A New Alert Correlation Framework Based on Entropy

Mohammad GhasemiGol
Department of Computer Engineering
Ferdowsi University of Mashhad
Mashhad, Iran
ghasemigol@wali.um.ac.ir

Abbas Ghaemi-Bafghi
Department of Computer Engineering
Ferdowsi University of Mashhad
Mashhad, Iran
ghaemib@um.ac.ir

***Abstract*— With the development of computer networks, security devices produce a large volume of low-level alerts. Analysis and management of these intrusion alerts is troublesome and time consuming task for network supervisors and intrusion response systems. The alert correlation methods find similarity and causality relationships between raw alerts to reduce alert redundancy, intelligently correlate security alerts and detect attack strategies. Several different approaches for alert correlation have been proposed which are desired for detecting known attack scenarios. This paper presents a new alert correlation framework without using predefined knowledge. For this purpose, we define the concept of partial entropy for each alert to find the alert clusters with the same information. Then we represent the alert clusters by intelligible notation called hyper-alert. Finally a subset of hyper-alerts is selected based on the entropy maximization. The results of experiments clearly show the efficiency of the proposed framework. We achieved the promising reduction ratio of 99.83% in LLS_DDOS_1.0 attack scenario in DARPA2000 dataset while the constructed hyper-alerts have the enough information to discover the attack scenario.**

Keywords-intrusion detection; alert correlation; entropy; hierarchical clustering method.

I. INTRODUCTION

In recent years, the security threats such as worms and distributed denial of service (DDoS) attacks are increasing. To protect networks and hosts on the Internet, many security devices such as intrusion detection systems (IDSs) are widely deployed. There are two kinds of IDSs: signature based and anomaly based. The former uses a database of known attack signatures for detection while the latter uses a model of normal behaviors. If an intrusion is detected, an IDS generates a warning known as alert or alarm [1]. IDSs could generate overwhelming number of alerts per day, among which false alerts are mixed with true ones. Analysis and management of these intrusion alerts is troublesome and time consuming task for network supervisors or intrusion response frameworks. To improve the representation of security threats, alert correlation is a necessary and critical process in intrusion detection and response [2].

Alert correlation is defined as a process that contains multiple components with the purpose of analyzing alert and providing high-level insight view on the security state of the network under surveillance. Researches on alert correlation are classified to the following techniques [3]:

- Alert Correlation Based on Feature Similarity
- Alert Correlation Based on Known Scenarios
- Alert Correlation Based on Prerequisite and Consequence Relationship

The similarity correlation methods correlate alerts based on the similarities of some selected features, such as source IP addresses, destination IP addresses, protocols, and port numbers. Alerts with higher degree of overall feature similarity will be correlated. The common weakness of these approaches is that they cannot fully discover the causal relationships between related alerts [3].

The Known Scenarios correlation methods correlates alerts based on the known attack scenarios. An attack scenario is either specified by an attack language such as STATL [4] or LAMDBA [5], or learned from training data sets using data mining approach [6]. Whenever a new alert is received it is compared with the current existing scenarios and then added to the most likely candidate scenario [7]. A common weakness of the scenarios correlation techniques is that they are all restricted to known situation. In other words, the scenarios have to be predefined by a human expert or learned from labeled training examples [3].

The Prerequisite and Consequence Relationship Alert Correlation is based on the assumption that most alerts are not isolated, but related to different stages of attacks, with the early stages preparing for the later ones. Intuitively, the prerequisite of an attack is the necessary condition to launch an attack successfully, and the consequence of an attack is the possible outcome if an attack succeeds [8]. This kind of approach requires specific knowledge about the attacks in order to identify their prerequisites and consequences. Based on this information, alerts are considered to be correlated by matching the consequences of some previous alerts and the prerequisites of later ones [9]. However, the major limitation of this class of approaches is that they cannot correlate unknown attacks since its prerequisites and consequences are not defined. Even for known attacks, it is difficult to define all prerequisites and all of their possible consequences [3].

This paper proposes a new similarity correlation framework based on entropy. The main idea of this paper is that the huge number of raw alerts contains some information which can be displayed by fewer hyper-alerts. At first we defined the concept of partial entropy for each alert to find the alert clusters with the same information. Then we represent the alert clusters by intelligible notation called hyper alerts. Finally the hyper alerts are reduced based on their entropy maximization.

In Section 2 some of the related works in alert correlation are reviewed. The detail of proposed correlation framework is presented in Section 3, whilst its

performance in alert correlation is discussed in Section 4. Finally, the conclusions and some suggestions for future work are given in Section 5.

II. Related Works

Here, we review the related work in the literature, which address alert correlating techniques. In the similarity correlation methods alerts are put into a group based on the similarity of their corresponding features. The most common attributes of alerts are Source IP, Destination IP, Source Port, Destination Port, Attack Class and Timestamp. According to Valdes et al. a probabilistic approach to alert correlation correlates attacks over time, over multiple attempts and from multiple sensors. Their used features are based on alert content that anticipates evolving IETF standards. Their probabilistic approach provide a unified mathematical framework for correlating alerts that match closely but not perfectly, where the minimum degree of match required to fuse alerts is controlled by a single configurable parameter. Only features in common are considered in the fusion algorithm. For each feature they define an appropriate similarity function. The overall similarity is weighted by a specifiable expectation of similarity [10].

Julisch proposed a clustering technique for grouping all the alerts which share the same root causes. The clustering technique proposed by Julisch has hierarchy structures which decompose the attributes of the alerts from the most general values to the most specific ones. These generalization hierarchies are later used for measuring the distance between alerts in a clustering algorithm [11]. We use the Julisch's generalization hierarchies to represent the hyper-alerts in our correlation method. Siraj et al. proposed a hybrid clustering model based on Improved Unit Range (IUR), Principal Component Analysis (PCA) and unsupervised learning algorithm to aggregate similar alerts and to reduce the number of alerts [12]. Perdisci et al. proposed a new on-line alarm clustering framework to introduce a concise view about attacks and to reduce the volume of alarms. Their proposed framework consist of three main modules, namely an alarm management interface (AMI), an alarm classifier and an alarm clustering module. The AMI receives alarms from multiple IDS and translates them in a standard format. Then, the alarm classifier assigns a class label to the received alarms and sends them to the alarm clustering module, where the alarms are fused to obtain meta-alarms [13].

In the known scenarios correlation methods, whenever a new alert is received it is compared with the current existing scenarios and then added to the most likely candidate scenario. Some of the previous works in this category have used formal models for specifying attack scenarios, like LAMBDA, STATL, ADeLe [4, 5, 14]. However, some correlation research works are based on pre-defined attack scenarios. For example, Dain et al. proposed a real-time alert clustering scheme which fuses the alerts produced by multiple intrusion detection systems into scenarios. In this framework, they use a probabilistic algorithm that a new alert belongs to a given scenario (the scenario constructed by their algorithm does not necessarily indicate malicious behavior). Whenever a new alert is received it is compared with current existing scenarios and then assigned to the scenario that yields highest probability score [15].

In the prerequisite and consequence relationship alert correlation, we require specific knowledge about the attacks in order to identify their prerequisites and consequences. Based on this information, alerts are considered to be correlated by matching the consequences of some previous alerts and the prerequisites of later ones. Ning et al. [16] proposed an alert correlation method to identify the prerequisites (e.g., existence of vulnerable services) and the consequences (e.g., discovery of vulnerable services) of each type of attacks and correlate the attacks by matching the consequences of some previous attacks and the prerequisites of some later ones. For example, if a UDP port scan followed by a buffer overflow attack against one of the scanned ports, they can be correlated as the same series of attacks. They introduce the notion of *hyper-alert type*, which is used to represent the prerequisite and consequence of each type of alerts. A hyper-alert type T is a triple (fact, prerequisite, consequence) where *fact* is a set of attribute names, each with an associated domain of values, *prerequisite* is a logical formula whose free variables are all in fact, and *consequence* is a set of logical formulas such that all the free variables in consequence are in fact.

Generally, the scenario-based and prerequisite-consequence methods are limited to a predefined knowledge base, whereas the similarity techniques are capable of correlating alerts that may contribute to unknown attacks. On the other hand, the common weakness of the similarity approaches is that they cannot fully discover the causal relationships between related alerts.

III. The Proposed Alert Correlation Framework

In this section we proposed a new similarity correlation framework based on entropy. The main idea of this work is that the massive number of alerts correlated, so that the correlated alerts have the same quantity of information as the original. Fig. 1 shows the architecture of framework which has the following procedure:

Input: Raw alerts
Output: Correlated alerts (hyper-alerts), Selected hyper-alerts

Step 1: the partial entropy of alerts is calculated for any alert.
Step 2: hierarchical clustering algorithm is used to cluster the raw alerts based on their obtained partial entropy.
Step 3: display the alert clusters by intelligible notation called hyper-alert.
Step 4: the network supervisor select a subset of hyper-alerts with the desired number of elements. He uses the principle of maximum entropy to find a subset of hyper-alerts that contain the most of information about the set of alerts.

We will describe the component of proposed correlator framework in greater detail in the following subsections.

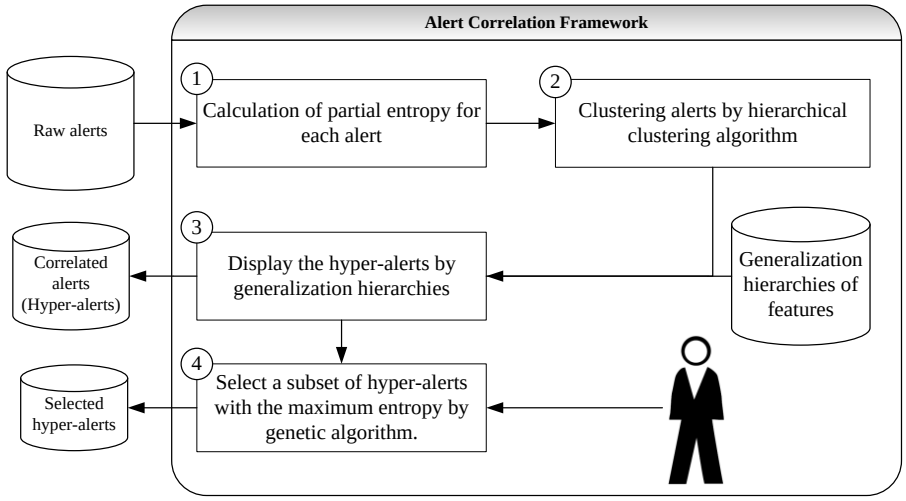

Figure 1: The proposed alert correlation framework

A. *Calculation of Partial entropy for each alert*

We first introduce the concept of entropy, which is a measure of the uncertainty of a random variable. Let X be a discrete random variable with alphabet X and probability mass function $P(x) = \Pr\{X = x\}$, $x \in X$. The entropy of a this random variable with the probability mass function $P(x)$ is defined as [17]:

$$H(X) = -\sum_{x \in X} P(x) \log_2 P(x) \quad (1)$$

According to this equation each value $x \in X$ has its portion in obtained entropy. We named each of these portions as partial entropy. The formal definition of partial entropy is given below.

Definition 1. Consider a discrete random variable with alphabet X and probability mass function $P(x)$. Let $H(X)$ be its entropy. The partial entropy of X is the portion of each value $x \in X$ in $H(X)$, it can be written as:

$$H_P(X = x) = -P(x) \log_2 P(x) \quad (2)$$

Now we use this concept as a similarity measure for alert correlation process. Suppose that the set of alerts is defined as $\psi = \{A_1, A_2, \ldots, A_n\}$, and the set of alert features such as source IP address, destination IP address, protocol, source port number, destination port number, time, duration and etc., is shown by $F = [F_1, F_2, \ldots, F_k]$. Each feature F_j is a discrete random variable with the set of value $\{f_j\}$ and the probability mass function of $P_j(f_j)$. Hence, we can calculate the entropy of feature F_j as the following equation:

$$H(F_j) = -\sum_{f_j \in F_j} P_j(f_j) \log_2 P_j(f_j) \quad (3)$$

According to Eq. 2, the partial entropy of feature F_j for $f_j \in F_j$ is defined as:

$$H_P(F_j = f_j) = -P_j(f_j) \log_2 P_j(f_j) \quad (4)$$

Definition 2. Suppose that we have the set of alerts $\psi = \{A_1, A_2, \ldots, A_n\}$, and the set of alert features is shown by $F = [F_1, F_2, \ldots, F_k]$. For each alert $A_i = [f_{i1}, f_{i2}, \ldots, f_{ik}]$ its partial entropy is a vector which is defined as:

$$H_P(\psi = A_i) = H_P([F_1, F_2, \ldots, F_k] = [f_{i1}, f_{i2}, \ldots, f_{ik}]) \quad (5)$$
$$= [H_P(F_1 = f_{i1}), H_P(F_2 = f_{i2}), \ldots, H_P(F_k = f_{ik})]$$

Whereas $f_{ij} \in F_j$ and,

$$H_P(F_j = f_{ij}) = -P_j(f_{ij}) \log_2 P_j(f_{ij}) \quad (6)$$

So we can calculate the partial entropy of alerts and fill the following matrix from the set of alerts:

$$\begin{array}{c} \\ A_1 \\ A_2 \\ \vdots \\ A_n \end{array} \begin{bmatrix} H_P(F_1 = f_{11}) & H_P(F_2 = f_{12}) & \cdots & H_P(F_k = f_{1k}) \\ H_P(F_1 = f_{21}) & H_P(F_2 = f_{22}) & \cdots & H_P(F_k = f_{2k}) \\ \vdots & \vdots & \cdots & \vdots \\ H_P(F_1 = f_{n1}) & H_P(F_2 = f_{n2}) & \cdots & H_P(F_k = f_{nk}) \end{bmatrix}$$

The alerts with the same information have the similar partial entropy. Hence we find the unique rows of above matrix to construct the hyper-alerts. We can also reduce the number of hyper-alerts by a simple clustering algorithm.

B. *Review of hierarchical clustering method*

After calculating the mentioned matrix, we need a method to cluster the alerts based on their partial entropy. Hierarchical clustering procedures are the most commonly used method of summarizing data structure. A hierarchical tree is a nested set of partitions represented by a tree diagram or dendrogram. Sectioning a tree at a particular level produces a partition into g disjoint groups. If two groups are chosen from different partitions (the results of partitioning at different levels) then either the groups are disjoint or one group wholly contains the other [18]. The hierarchical algorithm contained the following procedure, where c is the desired number of final clusters. If c=1 then the dendrogram could be created.

Algorithm (Agglomerative hierarchical clustering)

Inputs: c, $\{x_1, x_2 ... x_n\}$

Outputs: $\{D_1, D_2 ..., D_c\}$

Begin

initialize $\hat{c} \leftarrow n$

for $i = 1, ..., n$

$D_i \leftarrow \{x_i\}$

do $\hat{c} \leftarrow \hat{c} - 1$

Find nearest clusters according to single-link or complete-link methods, say, D_i *and* D_j

Merge D_i *and* D_j

until $c = \hat{c}$

return c clusters of $\{D_1, D_2 ..., D_c\}$

end

By this technique we will draw cluster's dendrogram and use it to specify clusters. So at first, dissimilarity matrix is created. This matrix shows distances between each pair of samples. Suppose that at the beginning, every sample is a cluster with one sample. Then in each step two clusters that are closer to each other get selected and joined as a new cluster. At the end, we have a nested set of clusters that can be analyzed.

In the hierarchical method we use several mechanisms to obtain the distance of two clusters. One of which is single-link method. In this method the distance between two clusters is defined as the distance between their closest members of two clusters. In other words the distance between two groups, *A* and *B*, is defined as:

$$d_{AB} = \min_{i \in A, j \in B} (d_{ij}) \quad (7)$$

Another mechanism is complete-link. In this method the distance of two clusters is defined as the distance between their furthest members of two clusters, i.e., the distance between two groups, A and B, is:

$$d_{AB} = \max_{i \in A, j \in B} (d_{ij}) \quad (8)$$

In this method, we make sure that other samples of two clusters are closer than the distance between of them.

C. Display the hyper-alerts

In this section we want to display the alerts located in a cluster by intelligible notation called hyper-alert. For this reason we define a meaningful hierarchy for each alert' feature. A hierarchy defines a sequence of mappings from a set of concepts to their higher-level correspondences [19]. A good example of this technique proposed by Pietraszek [20] as generalization hierarchies. He labeled the IP addresses according to their role (Workstation, Firewall, HTTPServer, etc.), then grouped according to their network location (Intranet, DMZ, Internet, etc.) with a final top-level generalized address AnyIP (see Fig. 2). When these classification hierarchies are not known, the IP addresses can be generalized according to the hierarchies in the addressing structure; For example, an IP address 195.176.20.45 can be generalized to the corresponding class C network: 195.176.20.0/24, followed by the class-B generalization 195.176.0.0/16, class-A generalization 195.0.0.0/8 and finally AnyIP.

Furthermore, the other attributes will have different generalization hierarchies, depending on the type and our interests. For example, the source and destination ports of port-oriented IP connections can be generalized into Privileged (0-1023) and Non-Privileged (1024-65535), with a top-level category of AnyPort. In addition, the well-known destination ports (0-1023) can comprise a number of hierarchies describing their function, e.g., httpPorts (80, 443, 8080, 9090), mailPorts (25, 110, 143, 993, 995), chatPorts (194, 258, 531, 994). By this generalization hierarchy, the hyper-alerts can be constructed from the set of alert clusters.

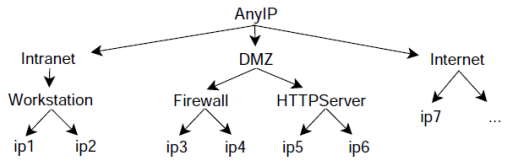

Figure 2: A Sample generalization hierarchies for IP address.

D. Select the best hyper-alerts with the maximum entropy

Now, we want to reduce the displayed hyper-alerts by selecting the specified number of hyper-alert that contain the most of information about the set of alerts. According to the principle of maximum entropy, when estimating the probability distribution, you should select that distribution which leaves you the largest remaining uncertainty consistant with your constraints.

Here we want to estimate the hyper-alerts probability distribution. So we should select a subset of hyper-alerts with the maximum entropy subject to the constraint that the number of desired hyper-alerts is fixed. In other words, we have the following optimization problem:

$$\max \quad Entropy(hyperalerts) \quad (9)$$

$$s.t. \quad number\ of\ desired\ hyperalerts = x$$

We can use the genetic algorithm to solve the above optimization problem. But before that the entropy of hyper-alerts should be defined.

Definition 3. Suppose that the set of hyper-alerts is defined as $\hbar = \{h_1, h_2, ..., h_l\}$, and the set of hyper-alert features such as source IP address, destination IP address, protocol, source port number, destination port number, time interval, duration, contained number of alerts and etc., is shown by $\Gamma = \{\Gamma_1, \Gamma_2, ..., \Gamma_m\}$. So each hyper-alert h_i is displayed by a vector with m different features ($h_i = [\gamma_{i1}, \gamma_{i2}, ..., \gamma_{im}]$). Now suppose that each feature Γ_j is a discrete random variable on the set $\gamma_j = \{v_{1j}, v_{2j}, ..., v_{pj}\} \cup \{g_{1j}, g_{2j}, ..., g_{qj}\}$ so that $\{v_{1j}, v_{2j}, ..., v_{pj}\}$ is the non-generalized value set and $\{g_{1j}, g_{2j}, ..., g_{qj}\}$ is the generalized one for feature Γ_j. First we should find the generalized value for $\{v_{1j}, v_{2j}, ..., v_{pj}\}$ to

calculate the probability of each value of Γ_j with more accurately. Hence, Γ_j is distributed on the set of $\gamma_j = \{v_{1j}, v_{2j}, ..., v_{pj}\} \cup \{g(v_{1j}), g(v_{2j}), ..., g(v_{pj})\} \cup \{g_{1j}, g_{2j}, ..., g_{qj}\}$. where $g(v_{lj})$ is the generalized value for v_{lj}. Now we can calculate the entropy of Γ_j as the following equation:

$$H(\Gamma_j) = -\sum_{\gamma_j \in \Gamma_j} P(\gamma_j) \log_2 P(\gamma_j) \quad (10)$$

IV. EXPERIMENTS

In this section, we test the proposed entropy based correlator by using DARPA2000 [21] dataset to demonstrate how it works. Even after thirteen years of its generation, it has been used in many papers since it was introduced and it is the only choice to compare alert correlation methods [3, 22-25].

A. *DARPA2000 dataset*

DARPA 2000 is a well-known IDS evaluation dataset created by MIT Lincoln Laboratory. There are two attack scenarios in DARPA2000 dataset, LLS_DDOS_1.0 and LLDOS2.0.2. In the both scenarios, the attacker tries to install components necessary to run a Distributed Denial of Service, and then launch a DDoS at a US government site. The main difference between them is that the attacker uses IPSweep and Sadmind Ping to find out the vulnerable hosts in LLS_DDOS_1.0 while DNS HInfo is used in LLDOS2.0.2; second, the attacker attacks each host individually in LLS_DDOS_1.0, while in LLDOS2.0.2, the attacker breaks into one host first and then fans out from it.

In this paper we only show the results of evaluation on LLS_DDOS_1.0. In this scenario, the attacker first sends ICMP echo-requests to many IP addresses and listens for ICMP echo-replies to determine which hosts are "up", then uses the "ping" option of the sadmind exploit program to determine which of the discovered hosts are running the sadmind service. In the next phase, the attacker tries to break into the hosts found to be running the sadmind service in the previous phase, and launches the sadmind Remote-to-Root exploit several times against each host, each time with different parameters. After gaining root access in each host, the attacker uses telnet, rcp and rsh to install a DDoS program in the compromised machines.

Here we have performed the experiments, with the DMZ network traffic of LLS_DDOS_1.0 that contains 34819 alerts which is indicated the five steps of DDoS attack on the target IP address 131.84.1.31. The mentioned alert correlation framework is applied to this set of alerts. The results in Table I show the promising reduction ratio of 99.83% in LLS_DDOS_1.0 attack scenario before running Step 4, whereas Sadoddin [26] achieved the reduction ratio of 96% in his experiments. Moreover the obtained 58 hyper-alerts cover the general information existed in each of the five phases of attack scenario and provides a more global view of what is happening in the network. The network supervisor can also select a subset of hyper-alerts with the desired number of elements by running Step 4.

TABLE I: THE RESULT OF PROPOSED ALERT CORRELATION FRAMEWORK ON LLS_DDOS 1.0 ALERTS.

	Number of raw alerts	Number of hyper-alerts in each phase (before running Step 4)	Reduction Ratio (%)	Number of hyper-alerts in each phase (after running Step 4 to find the 10 tops of hyper-alerts)
Phase1	785	16 (1 of them is shared between Phase1 & Phase2)	98.09	2
Phase2	25	15 (1 of them is shared between Phase1 & Phase2) (2 of them are shared between Phase2 & Phase4) (4 of them are shared between Phase2 & Phase3)	44.00	1
Phase3	80	13 (2 of them are shared between Phase3 & Phase4) (4 of them are shared between Phase2 & Phase3)	86.25	4
Phase4	19	13 (1 of them is shared between Phase4 & Phase5) (2 of them are shared between Phase3 & Phase4) (2 of them are shared between Phase2 & Phase4)	31.57	2
Phase5	33910	11 (1 of them is shared between Phase4 & Phase5)	99.97	1
The total of raw alerts	**34819**	**58**	**99.83**	**10**

TABLE II: THE RESULT OF SELECTED HYPER-ALERTS WITH MAXIMUM ENTROPY FOR ATTACK SCENARIO LLS_DDOS_1.0 IN DARPA 2000 DATASET

Number of alerts	Duration	UTime	LTime	Fr	Dport	Fr	Sport	Fr	Protocol	Fr	Destination IP	Fr	SourceIP
391	0	' 9:51:40 AM'	' 9:51:36 AM'	391	-1	391	-1	391	icmp-echo-request	233 158	Zone3 Zone4	391	202.77.162.213
2	0	' 9:51:38 AM'	' 9:51:38 AM'	2	-1	2	-1	2	icmp-echo-reply	2	202.77.162.213	1 1	Zone3 Zone4
4	0	'10:15:09 AM'	'10:11:09 AM'	4	111	4	privileged	4	udp	1 3	Zone2 Zone3	4	202.77.162.213
2	0	'10:50:38 AM'	'10:50:38 AM'	2	514	2	1023	2	tcp	2	202.77.162.213	2	172.16.112.50
3	9.26E-05	'10:34:31 AM'	'10:34:14 AM'	3	23	3	registered	3	tcp	3	172.16.114.30	3	202.77.162.213
6	0	'10:33:09 AM'	'10:33:09 AM'	6	dynamic	6	privileged	6	udp	6	172.16.115.20	6	202.77.162.213
2	9.26E-05	'10:33:57 AM'	'10:33:57 AM'	2	23	2	registered	2	tcp	2	172.16.114.20	2	202.77.162.213
4	0	'10:50:00 AM'	'10:50:00 AM'	4	514	4	1023	4	tcp	4	202.77.162.213	4	172.16.115.20
1	0	'10:50:07 AM'	'10:50:07 AM'	1	1022	1	1023	1	tcp	1	202.77.162.213	1	172.16.115.20
75	0	'11:27:55 AM'	'11:27:55 AM'	1 74	invalid privileged	75	registered	75	tcp	75	131.84.1.31	75	Outside

To display the hyper-alert, we use a vector with 14 features (Source IP address, Frequency of source IP address, Destination IP address, Frequency of destination IP address, Protocol, Frequency of protocol, Source port number, Frequency of source port number, Destination port number, Frequency of destination port number, Time interval (Lower bound of time), Time interval (Upper bound of time), Duration and Contained number of alerts). According to the network topology used to capture the DARPA dataset, we suppose the following generalization hierarchies for some of the mentioned features.

- The IP addresses can be generalized into Zone1, Zone2, Zone3, Zone4, and outside.
- The source and destination ports of port oriented IP connections can be generalized into privileged (1-1024), registered (1025-49151), and dynamic (49152-65535).

The full list of generated hyper-alerts for LLS_DDOS_1.0 is shown in Table II.

V. CONCLUSION

This paper presents a new alert correlation framework based on entropy. The main idea of the proposed framework is to correlate the raw alerts, so that the correlated alerts have the same quantity of information as the original. For this purpose, we defined the concept of partial entropy for each of generated alerts. The alerts with the similar partial entropy indicate the same information, hence we can correlate them into a specific cluster and report them by an intelligible hyper-alert which is provide a more global view of network status. By principle of entropy maximization the network supervisor can also select a subset of hyper-alerts that has more information about the alerts. We validated the framework on attack scenario LLS_DDOS_1.0 in DARPA 2000 dataset. The reduction ratio with the experiments was 99.83% while the generated hyper-alerts have the enough information to discover the attack scenario.

REFERENCES

[1] H. W. Njogu, L. Jiawei, J. N. Kiere *et al.*, "A comprehensive vulnerability based alert management approach for large networks," *Future Generation Computer Systems*, vol. 29, pp. 27-45, 2013.

[2] D. Xu, and P. Ning, "Privacy-Preserving Alert Correlation: A Concept Hierarchy Based Approach," in The 21st Annual Computer Security Applications Conference 2005, pp. 537-546.

[3] A. A. Ghorbani, W. Lu, and M. Tavallaee, *Network Intrusion Detection and Prevention Concepts and Techniques*: Springer, 2009.

[4] S. T. Eckmann, G. Vigna, and R. A. Kemmerer, "STATL: An Attack Language for State-based Intrusion Detection," in 1st ACM Workshop on Intrusion Detection Systems, Athens, Greece, 2000.

[5] F. Cuppens, and R. Ortalo, "A language to model a database for detection of attacks," in Recent Advances in Intrusion Detection, Toulouse, France, 2000.

[6] G. Xiang, X. Dong, and G. Yu, "Correlating Alerts with a Data Mining Based Approach," in The 2005 IEEE International Conference on e-Technology, e-Commerce and e-Service, 2005.

[7] S. O. AL-MAMORY, and H. L. ZHANG, "A Survey on IDS Alerts Processing Techniques," in 6th WSEAS International Conference on Information Security and Privacy, 2007.

[8] A. E. E. Taha, "Intrusion Detection Correlation in Computer Network Using Multi-Agent System," Ain Shams University, 2011.

[9] P. Ning, D. S. Reeves, and Y. Cui, *Correlating Alerts Using Prerequisites of Intrusions*, North Carolina State University, Department of Computer Science, 2001.

[10] A. Valdes, and K. Skinner, "Probabilistic Alert Correlation," in RAID, 2001, pp. 54-68.

[11] K. JULISCH, "Clustering Intrusion Detection Alarms to Support Root Cause Analysis," *ACM Transactions on Information and System Security*, vol. 6, no. 4, pp. 443-471, 2003.

[12] M. M. Siraj, M. A. Maarof, and S. Z. M. Hashim, "A Hybrid Intelligent Approach for Automated Alert Clustering and Filtering in Intrusion Alert Analysis," *International Journal of Computer Theory and Engineering*, vol. 1, no. 5, pp. 539-545, 2009.

[13] G. G. Roberto Perdisci, Fabio Roli, "Alarm clustering for intrusion detection systems in computer networks," *Engineering Applications of Artificial Intelligence*, vol. 19, pp. 429–438, 2006.

[14] B. V. Eric Totel, Ludovic Mé, "A language driven ids for event and alert correlation," in SEC, 2004, pp. 209-224.

[15] O. Dain, and R. K. Cunningham, "Fusing a Heterogeneous Alert Stream into Scenarios," in ACM Computer and Communications Security, Philadelphia, Pennsylvania, USA, 2001.

[16] P. Ning, and Y. Cui, *An intrusion alert correlator based on prerequisites of intrusions*, 2002.

[17] T. M. COVER, and J. A. THOMAS, *ELEMENTS OF INFORMATION THEORY*, New Jersey: John Wiley & Sons, 2006.

[18] A. R. Webb, *Statistical Pattern Recognition*, New York: Wiley 2002.

[19] S. O. Al-Mamory, and H. Zhang, "Intrusion detection alarms reduction using root cause analysis and clustering," *Computer Communications*, vol. 32, pp. 419–430, 2009.

[20] T. Pietraszek, "Alert classification to reduce false positives in intrusion detection," Institut für Informatik, Albert-Ludwigs-Universität Freiburg, Germany,, 2006.

[21] "Darpa 2000 intrusion detection evaluation datasets," http://www.ll.mit.edu/mission/communications/cyber/CSTcorpora/ideval/data/2000data.html.

[22] S. H. Ahmadinejad, S. Jalili, and M. Abadi, "A hybrid model for correlating alerts of known and unknown attack scenarios and updating attack graphs," *Computer Networks*, vol. 55, pp. 2221–2240, 2011.

[23] F. Xuewei, W. Dongxia, M. Guoqing *et al.*, "Research on the key technology of reconstructing attack scenario based on state machine " in 3rd IEEE International Conference on Computer Science and Information Technology (ICCSIT), 2010, pp. 42-46.

[24] D. Xu, "Correlation Analysis of Intrusion Alerts," North Carolina State University, 2006.

[25] M. Soleimani, and A. A. Ghorbani, "Multi-layer episode filtering for the multi-step attack detection," *Computer Communications*, vol. 35, pp. 1368–1379, 2012.

[26] A. A. G. Reza Sadoddin, "An incremental frequent structure mining framework for real-time alert correlation," *computers & security*, vol. 28, pp. 153-173, 2009.