A NOVEL ALERT CORRELATION TECHNIQUE FOR FILTERING NETWORK ATTACKS

Jane Kinanu Kiruki[1,3], Geoffrey Muchiri Muketha[2] and Gabriel Kamau[1]

[1]Department of Information Technology, Murang'a University of Technology, Kenya
[2]Department of Computer Science, Murang'a University of Technology, Kenya
[3]Department of Computer Science, Chuka University, Kenya

ABSTRACT

An alert correlation is a high-level alert evaluation technique for managing large volumes of irrelevant and redundant intrusion alerts raised by Intrusion Detection Systems (IDSs).Recent trends show that pure intrusion detection no longer can satisfy the security needs of organizations. One problem with existing alert correlation techniques is that they group related alerts together without putting their severity into consideration. This paper proposes a novel alert correlation technique that can filter unnecessary and low impact alerts from a large volume of intrusion. The proposed technique is based on a supervised feature selection method that usesclass type to define the correlation between alerts. Alerts of similar class type are identified using a class label. Class types are further classified based on their metric ranks of low, medium and high level. Findings show that the technique is able detect and report high level intrusions.

KEYWORDS

Alert correlation, merging, classification, alert clustering, alert prioritization, pre-processing

1. INTRODUCTION

Network attacks have become more serious in recent years, forcing the deployment if appropriate security devices such as firewalls and Intrusion Detection Systems (IDSs). IDSs inspect network activity with the aim of identifying suspicious behaviour, and if found, report the same in the form of alerts. Two common methods for reporting intrusion alerts are onscreen or email. The onscreen method is slow and can only be accessed from a physical site while thee mail method is fast and can be accessed both internally and remotely, and with internet connectivity. With email alerting, the IDS is linked to mail Gateway to send alerts notifications during intrusions[1]. There are two common types of IDSs depending on the method employed for traffic inspection, namely, signature-based and anomaly-based IDSs. However, both types suffer from the problem of generating numerous unsorted, unverified, noisy and dirty alerts per day. Additionally, most of the alerts are false alerts resulting from non-existing intrusions thereby diminishing the value of interesting alerts. Usually, analysts are overwhelmed with the voluminous alerts hence not likely to look at them until a sign is reported by other security means. This is because identifying interesting alerts and reporting network status is a laborious and challenging task[2][3].There is a need for a proactive network monitoring tool that can be used to continuously analyse network performance issues and bottlenecks.

To handle the problem at hand, we propose low-level and high-level alert evaluation operations. Low-level alert operations deal with each alert individually to enrich its attributes or assign a score to it based on the potential risk. High-level alert evaluation operations deal with groups of alerts and give an abstraction of each. The proposed approach is based on a supervised feature

selection method that uses similarity approach by class type. Metrics are used to rank the class types into low level, medium level and high level. A threshold is then used to eliminate variables less the set threshold hence low level alerts are discarded and high level intrusions are reported through short message services whenever encountered.

Results show an improved learning performance with better learning precision, lower computational time, and improved technique understand ability. However, the removal of irrelevant features help learn a better model, as irrelevant features confuse the learning system and cause memory and computation inefficiency [4].

The rest of this paper is structured as follows. Section 2 presents related work, section 3 presents our Methodology, section 4 presents the proposed alert correlation technique, section 5 presents the results, section 6 presents the discussion, and section 7presents the conclusions and future work.

2. RELATED WORKS

Alert correlation is a technique used to determine any association between alerts in relation to launched attacks. Alert correlation techniques have been classified into the following types: predefined attack scenarios-based approaches, similarity-based approaches, prerequisites and consequences-based approaches, and hybrid approaches. The main objective of these approaches is to categorize alerts and to reduce false positive ones [5][6].

Alert correlation has been further classified according to four criteria, namely, the number of information data sources, type of application domain, correlation methods and architecture [6-9].The number of information data sources can be single source or multiple source. Single source data has the advantage of its simplicity but fails to achieve optimal results from correlation. As such, it not the best solution for collaborative monitoring systems. Multiple data source has high cost mainly due to the heterogeneity of the different inputs but give better results than single data source. The type of application domain though versatile has mainly been employed in network management systems that aim at allowing operators to monitor the system by generating alerts for warning about problems in the network and has also been used in IT security to produce attack reports that capture a coherent view of the activity on the network or systems without losing security-relevant information and process control in manufacturing systems to identify the root cause of problems or process disturbances. Finally, Correlation methods have been classified as similarity-based, sequential-based and case-based while architecture has been classified as centralized, distributed and hierarchical.

Similarity based methods usually try to reduce the total number of similar alerts through clustering and aggregation. Researchers[7]have proposed a new alert correlation technique that works by extracting network flows. The technique was built without requiring pre-defined knowledge and pre-conditions to allow the discovery of new correlation relationships between alerts. The method used two-steps in analysing the feature of alert flows called low-level alert analysis and high level alert analysis. Analysed alerts were generated by the IDS system using Snort although other researchers [11] have proposed the use of a wide variety of sources of information in order to achieve the goals of alert correlation effectively and accurately.

Other related works include case based methods which rely on the presence of knowledge to signify well-defined scenarios. These methods try to correlate alerts based on known scenarios [8].They are efficient for solving well-known problems specifying a complete action plan or previously observed scenarios. However, it is not easy sometimes to exhaustively list all scenario

templates and build a database containing a comprehensive set of problems solutions. In addition, time inefficiency may make them unusable in real-time alarm correlation.

Further, researchers [9] have proposed the prerequisite and consequence relationship technique, applies a sequential based correlation method that deals with the relationship between alerts based on pre and post conditions. The assumption is that previous alerts prepare for later ones. Advantages of sequential-based methods are that they are scalable, can potentially uncover the causal relationship between alerts, and are not restricted to known attack scenarios. However, correlation results may contain a large number of false correlations, this being for two possible reasons: either the logical predicates are not well configured or the quality of the sensor alerts is not adequate.

Other researchers used different methods to determine alert association. In[10],a combination of conditional rough entropy and knowledge granularity calculation to find important attributes and their weights was used. Alerts were aggregated based on weight similarity. However, the researcher did not use real-world attacks for better outcome hence current attacks cannot be captured.

In [11],statistical-based algorithms were used to store causal relationships between alerts and analyse their frequency of occurrence. The method also takes into consideration of the previous attack data to generate attack steps. The attacks relationship knowledge is used to correlate different attack stages. Computing the alert relationship using this algorithm is nearly impossible in cases where sensors are providing incomplete data.

In[12],the destination port and alert type are combined together as parameters for analysis. If two alerts with the destination pot and alert type occurred in sequence, they were grouped as similar alerts, otherwise grouped as different alerts. Time-lag based Sequence Splitting (TSS) and SPA Sequence Pruning Algorithm (SPA) was used. TSS was used to split long attack sequences while SPA was used to eliminate the duplicated sessions extracted by TSS. Graphs were generated from the results. However, it is not easy to interpret the intrusion sequence because the attack sequence was interfered with while filtering out some of the repetitive attack steps.

Sequence-based algorithms depend on preconditions which are the main determinants for a successful attack. This makes it hard for such algorithms to determine the relationship between attacks in an operational network since attack patterns cannot be determined before attacks happen. To solve this problem,[13] proposed the use of time prefix span algorithm in alert correlation. The algorithm reduced the datasets, used time sequence and time intervals between attacks which improved its efficiency. However, by reducing the datasets there is a challenge of maintaining data quality and accurate results[14].

3. METHODOLOGY

In this study, the quantitative approach based on simulation experiment is used due to its ability to use numbers and figures in data analysis and assignment of scores that measure distinct attributes. The research uses deductive logic to addresses the issues of improving the quality of alerts that are generated by multiple sensors. Data derived from the metrics in our previous research is used for further analysis to get rid of the redundant alerts. To do this, similarity-based clustering is used to detect redundancies and merge the alert cluster. A multi-level threshold is used that assist in classifying intrusions according to their severity and reporting high level intrusion using short message services whenever encountered.

3.1. Data Collection

Low level, medium level and high level alerts from scored alert database composed of IDS, firewall and honey pot alerts was used. Alert attributes consist of several fields, including message field, class type, security identifier (SID)and alert identifier (ID) that provide information about the attack. An example is Id system Attack ET ATTACK RESPONSE Net User Command Response successful user 20170whereIdsystem Attack is the alert identifier, ET ATTACK RESPONSE Net User Command Response is the message and 2017025 is the SID. This information varies from one IDS product to another.

Three criteria were specified to determine the appropriate sample size, including the level of precision, the level of confidence or risk, and the degree of variability in the attributes being measured[15]. We determined our sample size using Yamane's formula [16] as shown below.

$$n = \frac{N}{1 + N(e)^2}$$

Where n = Desired sample size
N = Population of the study
e = precision of sampling error (0.03)

Therefore, we used a sample of 1000 alerts from a total population of 24,050.

3.2. Alert Management

In order to overcome IDS drawbacks, alert management systems were established. Alert management systems became useful when handling alerts and when generating an overview of intrusion. In our study, we categorised alerts into two groups, namely, low level alert management and high level alert management.

3.2.1. Low-Level Alert Process

A low-level alert process involves alerts from various multiple sources including IDS database, firewall database and honey pot database that were centrally collected in a MHN server. Alerts attributes were selected and scored using common vulnerability scoring (CVS) database and expert opinion based on its potential risk. The alerts were then forwarded to the alert prioritization metric[17] for rescoring. The reason for rescoring was to identify alert weight for each alert and later classify them into low medium and high level classification so that low scored alerts could be discarded. Therefore, low-level assessment methods are required to automatically analyse huge numbers of alerts and prioritize them for further processes. The prioritized alerts dataset lead to a more accurate high-level alert analysis [18][19].Figure 1 shows the Low-level and High-level alert evaluation management.

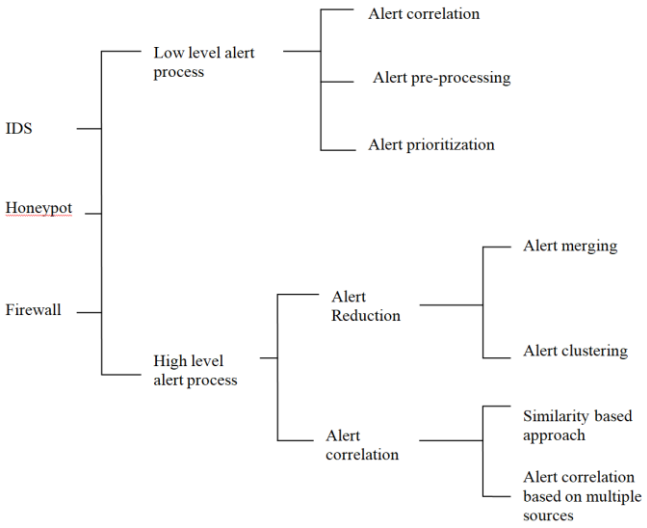

Figure 1. Low-level and High-level alert evaluation

3.2.2. High-Level Alert Process

High-level alert process techniques, such as merging, clustering and correlation, were proposed to deal with sets of alerts and provide an abstraction of them. To study the relationships between alerts, it is indispensable to analyse their features. Hence, identifying the main features is an essential step for further processing. The reduction of the input variables set can be performed through feature selection which selects a subset from the original features hence reducing the dimensionality of data while ensuring that the same is not transformed into a new set

The alert correlation techniques are composed of multiple correlation approaches to improve the result [20]. These include rule based, probabilistic, case based and multiple data source alert correlation techniques. In rule based alerts with similar source address and destination address are classified together. In probabilistic technique each alert is represented as a fraction of the total number of alerts. In case based alerts are correlated based on known attack cases in the knowledge base.

4. THE PROPOSED ALERT CORRELATION TECHNIQUE

The proposed alert correlation technique is focused on high level alert management. Alerts obtained from low level alert management are classified, merged and aggregated to eliminate alert redundancy. To obtain optimal results, a combination of rule based, probabilistic and case based approaches have been used. The approaches are used in various stages of the alert correlation architecture. The rule based approach is used both in alert classification and alert merging while the case based approach is used to come up with the alert scales at the alert aggregation stage.

4.1. Architecture

In this study multiple data source have been used that consists of scored IDS alerts, firewall alerts, honey pot alerts, CSV database and collected data from questionnaire. A two-steps analysis of similarity based alert correlation[7][21].The first step was to reduce the number of alerts by merging them based on their class type while the second step was to rank them into low level, medium level and high level. Our assumption was that similar alerts usually have the same

cause and effects on the network and systems. Figure 2 describes the proposed alert correlation architecture.

The proposed solution has six major components, namely, alert collection, alert prioritization, fuzzification, alert classification, alert merging and alert aggregation. When a new alert is collected, the associated scoring metrics are computed in relation to the expert opinion and vulnerability database. A fuzzy logic inference is then implemented in order to allocate a score to the alert in relation to the metrics values. The alert is then recorded in the alert database with its score for high level alert processing. At this level feature selection, alert merging, classification and aggregation is carried out to group alerts into low level, medium and high level. All medium and high level alerts are taken to the next stage for further analysis while low level alerts are discarded.

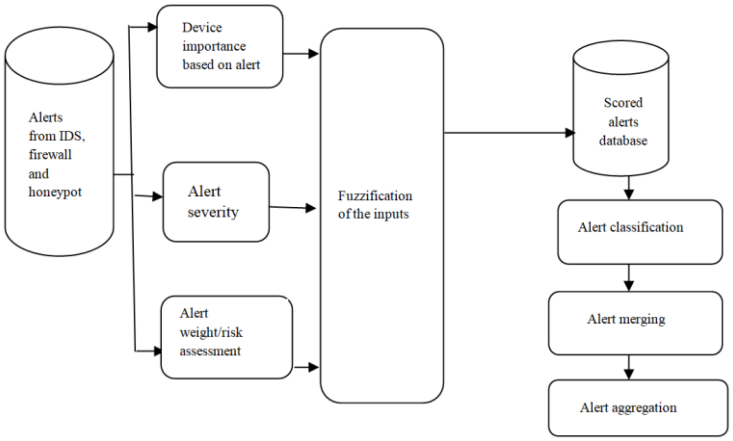

Figure 2. Proposed alert correlation architecture

4.2. Feature Selection

IDSs usually produce an enormous amount of alerts when unusual activity is identified. Inspecting and examining all relayed alerts manually is a difficult, error-prone, and time-consuming task. Additionally, overlooking alerts could result into successful intrusions being missed, hence the need for feature selection.

Feature selection focuses in isolating a small group of important attributes from the original ones by eliminating irrelevant, redundant, or noisy attributes. Selection results in an improved learning outcome, i.e., high learning precision, lower analytical cost, and improved technique interpretability as irrelevant features will confuse the learning system and cause memory and computation inefficiency.

In this study, study, we use supervised feature selection o identify only relevant alerts from the alerts collected in the high level alert management database in order to get rid of noise in the dataset and improve technique accuracy. This involved similarity based alert correlation where alerts belonging to the same class type were identified using a class label. The label information provides discriminative information to select relevant features thus, the availability of class labels allowed backwards sequential filter method to effectively select one alert from every class type and discard the redundant alerts. Selected features with label information were used to train a classifier for prediction.

4.3. Alert Merging

The role of the merging module is to group alerts with similar characteristics such as the source IP, the target IP, the type of the attack, etc. The module attempts to combine a group of alerts to reduce the volumes of redundant and isolated alerts belonging to the same attack activity within a particular time window for different IDS products. We merged alerts based on Valdes approach [10], that was based on attribute similarity and in our case we also incorporated alert class type to decrease repetitive ones.

4.4. Rule based Classification and Aggregation

In this phase, we defined similarity membership using a Likerts scale based on alert scores as defined in the low level alert management. Alerts from different classes with similar scores were aggregated together within the set thresholds into low level relevance, medium level relevance and high level relevance.

4.5. Application of the Technique

In this study, we configured threshold limits to help identify alert severity level on the resources, services and applications running on servers and network devices proactively. Every intrusion alerts had a threshold value set based on its severity on network resources. A multilevel threshold was used to help in classifying intrusions according to their severity and reporting high level intrusions through short message services whenever encountered. Utilizing alert thresholds, high level severity alerts can be reported before the device goes down or reach a critical value. The most important part with this monitoring system is that it is done in real time.

5. RESULTS

5.1. Risk Assessment Intrusion based on Alert Impact

Risk assessment was carried out on servers, switches, routers, access points and end user computers. The assessment was based on the alert impact on the network resources and demonstrated its efficiency in high level alert management. Alert priority, value of network resources (device value) and alert reliability parameters obtained from the survey[17] were used to analyze the alert impact. The alert priority value shows the importance of an alert in terms of its severity and its value ranged from 0 to 5. The device value shows the value of network resources within the environment and its value also ranged from 0 to 5. The alert reliability value ranged from 0 to 10 and shows the level to which an alert is not a false positive. Data showing the probability of attacks happening was collected through an experiment. The probability of attacks happening, alert priority, device value and alert reliability data are shown in Table 1.

Table 1. Shows attack impact values based on a survey carried on the network security experts

Attacks types	**Probability of attacks happening**	**Device value**			**Alert priority**			**Alert reliability**		
		Mail Server	Koha	ERP	MailServer	Koha	ERP	Mail Server	Koha	ERP
Web application attack	0.637	5	4	5	4	3	4	8	7	8

Suspicious login	0.004	5	4	5	4	3	4	8	7	8
Denial of service	0.000	5	4	5	4	3	4	8	7	8
Non standard protocol	0.002	4	3	4	3	2	3	7	6	7
Successful admin	0.001	5	4	5	4	3	4	8	7	8
Trojan	0.176	4	3	4	3	2	3	7	6	7
Attempted user	0.003	4	3	4	3	2	3	7	6	7
Policy violation	0.069	4	3	4	3	2	3	7	6	7
Misc attack	0.006	3	2	3	2	1	2	6	5	6
Service probe	0.001	3	2	3	2	1	2	6	5	6
Unsuccessful admin	0.000	4	3	4	3	2	3	7	6	7
Unsuccessful user	0.001	4	3	4	3	2	3	7	6	7
Attempted recon	0.012	3	2	3	2	1	2	6	5	6
Miscellaneous activity	0.029	3	2	3	2	1	2	6	5	6
Not suspicious	0.004	3	2	3	2	1	2	6	5	6
Protocol command decode	0.028	4	3	4	3	2	3	7	6	7
Attempted Dos	0.003	4	3	4	3	2	3	7	6	7
Bad unknown	0.004	4	3	4	3	2	3	7	6	7
String detect	0.010	4	3	4	3	2	3	7	6	7
Successful recon limited	0.004	4	3	4	3	2	3	7	6	7

Using the parameters in Table1, the device value, severity score, frequency and alert weight for each alert were computed. The results were then fuzzified to obtain the relevance of each alert. For example, the web application attack had a device value of 1.911, severity score of 1.815, frequency 0.637, alert weight of 10 and after fuzzification we obtained an alert relevance of 14.2. This information is also shown in Table 2.

Table 2. Metric and fuzzy results

Alert name	**Web application Attack**	**Trojan**	**Misc activity**	**Policy Violation**	**Non standard protocol**	**Suspicious login**	**Misc attack**	**Service probe**	**Attempted user**	**Not suspicious**
Device score	1.911	0.53	0.087	0.207	0.006	0.012	0.018	0.003	0.009	0.008
Severity score	1.815	0.502	0.079	0.189	0.006	0.011	0.017	0.003	0.008	0.011

Frequency/probability of alert happening	0.637	0.176	0.029	0.069	0.002	0.004	0.006	0.001	0.003	0.004
Alert weight	10	6.96	3.04	6.959	6.959	9.4	3.04	3.04	6.959	3.04
Alert relevance from fuzzy	14.2	8.5	2.83	2.83	2.89	2.89	2.87	2.89	2.89	2.89

The alerts were grouped according to class type. Each class type had more than one alert. Example, the web application attack had 637 attack alerts. This meant that there were 636 redundant alerts that were eliminated. Only one attack alert was left to represent the class type. Each alert is described by its ID system, message, class type, is_active, attack value and alert level. The ID system represent its position number within the span of attack collection, message represents the name of the attack and class type is the group in which an attack belonged. Is_active shows whether the alert is active or not. In this case YES refers to active alert. Every class type was given a number that represented all the alerts that belonged to it. This is the attack value. Example, all the web application attack alerts were represented by attack value of 7. The alert level represented the alert relevance ranges. This information is also shown in Table 3.

Table 3. Data filtering by class type

Id system Attacks	**Message**	**Class type**	**Sid**	**is_active**	**Attack Value**	**Alert level**
1226	ET ATTACK_RESPONSE Net User Command Response	successful-user	2017025	YES	18	Low relevance
2009	ET POLICY WebRTC IP tracking Javascript	successful-recon-limited	2021089	YES	21	Low relevance
4645	ET INFO SUSPICIOUS .LNK File Inside of Zip	Unknown	2035026	YES	14	Low relevance
4693	ET TROJAN Win32/Pterodo Activity (POST)	trojan-activity	2035220	YES	1	Medium relevance
4697	GPL CHAT Jabber/Google Talk Outgoing Traffic	not-suspicious	2100230	YES	6	Low relevance
4801	GPL SHELLCODE sparcsetuid 0	system-call-detect	2100647	YES	25	Low relevance
4929	GPL FTP CWD .... attempt	denial-of-service	2101779	YES	20	Low relevance
4937	GPL ATTACK_RESPONSE id check returned apache	bad-unknown	2101886	YES	11	High relevance
4946	GPL SCAN SolarWinds IP scan attempt	network-scan	2101918	YES	13	Low relevance

4953	GPL FTP shadow retrieval attempt	suspicious-filename-detect	2101928	YES	19	Low relevance
4973	GPL WEB_SERVER perl post attempt	web-application-attack	2101979	YES	8	High relevance
4992	GPL EXPLOIT rsyncd module list access	misc-activity	2102047	YES	4	Low relevance
4994	GPL RPC portmaprpc.xfsmd request TCP	rpc-portmap-decode	2102082	YES	24	Low relevance
4997	GPL EXPLOIT Microsoft cmd.exe banner	successful-admin	2102123	YES	16	Low relevance
5005	GPL NETBIOS SMB DCERPC invalid bind attempt	attempted-dos	2102191	YES	9	Low relevance
5007	GPL SMTP AUTH LOGON brute force attempt	suspicious-login	2102275	YES	22	Low relevance
5009	GPL SHELLCODE x86 0x71FB7BAB NOOP Unicode	shellcode-detect	2102313	YES	17	Low relevance
5048	GPL MISC HP Web JetAdmin file write attempt	web-application-activity	2102549	YES	7	Low relevance
5049	GPL SCAN nessus 2.x 404 probe	attempted-recon	2102585	YES	5	Low relevance
5050	GPL P2P eDonkey server response	policy-violation	2102587	YES	3	Low relevance
5057	GPL SQL TO_CHAR buffer overflow attempt	attempted-user	2102699	YES	2	Low relevance
5090	GPL SQL sa brute force failed login attempt	unsuccessful-user	2103152	YES	23	Low relevance
5103	GPL NETBIOS SMB-DS OpenKey overflow attempt	attempted-admin	2103226	YES	15	Low relevance
5125	GPL NETBIOS SMB CoGetInstanceFromFile attempt	protocol-command-decode	2103425	YES	10	Low relevance
5168	ET 3CORESec Poor Reputation IP UDP group 20	misc-attack	2525039	YES	12	Low relevance

The results obtained after filtering constituted of low, medium and high relevance alerts without duplicates. All the alerts whose relevance fell between 0.0 and 5.66 were categorised as low relevance and were discarded. However, alerts whose relevance fell between 5.67 and11.32 and between 11.33 and17.0 were categorised as medium and high relevance respectively and were accepted for further analysis. This information is also shown in Table 4.

Table 4. Alert relevance ranges

Alert relevance	**Ranges**
Low relevance	0 - 5.66
Medium relevance	>5.67 - 11.32
High relevance	>11.33 - 17.0

High relevance alerts are those attacks that could potentially make data confidentiality, integrity and availability highly vulnerable. Example, bad-unknown and web-application-attack had high relevance of 15 and 14.2 respectively. Appropriate preventive measures should be taken in order to avoid the escalation of high relevance attacks. Medium alert relevance is attack that happens and has medium effect on the system confidentiality, integrity and availability. These alerts need only preventive measures to be taken. This includes the Trojan-activity with a relevance of 8.5.Low alerts relevance are alerts of low severity that needs to be discarded. Some of them are the attempted-user and not-suspicious alerts that had a relevance of 2.89 each. This information is also shown in Table 5.

Table 5. The low, medium and high level relevance alerts

Id system Attacks	Message	Class type	Sid	is_active	Attack Value	Alert names	Alert Relevance
4693	ET TROJAN Win32/Pterodo Activity (POST)	trojan-activity	2035220	YES	1	Medium relevance	8.5
4697	GPL CHAT Jabber/Google Talk Outgoing Traffic	not-suspicious	2100230	YES	6	Low relevance	2.89
4937	GPL ATTACK_RESPONSE id check returned apache	bad-unknown	2101886	YES	11	High relevance	15
4973	GPL WEB_SERVER perl post attempt	web-application-attack	2101979	YES	8	High relevance	14.2
4992	GPL EXPLOIT rsyncd module list access	misc-activity	2102047	YES	4	Low relevance	2.83
5007	GPL SMTP AUTH LOGON brute force attempt	suspicious-login	2102275	YES	22	Low relevance	2.89
5050	GPL P2P eDonkey server response	policy-violation	2102587	YES	3	Low relevance	2.83
5057	GPL SQL TO_CHAR buffer overflow attempt	attempted-user	2102699	YES	2	Low relevance	2.89
5168	ET 3CORESec Poor Reputation IP UDP group 20	misc-attack	2525039	YES	12	Low relevance	2.87

5.2. High Relevance Alerts

An SMS is generated from alert server whenever a high relevance alert is encountered. This SMS notification is in the form of text file that is generated in real time and sent to a mobile phone

hence date and time of attack is known however, the attack type is not indicated in the notification. Table 6shows all the high relevance alerts.

Table 6. High relevance alerts

Idsystem attacks	Message	Classtype	Sid	is_active	Attack Value	Alert names	Alert relevance
4937	GPL ATTACK_RESPONSE id check returned apache	bad-unknown	2101886	YES	11	High relevance	15
4973	GPL WEB_SERVER perl post attempt	web-application-attack	2101979	YES	8	High relevance	14.2

5.3. Medium Relevance Alerts

Medium relevance alerts do not trigger real time SMS alerts but they are blocked by the firewall. These attacks are the possible results of a prerequisite and consequence that are required for a later attack to be successful. They are seen as incidents that build on each other to generate the possible network of attacks. The alerts of this level are not discarded but maintained in a database for further examination of their behaviour. This medium relevance alert is also shown in Table 7.

Table 7. Medium relevance alerts

Id system Attacks	Message	Class type	Sid	is_active	Attack Value	Alert names	Alert Relevance
4693	ET TROJAN Win32/Pterodo Activity (POST)	trojan-activity	2035220	YES	1	Medium relevance	8.5

5.4. Low Relevance Alerts

The low level alerts do not trigger real time alert notification via SMS. These are known malicious acts such as normal probing and other low risk activities such as misc-attack, not-suspicious and misc-activity on the network. These alerts are discarded and regular preventive measures such as the use of antivirus, security monitoring and employee training on cyber security done. This information is also shown in Table 8.

Table 8. Low relevance alerts

Idsystem Attacks	Message	Classtype	Sid	is_active	Attack value	Alert Names	Alert relevance
4697	GPL CHAT Jabber/Google Talk Outgoing Traffic	not-suspicious	2100230	YES	6	Low relevance	2.89
4992	GPL EXPLOIT	misc-activity	2102047	YES	4	Low relevance	2.83

	rsyncd module list access						
5007	GPL SMTP AUTH LOGON brute force attempt	suspicious-login	2102275	YES	22	Low relevance	2.89
5050	GPL P2P eDonkey server response	policy-violation	2102587	YES	3	Low relevance	2.83
5057	GPL SQL TO_CHAR buffer overflow attempt	attempted-user	2102699	YES	2	Low relevance	2.89
5168	ET 3CORESec Poor Reputation IP UDP group 20	misc-attack	2525039	YES	12	Low relevance	2.87

5.5. Short Message Services

A gateway is configured using PHP programming language to echo the username, API key, recipient's number, charges and the message such as "Caution high level intrusion happening" is generated whenever a high relevance alert occur. In the case of low relevance no message is sent because the threshold level is not met. If an error occurs within the network, an error message such as "encountered an error while sending" is generated.

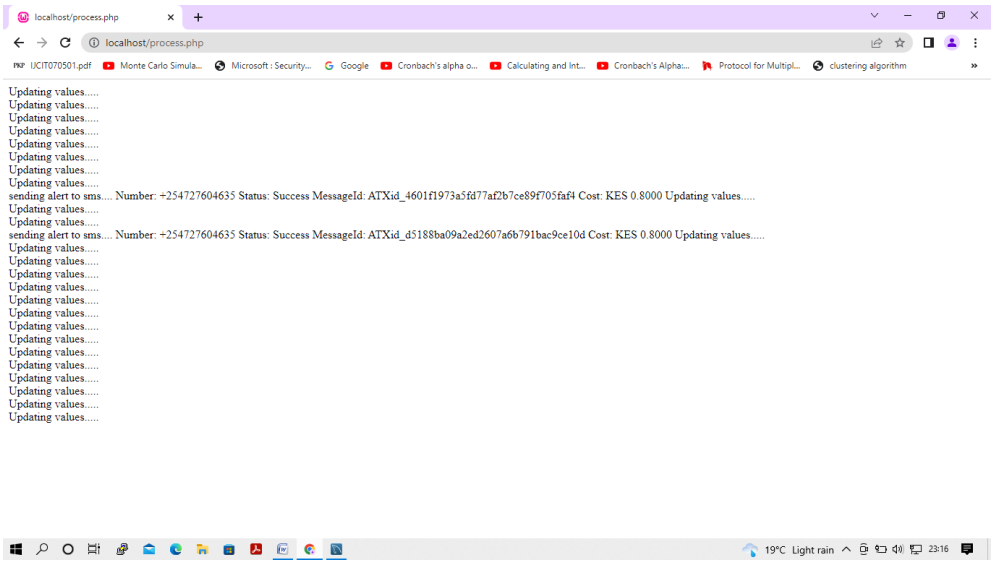

Figure 3.Showing short message services

6. DISCUSSION

The implication of this result is that since the number of alerts were minimised through alert grouping less time was taken to analyse them and take appropriate action. This was proven by the 25 different alert types generated. This implied that with less number of alerts to be analysed, the network activities became easier to monitor and secure. This in turn resulted to secured systems data. A real time network monitoring also contributed to faster incidence response which improved network security. This is evidenced by a real time Short Message Service intervention that generated a warning whenever there was a high impact attack on the target system. This enhanced network throughput due to early detection of intrusion before they escalated over the network.

7. CONCLUSIONS, LIMITATIONS AND FUTURE WORK

From the results of the research conducted, we conclude that the proposed alert correlation technique and the similarity approach were able to establish a relationship between types of alerts. This was achieved through the following alert analysis. Different classes but with similar scores were aggregated together into low level, medium level or high level relevance. Low level alert relevance was discarded. From the trials we conducted with IDS alert dataset of 1000, found 25 different types of alerts after merging. We then obtained 3 classes for mapping alerts based on relevance score similarity approach after aggregation. And from the similarity measurement of a sample size of 9 alerts, there were2 alerts with high relevance, 1 with medium relevance and 6 with low relevance which were discarded. Medium relevance and high relevance were taken to the next stage for further processing.

A real-time SMS alert is sent for high priority intrusions but lacks the ability to stop an attack from escalating. Human intervention is required to prevent the attack. The alert sent does not show the type of attack or data trail before that attack.

In future, we will focus on an additional reactive prevention response by advancing the technique to stop the intrusion wherever a high relevance alert is detected. Instead of just issuing a real time alert and waiting for human intervention we will focus on the use of bots for intervention. This could involve blocking the source address of the attacker, restarting a server or service or closing connections or ports, and resetting TCP sessions without human intervention which is not the case with the present security.

REFERENCES

[1] R. Muwardi, H. Gao, H. U. Ghifarsyam, and M. Yunita, “Network Security Monitoring System Via Notification Alert,” vol. 1, no. 2, pp. 113–122, 2021.

[2] T. H. Nguyen, J. Luo, and H. W. Njogu, “An Approach to Verify , Identify and Prioritize IDS Alerts,” vol. 7, no. 6, pp. 395–410, 2014.

[3] C. T. Kawakani, M. Cukier, B. B. Zarpelão, and R. S. Miani, “Intrusion Alert Correlation to Support Security Management Categories and Subject Descriptors,” pp. 313–320, 2016.

[4] S. Wang, J. Tang, and H. Liu, “Encyclopedia of Machine Learning and Data Science,” *Encycl. Mach. Learn. Data Sci.*, no. January, 2020, doi: 10.1007/978-1-4899-7502-7.

[5] W. Alhakami, “Alerts Clustering for Intrusion Detection Systems : Overview and Machine Learning Perspectives,” vol. 10, no. 5, 2019.

[6] A. B. Palekar and S. S. Dhande, “Study of Alert Correlation Technique,” vol. 6, no. 3, pp. 640–644, 2017, doi: 10.17148/IJARCCE.2017.63150.

[7] D. P. Hostiadi and R. R. Huizen, “A New Alert Correlation Model Based On Similarity Approach,” *2019 1st Int. Conf. Cybern. Intell. Syst.*, vol. 1, no. August, pp. 133–137, 2019.

[8] W. Li and S. Tian, “Expert Systems with Applications An ontology-based intrusion alerts

correlation system," *Expert Syst. Appl.*, vol. 37, no. 10, pp. 7138–7146, 2010, doi: 10.1016/j.eswa.2010.03.068.

[9] S. Haas and M. Fischer, "On the alert correlation process for the detection of multi-step attacks and a graph-based realization," *ACM SIGAPP Appl. Comput. Rev.*, vol. 19, no. 1, pp. 5–19, 2019, doi: 10.1145/3325061.3325062.

[10] K. Granularity, "entropy An E ffi cient Alert Aggregation Method Based on Conditional Rough Entropy and," pp. 1–23, 2020.

[11] S. A. Mirheidari, S. Arshad, and R. Jalili, "Alert Correlation Algorithms : A Survey and Taxonomy," pp. 183–197, 2013.

[12] K. Zhang, F. Zhao, S. Luo, Y. Xin, and H. Zhu, "An Intrusion Action-Based IDS Alert Correlation Analysis and Prediction Framework," *IEEE Access*, vol. 7, pp. 150540–150551, 2019, doi: 10.1109/ACCESS.2019.2946261.

[13] Q. Ohuw, R. Ojrulwkp, D. Rq, O. Y. B. Dqol, K. Frp, and T. T. Frp, "$q $ohuw &ruuhodwlrq $ojrulwkp %dvhg rq wkh 6htxhqfh 3dwwhuq 0lqlqj".

[14] X. Chu, I. F. Ilyas, S. Krishnan, and J. Wang, "Data cleaning: Overview and emerging challenges," *Proc. ACM SIGMOD Int. Conf. Manag. Data*, vol. 26-June-20, pp. 2201–2206, 2016, doi: 10.1145/2882903.2912574.

[15] T. S. Nanjundeswaraswamy and S. Divakar, "Determination of Sample Size and Sampling Methods in Applied Research," *Proc. Eng. Sci.*, vol. 3, no. 1, pp. 25–32, 2021, doi: 10.24874/pes03.01.003.

[16] A. M. Adam, "Sample Size Determination in Survey Research," *J. Sci. Res. Reports*, no. June, pp. 90–97, 2020, doi: 10.9734/jsrr/2020/v26i530263.

[17] J. Kiruki, G. Muketha, and K. Gabriel, "Metrics For Evaluating Alerts In Intrusion Detection Systems," *IJNSA*, vol. Vol.15, no. No.1, 2023.

[18] E. M. Chakir, M. Moughit, and Y. I. Khamlichi, "Risk assessment and alert prioritization for intrusion detection systems," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 10542 LNCS, pp. 641–655, 2017, doi: 10.1007/978-3-319-68179-5_56.

[19] H. Sallay and S. Bourouis, "Intrusion detection alert management for high-speed networks: Current researches and applications," *Secur. Commun. Networks*, vol. 8, no. 18, pp. 4362–4372, 2015, doi: 10.1002/sec.1366.

[20] N. H. Al-A'araji, S. O. Al-Mamory, and A. H. Al-Shakarchi, "Classification and Clustering Based Ensemble Techniques for Intrusion Detection Systems: A Survey," *J. Phys. Conf. Ser.*, vol. 1818, no. 1, 2021, doi: 10.1088/1742-6596/1818/1/012106.

[21] T. A. Alhaj, M. M. Siraj, A. Zainal, H. T. Elshoush, and F. Elhaj, "Feature selection using information gain for improved structural-based alert correlation," *PLoS One*, vol. 11, no. 11, pp. 1–18, 2016, doi: 10.1371/journal.pone.0166017.