Alert Correlation Survey: Framework and Techniques

Reza Sadoddin
Network Security Laboratory
University of New Brunswick
Fredericton, New Brunswick, Canada
reza.sadoddin@unb.ca

Ali Ghorbani
Network Security Laboratory
University of New Brunswick
Fredericton, New Brunswick, Canada
ghorbani@unb.ca

ABSTRACT

Managing raw alerts generated by various sensors are becoming of more significance to intrusion detection systems as more sensors with different capabilities are distributed spatially in the network. Alert Correlation addresses this issue by reducing, fusing and correlating raw alerts to provide a condensed, yet more meaningful view of the network from the intrusion standpoint. Techniques from a divers range of disciplines have been used by researchers for different aspects of correlation. This paper provides a survey of the state of the art in alert correlation techniques. Our main contribution is a two-fold classification of literature based on correlation framework and applied techniques. The previous works in each category have been described alongside with their strengths and weaknesses from our viewpoint.

1. INTRODUCTION

Complementary security tools are widely used in networks to provide a better detection coverage of network intrusions. Intrusion detection systems (IDSs), firewalls, antivirus tools and file integrity checkers are employed in different layers. These tools may differ from each other in types of intrusions they detect or in the techniques they employ. For example, firewalls focus on accepting, logging, or dropping network traffic, IDSs focus on detecting known attack patterns (signature-based IDSs) or abnormal behaviors (anomaly-based IDSs), antivirus tools focus on scanning viruses based on pre-defined virus signatures, and file integrity checkers monitor the activities on file systems such as file addition and deletion.

A higher level management is required for analyzing low-level alerts produced by these devices before reporting them to the next layer. This is needed for several reasons. First, it is not easy to locate the source or target of the intrusion or fault in the network by looking into low-level alerts. Secondly, the low-level sensors consider raw alerts in isolation and raise alarms for each of them, without considering logical connections between alerts. Also, any automatic response would be inefficient with these raw alerts as input. In addition, in a typical environment there are a lot of false alerts reported by traditional IDSs, which are mixed with true alerts.

To address the above-mentioned problems, some efforts have been made both in academic and industrial communities referred to as Alert Correlation. The primary goal of alert correlation is to provide system support for a global and condensed view of network attacks by analyzing raw alerts. Alert correlation is too complex to be tackled in a single phase. Instead, it has been accepted as a framework consisted of several components, which accepts raw alerts as input and produces high level attack scenarios as output.

Different techniques from various disciplines have been used for each of the components of the alert correlation frameworks. To the best of our knowledge, the only survey on alert correlation has been reported in 2003 by Pouget and Dacier reported in [23]. They have made a comprehensive review on different disciplines and provided a high level description of the previous methods. In this paper, our goal is to present a state of the art survey on the previous works from a different point of view. The emphasize of our survey is on mapping between framework components and employed techniques. We believe that our approach gives a better understanding of this area as more literature works are presented based on an underlying framework whose components are implemented using different techniques.

The components involved in an alert correlation process are not same for different correlation approaches. Table 1 summarizes components of three alert correlation frameworks introduced in previous works.

In this paper, a framework of six components is considered. These components include *Normalization*, *Aggregation*, *Correlation*, *False Alert Reduction*, *Attack Strategy Analysis* and *Prioritization*. Although there is no one-to-one mapping between our framework's components and those of others, we believe that it covers all aspects of correlation which have been addressed in the literature.

This paper is organized as follows. In the next section, we present an overall view of the framework components and the corresponding approaches. In Section 3, we describe previous works which have addressed each of the components. Finally, in Section 4 we conclude with summarizing current research trends and issues in this ares. Appendix A contains a table which summarizes previous works considered in this survey.

PST 2006 ,Oct 30-Nov 1, 2006, Markham, Ontario, Canada

[4]	[34]	[32]
Alert-based Management	Alert Aggregation	Normalization
Alert Clustering	Alert Clustering	Pre-Processing
Alert Merging	Alert prioritization	Alert Fusion
Alert Correlation	Alert time Series Formulation	Alert Verification
Intention Recognition	Alert Correlation Scenario Construction	Thread Reconstruction
		Attack Session Reconstruction
		Focus Recognition
		Multi-step Correlation
		Impact Analysis
		Prioritization

Table 1: Components of sample alert correlation frameworks

2. GENERAL VIEW OF THE FRAMEWORK AND APPROACHES

Diverse challenges of this area has motivated researchers to take advantage of quite a large variety of disciplines. Data mining techniques have been used extensively to facilitate the task of analyzing large volume of data and mining frequent event patterns. Machine learning methods have also been employed for learning indeterministic relationships between alerts and adaptation to environment changes. The input knowledge to the system are received from different devices with different degrees of confidence and even with contradictory evidences. The presence of uncertainty directed the researchers to employ fuzzy techniques in order to compensate for imprecision of data. The necessity for modeling temporal relationships between alerts has emerged to using time reasoning techniques. Figure 1 shows an overall view of the approaches which have been used for different components of an alert correlation framework.

The framework components are the main aspects of alert correlation being addressed in the literature. Although in some of the works these aspects have been mixed with each other, there is enough distinction between their main objectives as we will discuss more thoroughly in the following sections. In Figure 1, each framework component is connected to the techniques used for it. The techniques are limited to the works we have considered in this survey.

3. APPROACHES TO FRAMEWORK COMPONENTS

In this Section, we introduce different approaches which have been proposed in previous works based on our correlation framework.

3.1 Normalization

The alerts may come from different sensors and security systems. Since they can be encoded in different formats, it is necessary to translate each alert into a standardized format that is understood by correlation components. The Internet Engineering Task Force proposed the Intrusion Detection Message Exchange Format (IDMEF) [7] data model as a way to establish a standard representation of intrusion alerts. This work seems to be the most notable effort in standardizing alert formats. Currently, a lot of intrusion detection systems use the draft message format. IDMEF is object-oriented and is implemented in the Extensible Markup Language (XML). One of the main design goals of the IDMEF data model is to be able to express relationships between alerts which is actually an essential requirement of alert correlation. Currently, the top level class *Message* is inherited by two subclasses, the *Heartbeat* and the *Alert* messages. But, we are more interested in two subclasses called *Correlation-Alert* and *Tool-Alert*. The Correlation-Alert class encodes the relations between previously sent alerts, while the Tool-Alert class provides additional information related to attack tools or malicious programs such as Trojan horses which could have possibly been used for launching the attack. The latter piece of information can be used by the analyzer which is able to identify these tools.

Although IDMEF provides a common enriched data model for exchanging alerts between IDS components, there are still issues to be addressed for true inter-operability and wide acceptance among intrusion detection community. Different IDSs use their own naming conventions to fill the classification fields of the alerts. A common taxonomy (or classification) of attacks is required to be adopted by detectors and analyzers in order to cooperate with each other easily. In lack of such a common taxonomy, attack classes should be mapped to each other locally by maintaining huge databases for all low-level sensors supported by an analyzer. Clearly, this is not satisfactory. Some efforts have been made to address this gap [15, 13, 14, 30], but they are still far from wide acceptance. Instead, some other publicly available databases of attacks and vulnerabilities have been used for the purpose of inter-operability between different IDSs and their components though they do not provide a taxonomy. CVE (Common Vulnerability Exposure) [33] and Bugtraq-IDs [2] are two well-known databases of this kind.

3.2 Aggregation

In aggregation, the purpose is to group all similar alerts together. Previous works in literature have given different definitions of alert aggregation. In some of them alerts are considered similar to each other if they match in all attributes except with little time difference while some other works extend the concept of aggregation as clustering all the alerts sharing root causes. The latter interpretation of similarity may involve aggregating alerts based on several attributes. Throughout this document, we use the extended view of aggregation. Due to large number of alerts produced by low-level sensors for a single malicious activity, alert aggregation has proven to be highly effective in reducing alert volume. Similar alerts tend to have similar root causes or

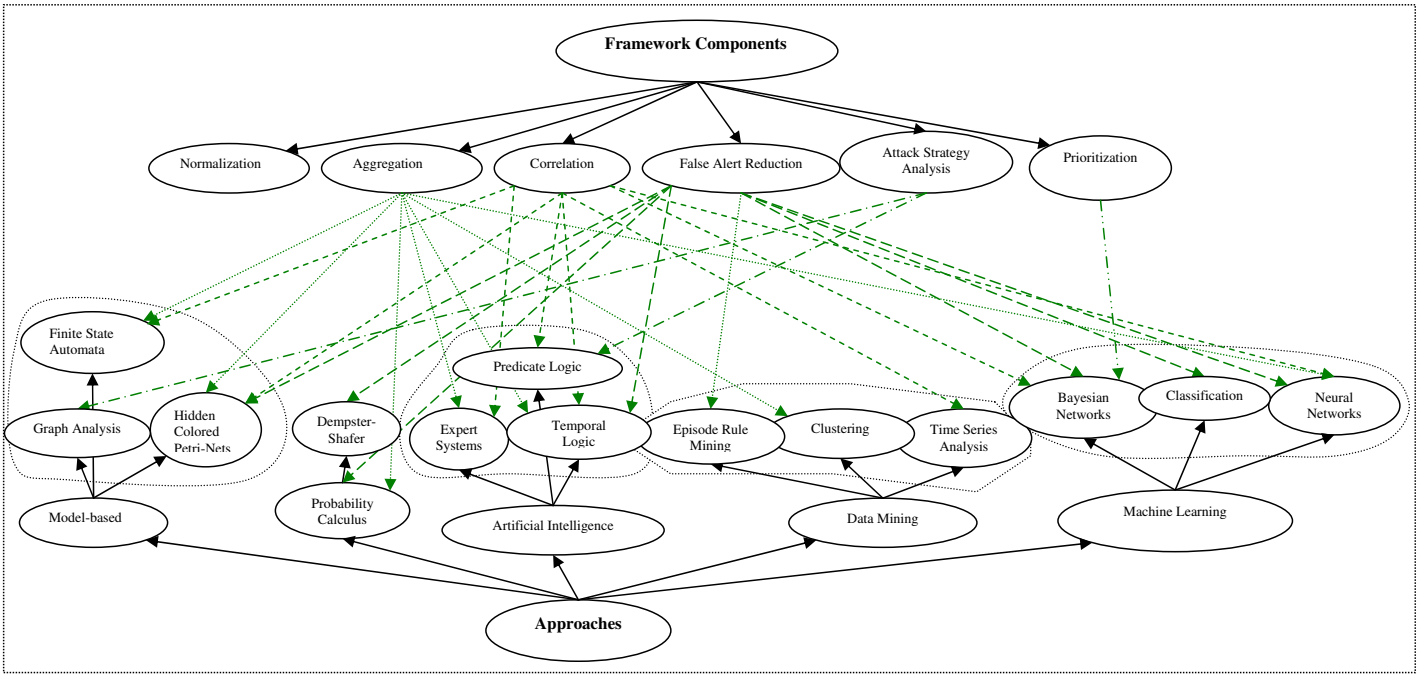

Figure 1: Mapping between Framework Components and Approaches

similar effects on network resources. Clustered alerts are not only more suitable for administrator analysis, but also facilitate causality or false positive analysis of the alerts.

In [31], the similarity of alerts has been estimated based on similarity between respective alert attributes. Similarity between values of each attribute (IP, port, time, attack class,...) has been defined based on the characteristics of that attribute and expected similarity between values of attributes have been used as the weight of that attribute in overall alert similarity.

A clustering technique has been used by Julisch in [12] for grouping all the alerts which share same root causes. Intuitively, root cause of an alert is the reason for which the alert has been raised. Central to clustering technique proposed by Julisch are the hierarchy structures (*generalization hierarchies* as they are called), which decompose the attributes of the alerts from the most general values to the most specific ones. Based on the generalization hierarchies, the dissimilarity of two attribute is defined to be the longest path between values of that attribute in the corresponding generalization hierarchy. The dissimilarity of two alerts is defined to be the sum of the dissimilarity of their attributes. A clustering algorithm has been proposed in [12] which takes as input the alert databases and min_size (minimum size of the resulting clusters) and returns all the clusters which contain at least min_size number of alerts. The idea of generalization hierarchies seems promising for the purpose of aggregation, but the proposed method has some drawbacks. The first and foremost is that a fix predefined size is assumed for all types of clusters which are created. It does not reflect the fact that alerts of different types tend to create clusters of different sizes. In addition, the clustering technique does not distinguish between different attributes in selecting the feature on which the generalization is performed. In contrast, it is known that different features contribute with different significance in overall similarity between two or more alerts. Even for a single attribute, the similarity between two alerts is dependent on attack class. For example, for attacks which are possible with spoofed IP addresses, the weight of IP similarity is different from those attacks in which IP spoofing is not possible.

The CRIM correlation framework [4] which is designed within the MIRADOR project includes two components dealing with alert aggregation. The received alerts from different sensors (which are assumed to be compliant with the IDMEF) are stored in a relational database. The clustering method is based on predicate logic in which the similarity between two alerts is specified in terms of similarities between alert fields (attack class, source, target, ...). Similarities between alert fields are, in turn, specified by domain-specific expert rules. Alert clustering component creates clusters of similar alerts using expert rules. Following that, alert merging component produces the representative alerts for a cluster. One interesting point mentioned in this work is the concept of cluster stability. The stability of a cluster determines when the representative alert can be reported on behalf of the aggregation module. The authors acknowledge that required time for cluster stability is different for various attacks, but assume the maximum delay times are provided statically in a database for different attacks.

Data mining techniques have been used in [8] to learn how to group alerts based on their similarities. In [38], two machine learning techniques have been used to estimate the probability of similarity between alert pairs. Some of the

works in this category propose a unified method both for aggregation and correlation [31, 8, 38]. This involves the causal relationships between two alerts to be reflected somehow in the similarity estimation model. In [31], similarity between alert types are defined in a static matrix which encodes the probability that an alert of type x be followed by an alert of type y. It is assumed that similarities between alert classes are provided by security experts. Using a machine learning approach in [38] allows the authors to use some derived features as input to correlation engine (in addition to simple features of alerts being compared). The similarity between destination IP address of the first alert and source IP address of the second one is a sample derived feature. An important contribution of this work is the idea of Alert Correlation Matrix (ACM). ACM encodes the average correlation between alert classes which is computed adaptively based on statistical analysis of consecutive input alerts (while this knowledge is provided by experts in most similar works such as [31]). The adaptation characteristic of this method makes it possible to start with initial (and maybe immature) correlation probability values and learn more from the environment as the operation proceeds.

The similarity-based techniques cannot find the causal relationship between alerts as these methods group alerts based on similarities between their attributes. It is the main weakness of the models which use this method both for the purpose of correlation and aggregation.

3.3 Correlation

The goal of correlation is to find the causal relationships between alerts in order to reconstruct attack scenarios from the isolated alerts. Although it may not have a significant affect in reducing the number of alerts, the role of correlation component is essential in providing a higher level view of the actual attacks. Previous works on correlation can be divided into the following categories based on the criterion used for relating alerts:

- Scenario-based correlation: In this category of works, causality relationships between alerts are specified in terms of scenarios. In other words, alerts are correlated with each other if they can be combined to construct an attack scenario.
- Rule-based correlation: In this type of works, relationships between alerts is specified rules. In other words, alert 'A' is correlated with alert 'B' if the first one prepares for the second alert in a precondition-postcondition relationship.
- Statistical correlation: Methods in this group correlate two alerts if they are statistically related to each other.
- Temporal correlation: Two alerts are correlated with each other based on their temporal relationships.

In the following sections, we review the proposed techniques for each of these categories.

3.3.1 Scenario-based Correlation

Most of the previous works in this category have used formal models for specifying attack scenarios or have employed machine learning techniques to learn this knowledge. LAMBDA [6], STATL [10] and ADeLe [29] are examples of formal correlation and detection languages which have been used for attack scenario specification. In [9], attack scenarios are specified by *consequence* rules. Consequence rules encode ordered relationships between attack classes which are used for correlating alerts. Temporal logic formalism is used in [17] for modeling attack scenarios. In this work, *chronicles* serve as correlation building blocks. Each chronicle can be viewed as set of patterns with temporal and contextual constraints over them. As new alerts reach to the correlation component, they are compared with chronicles and the state of the chronicles are changed (or new ones are constructed) if matching occurs.

Specifying scenarios for all possible attacks requires extensive prior knowledge and can be labor intensive. Machine learning techniques have been used in [8] for learning attack scenarios during training step from the labeled alerts. The alerts have been gathered in a real world network containing attack traffic and labeled manually by authors. The labels represent the attack scenario the alerts belong to. The proposed features for classification are interesting and reflect alerts relationships in real world attacks. This approach can build models for alert correlation automatically; however, it requires training in every deployment, and the resulting models may over-fit the training data and cannot correlate attack scenarios not seen in the training data sets. The major limitation of this category of methods is that they are restricted to known attack scenarios.

3.3.2 Rule-based Correlation

Attack sub-goals can be satisfied in different ways by using various exploits. This leads to a large number of attack scenario variations which makes it difficult to maintain a comprehensive attack database. Rule-based approaches [28, 5, 19, 3] mitigate this problem by providing a finer grained correlation method. These methods correlate alerts by matching prerequisites (pre-conditions) and consequences (post-conditions) of attack steps. In this sense, rule-based correlation can be considered as a special scenario-based correlation. In [28], an attack description language, named JIGSAW, has been introduced for specifying attack steps (or concepts as they are called) based on their pre-conditions and post-conditions (capabilities). The concepts are constructed as soon as all the required capabilities are satisfied, which in turn, provide new capabilities through which attack scenario can proceed. JIGSAW requires all the preconditions be satisfied. Therefore, it can not construct attack scenarios for which some alerts are missed by sensors. MIRADOR correlation method [5] allows partial satisfaction of prerequisites, which addresses the mentioned limitation in JIGSAW. In [5], attacks are specified in LAMBDA language which is consisted of five fields. The most important fields from the correlation point of view are as follows:

- Attack pre-condition: a logical condition that specifies the conditions to be satisfied for the attack to succeed.
- Attack post-condition: a logical condition that specifies the effect of the attack when this attack succeeds
- Description scenario: the combination of events which are necessary to detect an occurrence of the attack.

In this approach, alerts are converted from IDMEF format into a set of logical facts. In order for two alerts A and B to be correlated directly, (at least) one predicate in the postcondition of A should be unified with one predicate in

the precondition of B. In this way, the authors propose a correlation method which is consisted of two steps. In the offline phase, logical links between alerts are discovered and correlation rules are extracted which encode the conditions under which an alert may prepare for the second one. In the operation phase, the simplified correlation rules are used for constructing attack scenarios.

In [19], a similar approach is used to model precondition-postcondition relationships between alerts . The concept of *hyper alert* is introduced in this work which encodes the precondition and postcondition of an alert in its prerequisite and consequence fields, respectively. Alerts are processed by the alert-preprocessor to generate hyper alerts and instantiate the prerequisite and consequence sets of each hyper alert. After this step, prerequisites and consequences of hyper alerts are saved in two tables of a relational database system alongside with their unique hyper alert IDs. The main correlation step is just a matching between consequences of an alert with prerequisites of another alert by issuing SQL queries considering the time constraints. The correlated alerts are demonstrated in a hyper alert correlation graph which denotes the "prepare for" relationships between the hyper alerts.

3.3.3 Statistical Correlation

In [24], a statistical correlation technique has been used as a complementary approach to the knowledgebase correlation method. In this work, Bayesian network is used to model the causality relationship between alerts. In this model, alerts are denoted by nodes and their causality relationships are represented by edges. Continues alerts are divided along equal time slots and the state of each node corresponding to an alert is a binary value representing presence of the alert in the time slot. Based on this model, the goal is to determine which alert types may cause an arbitrary alert of type A and how the conditional probability of A is related to its parents (causes). A simple algorithm has been proposed in [24] based on *mutual information* between alerts to find the structure of the Bayesian network, i.e. causal relationships between alerts. Knowing the structure of the Bayesian network, it is easy to fill the CPTs (Conditional Probability Tables) using the statistical relationships between alerts divided along time slots.

3.3.4 Temporal Correlation

In [34], the causality relationships between alerts are analyzed using *Granger Causality Test*. Granger Causality Test (GCT) is a time series-based statistical analysis method that aims to test if a time series variable X correlates with another time series variable Y by performing a statistical hypothesis test. To apply time series analysis to the problem of alert correlation, *hyper alerts* are constructed by aggregating all the alerts which have same values of all features except some time differences. Time series variables are modeled for all hyper alerts based on the number of alerts per time unit and Granger Causality Test is used to verify whether variable X provides any significant information about variable Y. If yes, alert of type X may be the cause of the alert Y. The basic idea in GCT is to estimate the value of variable Y by two models:

- *AR* (Autoregressive) model which estimates the value of Y in time t based on its previous values.
- *ARMA* (Autoregressive Moving Average) model which estimates the value of Y in time t based on its previous values and previous values of variable X as well.

The errors of above estimations are compared with each other. If ARMA model provides a better estimation, it means that alert of type X is causally related to alert of type Y based on GCT. An important assumption about applicability of this technique is that alerts should present some kind of temporal relationship. This assumption leads to a major limitation of this method which fails to correlate alerts with random time delays between them.

Generally, the scenario-based and rule-base methods are restricted to knowledgebase which should be hard-coded in the system or learned through labeled data while the statistical and temporal methods are capable of correlating alerts which may contribute to unknown attacks. On the other hand, these techniques seem to be much more time consuming comparing to knowledgebase methods in practice. In addition, statistical and temporal techniques are highly sensitive to deliberate delays or planned noisy alerts in the middle of actual attack scenarios. Accordingly, they are complementary to each other in correlation component.

3.4 False Alert Reduction

The goal of this component is to distinguish between false positive and true positive alerts. Different sensors have their own strengths and weaknesses in detecting various attacks and it is a well-know problem of low-level sensors to generate alerts with high rate of false positives. As a result, the role of this component is critical in reducing false alerts.

The idea of *frequent episode rules* have been used in [16] for mining normal alert patterns. In general, a frequent episode rule is specified by the following expression:

$$L_1, L_2, \ldots, L_m \rightarrow R_1, R_2, \ldots, R_n(s, c, w) \qquad (1)$$

where Li's represent the left-hand patterns which may trigger a rule, R_i's represent the right-hand patterns which are expected to be seen most frequently in a time window no larger than w and s and c have their classic definitions as *support* and *confidence* values in an *association rule*. Applied to the context of alert correlation, frequent episode rules encode the normal sequence of alerts which are seen with adequate support and confidence and are suspicious of being false positives. The frequent episode rules are generated during a learning process with an attack free traffic. During the operation mode, sequence of alerts which match a frequent episode rule are considered as false positives.

In another reported work in [21], a classification technique has been used for false positive analysis of alerts. The classification engine takes as input three categories of information. These information include network topology, alert context (other similar alerts related to this alert) and alert semantics (evidences on alert applicability based on the vulnerabilities of the target network). Some appropriate features have been proposed which encode the required input information to the classification engine. A cost-sensitive variation of *Ripper* rule learner has been used as the classification algorithm which returns a confidence score for each of two output classes. In the operation mode, feedback from security administrator is required for true classification if the confidence of alert is not enough for each of true of false classes. This feedback serves as an adaptation technique as

well. The feedback of the administrator is actually a labeled data which helps the classification engine to adapt to previously unseen cases.

Confidence fusion is essential to reducing false alerts when alerts are reported with different confidences by low-level sensors. Here the goal is to estimate the overall confidence of alerts based on low-level evidences of alerts.

In [37], a reasoning framework for alert confidence is presented by employing security state information of the system alongside with alerts . The security state information is provided by system monitoring and vulnerability scanning tools and alerts are issued by IDS sensors. The causal relationships between alerts and system attributes (security state of the system) are modeled by *alert-attribute* graph. In this graph, nodes represent the alerts or system attributes and edges represent precondition-postcondition relationships between alerts and system attributes. They take advantage of Bayesian networks to verify evidences (Both alerts and attributes of the system). The Conditional Probability Tables (CPTs) ,corresponding to each node of alert-attribute graph, encode probability estimation of alerts or system attributes based on parent node's states. The authors have not discussed about initializing Conditional Probability Tables (CPTs).

In [27], the problem of aggregation and correlation of alerts in the presence of uncertainty has been addressed using subjective logic. It is assumed that alerts are issued by low-level sensors which provide their opinion (containing degree of *belief*, *disbelief*, *uncertainty* and *relative atomicity*) on events based on their views of the activity triggered the alert. At a higher level resides sensor fusion component which receives the opinions from the sensors and uses the *consensus* or *multiplication* operation of subjective logic to provide the confidence of the aggregated or correlated alerts, respectively. The decision is made based on a predefined threshold of confidence for issuing the final alarms on behalf of the correlation framework.

In [35], one basic drawback of the previous works on alert correlation is addressed. The problem the authors refer is that most rule-based correlation techniques do not distinguish between attacker actions and issued alerts, which leads to high rate of false positive and false negative alerts. *Hidden Colored Petri Net* is introduced in [35] with its formal definition as extension to *Colored Petri Net* to distinguish between attacker actions (*transitions*) and alerts (*observables*) in the new model. Attack scenarios are represented in this model based on the precondition and posconditions of attack steps. The required algorithms for *inference* (deciding about most probable transitions in the model given a sequence of observables) and learning parameters of the model are presented. Based on these facilities, it is possible to estimate the most likelihood hypothesis about attacker actions given observed alerts. The final report issued to the administrator contains the compromised resources which are much limited than individual alerts. Another advantage of this method, as mentioned by the authors, is the probabilistic confidence score associated with compromised resources rather than strict alerts. The basic limitation of this work is its reliance on expert knowledge for specifying attack rules which is a serious one considering large volume of attacks needed to be encoded.

In [36], *Weighted Dempster-Shafer* has been extended from the basic *Dempster-Shafer* theory. In the new model, *Dempster rule* takes into account different weights of the confidences provided by various sources (e.g. alerts reported by a signature-based sensor receive more confidence compared to those reported by an anomaly-based sensor). The *confidence fusion* component takes as input the correlated alerts by lower-level analyzers and resolves contradictory information in different analyzers. Finally, the overall confidence level of compromised resources (as is the output of the Hidden Colored Petri-nets) is reported to automatic response module or administrator for further analysis.

In another reported work [26] in this category, *Fuzzy Cognitive Map* has been used for the purpose of correlation and false positive reduction. The authors model attack scenarios using two types of events. *Cause events* represent actions taken on behalf of attacker (which are observed by alerts received from IDS's) and *effect events* which correspond to security incidents in the system. Based on this observation, Fuzzy Cognitive Maps (FCM) is used for modeling attack scenarios in which causal relationships between events are mapped to causal relationships between *concepts* of the FCM. The fuzzy concept has been reflected in the model both in concept evidences and their causal relationships (both of them are fuzzy values). Different incidents may be activated with various degrees for every resource (e.g. host). The concept of *Incident Alert Level* (IAL) is defined to fuse the overall impact of different incidents for each resource. Incident alert level is modeled by a FCM which encodes the degree of affect for every activated incident on IAL. A simple formula is used to compute the value of IAL based on fuzzy values of incidents and their impacts. This approach is again highly dependent on expert knowledge to specify attack scenarios in terms of cause and effect events and impact values in cognitive model. In [1], a similar approach has been used for estimating *Cluster Association Strength* from different clusters activated for a resource.

3.5 Attack Strategy Analysis

The aim of attack strategy analysis is to reason about attacker actual intention. The input to this component is the low-level correlated alerts. The motivation behind further analysis is that low-level correlated alerts may not represent a complete attack strategy planned by attacker as a result of missed alerts by sensors (false negatives). This leads to isolated attack scenarios which require higher level correlation. In addition, the initial attack steps may pave the way for further attacks. Predicting future attack steps would be helpful and important for security operators or automatic response modules to take actions in advance to avoid further damage.

In [20], a complementary approach to alert correlation has been proposed for hypothesizing and reasoning about attacks missed by IDSs. The basic idea is to consider indirect causal relationship between intrusion alerts and the constraints they must satisfy (prerequisites and consequences). The correlation framework uses two low-level correlation techniques: causality correlation which is based on prerequisites and consequences of attacks and correlation based on similarity between alert attribute values (which is called clustering in their work). Alerts are correlated with each other independently through each of correlation techniques. The result of causality correlation is a set of correlation graphs which may have been separated from each other due to possibly missed alerts by the IDSs. Correlation graphs

which share similar alerts (based on similarity-based correlation results) are considered for further analysis. The idea is to add possibly missed alerts between two similar alerts if the first alert can prepare for the second one by considering a chain of virtual alerts satisfying the causality relation.

In another reported work [5], the notation of *abductive rules* has been used to deal with missed alerts. If an alert, which is the known consequence of an event that should have generated another alert, is received by the correlation engine and if that other event has not been received by the same correlation engine, by means of a simple abductive reasoning the missed event can be identified.

A hierarchical approach for correlation has been proposed in [25], in which raw alerts are correlated with each other based on classic correlation techniques (knowledgebase or statistical). Following that, low-level correlated alerts are considered for further analysis in order to discover missing alerts or predict possible future actions on behalf of the attacker. In this work, it is assumed that attack plans are maintained by *Attack Trees* which encode attack goals and sub-goals hierarchically. The leaves of the attack tree contain the most specific actions performed by attacker, the internal nodes represent attack sub-goals and attacker ultimate goal is represented by the root node. To reason about missed alerts, two set of correlated alerts are compared against the attack plan(s) containing two alert sets. An algorithm is presented to check how two set of alerts may be related to each other where an alert set is a sub-goal of the other one or both of them are sub-goals of the same goal. For the purpose of prediction, alert trees are converted to Bayesian networks which encode the probability of goals and sub-goals based on state of their parent nodes. In runtime, *inference* process is applied to Bayesian networks to compute the conditional probabilities of goals and sub-goals based on the observed alerts and satisfied sub-goals.

All these approaches are limited to knowledgebase of the system in the sense that they can not cope with missing events that are not linked to other events by means of cause-consequence relationships.

3.6 Prioritization

The purpose of alert prioritization is to classify alerts based on their severity and take appropriate actions for dealing with each alert class. Alert prioritization component should take into account various domain information in addition to alert types. Security policy, network topology, vulnerability analysis of the network services and installed softwares, and asset profiles are some of factors affecting priority of alerts. As a result, the focus of previous works in this category has been mainly on modeling the domain knowledge.

In [22], M-Correlator has been proposed as a framework for computing rank of the alerts and clustering them based on the ranks. The framework is supported by an internal database, called *Incident Handling Fact Base*, which provides some critical information regarding alert codes, their descriptions, and dependencies of alert types to their required OS versions, hardware platforms, network services and vulnerabilities. At first, relevance scores are assigned to alerts based on their corresponding attack applicability to the network. M-Correlator maintains an internal topology map of the protected network, which is dynamically managed by the analyst. Automated topology map generation is supported using nmap [11], through which M-Correlator can identify the available assets on the network, IP address to hostname mappings, OS type and version information, active TCP and UDP network services per host, and hardware type. Nmap is periodically run to update the topology database. In the next step, the priority of the alerts is computed based on the criticality of the data assets and services they are targeting and also the severity of the attack class they belong. The asset values should be determined by the security analyst. Following that, an overall incident rank is assigned to each alert, which brings together the priority of the alert with the likelihood of success. Incident rank is computed based on the alert *outcome*, *priority* and *relevance* score. A belief propagation technique based on Bayesian networks is used for estimating the incident rank (based on a 'low', 'medium' and 'high' scale). Incident rank is encoded in a Bayesian network based on the evidences provided by leaf nodes representing the underlying beliefs of outcome, priority and relevance scores. Based on computed parameters, M-Correlator selects the alerts that will impose the greatest risk to the monitored network. Although, the impact analysis method can automate alert priority analysis, the construction of a mission-impact based model requires substantial human intervention, and the constructed model is highly dependent on the monitored systems.

Another widely referred work in this category is M2D2 [18]. M2D2 can be seen as a formal model for representing security related information including *vulnerabilities*, *security tools*, *alerts* and *information system characteristics*. Although M2D2 provides the formalism for modeling security related information, specific mechanisms are still required for prioritizing alerts.

4. CONCLUSION AND FUTURE RESEARCH TRENDS

In this paper, we made a review of different techniques from various disciplines introduced in the literature for alert correlation. The techniques were presented in the context of a comprehensive correlation framework. As high-level comparison between techniques, either competitive or complementary to each, the pros and cons of the techniques were described. It is worthy to notice that issues in alert correlation are quite similar to those in network management systems. Both have to do with redundant or similar events, discover the causally related alerts and fuse the evidences distributed throughout the network. This makes the techniques applied in network management area potential candidates for alert correlation as well. Although this extends the domain of potential techniques, there are issues specific to alert correlation area which need special treatments. The most important difference is that correlating alerts of IDSs requires dealing with some kind of intelligence on behalf of the attacker, while it is not the case for the alarms generated by fault discovery tools. Subtle strategies utilized by attacker during, before or after the intrusion make some techniques ineffective in finding relationships between alerts.

Some highlights of our survey are as follows:

- The *generalization hierarchies* provide a suitable data model for the purpose of aggregation through which alerts can be clustered with arbitrary level of abstraction. One issue in this regard is how to adapt to alert type taxonomy and keep it current with the latest

known attacks.

- The current trend of correlation is towards complementary correlation engines. As discussed earlier, the knowledge-based correlation methods and statistical and temporal correlation approaches have their pros and cons. While the last two approaches are capable of correlating alerts which may contribute in unknown attacks, the scenario-based and rule-based correlation methods are solely dependent on the defined attack scenarios and rules in the knowledgebase of the system, respectively. On the other hand, the knowledge-based correlation methods are much more accurate and less time consuming. In addition, statistical and temporal correlation methods fail to discover causality relation between alerts had noisy alerts or deliberate delays planned by the attacker among the low-level alerts. Complementary methods are preferred to enjoy capabilities of the proposed techniques.

- Attack strategy analysis and plan recognition has received attention in research works to address the problem of isolated attack scenarios. This opportunity also extends the functionality of correlation module by predicting next actions of the attacker. This facility, in turn, can be used by higher level modules for attack prevention.

- Knowledge acquisition is an important issue in the area of alert correlation. Domain knowledge is required for specifying attack scenarios, defining correlation rules based on prerequisites and consequences of alerts, finding possible missed alerts by sensors, capturing the temporal and statistical relationships between alerts and many other purposes. It is appealing to take advantage of data mining and machine learning techniques for knowledge acquisition.

- Adaptation to new environments and changes of an individual environment is another important challenge in this area. No matter what the knowledge is and how it is gathered, special mechanisms should be in place to keep the knowledge up to date.

- Lack of publicly available datasets for the purpose of training and evaluation is another major issue in this area. Previous works have used ready datasets with limited attack scenarios or produced their own datasets by capturing traffic in real world networks. Providing labeled datasets of enriched attacks would be of great benefit to alert correlation research for intrusion detection systems.

5. ACKNOWLEDGMENTS

The authors generously acknowledge the funding from the Atlantic Canada Opportunity Agency (ACOA) through the Atlantic Innovation Fund (AIF) and through grant RGPN 227441 from the National Science and Engineering Research Council of Canada (NSERC) to Dr. Ghorbani.

6. REFERENCES

[1] R. B. V. Ambareen Siraj. Multi-level alert clustering for intrusion detection sensor data. In *Proceedings: North American Fuzzy Information Processing Society International Conference on Soft Computing for Real World Applications*, Ann Arbor, Michigan, June 2005.

[2] B. M. L. by Bugtraq. http://www.securityfocus.com/bid/bugtraqid/.

[3] S. Cheung, U. Lindqvist, and M. W. Fong. Modeling multistep cyber attacks for scenario recognition. In *DARPA Information Survivability Conference and Exposition*, pages 284–292, Washington, D.C., 2003.

[4] F. Cuppens. Managing alerts in a multi-intrusion detection environment. In *17th Annual Computer Security Applications Conference New-Orleans*, New-Orleans, USA, December 2001.

[5] F. Cuppens and A. Miege. Alert correlation in a cooperative intrusion detection framework. In *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, page 202, Washington, DC, USA, 2002. IEEE Computer Society.

[6] F. Cuppens and R. Ortalo. Lambda: A language to model a database for detection of attacks. In *RAID '00: Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection*, pages 197–216, London, UK, 2000. Springer-Verlag.

[7] D. Curry and H. Debar. Intrusion detection message exchange format: Extensible markup language (xml) document type definition, January 2003.

[8] O. Dain and R. Cunningham. Fusing heterogeneous alert streams into scenarios. In *Proceedings of the 2001 ACM Workshop on Data Mining for Security Applications*, pages 1–13, 2001.

[9] H. Debar and A. Wespi. Aggregation and correlation of intrusion-detection alerts. In *Recent Advances in Intrusion Detection*, pages 85–103, 2001.

[10] S. Eckmann, G. Vigna, and R. Kemmerer. Statl: An attack language for state-based intrusion detection, 2002.

[11] Fyodor. The art of port scanning, December 1997. http://www.insecure.org/nmap/.

[12] K. Julisch. Clustering intrusion detection alarms to support root cause analysis. *ACM Trans. Inf. Syst. Secur.*, 6(4):443–471, 2003.

[13] S. Kumar. *Classification and Detection of Computer Intrusions*. PhD thesis, Purdue, IN, 1995.

[14] C. E. Landwehr, A. R. Bull, J. P. McDermott, and W. S. Choi. A taxonomy of computer program security flaws, with examples. Technical report, Naval Research Laboratory, November 1993.

[15] U. Lindqvist and E. Jonsson. How to systematically classify computer security intrusions. In *IEEE Symposium on Security and Privacy*, pages 154–163, 1997.

[16] S. Manganaris, M. Christensen, D. Zerkle, and K. Hermiz. A data mining analysis of rtid alarms. *Comput. Networks*, 34(4):571–577, 2000.

[17] B. Morin and H. Debar. Correlation of intrusion symptoms: An application of chronicles. In *RAID*, pages 94–112, 2003.

[18] B. Morin, L. Mé, H. Debar, and M. Ducassé. M2d2: A formal data model for ids alert correlation. In *RAID*, pages 115–127, 2002.

[19] P. Ning, Y. Cui, and D. Reeves. Constructing attack scenarios through correlation of intrusion alerts, 2002.

[20] P. Ning, D. Xu, C. G. Healey, and R. S. Amant. Building attack scenarios through integration of complementary alert correlation method. In *NDSS*, 2004.

[21] T. Pietraszek. Using adaptive alert classification to reduce false positives in intrusion detection, 2004.

[22] P. A. Porras, M. W. Fong, and A. Valdes. A mission-impact-based approach to infosec alarm correlation. In *RAID*, pages 95–114, 2002.

[23] F. Pouget and M. A. Dacier. Alert correlation: Review of the state of the art, Technical Report RR-03-093.

[24] X. Qin. *A Probabilistic-Based Framework for INFOSEC Alert Correlation*. PhD thesis, Georgia Institute of Technology, 2005.

[25] X. Qin and W. Lee. Attack plan recognition and prediction using causal networks. In *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04)*, pages 370–379, Washington, DC, USA, 2004. IEEE Computer Society.

[26] A. Siraj and R. B. Vaughn. A cognitive model for alert correlation in a distributed environment. In *ISI*, pages 218–230, 2005.

[27] H. Svensson and A. Josang. Correlation of intrusion alarms with subjective logic. In *Proceedings of the sixth Nordic Workshop on Secure IT systems (NordSec2001),Copenhagen, Denmark*, Novenber 2001.

[28] S. J. Templeton and K. Levitt. A requires/provides model for computer attacks. In *NSPW '00: Proceedings of the 2000 workshop on New security paradigms*, pages 31–38, New York, NY, USA, 2000. ACM Press.

[29] E. Totel, B. Vivinis, and L. Mé. A language driven ids for event and alert correlation. In *SEC*, pages 209–224, 2004.

[30] J. L. Undercoffer, A. Joshi, and J. Pinkston. Modeling Computer Attacks: An Ontology for Intrusion Detection. In *The Sixth International Symposium on Recent Advances in Intrusion Detection*. Springer, September 2003.

[31] A. Valdes and K. Skinner. Probabilistic alert correlation. *Lecture Notes in Computer Science*, 2212:54–68, 2001.

[32] F. Valeur, G. Vigna, C.Kruegel, and R. Kemmerer. A Comprehensive Approach to Intrusion Detection Alert Correlation. *IEEE Transactions on Dependable and Secure Computing*, 1(3):146–169, July-September 2004.

[33] C. Vulnerabilities and Exposures. http://www.cve.mitre.org/.

[34] W. L. Xinzhou Qin. Statistical causality analysis of infosec alert data. *Lecture Notes in Computer Science*, 2820(2):73 – 93, 2003.

[35] D. Yu and D. A. Frincke. A novel framework for alert correlation and understanding. In *ACNS*, pages 452–466, 2004.

[36] D. Yu and D. A. Frincke. Alert confidence fusion in intrusion detection systems with extended dempster-shafer theory. In *IEEE Symposium on Security and Privacy*, Kennesaw, Georgia, 2005.

[37] Y. Zhai, P. Ning, P. Iyer, and D. S. Reeves. Reasoning about complementary intrusion evidence. In *ACSAC '04: Proceedings of the 20th Annual Computer Security Applications Conference (ACSAC'04)*, pages 39–48, Washington, DC, USA, 2004. IEEE Computer Society.

[38] B. Zhu and A. A. Ghorbani. Alert correlation for extracting attack strategies. *International Journal of Network Security*, 3(2):244–258, 2006.

Appendix A: Survey Summary

#Ref	Framework	Techniques	Knowledge Acquisition	Datasets
[28]	Co	Not Mentioned	Manual	
[3]CAML	Co	Expert Systems	Manual	DARPA GCP
[17]Chronicle	Ag	Temporal Logic	Manual	
[4]CRIM	Ag	Predicate Logic	Manual	Private datasets
[5]CRIM	Co, St	Predicate Logic	Manual	Private datasets
[8]	Ag, Co	Neural Networks, Decision Trees	Automatic(Learning)	DEF CON 8
[9]ACC	No, Ag, Co	Clustering, Consequence Alert Matching	Manual	
[10]AlertSTAT	Ag, Co	State Machine	Manual	
[12]	Ag	Clustering	Manual	
[19]TIAA	Ag, Co	Matching Predicates using SQL Queries	Manual	DARPA 2000, DEFCON 8
[21]ALAC	Fa	Classification(RIPPER rule learner)	Automatic	DARPA 99, Private dataset
[22]M-Correlator	Pr	Bayesian Networks	Both Manual and Automatic	
[34]	Co	Time Series Analysis	Automatic(Learning)	GCP, DEF CON 9
[31]	Ag, Co	Probability Estimation	Manual(Attack Similarity Matrix)	
[32]Comprehensive Framework		The main focus is on the framework itself rather than proposing new techniques	Manual	DARPA(1999 and 2000), CTV, DEFCON 9 and other datsets
[37]	Fa	Bayesian Networks	Manual	DARPA 99 for background traffic)
[20]	Co, St	Similarity-based and Knowledge-based	Manual	DARPA 2000, LLDoS 1.0
[25]	St	Attack Tree Analysis , Bayesian Networks	Manual	DARPA GCP
[38]	Ag, Co	Neural Networks	Automatic(Learning)	DARPA 2000
[27]	Fa	Subjective Logic	Not Required	
[26]	Co, Fa	Fuzzy Cognitive Maps	Manual	DARPA 2000
[1]	Ag, Fa	Fuzzy Cognitive Maps	Manual	DARPA 2000
[36]	Fa	Weighted Dempster-Shafer Theory	Not Required	DARPA 2000
[35]	Ag, Co, Fa	Hidden Colored Petri-Nets	Manual(Attack Scenarios) and Automatic(Learning)	DARPA 2000
[24]	Co	Bayesian Networks	Automatic(Learning)	GCP, DEF CON 9

No: Normalization, **Ag**: Aggregation, **Co**: Correlation , **Fa**: False Alert Reduction, **St**: Attack Strategy Analysis, **Pr**: Prioritization