A Survey on IDS Alerts Processing Techniques

SAFAA O. AL-MAMORY, HONG LI ZHANG
School of Computer Science,
Harbin Institute of technology,
CHINA, HARBIN, 150001
Safaa_vb@yahoo.com, zhl@pact518.hit.edu.cn

Abstract: - When an attacker tries to penetrate the network, there are many defensive systems, including intrusion detection systems (IDSs). Most IDSs are capable of detecting many attacks, but can not provide a clear idea to the analyst because of the huge number of false alerts generated by these systems. This weakness in the IDS has led to the emergence of many methods in which to deal with these alerts, minimize them and highlight the real attacks. It has come to a stage to take a stock of the research results a comprehensive view so that further research in this area will be motivated objectively to fulfill the gaps exists till now.

Key-Words: - Network security, intrusion detection, alert correlation, alert reduction, attacks, scenarios

1 Introduction

After about twenty years of IDS developing, the research results obtained have made the scientific community conclude that further research is needed to fine tune these systems. Large organizations and companies are already setting up different models of IDS from different vendors. The IDSs provide an unmanageable amount of alerts. Inspecting thousands of alerts per day [1] is unfeasible, especially if 99% of them are false positives [2].

Due to this, during the last few years research on IDSs has focused on how to handle alerts. The main objectives of these investigation works are: to reduce the amount of false alerts, study the cause of these false positives, recognize high-level attack scenarios, and finally provide a coherent response to attacks understanding the relationship between different alerts. To achieve good recognition of attacks, the data needs to be collected from various sources like Host IDS, Network IDS, Routers, anti-viruses and others as shown in Fig. 1.

As can be seen in Fig. 1, there are many sources that generate alerts and the IDMEF [3] is the language that standardizes (normalize) these alerts to unified format. Then alert pre-processing techniques are applied to mitigate the influence of false alerts. After that, the resulting alerts are correlated to build attacks scenarios and generate reports for the analyst to prevent completion of attacks (if possible).

In this paper, we will survey the main techniques of the two phases which appear in Fig. 1 that is *Alert Pre-Processing* and *Alert Correlation* and also we will discuss and classify different alert processing methods and algorithms from a theoretical point of view. We have listed many important limitations noted from the literature.

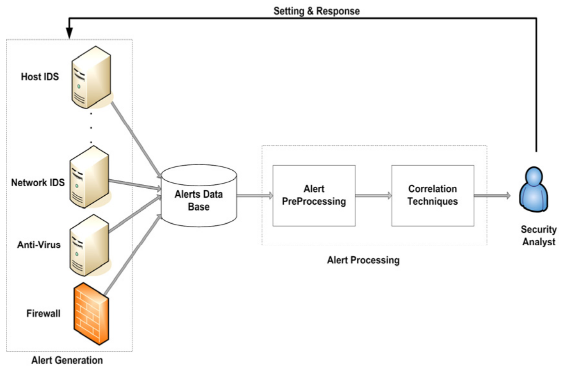

Fig.1 Alert processing position

2 Alert Processing

There are many terms that are usually misunderstood and should be differentiated between them that is *event*, *alert* and *alarm*. An *event* is a low level entity that is analyzed by the IDS, whereas an *alert* is generated by the IDS to notify parties of interesting events. A single event can cause many alerts (that is a problem) especially in a networked IDS environment, and a single alert can describe a set or sequence of events [4]. Every alert is suspicious but an event is not necessarily suspicious. An *alarm* is the user interface mechanism by which a user manages an alert [5].

Alert correlation is defined as a conceptual interpretation of multiple alerts such that new meanings are assigned to these alerts. In other words, correlation methods try to convert low level alerts to high level alarms. While *Attack correlation* is used in a very specific situation: some security experts try to model attacks by building some scenarios [6].

The classification of alert processing techniques is shown in Fig. 2. They can be classified into two main categories: *Alert Pre-Processing* and *Alert Correlation.* In other words, the *Alert Pre-Processing* works in the network layer while the attacker is in the application layer, so we need a process of alerts in the application layer that is *Alert Correlation*. The next two subsections describe these two categories in more details.

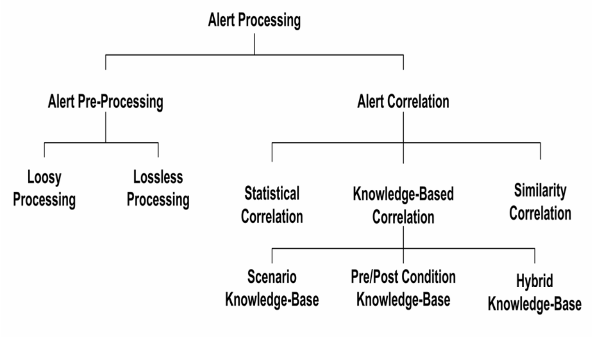

Fig.2 Classification of alert processing techniques

2.1 Alert Pre-Processing

This type of alert processing tries to mitigate the influence of false alerts and to make the next step (i.e. correlation process) more accurate. There are many methods in this class of processing, all of which try to remove the noise from the stream of alerts and make it more meaningful. These methods either loss some information (because it focuses on some events that occurred or not) or don't loss any information (because it uses additional knowledge). It is stated in the following two subsections.

2.1.1 Loosy Pre-Processing

The main techniques of this type of processing are the *alert prioritization* and *alert aggregation*, both of them try to reduce alert flooding and they are always used as components in the systems.

Alert prioritization is performed to assess the relative importance of alerts generated by the sensors. This method has to take into accounts the security policy and the security requirements of the site where the correlation system is deployed [7]. The significance of an alert can depend on many factors such as *Source/Target criticality*, *Attack criticality*, and *Alert confidence*. Therefore, prioritizing of alerts aids in substantial reduction of alert volume [8].

The main purpose of alert prioritization technique is to filter out lower priority alerts from higher priority alerts such that further analysis is not distracted by false positives or non-malicious data. Nonetheless, there are provisions for risks associated with such filtering in cases when carefully crafted less obvious attacks are appraised as a lower priority and ignored from analysis.

Alert Aggregation consists of detecting, from the observation of the alerts received in a given time window, multiple occurrences of the same alert and substituting the corresponding alerts, possibly indicating how many times the alert occurred during the observation period. Aggregation is mainly aimed at mitigating alert flooding. However, the generated alert may not contain information such as arrival time of each alert that was initially known before aggregation.

2.1.2 Lossless Pre-Processing

Sometimes this type of method is called *filters* and it mainly uses rules to filter the alerts. These rules are built either by experts or by automated programs. It is designed to remove the false alerts that make the correlation process inaccurate. In this subsection, we examine three of these methods that is *Alert Verification*, *Root Cause Discovery* and *Machine Learning.*

Alert verification using vulnerability analysis information has been advocated as an important tool to reduce the noise in the alert stream produced by intrusion detection sensors [9]. When the correlation process receives false positives as input, the quality of the results can degrade significantly. Correlating alerts that refer to failed attacks can easily result in the detection of whole attack scenarios that are nonexistent. The idea of alert verification is to differentiate between successful and failed intrusion attempts (both false and non-relevant positives). Identifying failed intrusion attempts allows other correlation components to reduce the influence of these alerts on their decision process [7].

These systems can operate either offline or online. Offline systems periodically perform vulnerability scans and update a database of network assets. This database is then accessed by the correlation system when processing the alerts. If an alert is received and the database indicates that the attacked service is not vulnerable the alert is suppressed. Online systems perform the vulnerability scans as the alerts arrive and do not rely on a database. The information about network assets is usually gathered using Nmap [10] and contains only information that is gathered by this specific tool (e.g., IP addresses, installed operating systems, and open ports).

Unfortunately, most alert verification systems do not support dynamic mechanisms for alert verification. Instead, they rely on information about the security configuration of the protected network that was collected at an earlier time. On the other hand, offline mechanisms have the drawback that they may rely on outdated data. Another drawback is the limitation of information type that can be gathered in advance.

Julisch introduces alarm (i.e., alert) clustering as a method to support *root cause discovery*. The root cause of an alarm is defined as the “reason for which it occurs.” He argues that root causes are primarily responsible for the large number of redundant alarms and 90% of these root causes are generated because of configuration problems and thus are fixable with manual interception. This work outlines a semi-automatic approach for reducing false positives in alarms by identifying the root causes automatically and then writing rules to filtering them. Such measures can drastically reduce future alarm load [11].

In this work, alarm clustering is performed by grouping together alarms whose root causes are generally similar. A generalized alarm for a specific alarm cluster represents a pattern that all of the alerts in the cluster must match in order to belong to that cluster. The alarm clustering algorithm based on an attribute-oriented induction method (AOI), attempts to find alarm clusters where all the alarms share the same root cause [11].

This approach focuses on identifying the root causes for large groups of alarms, which typically correspond to problems in the computing infrastructure that leads to many false positives (with the potential exception of large-scale automated attacks). It does not look for small, stealthy attacks in the alarm logs, but aims to reduce the noise in the raw alarms to make it easier to identify real attacks in the subsequent analysis.

When a filter was written to remove the root cause behind one of the largest identified alarm clusters, 82% of the original alarms were automatically discarded by the filter. This work outlines an effective approach to reduce false positives in sensor alert reports by clustering alerts with abstraction and then using the clusters to discover and understand the root causes of alerts.

This method is used to remove the false alerts generated from misconfigured equipments. However, in the small networks this method is useless because it is easy to configure all equipments in the network. But in the large networks it is a hard task to configure all equipments well so this method will be useful. Also, the written filters should be kept secret because the attacker may use it to evade detection.

The Adaptive Learner for Alert Classification (ALAC) [12] is an adaptive alert classifier based on the feedback of an intrusion detection analyst and machine-learning techniques. The classification of IDS alerts is a difficult machine-learning problem. ALAC was designed to operate in two modes: a recommender mode, in which all alerts are labeled and passed onto the analyst, and an agent mode, in which some alerts are processed automatically. In recommender mode, where it adaptively learns the classification from the analyst, false negative and false positive were obtained. Where in the agent mode, some alerts are autonomously processed (e.g., false positives classified with high confidence are discarded).

In this system, a fast and effective rule learner was used that is RIPPER. It can build a set of rules discriminating between classes (i.e. false and true alerts). Each rule consists of conjunctions of attribute-value comparisons followed by a class label, and if the rule evaluates it a true prediction is made. At the same time, the number of alerts for the analyst to handle has been reduced by more than 50%. This system has a disadvantage that is during a system's lifetime the size of the training set grows infinitely.

2.2 Alert Correlation

Generally speaking, three main categories in the correlation process are distinguished as can be seen in Fig. 2. The first category gathers all approaches that do not require a specific knowledge and there are many tools which currently are available to implement some of these approaches. This category is classified under statistical umbrella. The second category gathers the approaches that require a knowledge base. The last one focuses on similarities between alerts.

From another point of view, we can call the knowledge-based correlation as a *misuse correlation* because this type of correlation matches the alerts with a prior knowledge and search for fixed patterns of alerts (like misuse IDSs). Unlike misuse IDSs that often provide only "late warning" (they report when a system has been compromised), *misuse correlation* respond to attacks before its completion. In the same way, statistical correlation can be called as *anomaly correlation* due to their search for abnormal alerts in the huge number of alerts and it does not use a knowledge base (like anomaly IDS).

The most obvious shortcoming in all of the knowledge based algorithms is that they fail to correlate alerts of previously unknown attacks. In addition, it must be updated frequently to detect new attacks as they are discovered. This has led to development of totally different approaches that correlate alerts using statistical attack scenario analysis. These approaches can be used to find novel attacks. This capability is essential to protect critical hosts because new attacks and attack variants are constantly being developed. The statistical approaches suffer from noise in the results.

There is not a unique solution that is the 'best', in terms of precision and/or complexity, to solve a generic problem of alert correlation. Recent researches indicate a tendency for the adoption of combinations of different approaches for the solution of the problem in complex networks [13].

2.2.1 Statistical Correlation

Pure statistical causality analysis does not need predefined knowledge about attack scenarios, thus completely new attack scenarios can also be recognized. Qin *et al.* [14] present an alert correlation system combining a Bayesian correlation system (naive Bayes) with a statistical correlation system using Granger Causality Test (GCT), a time series-based causal analysis algorithm. Based on the results of this analysis the GCT module constructs a correlation graph. The motivation for statistical analysis is that every multi-step attack generates alerts that have statistical similarities in their attributes, and that attack steps have a causal relationship. In contrast, as the structure of the network is predetermined, the Bayes-based correlation module can discover alerts that have direct "causal" relationships according to domain knowledge.

This approach reports high false-causality rate, which suggests that the system can only be used by very specialized domain experts after manual tuning. As it is today, the statistical causality approach is not a feasible solution for the complete correlation process, but it can be utilized as a part of a larger system to provide meta-alert signatures. Another point is that all attributes of the alerts, particularly, the class name should be used to construct hyper alerts. But this technique can not take into account an anomaly IDS that generates alerts without classification.

2.2.2 Knowledge-Based Correlation

The most common approaches in the field of correlation are the knowledge-based methods. They can be divided into three groups as shown in

Fig. 2. They differ in the required type of knowledge. The following three subsections will present them in some detail.

2.2.2.1 Scenario Knowledge-Base Correlation

This class requires attack scenario knowledge such as the work of Dain *et al.* [15]. They use an alert clustering scheme that fuses alerts into scenarios using a "probabilistic in nature algorithm." In this system, scenarios are developed as they occur, i.e., whenever a new alert is received it is compared with the current existing scenarios and then assigned to the scenario that yields the highest probability score. If the score falls below a threshold, it starts its own scenario. This testing is done in a time proportional to the number of candidate scenarios.

For alert comparison, the new alert is compared with the most recent alert in a scenario [15]. The probability of this is computed as a product based on three factors that is *the strength of link between two alerts*, *the time between alerts* and *the source IP address range of the alerts*. They claim that this technique allows finding scenarios even if the attacker uses stealthy attack methods such as forged source IP addresses and long latencies between attacks. They also allege that combining only a few simple features is sufficient for satisfactory results. The alert data was used to estimate the parameters to be used in the probability estimation so that finding the likelihood of an alert in joining a scenario is optimized. Several data-mining techniques were applied in the system such as multi-layer perceptron and decision tree. The algorithm with decision trees proved to be the best.

Similarly to Valdes *et al.* [16] work, this method maintains a continuously updated list of alert groups called scenarios. The assignment of an alert to the scenario is final and irreversible. However, unlikely, in which the similarity is calculated based on set-valued attributes, in this approach the probability score is a function of a new alert and only the last alert in the existing scenario.

In this method, the authors have shown a methodology of how to learn correlation algorithms using labeled data. The problem however is that labeling alerts and grouping them into scenarios is very labor intensive and unlikely to be done in real environments.

In multi-sensor alert correlation, one of the early research efforts was led by Debar and Wespi [17]. This work concentrates on alert correlation more in terms of discovering structural relationships between alerts. The authors introduced the concept of an *Aggregation and Correlation Component (ACC)* that can analyze and correlate alerts generated by IDSs (i.e. probes) using an expert rule-based system. The algorithm used for correlation in the *ACCs* is a hybrid algorithm that takes into account both the predefined attack scenarios as well as the comparison between available alert attributes. The goal of the *ACCs* is to generate one alert per attack, even if the attack generates multiple alerts and to form groups of alerts by creating a small number of relationships. *ACCs* assign confidence values with each alert by considering the intrinsic inaccuracy of the probe in question. *ACCs* also take into account the severity of alerts during the assessment.

The algorithm takes into account only three alert attributes namely the source, the target, and the attack class. Only selected attributes are compared to form different views of the incident, thereby utilizing a very basic version of the minimum similarity and expected similarity concepts. Further, the algorithm only considers perfect matches of the alert attributes. Due to the absence of any correlation language the correlation rules are either entered manually or generated from the configuration files. Two kinds of pre-programmed correlation rules used by *ACCs* are *duplicates* and *consequences*. The Duplicate rules are employed to combine alerts representing same attack but that are generated from separate sources. The duplicate rule defines the various attack classes (specific names of known attacks) that are to be considered as duplicates and the specific attributes of the alerts that should match. The Consequence rules are employed to combine alerts that are known to occur in pairs in a multi-step attack. The consequence rule defines two attack classes, two probe IDSs, and a wait time representing the maximum time between the two alerts under consideration for correlation.

It is possible, therefore, to program

predefined attack scenarios into the system, but it will still be very tedious and will fail to be generic. Also this approach acknowledges that an alert stream may contain a large number of false positives, but it does not provide any specific technique to eliminate these spurious alerts.

In order to aggregate or correlate alerts, an empirical algorithm is used since it compares different alerts according to the different considered attributes. However, the using of wildcards in the aggregation relationship is of a great interest and could be used to aggregate alerts coming from anomaly IDSs.

2.2.2.2 Pre/Post Conditions Knowledge-Base Correlation.

One recently introduced powerful mechanism of expressing correlation criteria is defining pre/post-conditions for individual attacks. This can be seen as just another way of describing attack scenarios. When stating pre/post-conditions for distinct attacks, there is no need to know complete attack scenarios in advance and so do not have to insert huge amounts of correlation rules manually. It is sufficient to state the required conditions for a known attack and the possible outcomes of that attack [4].

Ning *et al.* [18] propose an alert correlation model based on the inherent observation that most intrusions consist of many stages, with the early stages preparing for the later ones. The correlation model is built upon two aspects of intrusions that are, *Prerequisites* (the necessary conditions for an intrusion to be successful) and *Consequences* (the possible outcomes of an intrusion). With knowledge of prerequisites and consequences, the correlation model can correlate related alerts by finding causal relationships between them, i.e., by matching the consequences of previous alerts with prerequisites of later ones.

Once the correlation model is able to identify time-lined sequences of the hyper alert instances, it can present a correlation chain where earlier alerts are shown to prepare for the later ones. Ning *et al.* used *hyper alert correlation* charts to visually represent the alerts. It is a connected chart where each node represents a hyper alert and the edges connect two hyper alerts if one prepares for the other one. Ning *et al.* [18] claim that the prerequisite-consequence model reveals structures of series of attacks, reduces false alerts, and predicts attacks in progress.

In [19], Ning *et al.* integrate the prerequisite-consequence model of alert correlation with alert clustering based on a match of alert's attribute values. In this respect, the authors correlate alerts to generate correlation charts separately and then integrate two correlation charts together if the charts involve the same destination IP address. The authors reason about missed attacks with the assumption that subsequent attacks in a correlation chain can be considered *directly related* and when any critical attack in the correlation chain is missing, the attacks in the chain are considered *indirectly related*. The authors define and use pre-defined constraints that must be satisfied by attacks to be considered as indirectly related.

The approach proposed by Cuppens and Miège in [20] also uses pre/post-conditions. In addition, it includes a number of phases including alert clustering, alert merging, and intention recognition. In the first two phases, alerts are clustered and merged using a similarity function. The intention recognition phase is referenced in their model, but has not been implemented. An interesting aspect of this approach is the attempt to generate correlation rules automatically. While it may seem appealing, this technique could generate a number of spurious correlation rules that, instead of reducing the number of alerts and increasing the abstraction level of the reports, could introduce the correlation of alerts that are close or similar by pure chance, in this way increasing the noise in the alert stream.

One limitation of the above approaches is that they are heavily dependent on prior knowledge of modeling each attack at the prerequisite and consequence level and therefore does not detect unknown attacks or variations of known attacks. Also, these methods are intuitively appealing and capture significant relations between alerts. However, it requires that prerequisites and consequences are associated with each alert generated by IDS. Since this information is not provided by IDSs vendors and often signatures are not sufficiently documented, such labeling might be a very difficult task. Another problem is the assumption that only attacks that are carried out in multiple steps are important. While it is reasonable to give high priority to alerts that have been detected as part of a multi-step attack, it is not wise to disregard all alerts that are not part of a multi-step attack.

2.2.2.3 Hybrid Knowledge-Base Correlation

This type of correlation methods tries to use most of the available information to leverage correlation reliability. An interesting and active method was proposed by Lingyu *et al.* [21] to correlate alerts and hypothesize the missed ones. The information used in this method is vulnerabilities, their dependencies, and network connectivity. The first step in this method is to build attack graph *AG* from the previous information. Then a new technique, namely queue graph *QG*, was suggested to correlate alerts in real time depending on *AG* and exploits. They show that this method can process alerts faster than an IDS can report them.

Each exploit is realized as a queue of length one, and each security condition as a variable. Every incoming alert is first matched with an exploit and placed in the corresponding queue. During this process, the results of correlation are collected as a directed graph, namely, the *result graph*.

The realization of edges is starting from each exploit, a breadth-first search is performed in the attack graph by following the directed edges. For each edge encountered during the search, a forward pointer is created to connect the corresponding queue and variable. Similarly, another search is performed by following the directed edges in their reversed direction, and a backward pointer is created for each encountered edge. Later, the backward edges were used for correlation purposes and the forward edges were used for prediction purposes [21].

The missing alerts cause inconsistency between the knowledge encoded in *AG*s and the facts represented by received alerts. By reasoning about such inconsistency, missing alerts can be plausibly hypothesized.

A queue graph only keeps in memory the latest alert matching each of the known exploits. The correlation between a new alert and those in-memory alerts is explicitly recorded. However, this method filters out any alerts not matching with existing vulnerabilities which cause to loss alerts that related to unknown and new vulnerabilities. Also, the existence of attack templates and the configuration file could be another vulnerability in itself. If these got into the wrong hands, they would be very valuable tools for the attacker.

2.2.3 Similarity Correlation

The similarity correlation methods try to group the alerts in a meaningful way to conclude the attacks. Three methods are presented in this subsection as examples of this type of correlation.

Probabilistic alert correlation finds similarity between alerts that match closely, if not exactly. According to Valdes *et al.*, probabilistic alert correlation correlates attacks over time, over multiple attempts and from multiple sensors. The alert correlation task consists of [16]: identifying alert threads, identifying incidents by clustering/correlating threaded alerts with meta-alerts and clustering/correlating meta-alerts with meta-alerts.

The probabilistic alert correlation system represents alert groups by means of meta-alerts, supporting set-valued attributes. All attributes of a meta-alert representing a group of alerts are a union of appropriate attributes. Valdes *et al.* [16] introduce the concept of *expectation of similarity* that serves as the normalizing weight of the similarity functions and the concept of *minimal similarity* that serves as the threshold for consideration of similarity. Construction of similarity functions to measure feature similarity is based on combination of expert rule-base and Bayes formalism.

For each new alert, it compares the alert to all existing meta-alerts and calculates similarities between them. The new alert is merged with the most similar meta-alert if the similarity value exceeds a user-defined minimum similarity threshold. Otherwise, it becomes a meta-alert to be considered for future alert correlation. If any feature pair fails to match the minimum criterion of similarity, the alert is excluded from consideration of overall similarity.

The methods of this type are ultimately not able to capture the true relationships between events. Carefully handcrafted similarity criteria (i.e. similarity matrices, similarity expectation values) can lead to well-behaving systems that capture the essence of many previously known attack types but the system in itself does not understand the attacks.

Another drawback of this method is the empirical definition of the similarity matrix. The heuristic technique with which this matrix is fulfilled is biased since we have a priori knowledge about attack classes. It is not completely implicit since we maintain a matrix whose values are fixed a priori between the different attacks. This technique is only based on known attacks. However, the proposed method will obviously fail to aggregate the novel attack that is detected by anomaly IDSs (which do not provide the attack class in the generated alerts content) because there is no entry in the similarity matrix for this new attack.

The work of Lee [22] is an expansion of Valdes *et al.* [16] work. It differs in many folds. Firstly, instead of correlating alerts, they correlate the events (pre-processed alerts) that results from a two-phase subsystem (i.e. *filter* and *aggregator*). Secondly, it uses the time information. Thirdly, detecting large-scale attacks such as DDoS or worm in the early stage was done by using the *situator*. Finally, the feed-back mechanism for the attack class similarity matrix that was conceptually described but was not constructed. The *situator* is a new component that has never been used before , and it has the ability to detect the large scale attacks like 1:N (e.g. network or service scan), N:1 (e.g. DDos), and M:N (e.g. worms).

In the French Defense Agency's MIRADOR project, the main objective was to develop a co-operation module between multiple IDSs to correlate alerts in order to reduce the alert volume and generate more global and synthetic alerts. The co-operation module is an expert rule-based system that supports logical reasoning with predicate logic. The functions of the system include [23]: *Alert Management*, *Alert Clustering*, *Alert Merging*, *Alert Correlation* and *Intention Recognition* (not developed). Only the first three components will be stated that are related to this subsection.

In this system, *Alert Management* deals with storing and managing alerts issued by different IDSs in a relational database. *Alert Clustering* refers to finding similarity of new alerts to existing alerts in a knowledge base. The alert similarity is determined by the similarity requirements specified by expert rules [23]. These rules are domain specific and are defined by examination of prior alerts generated by the IDSs considered. The rules are defined to express similarity between alert source, target, time and classification. Also, specific expert rules define unique cases where sources/targets can be considered similar based on alert classification.

While in Alert Merging, alert groups are incrementally constructed as alerts arrive. Each group has a so-called global alert, which contains combined information from all alerts in the group. Global alerts are generated with set-valued attributes. A new alert is compared to all global alerts in the system and added to similar groups. Note that an alert can be simultaneously added to more than one group.

3 Limitations and Enhancements

The detailed reviews in this paper demonstrate several problems or limitations of past research. They also suggest solutions that might overcome many of these limitations. The following points were noted from the literature:

- At present, most alert correlation techniques do not make full use of all the information that is available. For example, they tend to only use the alerts generated by security tools (like IDSs) and also specific features from these alerts. Using the vulnerabilities in correlation is a promising technique such as [21].
- The existing IDSs cannot detect all the attacks which cause a shortage in the alerts forwarded to the correlation system. This shortage leads to poor correlation results. The correlation systems must be more intelligent than IDSs by having the capability to avoid the shortage in the received data. So there is a need to develop a component in the correlation systems having the capability of recouping the missed alerts. For example, the missed attacks in one scenario lead to distinct scenarios.
- Research groups tend to focus on same sorts of techniques: for instance, a particular attention is given to scenario-based approaches. We believe, however, that there is an urgent need to find new techniques pass the weakness in this

type of approaches. Also most of the existing systems focus on a problem and neglect many others.

- Current research on correlation systems has mainly focused on serial attacks, i.e., sequences of atomic actions leading to a security breach. Although correlation analysis has been applied to analyze relationships between different ongoing attacks, little attention was paid to correlating individual actions across users in order to identify a single attack. In a coordinated attack, attackers' actions interfere with one another, making it difficult to analyze an action out of the context of other actions.
- Obtaining attack details is a big obstacle in the way of developing efficient correlation methods. A major weakness of many past approaches is that information used to describe pre/post-conditions for attack components must be entered by hand. This is labor intensive and difficult, especially to model attacks using the amount of detail required by many studies (e.g., [18, 19, 20, 21]).
- Most of the correlation techniques currently use only alerts from signature based IDSs. The using of anomaly IDSs techniques in the correlation process needs to be investigated. As a hybrid correlation technique, the combining of the unknown attacks generated by anomaly IDSs with the current correlation techniques should be performed.

4 Conclusions

Extensive research is going on in the field of alert processing and several techniques are already developed but their performance is poor. Researchers proposed several alert processing approaches and each approach have some limitations which enable the attacker to evade them easily. Because of limitation of each approach, here is an urgent need to arrive of a generic approach that handles almost all types of evasions. For that it is required to understand and analyze the techniques that are already investigated by several researchers. Keeping that in view here, we have made an attempt to review the well known alert processing approaches. Comparison of various approaches is made to show the strength and weakness of these approaches. We hope this study will be useful for researchers to carry forward research on system security for design of a correlation system that not only will have identified strengths but also overcome the drawbacks.

References:

[1] Manganaris S., Christensen M., Zerkle D., Hermiz K., A data mining analysis of RTID alarms, Computer Networks 34 (4), 2000, pp. 571-577.

[2] Julisch K., Clustering intrusion detection alarms to support root cause analysis, ACM Transactions on Information and System Security, ACM Press 6(4), 2003, pp. 443-471.

[3] Intrusion detection message exchange message format (IDMEF), http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-14.txt, 2005.

[4] Hätälä A., Särs C., Addams-Moring R., Virtanen T., Event data exchange and intrusion alert correlation in heterogeneous networks, Proceedings: 8th Colloquium for Information Systems Security Education West Point, NY, 2004, pp. 84-92.

[5] Sun B., Wu K., Pooch U. W., Alert aggregation in mobile ad hoc networks, ACM WiSE'03, San Diego, California, USA, Sep. 2003.

[6] Pouget F., Dacier M., Alert correlation: review of the state of the art, Research Report RR-03-093, Institut Eurecom, France, Nov 2003.

[7] Valeur F., Vigna G., Kruegel C., Kemmerer R. A., Comprehensive approach to intrusion detection alert correlation, IEEE Transactions on Dependable and Secure Computing 1(3), 2004,pp. 146-169.

[8] Siraj A., A unified alert fusion model for intelligent analysis of sensor data in an intrusion detection environment, Ph.D. thesis, Mississippi State University, Aug 2006.

[9] Kruegel C., Robertson W., Alert verification: determining the success of intrusion attempts, Proceedings: 1st Workshop on the Detection of Intrusions and Malware and Vulnerability Assessment (DIMVA 2004), Dortmund, Germany, Jul 2004.

[10] NMAP Network Mapping Tool. http://www.insecure.org/nmap/

[11] Julisch K., Mining alarm clusters to improve alarm handling efficiency, Proceedings: 17th Annual Computer Security Applications Conference (ACSAC'01), New Orleans, LA, Dec 2001.

[12] Pietraszek T., Using adaptive alert classification to reduce false positives in intrusion detection, Recent advances in intrusion detection (RAID2004). In: Lecture notes in computer science, vol. 3324. Sophia Antipolis, France: Springer-Verlag; 2004.

[13] Lewis L., Service level agreements for enterprise networks, Artech House: Norwood, MA, 1999, pp. 158-190.

[14] Qin X., Lee W., Statistical causality of INFOSEC alert data, Proceedings: Recent Advances in Intrusion Detection, LNCS 2820; Springer-Verlag, 2003, pp. 73-93.

[15] Dain O. M., Cunningham R. K., Building scenarios from a heterogeneous alert stream, IEEE Transactions on Systems Man and Cybernetics, 2002.

[16] Valdes A., Skinner K., Probabilistic alert correlation, Proceedings: Recent Advances in Intrusion Detection, LNCS 2212, 2001, pp. 54-68.

[17] Debar H., Wespi A., Aggregation and correlation of intrusion-detection alerts, Proceedings: 4th International Symposium on Recent Advances in Intrusion Detection, Davis, CA, USA, Oct 2001, pp. 85-103.

[18] Ning P., Reeves D., Cui Y., Correlating alerts using prerequisites of intrusions, technical report TR-2001-13, Department of Computer Science, North Carolina State University, 2001.

[19] Ning P., Xu D., Healey C. G., Amant R. A. S., Building attack scenarios through integration of complementary alert correlation methods, Proceedings: 11th Annual Network and Distributed System Security Symposium (NDSS '04), Feb 2004, pp. 97-111.

[20] Cuppens F., Miège A., Alert correlation in a cooperative intrusion detection framework, Proc. IEEE Symposium, Security and Privacy, May 2002.

[21] Wang L., Liu A., Jajodia S., Using attack graphs for correlating, hypothesizing, and predicting intrusion alerts, Computer Communications 29, Apr 2006, pp. 2917-2933.

[22] Lee S., Chung B., Kim H., Lee Y., Park C., Yoon H., Real-time analysis of intrusion detection alerts via correlation, Computers & Security 25, 2006, pp.169-183.

[23] Cuppens F., Managing alerts in a multi-Intrusion detection environment, Proceedings: 17th Annual Computer Security Applications Conference, New Orleans, LA, Dec 2001.