impede malicious traffic. To combat this, they heavily rely on a trio of key tools: Firewalls, SIEM (Security Information and Event Management), and IDSs (Intrusion Detection Systems), which stand as primary instruments for detecting and filtering suspicious traffic.

To scrutinize and identify potentially suspicious activities within network traffic using IDSs, two primary detection methods prevail: signature-based and anomaly-based detection. Signature-based or misuse detection methods employ pattern-matching techniques to identify pre-known attacks. The primary advantage lies in their high accuracy, ensuring minimal false positives or negatives when detecting previously recognized suspicious attacks. Anomaly-based detection methods necessitate an initial phase to comprehend normal traffic patterns, employing techniques like machine learning, statistical analysis, or knowledge-based methodologies. Any significant deviation between observed traffic and established norms is flagged as suspicious. The primary advantage lies in its capability to effectively identify unknown suspicious attacks with commendable accuracy.

The current state of the art presents a myriad of intriguing techniques (e.g., [1]-[4]) and tools that have notably bolstered network security by effectively detecting and thwarting malicious traffic. Nevertheless, the challenge persists: cyberattacks persistently wreak havoc, inflicting substantial damage. Hence, any novel contribution that mitigates the risks associated with network traffic would be immensely valued.

This paper introduces a novel technique employing differential analysis to discern suspicious network traffic. The approach initially segments traffic into small-time slices, transforming each of them into a value in $\mathbb{R}^n$. Subsequently, it computes the divergence between neighboring slices to unveil abrupt changes in traffic behavior. After that, clustering techniques are applied to abstracted intervals to validate traffic homogeneity (a single class) or detect significant variations (multiple classes), indicating potential suspicious activities.

The approach we introduce is geared towards enhancing the efficiency of Security Information Event Management (SIEM) [5], an integral component of a

Security Operations Center (SOC) [6]. A (SIEM), such as Wazuh [7], encapsulates a suite of functionalities aimed at gathering, analyzing, and presenting information sourced from network and security devices. It essentially integrates two vital components: Security Information Management (SIM) and Security Event Management (SEM). SIM focuses on storing, analyzing, and reporting log files, while SEM is responsible for real-time monitoring, event correlation, notifications, and console views.

The rest of this paper is organized as follows. Section 2 delves into related works within the field. Section 3 details the methodology of the approach. Section 4 presents three case studies. Finally, concluding remarks are presented in Section 5.

2. Related Work

The state of the art contains many valuable techniques that have significantly contributed to the improvement of the security of network services and applications. Here, the study focuses on anomaly-based detection techniques and methods that try to detect suspicious traffic based on IP packets information such as IP address (layer 3 in the TCP/IP Model), TPC or UDP ports (layer 4) and web application data (layer 5).

Najafabadi *et al.* proposed in [8] an anomaly detection mechanism for detecting HTTP GET flood attacks. They used the Principal Component Analysis (PCA)-subspace method on the browsing behavior instances extracted from HTTP server's logs in order to detect abnormal behaviors. They apply the approach to detect some DDoS and HTTP GET flood attacks. This approach used the supervised machine learning techniques.

In [9], Betarte *et al.* proposed a method based on machine learning to enhance the famous ModSecurity [10], a Web Application Firewall provided by OWASP, by using one-class classification and n-gram techniques on three datasets. The proposal method used the supervised machine learning techniques and provides better detection and false positive rates than the original version of ModSecurity.

Wang *et al.* presented in [11] a new web anomaly detection method which uses Frequent Closed Episode Rules Mining (FCERMining) algorithm to analyze web logs and detect new unknown web attacks. The method used the supervised machine learning techniques and has a detection rate of 96.67% and a false alarm rate of 3.33% for detecting abnormal users.

In [12], Brontë *et al.*, proposed an anomaly detection approach that uses the cross-entropy technique to calculate three metrics: cross entropy parameters (CEP), cross entropy value (CEV) and cross entropy data type (CET). These metrics aim to compare the deviation between learned request profiles and a new web request. The cross-entropy approach performs better than Value Length and Mahalanobis distance approach. This approach used the supervised machine learning techniques, focused on detecting four types of web attacks: SQLI, XSS, RFI, and DT and has a detection rate of 66.7%.

Ren *et al.* presented in [13] a method based on the bag of words (BOW) model to extract features and efficiently detect web attacks with hidden Markov algorithms. BOW has higher detection rate and lower false alarm rate when compared with N-gram feature-extraction algorithms. This approach used the supervised machine learning techniques to detecting SQL injection and cross-site scripting attacks. The accuracy increased to 96%, but the false alarm rate still remained low.

In [14], Pukkawanna *et al.* proposed a method using port pair distribution and Kullback-Leibler (KL) divergence to detect suspicious flows when the KL divergence deviates from an adaptive 3-sigma rule-based threshold. This approach used the unsupervised machine learning techniques to detecting mimicry attacks. The approach does not need any previous learning step.

Hounkpevi proposed in [15] a method using K-means, port pair distribution and Kullback-Leibler (KL) algorithm that improves [14]. The approach compares the traffic of current time intervals with the nearby ones by applying the k-mean algorithm. Any significant divergence means that the current time interval traffic is suspicion. This approach used the unsupervised machine learning techniques to detecting mimicry attacks. The proposal approach seems more efficient than [14].

In [16], Munz *et al.* presented a novel Network Data Mining approach that applies the K-means clustering algorithm to feature datasets extracted from flow records. Training data containing unlabelled flow records are separated into clusters of normal and anomalous traffic. This approach used the unsupervised machine learning techniques to detecting Port scans and D/oS attacks. In this approach there is a challenge to determine the optimum number of clusters.

Asselin *et al.* presented in [17] an anomaly detection model based on crawling method and n-gram model that is effective in reducing the access to the log file generated by the web servers. It has shown to be a good solution for web applications black-box analysis but it is not efficient for detecting attacks that use cookie or post data. This approach used the unsupervised machine learning techniques to detecting brute force, DDoS, Crawler Miss, High Load, Anomalous Query attacks and has a detection rate of 95%.

Swarnkar and Hubballi described, in [18], a new method for payload-based anomaly detection that learns normal behavior and detects deviations. The approach makes a frequency range of occurrences of n-grams from packets in training phase and count the number of deviations from the range to detect anomalies. The approach showed lower false positives and higher detection rate when compared to Anagram methods.

Kang *et al.* [19] described a one-class classification method for improving intrusion detection performance for malicious attacks. Results scores were evaluated based on artificially generated instances in two-dimensional space. In the detection phase, the approach based on simple logic, the center of the normal patterns was determined at (0, 0), and two malicious class centers were at (1, 1) and (−1, −1), respectively. Experimental results on simulated data show better

performance.

Camacho *et al.* [20] developed a framework that used a PCA-based multivariate statistical process control (MSPC) approach. The framework monitors both the Q-statistic and D-statistic. Thereby, it was possible to establish control limits in order to detect anomalies when they became consistently exceeded.

Yoshimura *et al.* [21] proposed a new model called DOC-IDS, which is an intrusion detection system based on Perera's deep one-class classification. This approach used the supervised machine learning techniques to detecting Multi-attacks and has a detection rate of 97%.

Zavrak *et al.* [22] proposed an intrusion detection and prevention architecture called SAnDet which is based on an anomaly-based attack detection module that uses the EncDecAD method to detect attacks. This approach used the semi-supervised machine learning techniques to detecting DoS and Portscan attacks and has a detection rate of 99.3%.

The evaluation of the previous approaches according to cited criteria is illustrated by Table 1.

Table 1. Evaluation of the approaches.

Author	Techniques	Attacks types	Target	Learning types	Logic rules	Training is not required	Multi-target	Detection rate
Pukkawanna *et al.* [14], 2015	Kullback-Leibler (KL) Divergence	Mimicry attacks	TCP/ UDP-Ports	unsupervised learning	×	✓	×	12.5%
Hounkpevi [15], 2020	- Kullback-Leibler (KL) Divergence. - k-mean algorithm.	Mimicry attacks	TCP/ UDP-Ports	unsupervised learning	×	✓	×	66.7%
Najafabadi *et al.* [8], 2017	PCA (Principle Component Analysis)-Subspace method	detecting HTTP GET flood attacks DDOS	HTTP.Url	supervised learning	×	×	×	-
Betarte *et al.* [9], 2018	- one-class classification - n-gram	Multi attacks	HTTP.Url	supervised learning	×	×	×	90%
Wang *et al.* [11], 2017	FCER (Frequent Closed Episode Rules) Mining algorithm	Unknown web attacks.	HTTP.Url	supervised learning	×	×	×	96.67%
Bronte *et al.* [12], 2016	Cross Entropy.	SQLI, XSS, RFI, and DT.	HTTP.Url	supervised learning	×	×	×	66.7%
Ren *et al.* [13], 2018	-Bag of words (BOW) model - Hidden Markov algorithms.	SQL injection and cross-site scripting	HTTP.Url	supervised learning	×	×	×	96%
Munz *et al.* [16], 2007	K-mean algorithm.	Port scans and D/oS attacks.	TCP/ UDP-Ports	unsupervised learning	×	✓	×	-

Continued

Asselin *et al.* [17], 2016	black-box approach (crawling based) N-gram model.	brute force, DDoS, Crawler Miss, High Load, Anomalous Query	HTTP.Url	unsupervised learning	×	✓	×	95%
Yoshimura *et al.* [21], 2022	one-class classification.	Multi attacks	-	supervised learning	×	×	×	97%
Zavrak *et al.* [22], 2023	EncDecAD. LSTM.	DoS Portscan	-	semi-supervised learning	×	×	×	99.3%

The existing approaches could be evaluated according to many criteria such as:

- Attack Types: The different types of attacks detected by the approach
- Target: The fields of the IP packet that are analyzed by the approach to detect suspicious behaviors such as IP address, HTTP.Url and TCP-UDP Port.
- Learning Types: If the approach uses any supervised or unsupervised machine learning techniques.
- Logic Rules: It is useful if the approach provides an expressive language such as temporal logic to specify a rich variety of malicious traffics (fine-grained specification).
- Training is not required: Most of existing approaches require a training step, but some few others do not.
- Multi-Target: It is related to the ability of the approach to detect suspicious traffic that requires the analysis of many fields in IP packets in the same time.
- Detection Rate: It gives the percentage of detected bad traffics.

3. Methodology

The detection of suspicious traffic is based on the following simple observation: the nature of the traffic should not change suddenly. If this happens, it will be suspicious. For example, there is no reason that the nature of the traffic between the period P_1 = [10 am - 10:30 am] will be so different from the period P_2 = [10:30 am - 11 am]. However, distinctions might reasonably exist between daytime and nighttime traffic patterns, as well as between traffic from different years.

Let $\mathcal{F}:\mathbb{R}\to\mathbb{R}$ be a function such that $y=\mathcal{F}(x)$ measures a particular feature related to the network traffic.(e.g., x is time and y is the number of packets coming from a specific country). Assume that the curve of $\mathcal{F}$ is as shown by **Figure 1**, then it is clear that there exists a sudden variation from $f(4)$ to $f(5)$ which is suspicious.

More precisely, the traffic τ will be scattered to one or many sequences of ordered slices. On each of these slices, we apply a function $\mathcal{F}$ that measure some of its features. After that, we compute the distance between successive values of $\mathcal{F}$ as shown in **Figure 2**. The sudden changes of $\mathcal{F}$ appears, if there exist a big deviations between the measured distances.

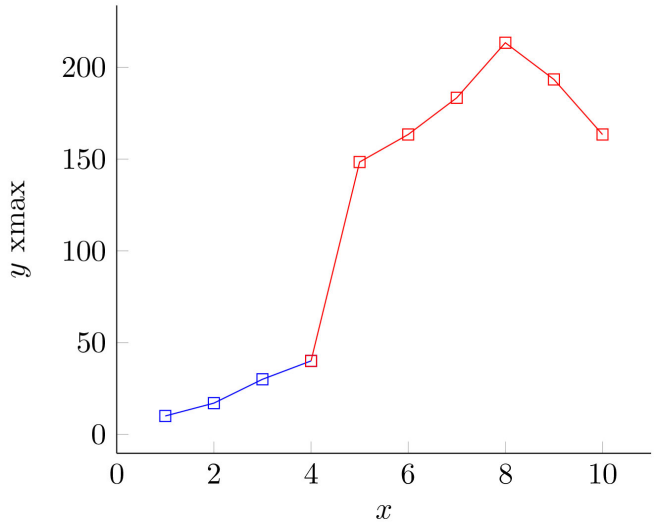

Figure 1. Sudden variation in traffic.

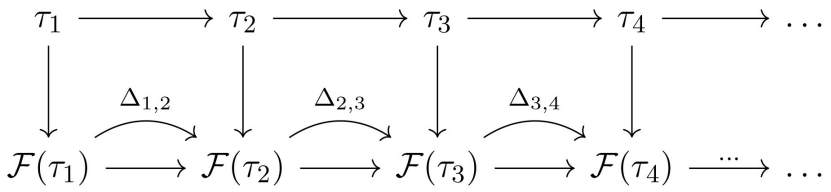

Figure 2. Looking for sudden variation in traffic.

The function F may not solely yield a singular real value within $\mathbb{R}$; instead, its outputs could exist within $\mathbb{R}^n$. For example, it might produce a complete distribution that assesses various characteristics across analyzed slices of the trace. In such scenarios, assessing the disparity between F values could involve employing measures like KL-divergence or Euclidean distance.

Furthermore, in determining whether the variation between successive F values exhibits abrupt changes or unacceptable deviations, clustering analysis could be valuable. If the resultant clusters surpass one in number, and the expectation dictates smooth change in traffic distributions across successive slices, we conclude that the analyzed traffic is suspicious.

In the subsequent sections, we elaborate on and formalize all of these analyses.

To maintain simplicity in presenting the approach, we concentrate solely on network traffic. However, it's important to note that the same concept can be extended to analyze any type of log file.

3.1. Preliminary Notations

In order to articulate the definition of suspicious traffic formal and more succinctly, it's essential to establish a set of initial notations.

We assume that network traffic is represented by a sequence of stamped IP packets or messages where each one of them is a structure that contains a header and a payload. We suppose that we have access to any field (e.g., IP addresses,

ports and protocols) to any non-encrypted header of the network protocols (e.g., IP, TCP and UDP) inside an intercepted traffic.

Definition 1 (Messages). We denote by $\mathcal{M}$ the set of messages that could be found in the network traffic.

- f_n: we use f_n to range over the possible fields in messages of $\mathcal{M}$. Examples of f_n are given in Table 2.
- $m@f_n$: if m is a message and f_n is an attribute, we denote by $m@f_n$ the value of f_n in m.

Table 2. Examples of attributes.

		Field Name
f_n	::=	T | IP.SourceAdd | IP.DestAdd | IP.Prot | IP.TTL | ...
	|	TCP.SourcePort | TCP.DestPort
	|	TCP.Flag.Syn | TCP.Flag.Fin | ...
	|	UDP.SourcePort | UDP.DestPort | ...
	|	HTTP. Url | HTTP.Method |
	|	HTTP.Cookies | HTTP.Payload ...

Stamped messages are called events and are defined as follows:

Definition 2 (Events). We denote by $\mathcal{E}$, the set of the possible events built from $\mathcal{M}$ as follows:

$$e ::= \langle t, m \rangle$$
$$t ::= time$$
$$m \in \mathcal{M}$$

- $e@f_n$: we denote by $e@f_n$ the value of f_n in e. It is defined as follows: $\langle t, m \rangle @\mathrm{T} = t$ and $\langle t, m \rangle @f_n = m@f_n$, if $f_n \neq \mathrm{T}$.

A sequence of stamped events forms a trace.

Definition 3 (Trace). A trace τ over $\mathcal{E}$ is defined using the following BNF grammar:

$$\tau ::= \epsilon \mid e \mid e.\tau$$
$$e \in \mathcal{E}$$

where ϵ is the empty trace. The "." represents the chronological order, *i.e.*, if e appears before e' in a trace τ, then necessarily e happened at a previous time than e'.

We introduce the following propositional logic allowing to verify whether an event in a trace respects some conditions. The main purpose of this language is to define specific patterns of messages we are looking for within the trace, such as message having a given source or destination IP addresses or ports.

Definition 4 (Propositional Event Logic). Let f_n be a field name and v be a value, we introduce the Propositional Event Logic (PEL) as follows:

$$p, q ::= \mathrm{true} \mid \mathrm{false} \mid f_n \; op \; v \mid p \vee q \mid p \wedge q \mid \neg p$$
$$op ::= = \mid \neq \mid \leq \mid \geq \mid < \mid >$$

An event e respects a proposition p, and we say that $p(e) = \mathrm{true}$, if one of the

following conditions holds:

$$\begin{aligned}
&\text{true}(e) = \text{true}\\
&(\neg p)(e) = \neg p(e)\\
&(p \vee q)(e) = p(e) \vee q(e)\\
&(p \wedge q)(e) = p(e) \wedge q(e)\\
&(f_n \; op \; v)(e) = (e@f_n) \; op^{?} \; v
\end{aligned}$$

For instance, to know if (TCP.DestPort = 80)(e), we check if (e@TCP.DestPort) $=^{?} 80$.

3.2. Trace Slicing

This step requires meticulous attention to ensure the approach's effectiveness is maximized. It's important to decompose the trace into one or multiple sequences of slices characterized by smooth variations. The end user must have a clear understanding of their activity's nature to identify instances where sudden changes should not occur. Below, we provide some illustrative examples:

- Significant and sudden fluctuations in traffic volume are often indicative of potential Denial of Service (DoS) attacks. To detect this activity, it's appropriate to divide the traffic trace τ into successive discrete slices, denoted as $\tau_1, \cdots, \tau_n$, each representing a predefined time window, such as 10 minutes.
- The previous analysis will be more precise and efficient if we separate the traffic of different *IP* addresses. Also input traffic can be separated from output. Sudden variation in input traffic can be du to DoS attack but variation of output traffic can be generated by a malware (e.g. botnet) activity. Therefore this kind of separation allow us either to know the *IP* address in the suspicious traffic as well as the nature of the attack.
- Input and output traffic of different *IP* address can be further separated into traffics related to different IP protocols and TCP ports.
- The previous divisions can be further refined as we will show in the case study section. For instance, we can separate the traffic of different days of the weak. By doing so, we assume that traffic related to successive Monday should not present a sudden change.

The forthcoming definition introduces a slicing function designed to partition a trace, catering to diverse scenarios and requirements.

Definition 5 (Slicing). Let p be a propositional formula in PEL and τ be a trace in $\mathcal{T}$. We inductively introduce a slicing function $\mathcal{S}_p(\tau)$ as follows:

$$\begin{aligned}
&\mathcal{S}_p(\epsilon) ::= \epsilon\\
&\mathcal{S}_p(e) ::= \begin{cases} \epsilon & \text{if } p(e) = \text{false} \\ e & \text{if } p(e) = \text{true} \end{cases}\\
&\mathcal{S}_p(e.\tau) ::= \mathcal{S}_p(e).\mathcal{S}_p(\tau)
\end{aligned}$$

Let $p_1, \cdots, p_n$ denote propositions. We extend the selection function to operate on sets of sequences of propositions as follows:

$$\mathcal{S}_{\{p_1,\cdots,p_n\}}(\tau)=\{\mathcal{S}_{p_1}(\tau),\cdots,\mathcal{S}_{p_n}(\tau)\}$$
$$\mathcal{S}_{\langle p_1,\cdots,p_n\rangle}(\tau)=\langle\mathcal{S}_{p_1}(\tau),\cdots,\mathcal{S}_{p_n}(\tau)\rangle$$

If $p(i)$ is a proposition that depends on i, we use the notation $\langle p(i)\rangle_{sart,\,jmp}^{end}$ as an abbreviation of

$$\langle p(start), p(start+jmp),\cdots, p(start+n*jmp)\rangle$$

where n is the natural number such that $n*jmp\le end$ and $(n+1)*jmp$. For instance:

- $\langle p(i)\rangle_{1,2}^{8}$ is same as $\langle p(1), p(3), p(5), p(7)\rangle$, and
- $\langle(\mathrm{T}\ge 10.00.j)(\wedge\mathrm{T}\le 10.00.(j+10))\rangle_{j=0,10}^{60}$ is same as $\langle p_1,\cdots, p_6\rangle$, where:

$$p_1=(\mathrm{T}\ge 10.00.00)(\wedge\mathrm{T}\le 10.00.10)$$
$$p_2=(\mathrm{T}>10.00.10)(\wedge\mathrm{T}\le 10.00.20)$$
$$p_3=(\mathrm{T}>10.00.20)(\wedge\mathrm{T}\le 10.00.30)$$
$$p_4=(\mathrm{T}>10.00.30)(\wedge\mathrm{T}\le 10.00.40)$$
$$p_5=(\mathrm{T}>10.00.40)(\wedge\mathrm{T}\le 10.00.50)$$
$$p_6=(\mathrm{T}>10.00.50)(\wedge\mathrm{T}\le 10.00.60)$$

Example 1 (Selection). Let τ be the trace containing the traffic captured between 10:00:000 and 10:00:052 focusing on IP.Prot as shown by Table 3.

Table 3. Captured traffic.

τ	
$<10:00:000, m_1$@IP.Prot $=1>$.	$<10:00:027, m_{14}$@IP.Prot $=17>$
$<10:00:001, m_2$@IP.Prot $=6>$.	$<10:00:030, m_{15}$@IP.Prot $=17>$
$<10:00:003, m_3$@IP.Prot $=6>$.	$<10:00:031, m_{16}$@IP.Prot $=4>$.
$<10:00:007, m_4$@IP.Prot $=17>$.	$<10:00:033, m_{17}$@IP.Prot $=4>$.
$<10:00:010, m_5$@IP.Prot $=6>$.	$<10:00:035, m_{18}$@IP.Prot $=4>$.
$<10:00:011, m_6$@IP.Prot $=17>$.	$<10:00:038, m_{19}$@IP.Prot $=4>$.
$<10:00:013, m_7$@IP.Prot $=17>$.	$<10:00:040, m_{20}$@IP.Prot $=4>$.
$<10:00:015, m_8$@IP.Prot $=6>$.	$<10:00:043, m_{21}$@IP.Prot $=4>$.
$<10:00:018, m_9$@IP.Prot $=6>$.	$<10:00:045, m_{22}$@IP.Prot $=6>$.
$<10:00:020, m_{10}$@IP.Prot $=1>$.	$<10:00:048, m_{23}$@IP.Prot $=6>$.
$<10:00:0022, m_{11}$@IP.Prot $=6>$.	$<10:00:050, m_{24}$@IP.Prot $=17>$
$<10:00:023, m_{12}$@IP.Prot $=6>$.	$<10:00:051, m_{25}$@IP.Prot $=1>$.
$<10:00:024, m_{13}$@IP.Prot $=6>$.	$<10:00:052, m_{26}$@IP.Prot $=1>$

Let $\varphi=\langle(\mathrm{T}\ge 10.00.j)(\wedge\mathrm{T}\le 10.00.(j+10))\rangle_{j=0,10}^{60}$. When slicing τ using φ, we compute $S_\varphi(\tau)$, resulting in the sequence $\langle\tau_1,\cdots,\tau_6\rangle$, as illustrated in Table 4.

Table 4. Sliced captured traffic.

$\mathcal{S}_{\varphi}(\tau)$	
τ_1	$< 10:00:027, m_{14}$@IP.Prot $= 17 >$.
$< 10:00:000, m_1$@IP.Prot $= 1 >$.	$< 10:00:030, m_{15}$@IP.Prot $= 17 >$.
$< 10:00:001, m_2$@IP.Prot $= 6 >$.	τ_4
$< 10:00:003, m_3$@IP.Prot $= 6 >$.	$< 10:00:031, m_{16}$@IP.Prot $= 4 >$.
$< 10:00:007, m_4$@IP.Prot $= 17 >$.	$< 10:00:033, m_{17}$@IP.Prot $= 4 >$.
$< 10:00:010, m_5$@IP.Prot $= 6 >$.	$< 10:00:035, m_{18}$@IP.Prot $= 4 >$.
τ_2	$< 10:00:038, m_{19}$@IP.Prot $= 4 >$.
$< 10:00:011, m_6$@IP.Prot $= 17 >$.	$< 10:00:040, m_{20}$@IP.Prot $= 4 >$.
$< 10:00:013, m_7$@IP.Prot $= 17 >$.	τ_5
$< 10:00:015, m_8$@IP.Prot $= 6 >$.	$< 10:00:043, m_{21}$@IP.Prot $= 4 >$.
$< 10:00:018, m_9$@IP.Prot $= 6 >$.	$< 10:00:045, m_{22}$@IP.Prot $= 6 >$.
$< 10:00:020, m_{10}$@IP.Prot $= 1 >$.	$< 10:00:048, m_{23}$@IP.Prot $= 6 >$.
τ_3	$< 10:00:050, m_{24}$@IP.Prot $= 17 >$.
$< 10:00:0022, m_{11}$@IP.Prot $= 6 >$.	τ_6
$< 10:00:023, m_{12}$@IP.Prot $= 6 >$.	$< 10:00:051, m_{25}$@IP.Prot $= 1 >$.
$< 10:00:024, m_{13}$@IP.Prot $= 6 >$.	$< 10:00:052, m_{26}$@IP.Prot $= 1 >$

3.3. Feature Measuring

Each slice, derived from the preceding step, undergoes transformation into an element in $\mathbb{R}^n$ $(n \geq 1)$ by quantifying certain characteristics through a predefined function F. For simplicity, we concentrate on a class of functions F that produce distributions by tallying events adhering to specified conditions, as delineated in the following definition:

Definition 6 (Feature Measuring Function). Let q be a propositional formula in PEL and τ be a trace in $\mathcal{T}$. We introduce a slicing function $\mathcal{F}_q(\tau)$ inductively as follows:

$$\mathcal{F}_q(\epsilon) ::= 0$$

$$\mathcal{F}_q(e) ::= \begin{cases} 0 & \text{if } q(e) = \text{false} \\ 1 & \text{if } q(e) = \text{true} \end{cases}$$

$$\mathcal{F}_q(e.\tau) ::= \mathcal{F}_q(e) + \mathcal{F}_q(\tau)$$

Broadly speaking, $\mathcal{F}_q(\tau)$ returns the number of packets in τ that satisfy the property q.

We also extend the selection function to operate on both a sequence of propositions $q_1, \cdots, q_n$ and a set of traces as follows:

$$\mathcal{F}_{\langle q_1, \cdots, q_n \rangle}(\tau) = \left\langle \mathcal{F}_{q_1}(\tau), \cdots, \mathcal{F}_{q_n}(\tau) \right\rangle$$

$$\mathcal{F}_q\left(\langle \tau_1, \cdots, \tau_n \rangle\right) = \left\langle \mathcal{F}_q(\tau_1) \cdots \mathcal{F}_q(\tau_n) \right\rangle$$

$$\mathcal{F}_q\left(\{\tau_1,\cdots,\tau_n\}\right)=\left\{\mathcal{F}_q\left(\tau_1\right)\cdots\mathcal{F}_q\left(\tau_n\right)\right\}$$

Example 2. Let's examine the trace provided in Example 1. Let $\psi=\langle q_1,q_2,q_3,q_4\rangle$ such that $q_1=(\text{IP.Prot}=1)$, $q_2=(\text{IP.Prot}=6)$, $q_3=(\text{IP.Prot}=17)$ and $q_4=(\text{IP.Prot}\neq 1)\wedge(\text{IP.Prot}\neq 6)\wedge(\text{IP.Prot}\neq 17)$, then when applying the function $\mathcal{F}_\psi$ to the slices $\tau_1,\cdots,\tau_6$ as depicted in Table 4, the resulting outcomes are as illustrated in Table 5.

Table 5. Quantification of slices using $\mathcal{F}$.

$\mathcal{S}_\varphi(\tau)$	$\mathcal{F}_\psi(\tau_i)$
τ_1	$\langle 1,3,1,0\rangle$
τ_2	$\langle 1,2,2,0\rangle$
τ_3	$\langle 0,3,2,0\rangle$
τ_4	$\langle 0,0,0,5\rangle$
τ_5	$\langle 0,2,1,1\rangle$
τ_6	$\langle 2,0,0,0\rangle$

For instance, $\mathcal{F}_\psi(\tau_1)=\langle 1,3,1,0\rangle$ indicates that in slice τ_1, there is 1 packet with IP.Prot = 1, 3 packets with IP.Prot = 6, 1 packet with IP.Prot = 17, and 0 packets with other IP.Prot values.

The distributions of these slices serve as inputs to algorithms like KL-Divergence, enabling the measurement of traffic divergence across distinct slices. However, in cases where certain events are absent during observation, their frequencies register as zero, posing a challenge for computing KL-Divergence and potentially leading to division by zero errors. To address this issue, we must either explore alternative divergence techniques or slightly adjust the data distribution through methods such as smoothing. The following definition illustrates one of the well-known smoothing techniques.

Definition 7 (Laplace Smoothing). Let $v=\langle v_1,\cdots,v_n\rangle$ be a sequence of real numbers. We denote by $\pi^k(v)$ the *k*-Laplace Smoothing Distribution (*k*-LSD) of a trace and we define it as follows:

$$\pi^k(v)=\left\langle\frac{k+v_1}{k+\sum_{i=1}^n v_i},\cdots,\frac{k+v_n}{k+\sum_{i=1}^n v_i}\right\rangle$$

We augment the function $\mathcal{F}$ with Laplace smoothing as follows:

Definition 8 (Feature Measuring Function with Smoothing). We denote by $\hat{\mathcal{F}}_p$, the smoothed version of $\mathcal{F}_p$ achieved through the application of the smoothing function π^1. More formally:

$$\hat{\mathcal{F}}_p=\pi^1\circ\mathcal{F}_p$$

Example 3. By applying π^1 to column 2 of Table 5, we obtain $\hat{\mathcal{F}}_\psi(\tau_i)$ as shown by column 3 of Table 6.

Table 6. Quantification and smoothing of slices using $\hat{\mathcal{F}}$.

$\mathcal{S}_\varphi(\tau)$	$\mathcal{F}_\psi(\tau_i)$	$\hat{\mathcal{F}}_\psi(\tau_i)$
τ_1	$\langle 1,3,1,0\rangle$	$\left\langle \frac{1+1}{1+5},\frac{1+3}{1+5},\frac{1+1}{1+5},\frac{1+0}{1+5}\right\rangle = \left\langle \frac{1}{3},\frac{2}{3},\frac{1}{3},\frac{1}{6}\right\rangle$
τ_2	$\langle 1,2,2,0\rangle$	$\left\langle \frac{1+1}{1+5},\frac{1+2}{1+5},\frac{1+2}{1+5},\frac{1+0}{1+5}\right\rangle = \left\langle \frac{1}{3},\frac{1}{2},\frac{1}{2},\frac{1}{6}\right\rangle$
τ_3	$\langle 0,3,2,0\rangle$	$\left\langle \frac{1+0}{1+5},\frac{1+3}{1+5},\frac{1+2}{1+5},\frac{1+0}{1+5}\right\rangle = \left\langle \frac{1}{6},\frac{2}{6},\frac{1}{2},\frac{1}{6}\right\rangle$
τ_4	$\langle 0,0,0,5\rangle$	$\left\langle \frac{1+0}{1+5},\frac{1+0}{1+5},\frac{1+0}{1+5},\frac{1+5}{1+5}\right\rangle = \left\langle \frac{1}{6},\frac{1}{6},\frac{1}{6},1\right\rangle$
τ_5	$\langle 0,2,1,1\rangle$	$\left\langle \frac{1+0}{1+5},\frac{1+2}{1+5},\frac{1+1}{1+5},\frac{1+1}{1+5}\right\rangle = \left\langle \frac{1}{6},\frac{1}{2},\frac{1}{3},\frac{1}{3}\right\rangle$
τ_6	$\langle 2,0,0,0\rangle$	$\left\langle \frac{1+2}{1+5},\frac{1+0}{1+5},\frac{1+0}{1+5},\frac{1+0}{1+5}\right\rangle = \left\langle \frac{1}{2},\frac{1}{6},\frac{1}{6},\frac{1}{6}\right\rangle$

When detecting suspicious activities within traffic data, it can be advantageous to prioritize specific positions within the values returned by $\hat{\mathcal{F}}$ in $\mathbb{R}^n$. For instance, if $\hat{\mathcal{F}}$ yields $(v_1,\cdots,v_n)$ where each v_i represents traffic originating from a specific country, these values might be weighted according to the respective country's reputation in cyberattacks, assigning greater weight to countries with negative reputations. Presently, there's a lack of a systematic approach to guide end users in determining these weight values. However, we believe that fine-tuning these weights based on intuition could enhance detection capabilities.

The subsequent definition formalizes the concept of weights.

Definition 9 (Weighting Function ω). We denote by ω a weighting function that accepts weights in $\left(\mathbb{R}^+\right)^n$, a tuple in $\mathbb{R}^n$, and returns a probability distribution, *i.e.*: $\omega:\left(\mathbb{R}^+\right)^n\times\mathbb{R}^n\to[0,1]^n$.

Let $V_1,\cdots,V_m$ be in $\mathbb{R}^n$). We extend ω to a set $\{V_1,\cdots,V_m\}$ and a sequences $\langle V_1,\cdots,V_m\rangle$ of tuples as follows:

$$\omega\left(\{V_1,\cdots,V_m\}\right)=\{\omega(V_1),\cdots,\omega(V_m)\}$$
$$\omega\left(\langle V_1,\cdots,V_m\rangle\right)=\langle\omega(V_1),\cdots,\omega(V_m)\rangle$$

The following definition provides an example of ω.

Definition 10. *(*Product Scalar Weighting Function) We define the scalar product weighting function, abbreviated as spw, as follows:

$$\text{spw}:\left(\mathbb{R}^+\right)^n\times\mathbb{R}^n\to[0,1]^n$$
$$\text{spw}(\text{w},v)=\left\langle\frac{\text{w}_1\times v_1}{\text{w}.v},\cdots,\frac{\text{w}_n\times v_n}{\text{w}.v}\right\rangle$$

where $w.u$ is the scalar product of the tow vectors w and u, *i.e.*:

$$\text{w}.u=\sum_{i=1}^{n}\text{w}_i\times v_i$$

We extend the function $\hat{\mathcal{F}}$ by incorporating a weighting function as follows:

Definition 11 (Feature Measuring Function with Smoothing and Weighting). Let ω a weighting function. In the sequel, we denote by $\hat{\mathcal{F}}_{p,\omega}$, the weighted version of $\hat{\mathcal{F}}_p$ using the weighting function ω. More precisely:

$$\hat{\mathcal{F}}_{p,\omega} = \omega \circ \hat{\mathcal{F}}_p$$

and for any trace τ and a weight vector w, we have:

$$\hat{\mathcal{F}}_{p,\omega}(\mathrm{w},\tau) = \omega\left(\mathrm{w}, \hat{\mathcal{F}}_p(\tau)\right)$$

Example 4. Let's examine the trace provided in Example 3. Suppose we aim to prioritize packets containing ports not in 1, 6, 17. As an example, we apply the weighting function $\omega = \mathrm{spw}$ with weights $\mathrm{w} = \langle 0.2, 0.2, 0.2, 0.4\rangle$. The results are illustrated in Table 7.

Table 7. Slice distribution.

$\mathcal{S}_\varphi(\tau)$	$\mathcal{F}_\psi(\tau_i)$	$\hat{\mathcal{F}}_\psi(\tau_i)$	$\hat{\mathcal{F}}_{\psi,\omega}(w,\tau_i)$
τ_1	$\langle 1,3,1,0\rangle$	$\left\langle \frac{1}{3},\frac{2}{3},\frac{1}{3},\frac{1}{6}\right\rangle$	$\langle 0.2,0.4,0.2,0.2\rangle$
τ_2	$\langle 1,2,2,0\rangle$	$\left\langle \frac{1}{3},\frac{1}{2},\frac{1}{2},\frac{1}{6}\right\rangle$	$\langle 0.2,0.3,0.3,0.2\rangle$
τ_3	$\langle 0,3,2,0\rangle$	$\left\langle \frac{1}{6},\frac{2}{6},\frac{1}{2},\frac{1}{6}\right\rangle$	$\langle 0.1,0.4,0.3,0.2\rangle$
τ_4	$\langle 0,0,0,5\rangle$	$\left\langle \frac{1}{6},\frac{1}{6},\frac{1}{6},1\right\rangle$	$\langle 0.067,0.067,0.67,0.8\rangle$
τ_5	$\langle 0,2,1,1\rangle$	$\left\langle \frac{1}{6},\frac{1}{2},\frac{1}{3},\frac{1}{3}\right\rangle$	$\langle 0.1,0.3,0.2,0.4\rangle$
τ_6	$\langle 2,0,0,0\rangle$	$\left\langle \frac{1}{2},\frac{1}{6},\frac{1}{6},\frac{1}{6}\right\rangle$	$\langle 0.429,0.143,0.143,0.286\rangle$

3.4. Divergence Measuring

After abstracting and transforming the traffic into smoothed distributions, the next step involves measuring the divergence between adjacent slices within each sequence. To achieve this, we employ a divergence function such as the KL-Divergence.

Definition 12 (Divergence Function). A divergence measuring function, denoted by Δ, can be any function with the following signature:

$\Delta : [0,1]^n \times [0,1]^n \to \mathbb{R}$.

Examples of divergence measuring functions are given in Table 8.

Table 8. Examples of divergence functions.

Divergence		
Δ	::=	KL-Divergence [23] | Cosine [24] |TF-IDF [25] |…

Notice that, since the KL-Divergence, usually denoted by D_{KL}, between two distributions $P=(p_1,\cdots,p_n)$ and $Q=(q_1,\cdots,q_n)$ is not commutative (*i.e.*, $D_{KL}(P\|Q)\neq D_{KL}(Q\|P)$ as shown by Equations (1) and (2)), we can consider $\Delta(P,Q)=KL(P,Q)=D_{KL}(P\|Q)+D_{KL}(Q\|P)$ as the divergence value.

$$D_{KL}(P\|Q)=\sum_{i=1}^{n} p_i\times\log_2\left(\frac{p_i}{q_i}\right) \tag{1}$$

$$D_{KL}(Q\|P)=\sum_{i=1}^{n} q_i\times\log_2\left(\frac{q_i}{p_i}\right) \tag{2}$$

Example 5. We apply the KL-Divergence to the trace of Example 4. The result is shown by Table 9.

Table 9. Slice distribution.

$\mathcal{S}_\varphi(\tau)$	$\mathcal{F}_\psi(\tau_i)$	$u_i=\hat{\mathcal{F}}_{\psi,\omega}(w,\tau_i)$	$PKL(u_i,u_{i+1})$	$PKL(u_{i+1},u_i)$	$KL(u_i,u_{i+1})$
τ_1	$\langle 1,3,1,0\rangle$	$u_1=\langle 0.2,0.4,0.2,0.2\rangle$	0.049	0.51	**1**
τ_2	$\langle 1,2,2,0\rangle$	$u_2=\langle 0.2,0.3,0.3,0.2\rangle$	0.0755	0.066	**0.142**
τ_3	$\langle 0,3,2,0\rangle$	$u_3=\langle 0.1,0.4,0.3,0.2\rangle$	1.3435	1.2440	**2.597**
τ_4	$\langle 0,0,0,5\rangle$	$u_4=\langle 0.067,0.067,0.67,0.8\rangle$	0.5107	0.6265	**1.137**
τ_5	$\langle 0,2,1,1\rangle$	$u_5=\langle 0.1,0.3,0.2,0.4\rangle$	0.4024	0.5388	**0.941**
τ_6	$\langle 2,0,0,0\rangle$	$u_6=\langle 0.429,0.143,0.143,0.286\rangle$	-	-	-

3.5. Divergence Clustering

After quantifying the divergence between successive slices of traces, the next step is to ascertain if significant abrupt changes have occurred. To accomplish this, we estimate the number of clusters generated by the divergence values. If this count exceeds one, we infer that the trace contains suspicious traffic.

Definition 13 (Clustering). Let $\mathcal{C}_n:2^{\mathbb{R}}\rightarrow$ true, false be a clustering algorithm that estimates the optimal number of clusters N associated with a dataset in $2^{\mathbb{R}}$. It returns true if the number $N\geq n$, indicating that the threshold for suspicious activity has been surpassed, and false otherwise.

We are particularly interested in $\mathcal{C}_2$. When $\mathcal{C}_2$ returns true, it indicates that the traffic is considered suspicious. Examples of the $\mathcal{C}_2$ function are provided in Table 10.

Table 10. Examples of clustering functions.

		Clustering
$\mathcal{C}_2$	::=	HC [26] | KM [27] | EM [28] |…

Example 6. Let's apply the K-means algorithm with the Elbow Method to compute $\mathcal{C}_2$ on the trace from the previous example, as illustrated in Table 11.

Table 11. K-means results.

Cluster 1	Cluster 2
0.100, 0.142, 0.941, 1.137	2.587

3.6. Suspicious Traffic Detection

Now, we have all the necessary ingredients to define a suspicious traffic.

Definition 14 (Suspicious Traffic)

- Let τ be a trace.
- Let $\varphi = \langle p_1, \cdots, p_n \rangle$ be a n sequences of propositions.
- Let $\psi = \langle q_1, \cdots, q_m \rangle$ be a m sequences of propositions.
- Let $\mathrm{w} = \langle \mathrm{w}_1, \cdots, \mathrm{w}_n \rangle$ be a weight vector in $\mathbb{R}^n$.
- Let $\Delta : [0,1]^n \times [0,1]^n \to \mathbb{R}$ be a divergence measuring function such as KL-Divergence.
- Let $\mathcal{C}_2 : 2^{\mathbb{R}} \to \{\text{true}, \text{false}\}$ be a clustering algorithm that estimated the best number of clusters N related to the set of data in $2^{\mathbb{R}}$ and returns true if $N \geq 2$, false otherwise.

We define $Suspicious^{\sigma,\omega}_{\Delta,\mathcal{C}_2}(\tau, \varphi, \psi, \mathrm{w})$, a generic function designed to detect suspicious traffic within an analyzed trace τ, as follows:

$$Suspicious^{\omega}_{\Delta,\mathcal{C}_2}(\tau, \varphi, \psi, \mathrm{w}) = \mathcal{C}_2\left(\Delta\left(\hat{\mathcal{F}}_{\psi,\omega}\left(\mathrm{w}, \mathcal{S}_{\varphi}(\tau)\right)\right)\right)$$

The *Suspicious* function integrates various analyses, conducted in the sequence depicted in Figure 3, and returns true if the traffic is deemed suspicious, and false otherwise. It requires three functions, ω, Δ, and $\mathcal{C}_2$, as well as four parameters: τ, w, φ, and ψ.

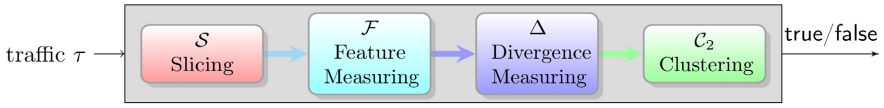

Figure 3. Steps involved in detecting suspicious traffic.

Example 7. Let's apply the *Suspicious* function to the trace provided in Example 1 to ascertain if there exists a sudden change. Based on the results shown in Table 11, where $\mathcal{C}_2$ generates more than one cluster, we deduce that:

$$Suspicious^{\omega}_{\Delta,\mathcal{C}_2}(\tau, \varphi, \psi, \mathrm{w}) = \text{true}$$

The suspicious traffic is triggered on slice τ_3.

4. Case Study

In this section, we present three cases of detecting suspicious activities using two distinct datasets comprising real traffic. The first case involves detecting suspicious activities based on daily patterns in the dataset from [29]. The second and third cases utilize the UNSW-NB15 dataset [30] to detect suspicious traffic by analyzing TCP and DNS traffic, respectively.

4.1. Detecting Suspicious Activities Based on Days of the Week

An example of an interesting dataset with a real traffic is available at [29]. It contains 21,000 rows and covers the traffics related to 10 workstations with local IP addresses over a period of three months. Half of these local IP addresses were hacked at some point during this period and became members of different botnets and generated abnormal traffic.

A screenshot of a part of the dataset is shown in Figure 4, where:

- **date**: yyyy-mm-dd (from 2006-07-01 through 2006-09-30);
- **l_ipn**: local IP address (coded as an integer from 0-9);
- **r_asn**: remote ASN (an integer which identifies the remote Autonomous System Network);
- **f**: flows (number of connections during the corresponding day).

date	l_ipn	r_asn	f
2006-07-01	0	701	1
2006-07-01	0	714	1
2006-07-01	0	1239	1
2006-07-01	0	1680	1
2006-07-01	0	2514	1
2006-07-01	0	3320	1
2006-07-01	0	3561	13
2006-07-01	0	4134	3
2006-07-01	0	5617	2
2006-07-01	0	6478	1
2006-07-01	0	6713	1
2006-07-01	0	7132	1
2006-07-01	0	9105	1
2006-07-01	0	10738	1
2006-07-01	0	10994	1
2006-07-01	0	12334	1
2006-07-01	0	12524	1
2006-07-01	0	12542	1
2006-07-01	0	13343	1
2006-07-01	0	13446	1
2006-07-01	0	13462	20

Figure 4. A part of the dataset provided by [29].

We try to detect the infected computer based on the following assumption for each workstation of the network: the nature of traffic may vary across different days of the week. For instance, weekend traffic could differ significantly from

that of weekdays. However, when we consider a specific day, such as Monday, there is no compelling reason for it to undergo substantial changes from one week to another. This implies that Monday's traffic should remain relatively consistent across all weeks. A similar pattern is expected for other days of the week, such as Tuesday, Wednesday, and so forth.

Based on the assumption, we proceed as follows: we segregate the traffic associated with each workstation and day of the week into distinct files. With ten workstations and seven days a week, this results in a total of 70 files. Subsequently, each of these files undergoes analysis to identify any abrupt changes.

Here are the values of the parameters required used within the function *GSuspicious* allowing to detect suspect traffic.

- τ (trace): the dataset available at [29].
- The trace is scattered into various slices, each exclusively comprising traffic linked to a specific IP address and a designated day of the week. To illustrate, for IP address 0, distinct slices are allocated for Mondays, Tuesdays, and so forth. Similar slices are build for IP addresses 1 to 9. By doing this division, we are implicitly making the assumption that for any IP address, the traffic of different Mondays should be quite similar and this should be the same for the other days of the week. More formally, the slicing will be based on the following set of propositions:

$$\varphi = \bigcup_{\substack{1\le i\le 9\\ 0\le j\le 6}} \left\{ p_{i,j} \right\}$$

where

$$p_{i,j} = \left\langle \left(IP = i \right) \wedge \left(date.dd = j \right) \right\rangle_{j,j+7}^{N}$$

and $N = 21000$ represents the number of events in the trace.

- Let $\psi = \left\langle \left(q_1 = v_1 \right), \cdots, \left(q_n = v_n \right) \right\rangle$ where $v_1, \cdots, v_n$ are the different values that appears in the column **r_asn** presented in ascending order.
- $\mathrm{w} = \left(\mathrm{w}_1, \cdots, \mathrm{w}_n \right) = \left(1, \cdots, 1 \right)$. This captures the fact that each element of the partition has the same weight.
- Δ is the KL-Divergence.
- Let $\mathcal{C}_2$ is the composition of K-means and Elbow Methods. The K-means do the clustering and the Elbow Methods estimate the best number of clusters.

All these fixed parameters will be the input of our *Suspicious* function to conclude whether the traffic is suspicious or not. This function proceed as follows:

- After applying the function $\mathcal{S}_\varphi$ to the dataset, we obtain a separate file for each IP address and each day of the week. For instance, for IP address 0 and Monday, we generate a file that will be analyzed independently for suspicious traffic. This file aggregates traffic not only from a single Monday but from multiple Mondays, and our objective is to detect any sudden changes in the distribution of traffic from one Monday to another. We repeat this process for the other days of the week and for the remaining IP addresses.

- The traffic from each IP address and each day of the week undergoes transformation through the function $\mathcal{F}_{\psi}$, resulting in a point in $\mathbb{R}^n$, where each dimension represents the number of connections related to every **r_asn**, and n is the total number of **r_asn**.
- Thanks to the function Δ, we quantify the divergence between every two successive Mondays for each IP address, and we repeat this process for the other days of the week as well.
- Using the function $\mathcal{C}_2$ (composition of the K-means and the Elbow Method), we estimate the number of clusters generated by the previous steps.
- If we observe two or more clusters for any analyzed sequence, we infer that the traffic is suspicious.

Below, we present the results obtained from the Elbow Method corresponding to the different days and IP addresses.

1) **Monday**: Based on the analysis of Monday traffic depicted in Figure 5, we identify five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8).

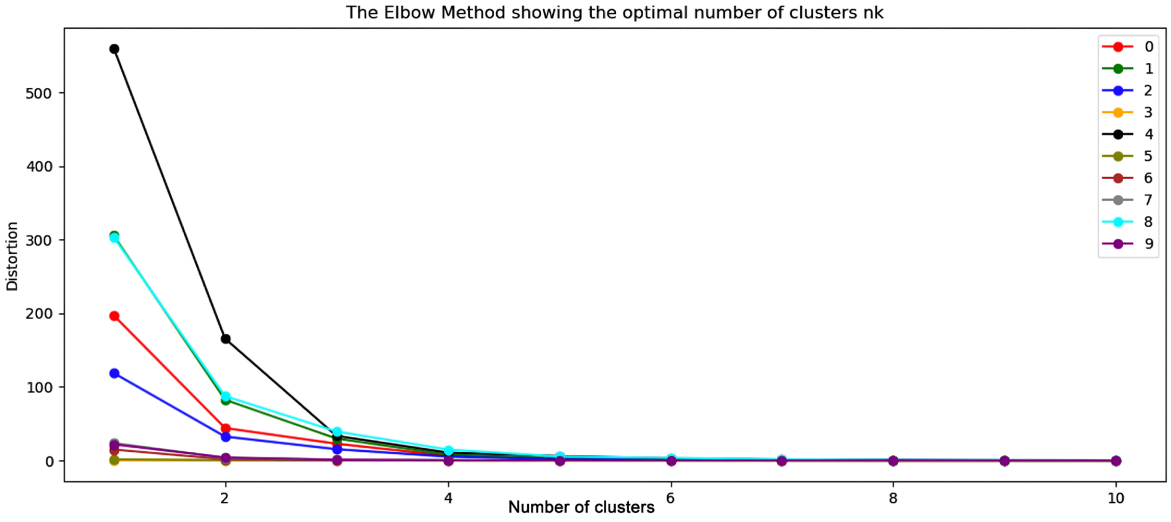

Figure 5. Elbow results for every Monday.

2) **Tuesday**: Based on the analysis of Tuesday traffic depicted in Figure 6, we identify five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8).

3) **Wednesday**: Based on the analysis of Wednesday traffic shown in Figure 7, we observe five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8).

4) **Thursday**: According to the analysis depicted in Figure 8, we identify five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8) on Thursday.

5) **Friday**: Based on the analysis presented in Figure 9, we observe five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8) on Friday.

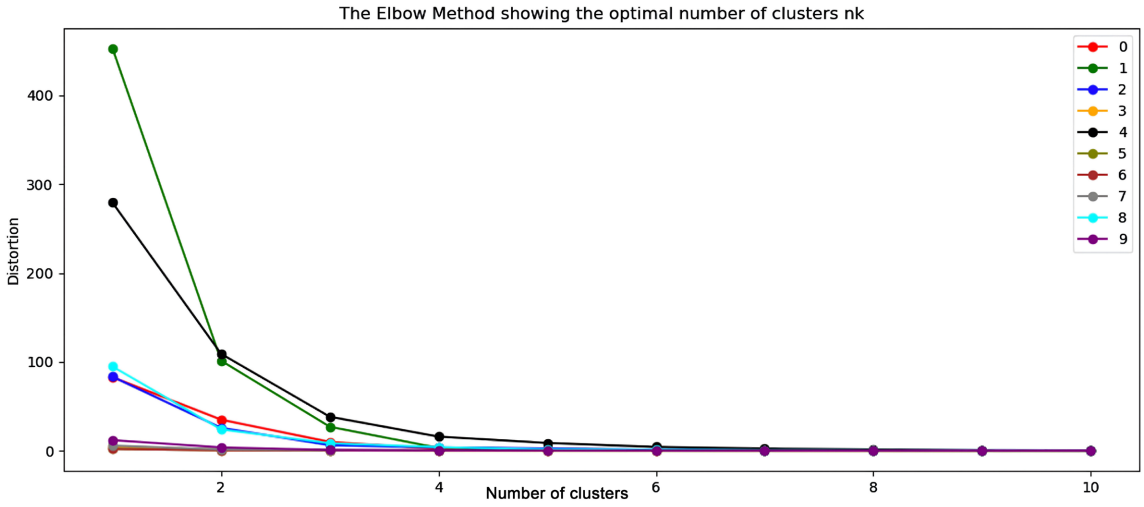

Figure 6. Elbow results for every Tuesday.

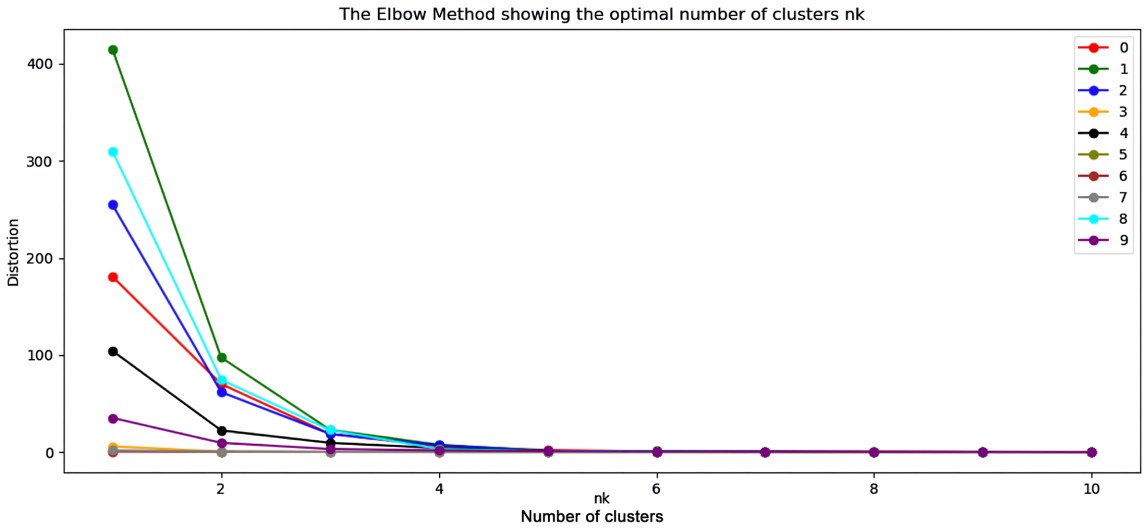

Figure 7. Elbow results for every Wednesday.

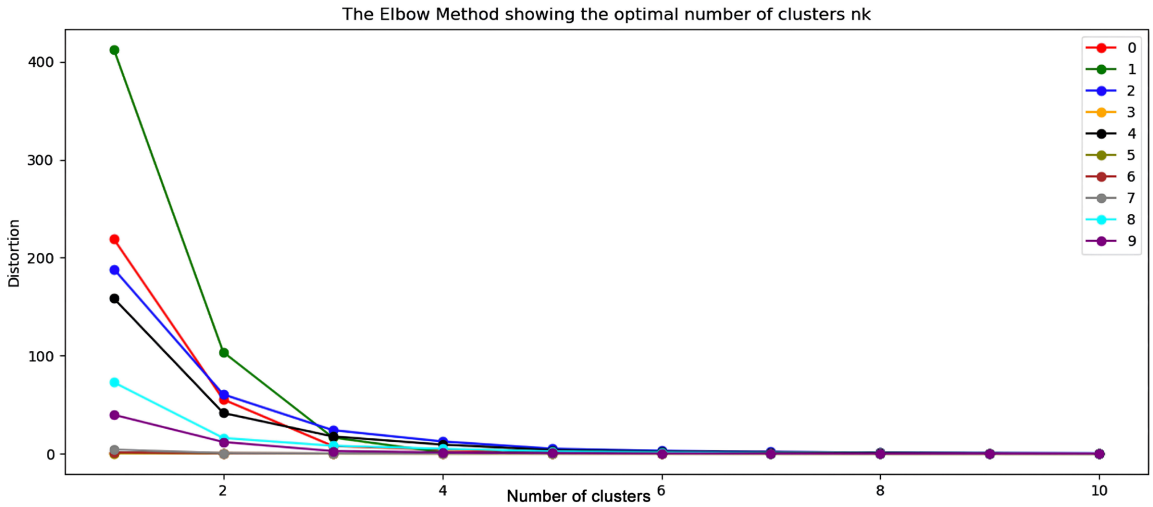

Figure 8. Elbow results for every Thursday.

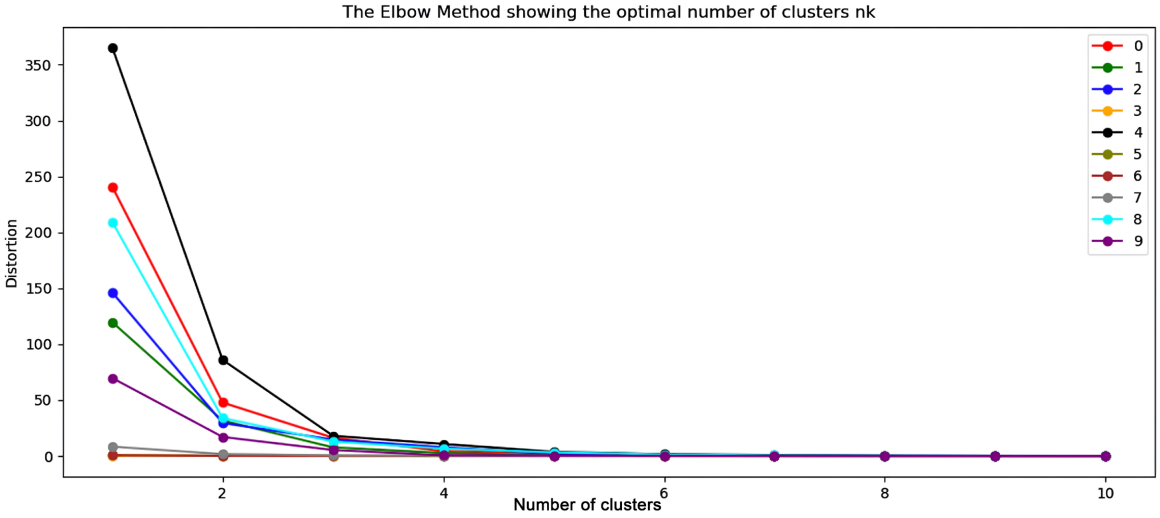

Figure 9. Elbow results for every Friday.

6) **Saturday**: According to the analysis shown in Figure 10, we can identify five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8) on Saturday.

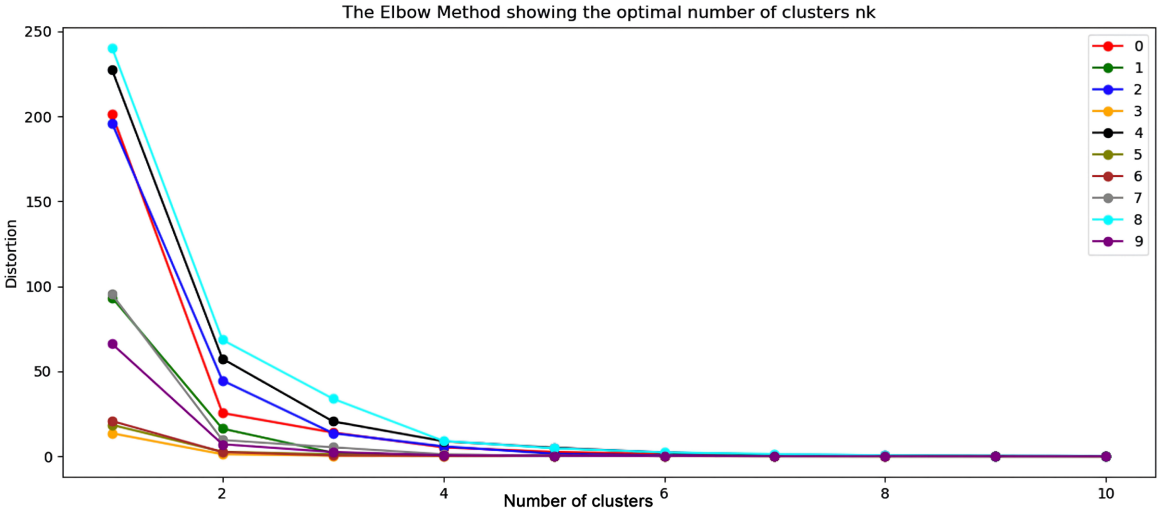

Figure 10. Elbow results for every Saturday.

7) **Sunday**: Based on the analysis presented in Figure 11, we observed five non-suspicious machines (3, 5, 6, 7, and 9) and five suspicious machines (0, 1, 2, 4, and 8) on Sunday.

Here are the conclusions extracted from Figures 5-11:

- There are five clear elbows showing that the number of clusters related to the traffics of the machines l_ipn values 0, 1, 2, 4, and 8 is greater than one and then they are the origins of the suspicious traffics shown in Table 12.
- There are five machines l_ipn values 3, 5, 6, 7, and 9 with no elbow, meaning that the number of their clusters is one, then they are not associated with any suspicious traffic showed in Table 12.

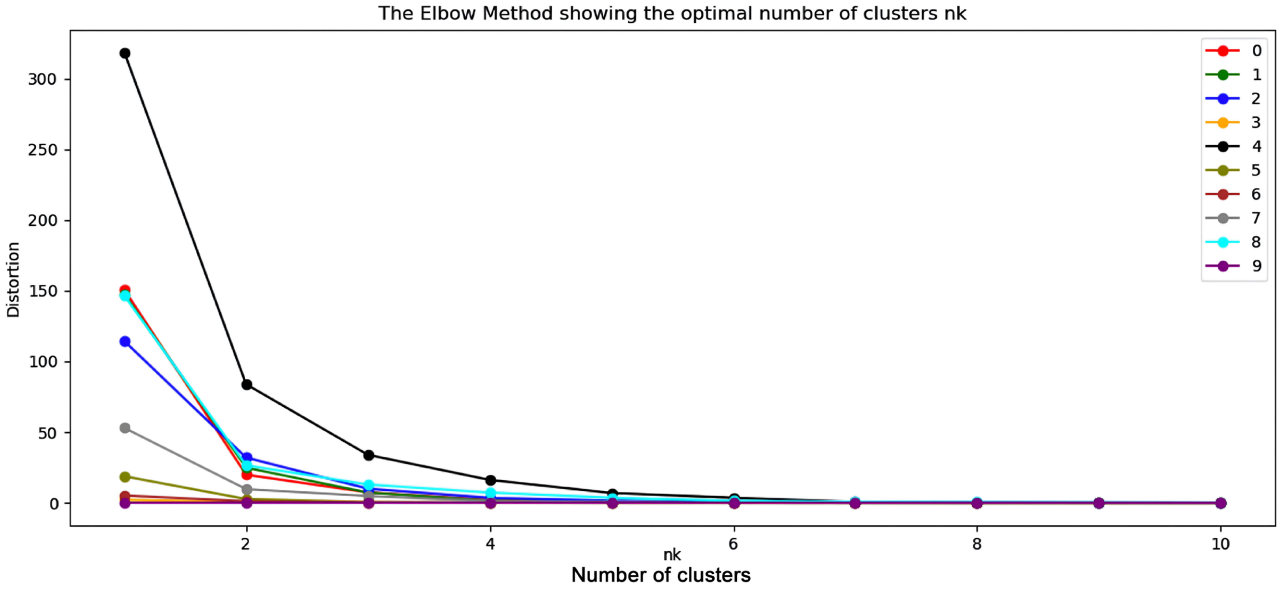

Figure 11. Elbow results for every Sunday.

Table 12. Detecting suspicious traffic based on days of the week.

	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday	Sunday
Unsuspicious	3, 5, 6, 7, 9	3, 5, 6, 7, 9	3, 5, 6, 7, 9	3, 5, 6, 7, 9	3, 5, 6, 7, 9	3, 5, 6.7, 9	3, 5, 6, 7, 9
Suspicious	0, 1, 2, 4, 8	0, 1, 2, 4, 8	0, 1, 2, 4, 8	0, 1, 2, 4, 8	0, 1, 2, 4, 8	0, 1, 2, 4, 8	0, 1, 2, 4, 8
Total Suspicious				0, 1, 2, 4, 8			

- The confusion matrix for our approach:

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative (TN)	False Negative (FN)
	Positive	False Positive (FP)	True Positive (TP)

Our approach predicted that 5/10 local IPs are botnets. Actually, only 5/10 local IPs are real botnets. Therefore:

		Predicted		
		Negative	**Positive**	Total
Actual	**Negative**	5	0	5
	Positive	0	5	5
	Total	5	5	10

It follows that:

- $\text{Accuracy} = \dfrac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} = \dfrac{10}{10} = 100\%$.

- Precision $= \frac{TP}{TP + FP} = \frac{5}{5} = 100\%$.
- Recall $= \frac{TP}{TP + FN} = \frac{5}{5} = 100\%$.
- False Negative (FN) = 0%.
- False Positive (FP) = 0%.
- Performance: Our code was executed on a Ubuntu virtual machine with a 2.3 GHz Intel Core i9 processor, equipped with 2 cores and 4GB of RAM. The total execution time to process the entire dataset, consisting of 21,000 rows covering 10 workstations over a three-month period, was approximately 51.7 seconds.

4.2. Detecting Suspicious Activities Based on DNS and HTTP Traffic

The UNSW-NB15 dataset [30] was generated using the IXIA PerfectStorm tool. It encompasses nine categories of modern attack types and incorporates realistic behaviors of normal traffic. Comprising 49 features across various categories, some of them are illustrated in Figure 12. Utilized as an attack tool, IXIA dispatches both benign and malicious traffic to different network nodes. A segment of certain fields from this traffic is demonstrated in Table 13.

No.	Name	Type	Description
1	srcip	nominal	Source IP address
2	sport	integer	Source port number
3	dstip	nominal	Destination IP address
4	dsport	integer	Destination port number
5	proto	nominal	Transaction protocol
7	dur	Float	Record total duration
8	sbytes	Integer	Source to destination transaction bytes
9	dbytes	Integer	Destination to source transaction bytes
10	sttl	Integer	Source to destination time to live value
11	dttl	Integer	Destination to source time to live value
12	sloss	Integer	Source packets retransmitted or dropped
13	dloss	Integer	Destination packets retransmitted or dropped
14	service	nominal	http, ftp, smtp, ssh, dns, ftp-data ,irc and (-) if not much used service

Figure 12. UNSW-NB15: example of features.

The network contains three sub networks as shown by Figure 13.

1) Sub network1 (server1): contains nodes with source IP addresses from 59.166.0.0 to 59.166.0.9.

2) Sub network2 (Server2): contains nodes with source IP addresses from 175.45.176.0 to 175.45.176.3.

Table 13. UNSW-NB15 samples.

srcip	sport	dstip	dsport	proto	state	dur	sbytes	dbytes	sttl	dttl	sloss	dloss	service	Sload	Dload	Spkts	Dpkts	swin	dwin	stcpb	dtcpb
59.166.0.6	49434	149.171.126.6	80	tcp	FIN	1.366053	1684	10168	31	29	3	5	http	9159.234375	56243.79297	14	18	255	255	1302136523	3540341804
59.166.0.1	32219	149.171.126.0	53	udp	CON	0.001028	146	178	31	29	0	0	dns	568093.375	692607	2	2	0	0	0	0
59.166.0.1	53228	149.171.126.4	53	udp	CON	0.001016	130	162	31	29	0	0	dns	511811	637795.25	2	2	0	0	0	0
59.166.0.3	2803	149.171.126.1	33687	tcp	FIN	0.040832	2750	25564	31	29	7	15	-	526645.75	4894788	44	44	255	255	2380039384	2398297455
59.166.0.6	50368	149.171.126.5	27436	tcp	FIN	0.021008	2542	22128	31	29	7	14	-	944021.3125	8226199.5	40	42	255	255	114650090	2264499374
59.166.0.0	23677	149.171.126.8	6881	tcp	FIN	9.887264	36762	1641360	31	29	55	583	-	29700.42969	1327021.125	660	1278	255	255	2642499349	513586157
59.166.0.0	44972	149.171.126.5	45817	tcp	FIN	0.022846	3984	2560	31	29	7	7	-	1317692.375	851615.125	18	20	255	255	414654029	2563301267
59.166.0.5	55280	149.171.126.1	55325	tcp	FIN	0.05809	2646	25564	31	29	7	15	-	355723.875	3440592	42	44	255	255	2142135751	2156847595
59.166.0.6	53913	149.171.126.3	52000	tcp	FIN	0.030868	2750	25564	31	29	7	15	-	696643.8125	6474796	44	44	255	255	114653315	127571083
59.166.0.0	60479	149.171.126.8	6881	tcp	FIN	10.374004	35516	1551472	31	29	52	551	-	27346.04688	1195440.375	638	1208	255	255	497113828	2659548381
59.166.0.5	2992	149.171.126.5	22	tcp	FIN	5.182327	26664	49682	31	29	78	103	ssh	41042.56641	76479.92969	342	356	255	255	3439197834	1289891225
59.166.0.0	6697	149.171.126.5	111	udp	CON	0.004779	568	320	31	29	0	0	-	713119.9375	401757.7188	4	4	0	0	0	0
59.166.0.1	43183	149.171.126.2	32912	tcp	FIN	0.006673	424	8824	31	29	1	4	ftp-data	444777.4688	9697588	8	12	255	255	2204241814	63057888
59.166.0.0	49549	149.171.126.3	80	tcp	FIN	1.92579	1684	10168	31	29	3	5	http	6497.074219	39896.35547	14	18	255	255	3507521501	1414139916
59.166.0.0	8107	149.171.126.5	2618	udp	CON	0.001685	520	304	31	29	0	0	-	1851632	1082492.625	4	4	0	0	0	0
59.166.0.1	4482	149.171.126.2	11680	tcp	FIN	0.004005	320	1938	31	29	1	2	ftp-data	533333.3125	3387765.25	6	8	255	255	2204240989	59453931
59.166.0.3	1305	149.171.126.5	53716	tcp	FIN	0.022919	2646	22128	31	29	7	14	-	901610.0625	7540294.5	42	42	255	255	2380036293	234923632
59.166.0.8	59043	149.171.126.7	1584	tcp	FIN	0.01934	2542	22128	31	29	7	14	-	1025439.563	8935678	40	42	255	255	2022133228	4171965882
59.166.0.9	64393	149.171.126.5	80	tcp	FIN	1.178003	1684	10168	31	29	3	5	http	10621.36523	65222.24609	14	18	255	255	3927523623	1808904086
59.166.0.1	49594	149.171.126.2	21	tcp	FIN	0.310721	2934	3742	31	29	11	15	ftp	74098.625	94567.14844	52	54	255	255	2204239569	2205191305
59.166.0.1	6180	149.171.126.2	44685	tcp	FIN	0.179295	8928	320	31	29	4	1	ftp-data	369937.8125	11913.32715	14	6	255	255	2204242698	2217236893
59.166.0.2	26238	149.171.126.5	24338	tcp	FIN	0.030853	2646	25564	31	29	7	15	-	669756.625	6477944	42	44	255	255	2082136174	2094967420
59.166.0.8	3622	149.171.126.4	53	udp	CON	0.001037	146	178	31	29	0	0	dns	563163	686595.9375	2	2	0	0	0	0

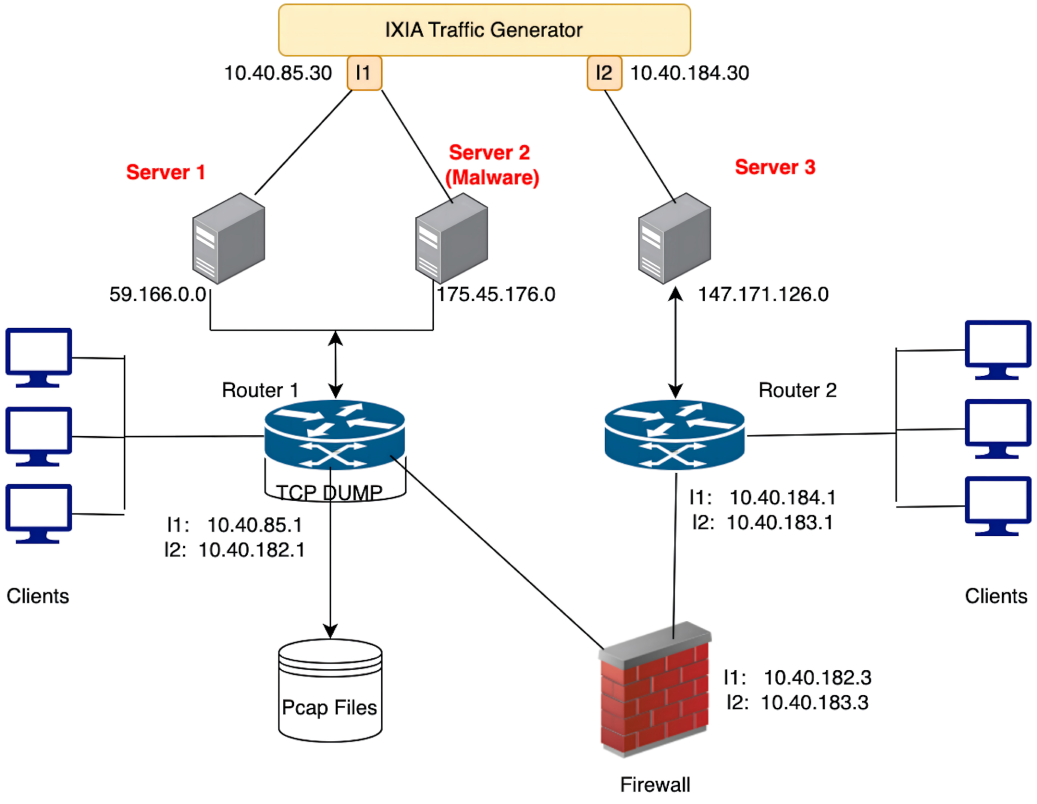

Figure 13. UNSW-NB15 network.

3) Sub network3 (Server3): contains nodes with source IP addresses from 149.171.126.0 to 149.171.126.19.

Subnetwork 1 (servers1) and subnetwork 3 (server3) are configured to exhibit

normal traffic patterns, whereas subnetwork 2 (server2) is associated with abnormal or malicious activities.

We employ our approach across various source IP addresses within all subnetworks. Our assumption is that the nature of the outbound traffic should not undergo sudden changes.

4.2.1. Detecting Suspicious Activities Based on DNS Traffic

Below are the values of the required parameters used within the *Suspicious* function for detecting suspicious traffic:

- τ (trace): it is the dataset available at [30].
- Let

$$\varphi = \langle (\text{IP.SourceAdd} = ips_1), \cdots, (\text{IP.SourceAdd} = ips_n) \rangle$$

where $\{ips_1, \cdots, ips_n\}$ are the different *IP* source addresses appearing in τ.

- Let $\psi = \langle (\text{IP.DestAdd} = ipd_1) \wedge (\text{UDP.DestPort} = DNS), \cdots, (\text{IP.DestAdd} = ipd_n) \wedge (\text{UDP.DestPort} = DNS) \rangle$, where $\{ipd_1, \cdots, ipd_n\}$ are the different *IP* destination addresses appearing in τ.
- $\text{w} = (\text{w}_1, \cdots, \text{w}_n) = (1, \cdots, 1)$.
- Δ is the KL-Divergence.
- Let $\mathcal{C}_2$ is the composition of K-means and Elbow Methods. The K-means do the clustering and the Elbow Methods estimate the best number of clusters.

Below, we present the results obtained from the Elbow Method corresponding to the different subnetworks (servers).

1) Sub network1 (Server1): the Source IP addresses from 59.166.0.0 to 59.166.0.9: shown in Figure 14.

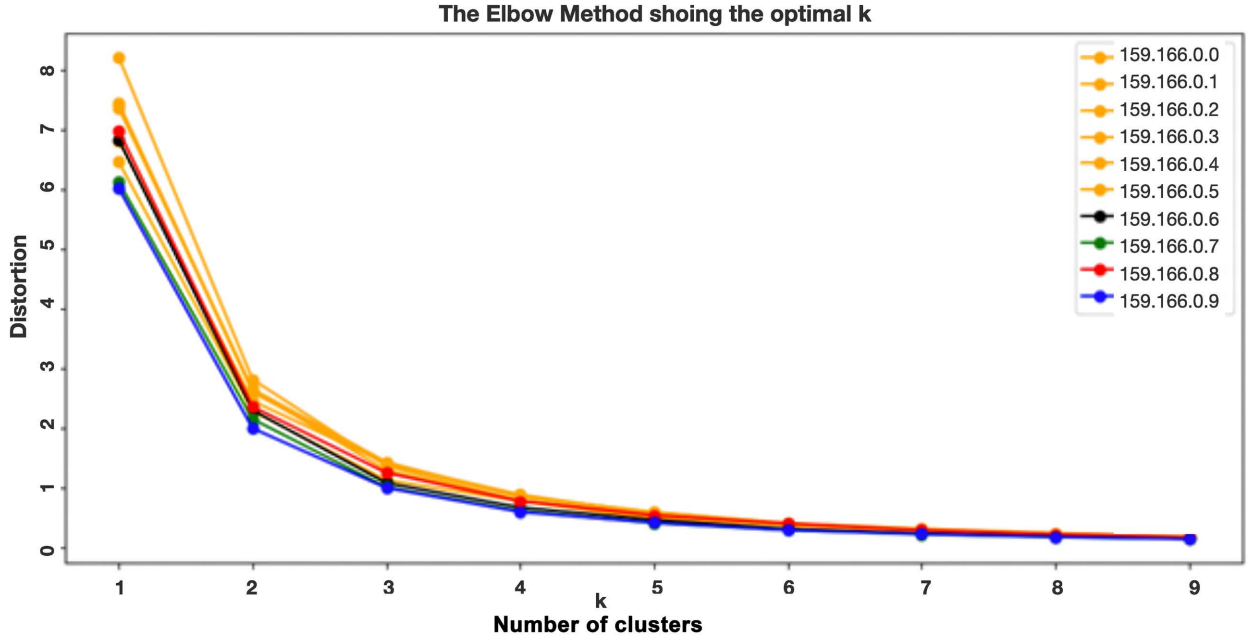

Figure 14. Elbow results for sub network1 (server1) based on DNS services.

2) Sub network2 (Server2): Source IP addresses from 175.45.176.0 to 175.45.176.3: shown in Figure 15.

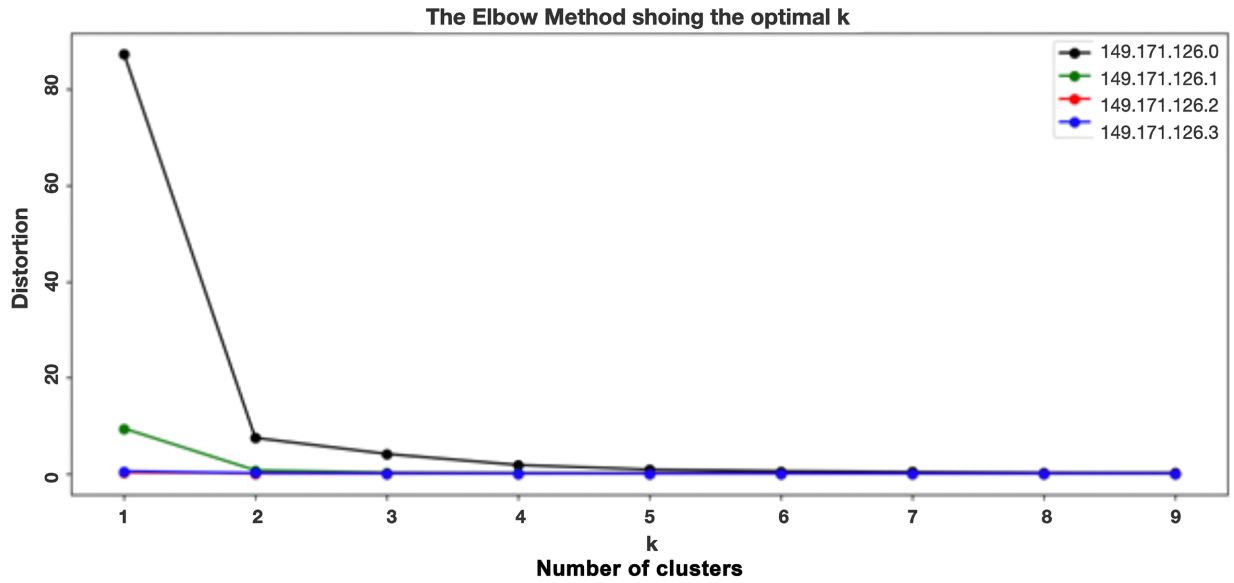

Figure 15. Elbow results for sub network2 (server2) based on DNS services.

3) Sub network3 (Server3): Source IP addresses from 149.171.126.0 to 149.171.126.19: shown in Figure 16.

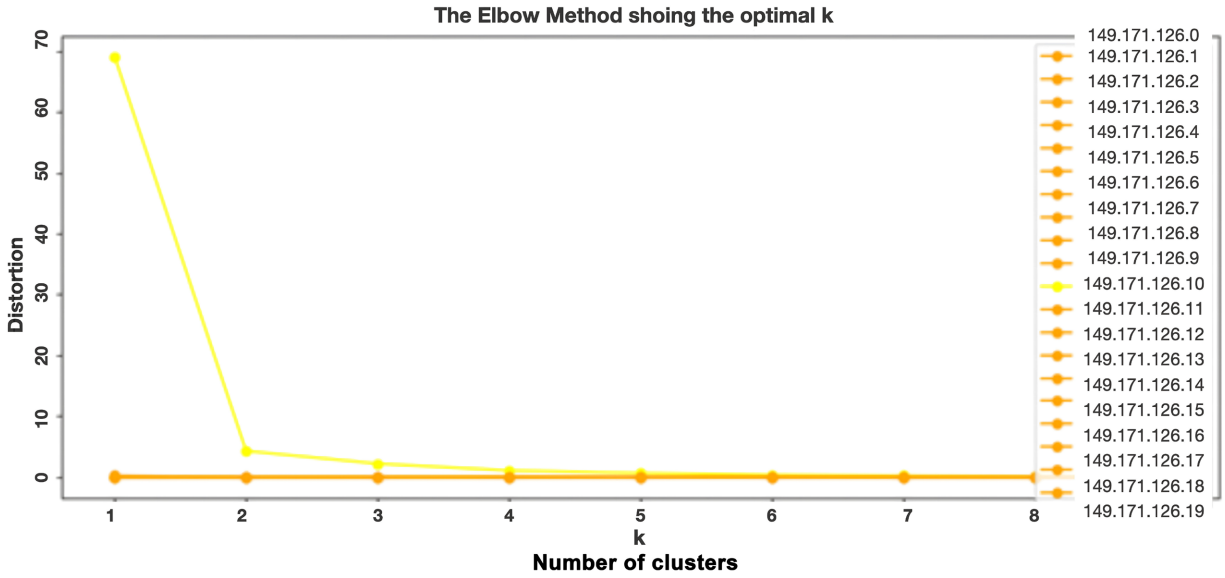

Figure 16. Elbow results for sub network3 (server3) based on DNS services.

From Figure 14, the maximum distortion value for subnetwork 1 (server1) is above 8. In Figure 15, the maximum distortion value for subnetwork 2 (server2) exceeds 80. Meanwhile, in Figure 16, the maximum distortion value for subnetwork 3 (server3) is above 70, but for only one IP address, whereas more than 20 IP addresses have a distortion value of zero.

Based on these findings, our approach predicts that subnetwork 2 (server2) is suspicious, as it exhibits abnormal or malicious activities in the network traffic. Therefore:

		Predicted		
		Negative	**Positive**	Total
Actual	**Negative**	2	0	2
	Positive	0	1	1
	Total	2	1	3

It follows that:

- $\text{Accuracy} = \dfrac{\text{TP}+\text{TN}}{\text{TP}+\text{TN}+\text{FP}+\text{FN}} = \dfrac{3}{3} = 100\%$.
- $\text{Precision} = \dfrac{\text{TP}}{\text{TP}+\text{FP}} = \dfrac{1}{1} = 100\%$.
- $\text{Recall} = \dfrac{\text{TP}}{\text{TP}+\text{FN}} = \dfrac{1}{1} = 100\%$.
- False Negative (FN) = 0%.
- False Positive (FP) = 0%.

4.2.2. Detecting Suspicious Activities Based on HTTP Traffic

Below are the values of the required parameters used within the *Suspicious* function for detecting suspicious traffic:

- τ (trace): it is the dataset available at [30].
- Let

$$\varphi = \left\langle \left(\text{IP.SourceAdd} = ips_1\right), \cdots, \left(\text{IP.SourceAdd} = ips_n\right) \right\rangle$$

where $\{ips_1, \cdots, ips_n\}$ are the different *IP* source addresses appearing in τ.

UDP.SourcePort

- Let $\psi = \left\langle \left(IP.DestAdd = ipd_1\right) \wedge \left(TCP.DestPort = HTTP\right), \cdots, \left(IP.DestAdd = ipd_n\right) \wedge \left(TCP.DestPort = HTTP\right) \right\rangle$, where $\{ipd_1, \cdots, ipd_n\}$ are the different *IP* destination addresses appearing in τ.
- $\mathrm{w} = \left(\mathrm{w}_1, \cdots, \mathrm{w}_n\right) = (1, \cdots, 1)$.
- Δ is the KL-Divergence.
- Let $\mathcal{C}_2$ is the composition of K-means and Elbow Methods.

Below, we present the results obtained from the Elbow Method for different subnetworks (servers).

1) Sub network1 (Server1): IP source addresses from 59.166.0.0 to 59.166.0.9 shown in Figure 17.

2) Sub network2 (Server2): IP source addresses from 175.45.176.0 to 175.45.176.3 shown in Figure 18.

3) Sub network3 (Server3): IP source addresses from 149.171.126.0 to 149.171.126.19 shown in Figure 19.

From Figure 17, the maximum distortion value for subnetwork 1 (server1) is

above 4. In Figure 18, the maximum distortion value for subnetwork 2 (server2) exceeds 2. Meanwhile, in Figure 19, the maximum distortion value for subnetwork 3 (server3) is above 0.008, but only for one IP address, while more than 20 IP addresses have a distortion value of zero.

Based on these findings, our approach predicts that subnetworks 1 (server1) and 2 (server2) are suspicious, as they exhibit abnormal or malicious activities in the network traffic. Therefore:

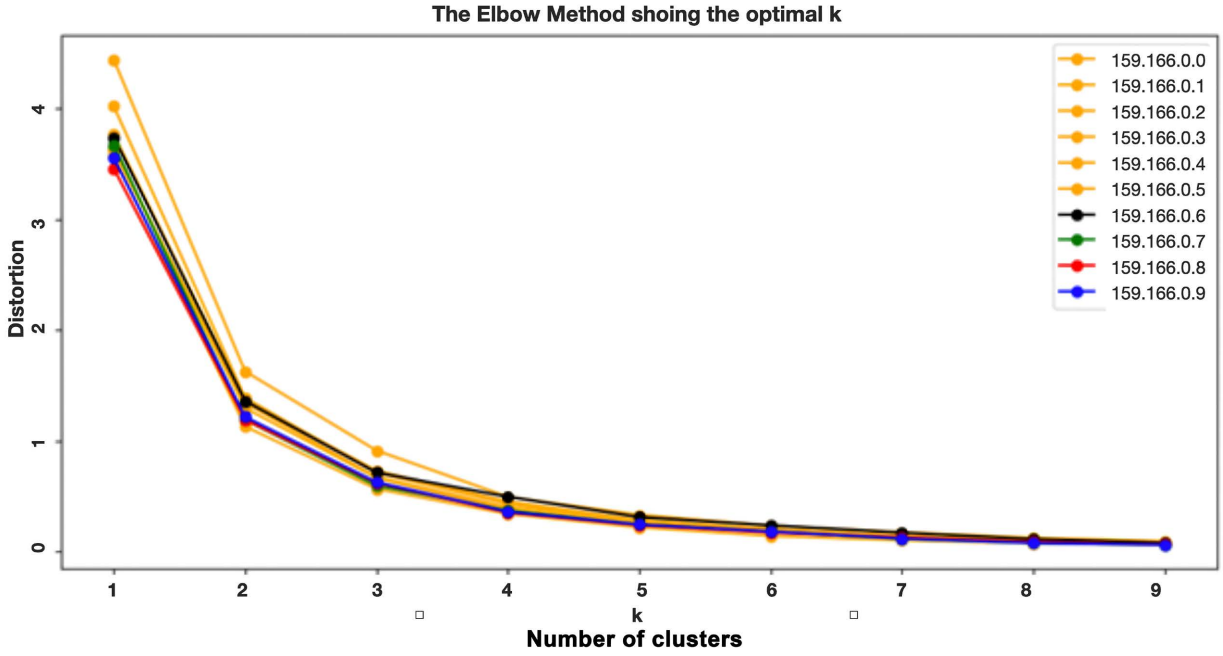

Figure 17. Elbow results for sub network1 (server1) based on HTTP services.

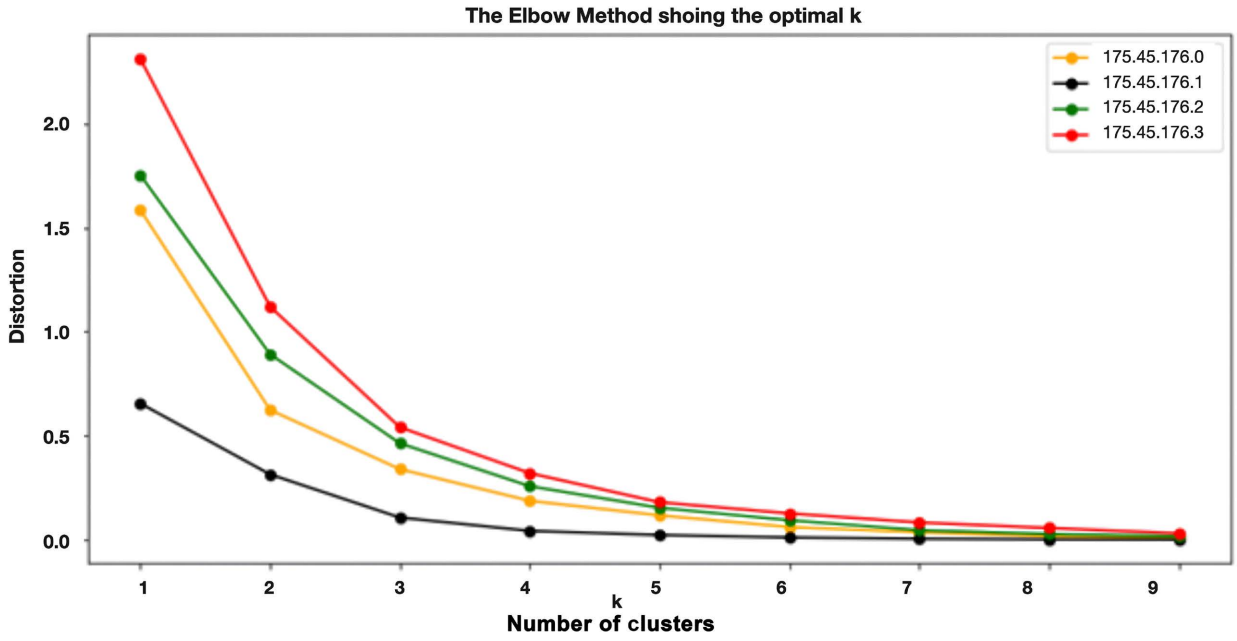

Figure 18. Elbow results for sub network2 (server2) based on HTTP services.

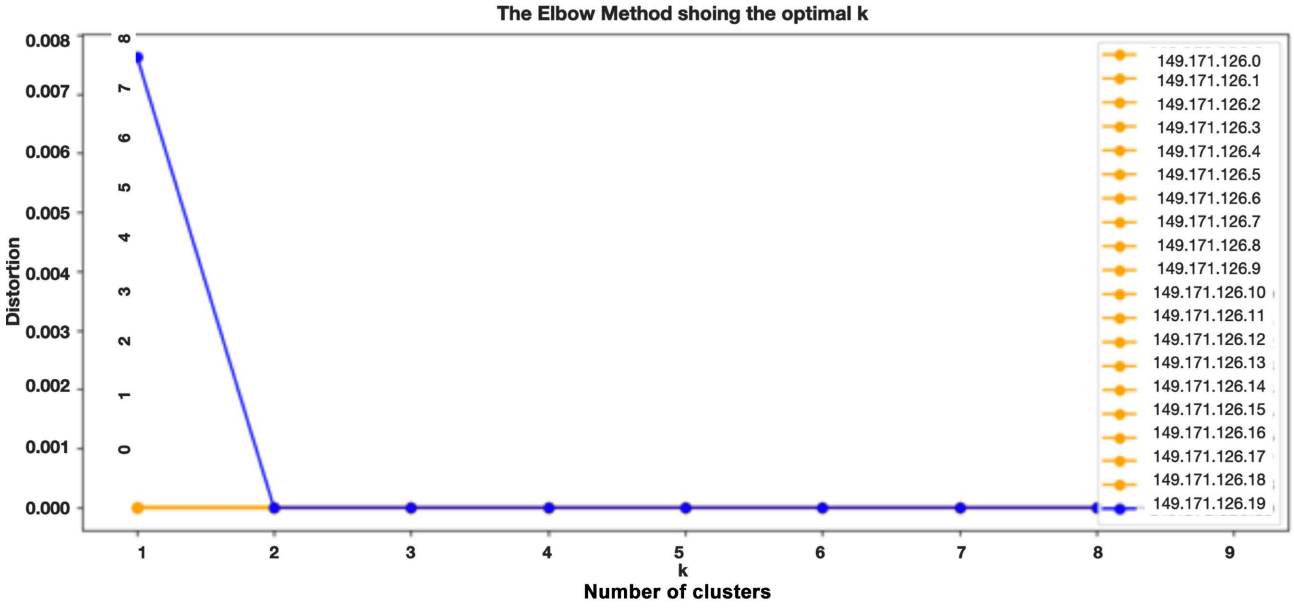

Figure 19. Elbow Results for sub network3 (server3) based on HTTP services.

		Predicted		
		Negative	Positive	Total
Actual	Negative	1	1	2
	Positive	0	1	1
	Total	1	2	3

It follows that:

- $\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} = \frac{2}{3} = 67\%$.
- $\text{Precision} = \frac{TP}{TP + FP} = \frac{1}{1} = 100\%$.
- $\text{Recall} = \frac{TP}{TP + FN} = \frac{1}{2} = 50\%$.
- False Negative (FN) = 0%.
- $\text{False Positive (FP)} = \frac{1}{3} = 33.33\%$.

4.3. Discussion

Table 14. Evaluation of the proposed approaches.

Techniques	Attacks types	Target	Learning types	Logic rules	Training is not required	Multi-target	Detection rate
-Kullback-Leibler (KL) Divergence. -Cosine Similarity -TF-IDF -k-mean algorithm.	suspicious attacks for different target	IP-Addresses TCP/UDP-Ports HTTP.Url others	unsupervised learning	✓	✓	✓	100%

Table 14 resumes the main features of the proposed approach. Although it has shown the best detection rate (100%), our experimental dataset remains small and we need to apply it on further representative datasets to have better precision on this parameter and other metrics.

5. Conclusions

This paper introduces a promising new technique for incident detection, leveraging differential analysis. Initially, the traffic undergoes dispersion via a slicing function $\mathcal{S}_{\varphi}$, partitioning it into sequences of slices based on propositional logical formulas φ, which are specified by the end-user. Subsequently, each slice undergoes transformation through a measuring function $\mathcal{F}_{\psi}$, mapping it to a point in $\mathbb{R}^n$ by quantifying select characteristics defined by the end-user via a formula ψ. following this, the distances between successive values returned by $\mathcal{F}_{\psi}$, associated with the same sequence, are evaluated using a designated function Δ (e.g., KL-Divergence). Lastly, employing a clustering technique (e.g., K-means), the values produced by Δ are clustered, and the number of clusters is estimated. If any sequence yields more than one cluster, it indicates suspicious activity.

The experimental results demonstrate significant promise, with a 100% accuracy achieved across both datasets used in the experiments. However, it's essential to note that this level of accuracy may not be guaranteed with other datasets and is contingent upon the parameters selected for analysis, such as φ and ψ.

In addition to its remarkable efficiency, the approach exhibits versatility in tackling a wide array of attacks spanning various activities, including those targeting networks, operating systems, and applications. Notably, it operates without necessitating any learning step or data.

Looking ahead, our future endeavors entail applying this methodology to diverse datasets encompassing log files that capture a spectrum of activities across networks, operating systems, and applications. Furthermore, we aspire to integrate this approach into an open-source Security Information and Event Management (SIEM) tool like Wazuh, thereby extending its accessibility and practicality within cybersecurity frameworks.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Khraisat, A., Gondal, I., Vamplew, P. and Kamruzzaman, J. (2019) Survey of Intrusion Detection Systems: Techniques, Datasets and Challenges. *Cybersecurity*, **2**, Article No. 20. https://doi.org/10.1186/s42400-019-0038-7

[2] Sureda Riera, T., Bermejo Higuera, J., Bermejo Higuera, J., Martínez Herraiz, J. and Sicilia Montalvo, J. (2020) Prevention and Fighting against Web Attacks through Anomaly Detection Technology. A Systematic Review. *Sustainability*, **12**, Article

4945. https://doi.org/10.3390/su12124945

[3] Aldwairi, M., Abu-Dalo, A.M. and Jarrah, M. (2017) Pattern Matching of Signature-Based IDS Using Myers Algorithm under Mapreduce Framework. *EURASIP Journal on Information Security*, **2017**, Article No. 7. https://doi.org/10.1186/s13635-017-0062-7

[4] Li, W., Tug, S., Meng, W. and Wang, Y. (2019) Designing Collaborative Blockchained Signature-Based Intrusion Detection in IoT Environments. *Future Generation Computer Systems*, **96**, 481-489. https://doi.org/10.1016/j.future.2019.02.064

[5] Detken, K., Rix, T., Kleiner, C., Hellmann, B. and Renners, L. (2015) SIEM Approach for a Higher Level of IT Security in Enterprise Networks. 2015 *IEEE 8th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications* (*IDAACS*), Warsaw, 24-26 September 2015, 322-327. https://doi.org/10.1109/idaacs.2015.7340752

[6] Madani, A., Rezayi, S. and Gharaee, H. (2011) Log Management Comprehensive Architecture in Security Operation Center (SOC). 2011 *International Conference on Computational Aspects of Social Networks* (CASoN), Salamanca, 19-21 October 2011, 284-289. https://doi.org/10.1109/cason.2011.6085959

[7] (2023) The Wazuh Manual. https://documentation.wazuh.com/current/user-manual/index.html

[8] Najafabadi, M.M., Khoshgoftaar, T.M., Calvert, C. and Kemp, C. (2017) User Behavior Anomaly Detection for Application Layer DDoS Attacks. 2017 *IEEE International Conference on Information Reuse and Integration* (*IRI*), San Diego, 4-6 August 2017, 154-161. https://doi.org/10.1109/iri.2017.44

[9] Betarte, G., Giménez, E., Martínez, R. and Pardo, Á. (2018) Machine Learning-Assisted Virtual Patching of Web Applications. arXiv: 1803.05529.

[10] Owasp.org (2021) OWASP ModSecurity Core Rule Set. https://owasp.org/www-project-modsecurity-core-rule-set/

[11] Wang, L., Cao, S., Wan, L. and Wang, F. (2017) Web Anomaly Detection Based on Frequent Closed Episode Rules. 2017 *IEEE Trustcom/BigDataSE/ICESS*, Sydney, 1-4 August 2017, 967-972. https://doi.org/10.1109/trustcom/bigdatase/icess.2017.338

[12] Bronte, R., Shahriar, H. and Haddad, H. (2016) Information Theoretic Anomaly Detection Framework for Web Application. 2016 *IEEE 40th Annual Computer Software and Applications Conference* (*COMPSAC*), Atlanta, 10-14 June 2016, 394-399. https://doi.org/10.1109/compsac.2016.139

[13] Ren, X., Hu, Y., Kuang, W. and Souleymanou, M.B. (2018) A Web Attack Detection Technology Based on Bag of Words and Hidden Markov Model. 2018 *IEEE 15th International Conference on Mobile Ad Hoc and Sensor Systems* (*MASS*), Chengdu, 9-12 October 2018, 526-531. https://doi.org/10.1109/mass.2018.00081

[14] Pukkawanna, S., Kadobayashi, Y. and Yamaguchi, S. (2015) Network-based Mimicry Anomaly Detection Using Divergence Measures. 2015 *International Symposium on Networks, Computers and Communications* (*ISNCC*), Yasmine Hammamet, 13-15 May 2015, 1-7. https://doi.org/10.1109/isncc.2015.7238570

[15] Clement, A. (2020) On Network-Based Mimicry Anomaly Detection Using Divergence Measures and Machine Learning. Master's Thesis, AIMS Senegal.

[16] Münz, G., Li, S. and Carle, G. (2007) Traffic Anomaly Detection Using K-Means Clustering. GI/ITG Workshop MMBnet.

[17] Asselin, E., Aguilar-Melchor, C. and Jakllari, G. (2016) Anomaly Detection for Web

Server Log Reduction: A Simple yet Efficient Crawling Based Approach. 2016 *IEEE Conference on Communications and Network Security* (*CNS*), Philadelphia, 17-19 October 2016, 586-590. https://doi.org/10.1109/cns.2016.7860553

[18] Swarnkar, M. and Hubballi, N. (2015) Rangegram: A Novel Payload Based Anomaly Detection Technique against Web Traffic. 2015 *IEEE International Conference on Advanced Networks and Telecommuncations Systems* (*ANTS*), Kolkata, 15-18 December 2015, 1-6. https://doi.org/10.1109/ants.2015.7413635

[19] Kang, I., Jeong, M.K. and Kong, D. (2012) A Differentiated One-Class Classification Method with Applications to Intrusion Detection. E*xpert Systems with Applications*, **39**, 3899-3905. https://doi.org/10.1016/j.eswa.2011.06.033

[20] Camacho, J., Pérez-Villegas, A., García-Teodoro, P. and Maciá-Fernández, G. (2016) PCA-Based Multivariate Statistical Network Monitoring for Anomaly Detection. *Computers & Security*, **59**, 118-137. https://doi.org/10.1016/j.cose.2016.02.008

[21] Yoshimura, N., Kuzuno, H., Shiraishi, Y. and Morii, M. (2022) DOC-IDS: A Deep Learning-Based Method for Feature Extraction and Anomaly Detection in Network Traffic. *Sensors*, **22**, Article 4405. https://doi.org/10.3390/s22124405

[22] Zavrak, S. and Iskefiyeli, M. (2023) Flow-Based Intrusion Detection on Software-Defined Networks: A Multivariate Time Series Anomaly Detection Approach. *Neural Computing and Applications*, **35**, 12175-12193.

[23] Joyce, J.M. (2011) Kullback-Leibler Divergence. In: Lovric, M., Ed., *International Encyclopedia of Statistical Science*, Springer, 720-722. https://doi.org/10.1007/978-3-642-04898-2_327

[24] Li, B. and Han, L. (2013) Distance Weighted Cosine Similarity Measure for Text Classification. In: Yin, H., *et al.*, Eds., *Intelligent Data Engineering and Automated Learning—IDEAL* 2013, Springer, 611-618. https://doi.org/10.1007/978-3-642-41278-3_74

[25] Sammut, C., and Webb, G. (2010) TF-IDF. In: Sammut, C. and Webb, G.I., Eds., *Encyclopedia of Machine Learning*, Springer, 986-987. https://doi.org/10.1007/978-0-387-30164-8_832

[26] Keogh, E., Lonardi, S. and Ratanamahatana, C.A. (2004) Towards Parameter-Free Data Mining. *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Seattle, 22-25 August 2004, 206-215. https://doi.org/10.1145/1014052.1014077

[27] Kanungo, T., Mount, D.M., Netanyahu, N.S., Piatko, C.D., Silverman, R. and Wu, A.Y. (2002) An Efficient K-Means Clustering Algorithm: Analysis and Implementation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **24**, 881-892. https://doi.org/10.1109/tpami.2002.1017616

[28] Dempster, A.P., Laird, N.M. and Rubin, D.B. (1977) Maximum Likelihood from Incomplete Data via the *em* Algorithm. *Journal of the Royal Statistical Society Series B*: *Statistical Methodology*, **39**, 1-22. https://doi.org/10.1111/j.2517-6161.1977.tb01600.x

[29] Crawford, C. Computer Network Traffic. https://www.kaggle.com/datasets/crawford/computer-network-traffic

[30] Moustafa, N. and Slay, J. (2015) UNSW-NB15: A Comprehensive Data Set for Network Intrusion Detection Systems (UNSW-NB15 Network Data Set). 2015 *Military Communications and Information Systems Conference* (*MilCIS*), Canberra, 10-12 November 2015, 1-6. https://doi.org/10.1109/milcis.2015.7348942