Statistical Causality Analysis of INFOSEC Alert Data

Xinzhou Qin Wenke Lee

College of Computing
Georgia Institute of Technology
Atlanta, GA 30332
{xinzhou, wenke}@cc.gatech.edu

Abstract. With the increasingly widespread deployment of security mechanisms, such as firewalls, intrusion detection systems (IDSs), antivirus software and authentication services, the problem of alert analysis has become very important. The large amount of alerts can overwhelm security administrators and prevent them from adequately understanding and analyzing the security state of the network, and initiating appropriate response in a timely fashion. Recently, several approaches for alert correlation and attack scenario analysis have been proposed. However, these approaches all have limited capabilities in detecting new attack scenarios. In this paper, we study the problem of security alert correlation with an emphasis on attack scenario analysis. In our framework, we use clustering techniques to process low-level alert data into high-level aggregated alerts, and conduct *causal analysis* based on statistical tests to discover new relationships among attacks. Our statistical causality approach complements other approaches that use hard-coded prior knowledge for pattern matching. We perform a series of experiments to validate our method using DARPA's Grand Challenge Problem (GCP) datasets and the DEF CON 9 datasets. The results show that our approach can discover new patterns of attack relationships when the alerts of attacks are statistically correlated.

Keywords: Intrusion detection, alert correlation, attack scenario analysis, time series analysis

1 Introduction

Information security (INFOSEC) is a complex process with many challenging problems. Deploying INFOSEC mechanisms, e.g., authentication systems, firewalls, intrusion detection systems (IDSs), antivirus software, and network management and monitoring systems, is just one of the necessary steps in the security process. INFOSEC devices often output a large amount of low-level or incomplete alert information because there is a large number of network and system activities being monitored and multiple INFOSEC systems can each report some aspects of the same (coordinated) security event. The sheer quantity of alerts from these security components and systems also overwhelms security administrators. The large number of low-level or incomplete alert information can prevent intrusion response systems and security administrators from adequately understanding and analyzing the security state of the network, and initiating appropriate response in a timely fashion. From a security administrator's point of view, it is important to reduce the redundancy of alarms, intelligently integrate and correlate security alerts, construct attack scenarios (defined as a sequence of related attack steps) and present high-level aggregated information from multiple local-scale events. Correlating alerts of the related attack steps to identify an attack scenario can also help forensic analysis, response and recovery, and even prediction of forthcoming attacks.

Recently there have been several proposals on alert correlation (e.g., [4, 7, 10, 22, 25, 27]). Most of these proposed approaches have limited capabilities because they rely on various forms of predefined knowledge of attack conditions and consequences. They cannot recognize a correlation when an attack is new (previously unknown) or the relationship between attacks is new. In other words, these approaches in principle are similar to *misuse detection* techniques, which use the "signatures" of known attacks to perform pattern matching and cannot detect new attacks. It is obvious that the number of possible correlations is very large, potentially a combinatorial of the number of (known and new) attacks. It is infeasible to know *a priori* and encode all possible matching conditions between attacks. To further complicate the matter, the more dangerous and intelligent adversaries will always invent new attacks and novel attack sequences. Therefore, we must develop significantly better alert correlation algorithms that can discover sophisticated and new attack sequences.

In this paper, we study the problem of INFOSEC alert analysis with an emphasis on attack scenario analysis. The analysis mechanism is based on time series and statistical analysis. We reduce the high volume of raw alerts by combining low-level alerts based on alert attributes. Clustering techniques are used to group low-level alert data into high-level alerts. We prioritize alerts based on the relevance of attacks to the protected networks and hosts and the impacts of attacks on the mission goals. We then conduct causality analysis to correlate alerts and construct attack scenarios. We perform a series of experiments to validate our method using DARPA's Grand Challenge Problem (GCP) datasets and the DEF CON 9 datasets. Our results show that our approach can discover new patterns of alert relationships without depending on prior knowledge of attack scenarios. Our statistical approach complements other approaches in that our correlation approach does not depend on the hard-coded prior knowledge for pattern matching and can discover new attack relationships when the alerts of attacks are statistically correlated.

The emphasis of this paper is on statistical causality analysis. The remainder of this paper is organized as follows. In Section 2, we introduce Granger Causality Test, a time series analysis method. Our alert correlation steps and algorithms are presented in Section 3. In Section 4, we report the experiments and results on the GCP datasets and the DEF CON 9 datasets. Section 5 discusses related work. We summarize our work and future work in Section 6.

2 Granger Causality Analysis

Time series analysis aims to identify the nature of a phenomenon represented by a sequence of observations. The objective requires the study of patterns of the observed time series data. Time series analysis has been widely used in many applications, e.g., earthquake forecasting and economy analysis. In this section, we introduce time series based causal analysis, and in particular, the Granger Causality Test [11].

2.1 Time Series Analysis

A time series is an ordered finite set of numerical values of a variable of interest along the time axis. It is assumed that the time interval between consecutively recorded values is constant. We denote a univariate time series as $x(k)$, where $k = 0, 1, \ldots, N-1$, and N denotes the number of elements in $x(k)$.

Time series causal analysis deals with analyzing the correlation between time series variables and discovering the causal relationships. Causal analysis in time series has been widely studied and used in many applications, e.g., economy forecasting and stock market analysis. Network security is another application in which time series analysis can be very useful. In our prior work [1,3], we have used time series-based causality analysis for pro-active detection of Distributed-Denial-of-Service (DDoS) attacks using MIB II [26] variables. We based our approach on the Granger Causality Test (GCT) [11]. Our results showed that the GCT is able to detect the "precursor" events, e.g., the communication between Master and Slave hosts, without prior knowledge of such communication signatures, on the attacker's network before the victim is completely overwhelmed (e.g., shutdown) at the final stage of DDoS.

In this work, we apply the GCT to INFOSEC alert streams for alert correlation and scenario analysis. The intuition is that attack steps that do not have well-known patterns or obvious relationships may nonetheless have some statistical correlations in the alert data. For example, there are one or more alerts for one attack only when there are also one or more alerts for another attack. We can apply statistical causality analysis to find such alerts to identify an attack scenario. We next give some background on the GCT.

2.2 Granger Causality Test

The intuition of Granger Causality is that if an event X is the cause of another event Y, then the event X should precede the event Y. Formally, the Granger Causality Test (GCT) uses statistical functions to test if *lagged* information on a time-series variable x provides any statistically significant information about another time-series variable y. If the answer is yes, we say variable x Granger-causes

y. We model variable y by two auto-regression models, namely, the Autoregressive Model (AR Model) and the Autoregressive Moving Average Model (ARMA Model). The GCT compares the residuals of the AR Model with the residuals of the ARMA Model. Specifically, for two time series variables y and x with size N, the Autoregressive Model of y is defined as:

$$y(k) = \sum_{i=1}^{p} \theta_i y(k-i) + e_0(k) \tag{1}$$

The Autoregressive Moving Average Model of y is defined as:

$$y(k) = \sum_{i=1}^{p} \alpha_i y(k-i) + \sum_{i=1}^{p} \beta_i x(k-i) + e_1(k) \tag{2}$$

Here, p is a particular lag length, and parameters α_i, β_i and θ_i $(1 \leq i \leq p)$ are computed in the process of solving the Ordinary Least Square (OLS) problem (which is to find the parameters of a regression model in order to have the minimum estimation error). The residuals of the AR Model is $R_0 = \sum_{k=1}^{T} e_0^2(k)$, and the residuals of the ARMA Model is $R_1 = \sum_{k=1}^{T} e_1^2(k)$. Here, $T = N - p$.

The AR Model, i.e., Equation 1, represents that the current value of variable y is predicted by its past p values. The residuals R_0 indicate the total sum of squares of error. The ARMA Model, i.e., Equation 2, shows that the current value of variable y is predicted by the past p values of both variable y and variable x. The residuals R_1 represents the sum of squares of prediction error.

The Null Hypothesis H_0 of GCT is $H_0 : \beta_i = 0, i = 1, 2, \cdots, p$. That is, x does not affect y up to a delay of p time units. We denote g as the Granger Causality Index (GCI):

$$g = \frac{(R_0 - R_1)/p}{R_1/(T - 2p - 1)} \sim F(p, T - 2p - 1) \tag{3}$$

Here, $F(a, b)$ is Fisher's F distribution with parameters a and b [14]. F-test is conducted to verify the validity of the Null Hypothesis. If the value of g is larger than a critical value in the F-test, then we reject the Null Hypothesis and conclude that x Granger-causes y. Critical values of F-test depends on the degree of freedoms and significance value. The critical values can be looked up in a mathematic table [15].

The intuition of GCI (g) is that it indicates how better variable y can be predicted using histories of both variable x and y than using the history of y alone. In the ideal condition, the ARMA model precisely predicts variable y with residuals $R_1 = 0$, and the GCI value g is infinite. Therefore, the value of GCI (g) represents the strength of the causal relationship. We say that variable $\{x_1(k)\}$ is more likely to be causally related with $\{y(k)\}$ than $\{x_2(k)\}$ if $g_1 > g_2$ and both have passed the F-test, where g_i, $i = 1, 2$, denotes the GCI for the input-output pair (x_i, y).

Applying the GCT to alert correlation, the task is to determine which hyper alerts among $B_1, B_2, \ldots, B_l$ are the most likely to have the causal relationship with hyper alert A (a hyper alert represents a sequence of alerts in the same cluster, see Section 3). For a hyper alert time series, say A, each $A(k)$ is the number of alerts occurring within a certain time period. In other words, we are testing the statistical correlation of alert instances to determine the causal relationship between alerts. For each pair of hyper alerts $(B_i, A), i = 1, 2, \ldots, l$, we compute the GCI value g_i. We record the alerts whose GCI values have passed the F-test as the candidates, and rank order the candidate alerts according to their GCI values. We can then select the top m candidate alerts and regard them as being causally related to alert A. These (candidate) relationships can be subject to more inspection by other analysis techniques such as probabilistic reasoning or plan recognition.

The main advantage of using statistical causality test such as GCT for alert correlation is that this approach does not require *a priori* knowledge about attack behaviors and how the attacks could be related. This approach can identify the correlation between two attack steps as long as the two have a high probability (not necessarily high frequency) of occurring together. We believe that there is a large number of attacks, e.g., worms, that have attack steps with such characteristics. Thus, we believe that causal analysis is a very useful technique. As discussed in [1, 3, 2], when there is sufficient training data available, we can use GCT off-line to compute and validate very accurate causal relationships from alert data. We can then update the knowledge base with these "known" correlations for efficient pattern matching in run-time. When GCT is used in real-time and finds a new causal relationship, as discussed above, the top m candidates can be selected for further analysis by other techniques.

3 Alarm Correlation

In this section, we describe our framework for alert correlation and attack scenario construction. Specifically, the steps include alert aggregation and clustering, alert prioritization, alert time series formulation, alert correlation, and scenario construction.

3.1 Alert Aggregation and Clustering

One of the issues with deploying multiple security devices is the sheer amount of alerts output by the devices. The large volume of alerts makes it very difficult for the security administrator to analyze attack events and handle alerts in a timely fashion. Therefore, the first step in alert analysis is alert aggregation and volume reduction.

In our approach, we use alert fusion and clustering techniques to reduce the redundancy of alerts while keeping the important information. Specifically, each alert has a number of attributes such as *timestamp*, *source IP*, *destination IP*, *port(s)*, *user name*, *process name*, *attack class*, and *sensor ID*, which are defined in the standard document "Intrusion Detection Message Exchange Format (IDMEF)" [12] drafted by the IETF Intrusion Detection Working Group.

In alert fusion, there are two steps. First, we combine alerts that have the same attributes except timestamps. The timestamps can be slightly different, e.g., 2 seconds apart. Second, based on the results of step 1, we aggregate alerts with the same attributes but are reported from different heterogeneous sensors. The alerts varied on time stamp are fused together if they are close enough to fall in a pre-defined time window.

Alert clustering is used to further group alerts after alert fusion. Based on various clustering algorithms, we can group alerts in different ways according to the *similarity* among alerts, (e.g., [27] and [17]). Currently, based on the results of alert fusion, we further group alerts that have same attributes except time stamps into one cluster. After this step, we have further reduced the redundancy of alerts.

A *Hyper Alert* is defined as a time ordered sequence of alerts that belong to the same cluster.

For example, after alert clustering, we have a series of alerts, $A_1, A_2...A_n$ in one cluster that have the same attributes along the time axis, and we use hyper alert A to represent this sequence of alerts.

3.2 Alert Prioritization

The next phase of alert processing is to prioritize each hyper alert based on its relevance to the mission goals. The objective is that, with the alert priority rank, security analyst can select important alerts as the target alerts for further correlation and analysis. Specifically, the priority score of an alert is computed based on the relevance of the alert to the configuration of the protected networks and hosts as well as the severity of the corresponding attack assessed by the security analyst. Porras et al. proposed a more comprehensive mechanism of incident/alert rank computation model in a "mission-impact-based" correlation engine, named M-Correlator [25]. Because we focus on alert correlation and scenario analysis instead of alert priority ranking, and alert prioritization is just an intermediate step to facilitate further alert analysis, we adapted the priority computation model of M-Correlator with a simplified design.

Figure 1 shows our priority computation model that is constructed based on Bayesian networks [24]. We use Bayesian inference to obtain a belief over states (hypotheses) of interests. A Bayesian network is usually represented as a directed acyclic graph (DAG) where each node represents a variable, and the directed edges represent the causal or dependent relationships among the variables. A conditional probability table (CPT) [24] is associated with each child node. It encodes the prior knowledge between the child node and its parent node. Specifically, an element of the CPT at a child node is defined by $CPT_{ij} = P(child\_state = j | parent\_state = i)$ [24]. The belief in hypotheses of the root is related to the belief propagation from its child nodes, and ultimately the evidence at the leaf nodes.

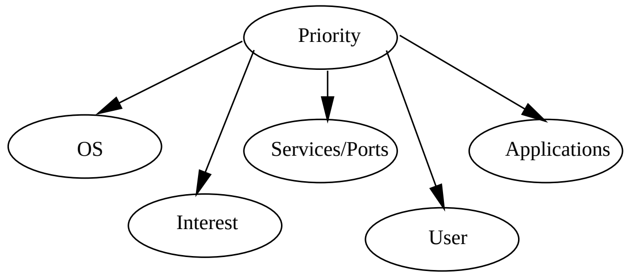

Fig. 1. Alert Priority Computation Model

Specifically, in our priority computation model, the root represents the priority with two hypothesis states, i.e., "high" and "low". Each leaf node has three states. For node "Interest", its three states are "low", "medium" and "high". For other nodes, the three states are "matched", "unmatched" and "unknown". The computation result is a value in [0,1] where 1 is the highest priority score.

We denote e^k as the k^{th} leaf node and H_i as the i^{th} hypothesis of the root node. Given the evidence from the leaf nodes, assuming conditional independence with respect to each H_i, the belief in hypothesis at the root is: $P(H_i \mid e^1, e^2, \ldots, e^N) = \gamma P(H_i) \prod_{k=1}^{N} P(e^k|H_i)$, where $\gamma = [P(e^1, e^2, \ldots, e^N)]^{-1}$ and γ can be computed using the constraint $\sum_i P(H_i|e^1, e^2, \ldots, e^N) = 1$. For example, for the hyper alert of *FTP Globbing Buffer Overflow* attack, we get evidence [high, matched, matched, unknown, unknown] from the corresponding leaf nodes, i.e., Interest, OS, Services/Ports, Applications and User, respectively. As Figure 1 shows, the root node represents the priority of hyper alert. Assume that we have the prior probabilities for the hypotheses of the root, i.e., $P(Priority = high) = 0.8$ and $P(Priority = low) = 0.2$, and the following conditional probabilities as defined in the CPT at each leaf node, $P(Interest = high|Priority = high) = 0.70$, $P(Interest = high|Priority = low) = 0.10$, $P(OS = matched|Priority = high) = 0.75$, $P(OS = matched|Priority = low) = 0.20$, $P(Services = matched|Priority = high) = 0.70$, $P(Services = matched|Priority = low) = 0.30$, $P(Applications = unknown|Priority = high) = 0.15$, $P(Applications = unknown|Priority = low) = 0.15$, $P(User = unknown|Priority = high) = 0.10$, $P(User = unkown|Priority = low) = 0.10$, we then can get $\gamma = 226.3468$, therefore, $P(Priority = high|Interest = matched, OS = matched, Service = matched, Applications = matched, User = unknown) = 0.9959$. We regard this probability as the priority score of the alert. The current CPTs are predefined based on our experience and domain knowledge. It is our future work to develop an adaptive priority computation model so that the CPTs can be adaptive and updated according to specific mission goals.

To calculate the priority of each hyper alert, we compare the dependencies of the corresponding attack represented by the hyper alert against the configurations of target networks and hosts. We have a knowledge base in which each hyper alert has been associated with a few fields that indicate its attacking OS, services/ports and applications. For the alert output from a host-based IDS, we will further check if the target user exists in the host configuration. The purpose of relevance check is that we can downgrade the importance of some alerts that are unrelated to the protected domains. For example, an attacker may launch an individual buffer overflow attack against a service blindly, without knowing if the service exists. It is quite possible that a signature-based IDS outputs the alert once the packet contents match the detection rules even though such service does not exist on the protected host. The relevance check on the alerts aims to downgrade the impact of such kind of alerts on further correlation analysis. The interest of the attack is assigned by the security analyst based on the nature of the attack and missions of the target hosts and services in the protected domain.

3.3 Alert Time Series Formulation

After the above processes, we formulate each hyper alert into a univariate time series. Specifically, we set up a series of time slots with equal time interval, denoted as T, along the time axis. Given a time range H, we can have $N = H/T$ time slots. Recall that each hyper alert A represents a sequence of alerts in the same cluster in which all alerts have the same attributes except timestamp, i.e., $A = [A_1, A_2, \ldots, A_n]$, where A_i represents an alert in the cluster. We denote $a(k)$, where $k = 0, 1, \ldots, N-1$, as the corresponding time series variable of hyper alert A. An element of the time series $a(k)$, denoted as a_i, is the number of alerts that fall in the i^{th} time slot. Therefore, each element of a hyper alert time series variable represents the number of alert instances within the corresponding time slot. We currently do not use categorical variables such as port accessed and pattern of TCP flags as time series variables in our approach.

3.4 GCT Alert Correlation

The next phase of alert processing is to apply GCT for pair-wise alert correlation. Based on alert priority value and mission goals, the security analyst can specify a hyper alert as a target (e.g., alert *Mstream_DDOS* against a database server) with which other alerts are correlated. The GCT algorithm is applied to the corresponding alert time series. Specifically, for a target hyper alert Y whose corresponding univariate time series is $y(k)$, and another hyper alert X whose univariate time series is $x(k)$, we compute $GCT(x(k), y(k))$ to correlate these two alerts. For the target alert Y, we compute such pair-wise correlation with all the other alerts. As described in Section 2.2, the GCT index (GCI) g returned by the GCT function represents the evidence strength if X is causally related to Y. We record the alerts whose GCI values have passed the F-distribution test as candidates of causal alerts, and rank order the candidate alerts according to their GCI values. We then select the top m candidate alerts and regard them as being causally related to alert Y. These candidate relationships can be further inspected by other techniques or security analyst based on expertise and domain knowledge. The corresponding attack scenario is constructed based on the correlation results.

In alert correlation, identifying and removing background alerts is an important step. We use *Ljung-Box* [20] test to identify the background alerts. The assumption is that background alerts have characteristic of randomness. The *Ljung-Box* algorithm tests for such randomness via autocorrelation plots. The Null Hypothesis is that the data is random. The test value is compared with critical values to determine if we reject or accept the Null Hypothesis.

However, in order to correctly remove the background alerts, expertise is still needed to verify that a hyper alert can be regarded as a background alert. In addition to expertise, we can also use other techniques, e.g., probabilistic reasoning, for further inspection and verification. This is part of our future work.

4 Experiments

To evaluate the effectiveness and validity of our alert correlation mechanisms, we applied our algorithms to the datasets of the Grand Challenge Problem (GCP) version 3.1 provided by DARPA's Cyber Panel program [6, 13], and datasets of the DEF CON 9 Capture The Flag (CTF) [9]. In this section, we describe our experiments with an emphasis on the GCP.

4.1 The Grand Challenge Problem (GCP)

The main motivation to use the GCP datasets is that the GCP has developed multiple innovative attack scenarios to specifically evaluate alert correlation techniques. In addition to the complicated attack scenarios, the GCP datasets also include many background alerts. This makes alert correlation and scenario construction more challenging. Other datasets, e.g., DEF CON 8 Capture The Flag (CTF) [8], have relatively simple scenarios [21]. In the GCP, multiple heterogeneous security systems, e.g., network-based IDSs, host-based IDSs, firewalls, and network management systems, are deployed in several network enclaves.

GCP alerts are in IDMEF (XML) format. We implemented our alert processing system in Java. It can consume XML format alerts directly.

As described in Section 3, we first fuse and cluster raw alerts into more aggregated and *hyper alerts*. In scenario I, there are a little more than 25,000 low-level raw alerts output by heterogeneous security devices in all enclaves. After alert fusion and clustering, we have around 2,300 hyper alerts. In scenario II, there are around 22,500 raw alerts that result in 1,800 hyper alerts.

The GCP definition includes complete information about the configuration of the protected networks and hosts including services, operating systems, user accounts, etc. Therefore, we can establish a configuration database accordingly. Information of mission goals enables us to identify the servers of interest and assign interest score for corresponding alerts targeting at the important hosts. The alert priority is calculated based on our model described in Section 3.2.

In formulating hyper alert time series, as described in Section 3, we set the time slot to 60 seconds. In the GCP, the whole time range is 5 days. Therefore, each hyper alert time series $x(k)$ has a size of 7,200, i.e., k=0, 1, 2, ..., 7,199.

In GCT alert correlation, the first step is to identify and remove the background alerts. As described in Section 3.4, we apply the *Ljung-Box* statistical test to all hyper alerts. We select the significance level $\alpha = 0.05$. However, in order to correctly remove the background alerts, expertise is still needed to verify that a hyper alert can be regarded as background alert. In the GCP, by using this mechanism, we can identify background alerts such as "HTTP_Cookie" and "HTTP_Posts". The next step is to select the alerts with high priority values as the target alerts. In this step, we set the threshold $\beta = 0.6$. Alerts with priority scores above β are regarded as important alerts and are selected as target alerts. We then apply the GCT to correlate each target alert with other alerts from which the background alerts identified by the *Ljung-Box* test are already excluded.

For performance evaluation, we define two measures: *true causality rate* $= \frac{\#\ of\ correct\ causal\ alerts}{total\ \#\ of\ causal\ relationships}$ and *false causal rate* $= \frac{\#\ of\ incorrect\ causal\ alerts}{total\ \#\ of\ causal\ alerts}$. Here, *causal alerts* refer to the causal alert candidates output by the GCT (i.e., passing the *F-test*) w.r.t. the target alerts. In experiments of the GCP, we refer to the documents with the ground truth to determine the causal relationships among the alerts.

$Alert_i$	Target Alert	GCT Index
HTTP_Java	Loki	22.25
DB_IllegalFileAccess	Loki	11.81
DB_NewClient	Loki	11.12
DB_NewClient_Target	Loki	10.84
DB_FTP_Globbing_Attack	Loki	10.84
HTTP_ActiveX	Loki	10.68

Table 1. Alert Correlation by the GCT on the GCP Scenario I. Target Alert: Loki

$Alert_i$	Target Alert	GCT Index
Loki	DB_NewClient	115.56
Plan_NewClient	DB_NewClient	14.50
Plan_Loki	DB_NewClient	13.06
HTTP_Java	DB_NewClient	12.84
DB_NewClient_Target	DB_NewClient	12.84
DB_FTP_Globbing_Attack	DB_NewClient	12.84
HTTP_ActiveX	DB_NewClient	12.84
DB_IllegalFileAccess	DB_NewClient	10.76

Table 2. Alert Correlation by the GCT on the GCP Scenario I: Target Alert: DB_NewClient

$Alert_i$	Target Alert	GCT Index
HTTP_Java	DB_IllegalFileAccess	22.23
DB_NewClient	DB_IllegalFileAccess	14.87
Loki	DB_IllegalFileAccess	11.24
Plan_Loki	DB_IllegalFileAccess	11.13
HTTP_ActiveX	DB_IllegalFileAccess	10.71
Plan_NewClient	DB_IllegalFileAccess	9.08

Table 3. Alert Correlation by the GCT on the GCP Scenario I: Target Alert: DB_IllegalFileAccess

In the GCP Scenario I, there are multiple network enclaves in which attacks are conducted separately. The attack scenario in each network enclave is almost the same. We selected a network enclave as an example to show the GCT correlation results.

In this network enclave, there are a total of 370 hyper alerts. Applying the *Ljung-Box* test on the hyper alerts, we identify 255 hyper alerts as background alerts. According to the alert priority values calculated based on the mission-goals and relevance to the protected networks and hosts, there are 15 hyper alerts whose priority values are above the threshold $\beta = 0.6$. Therefore, we have 15 hyper alerts as the target alerts, which are correlated with other alerts excluding the identified background alerts. As an example, we select three alerts that are related to the Database Server as the target alerts, i.e., *Loki*, *DB_NewClient* and *DB_IllegalFileAccess*. Alert *Loki* indicates that there is a stealthy data transfer via a covert channel. Alert *NewClient* means that a host on the network initiates a connection to a remote service that is suspicious and uncharacteristic. Therefore, alert *DB_NewClient* denotes the connection activity from the Database Server to an external suspicious site. Alert *DB_IllegalFileAccess* occurs when there is a file access (read or write) on the Database Server that violates the access policy. *DB* and *Plan* represent *Database Server* and *Plan Server* respectively. Table 1 shows the causal alert candidates correlated with target alert *Loki*. Table 2 shows the alert candidates that are causally related to target alert *DB_NewClient*. Table 3 shows the causal alerts related to target alert*DB_IllegalFileAccess*. Alert *DB_FTP_Globbing_Attack* indicates an *FTP Globbing buffer overflow* attack on the Database Server. Alert *DB_NewClient_Target* denotes an unusual connection activity from a host to the Database Server. Among the candidate alerts which have passed the F-test, we select the top 6 alerts according to their GCI values.

Figure 2 shows the correlation graph based on the correlation results of alerts *Loki*, *DB_NewClient* and *DB_IllegalFileAccess*. Here, some expert knowledge is needed to further inspect the causal alert candidates resulted from GCT correlation in order to construct the correlation graph. In this case, we do not include alerts such as *HTTP_Java* and *HTTP_ActiveX* in the scenario construction because they are not likely to be correct causal alerts. In the correlation graph, the directed edges represent the causal relationships and the arrows show the causal-

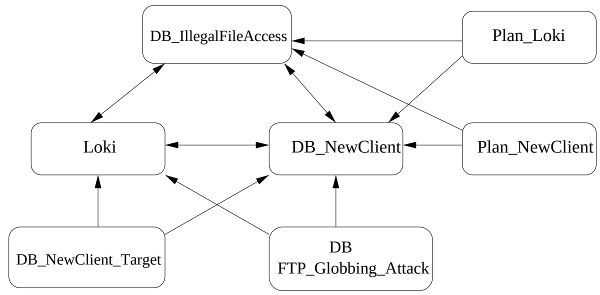

Fig. 2. The GCP Scenario I: Correlation Graph on Database Server

ity directions. For example, Table 1 shows that alert *DB_FTP_Globbing_Attack* is a causal alert candidate with regard to alert *Loki*. Such causal relationship is shown by a directed edge originated from *DB_FTP_Globbing_Attack* pointing to *Loki* in Figure 2. A bi-directional edge indicates a mutual causal relationship between two alerts.

Figure 2 shows that there are multiple types of relationships among the alerts. First, there is a *straightforward* causal relationship that is obvious because of the nature of corresponding attacks. In Figure 2, we can see that alert *DB_FTP_Globbing_Attack* is causally related to alerts *Loki* and *DB_NewClient*, so is alert *DB_NewClient_Target*. Such causality indicates that the corresponding activities represented by alert *DB_FTP_Globbling_Attack* and alert *DB_NewClient _Target* cause the activities indicated by alert *DB_NewClient* and *Loki*. The fact spreadsheet in the GCP document also supports the validity of such causality. The ground truth shows that the attacker first gets root access to the Database Server using the *FTP Globbling buffer overflow* attack, then transports the malicious agent to the Database Server. The activity of agent transfer is detected by an IDS that outputs alert *DB_NewClient_Target*. The buffer overflow attack and initial malicious agent transfer are followed by a series of forthcoming autonomous attacks from/against the Database Server. Such causal relationship is obvious and can also be discovered by other correlation techniques because once the attacker obtained the root access to the victim using the buffer overflow attack, he/she can easily launch other attacks from/against the target. Therefore, a simple rule is to correlate the buffer overflow attack with other following attacks at the same target.

Some *indirect* relationships among alerts can also be discovered by the GCT correlation. As shown in Figure 2, we can see that alerts *Plan_Loki* and *Plan NewClient* all have causal relationship with alerts *DB_IllegalFileAccess* (triggered by activities of illegal access to files at the Database Server) and *DB NewClient* (triggered by activities of connecting to a suspicious site). It is hard to correlate them together via traditional correlation techniques because they do not have a known relationship with the target alert *DB_NewClient*. From the ground truth in the GCP document, we can see that the attacker first compromises the Plan Server and then uses that host to break into the Database

Server. Alert *Plan_NewClient* indicates that the attacker downloads malicious agent from the external site to the *Plan_Server*. Alert *Plan_Loki* indicates the attacker uploads sensitive information from the *Plan_Server* to the external site. The malicious code is later transferred to the Database Server after a buffer overflow attack against the Database Server originated from the Plan Server.

Figure 2 also shows a pattern of loop relationships among alerts. We can see that alerts *DB_IllegalFileAccess*, *Loki* and *DB_NewClient* have mutual causal relationships with each other. Such pattern indicates that the occurrences of these three alerts are tightly coupled, i.e., whenever we see one alert, we expect to see another one forthcoming. The fact spreadsheet validates our results. The malicious agent autonomously gets access to the sensitive files and collects data (alert *DB_IllegalFileAccess*), uploads the stolen data to an external site (alert *Loki*), then downloads new agent software (alert *DB_NewClient*) and installs it (alert *DB_IllegalFileAccess*) on the Database Server, and then begins another round of the same attack sequence. GCT correlation results show a loop pattern of causal relationship among the corresponding alerts because these activities occur together.

When we correlate each target alert with other alerts using the GCT, we have some false causal alert candidates. For example, *HTTP_Java*, *HTTP_ActiveX* in Table 1. Overall, in this experiment, the true causality rate is 95.06% (77/81) and false causality rate is 12.6% (10/87) in this network enclave.

$Alert_i$	Target Alert	GCT Index
Plan_IIS_Generic_BufferOverFlow	Plan_Service_Status	20.21
Plan_Registry_Modified	Plan_Service_Status	20.18
IIS_Unicode_Attack	Plan_Service_Status	18.98
HTTP_Java	Plan_Service_Status	17.35
HTTP_Shells	Plan_Service_Status	16.28
HTTP_ActiveX	Plan_Service_Status	1.90

Table 4. Alert Correlation by the GCT on the GCP Scenario II: Target Alert: Plan_Service_Status

$Alert_i$	Target Alert	GCT Index
HTTP_Java	Plan_Host_Status	7.73
Plan_IIS_Generic_BufferOverflow	Plan_Host_Status	7.70
Plan_Registry_Modified	Plan_Host_Status	7.63
CGI_Null_Byte_Attack	Plan_Host_Status	7.56
Port_Scan	Plan_Host_Status	3.26
HTTP_RobotsTxt	Plan_Host_Status	1.67

Table 5. Alert Correlation by the GCT on the GCP Scenario II: Target Alert: Plan_Host_Status

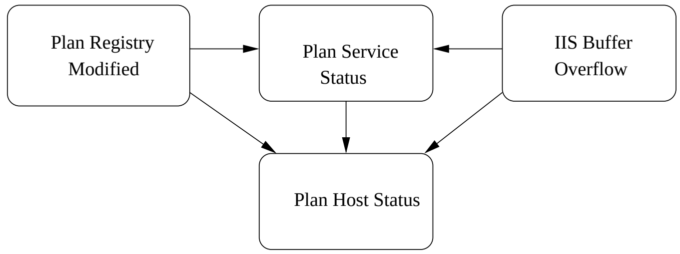

Fig. 3. The GCP Scenario II: Correlation Graph of Plan Server

We also use the same network enclave as an example to show our results in the GCP Scenario II. In this network enclave, there are a total of 387 hyper alerts. Applying the *Ljung-Box* test to the hyper alerts, we identify 273 hyper alerts as the background alerts. In calculating the priority of hyper alerts, there are 9 hyper alerts whose priority values are above the threshold $\beta = 0.6$. Therefore, we have 9 hyper alerts as the target alerts, which are correlated with other alerts excluding the identified background alerts. As before, based on the mission goals and alert priority, for example, we select two alerts, *Plan_Service_Status* and *Plan_Host_Status*, as the targets, then apply the GCT to correlate other alerts with them. Table 4 and Table 5 show the corresponding GCT results. We list the top 6 candidate alerts that have passed the F-test in the tables. The alerts *Plan_Host_Status* and *Plan_Service_Status* are issued by a network management system deployed on the network. The true causality rate is 93.15% (68/73) and false causality rate is 13.92% (11/79).

After finding the candidate alerts, we construct a corresponding correlation graph as shown in Figure 3. This figure shows that alerts *IIS_Buffer_Overflow* and *Plan_Registry_Modified* are causally related to alerts *Plan_Service_Status* and *Plan_Host_Status*. The GCP document verifies such relationship. The attacker launches *IIS Buffer Overflow* attack against the Plan Server in order to transfer and install the malicious executable code on it. The Plan Server's registry file is modified (alert *Plan_Registry_Modified*) and the service is down (alert *Plan_Service_Status*) during the daemon installation. Alert *Plan_Host_Status* indicates the "down" or "up" states of the Plan Server. The states are affected by the activities of the malicious agent installed on the Plan Server. Therefore, the ground truth described in the GCP document also supports the causal relationships among the corresponding alerts. These relationships are represented by directed edges pointing to alert *Plan_Host_Status* from alerts *IIS_Buffer_Overflow*, *Plan_Registry_Modified* and *Plan_Service_Status* in Figure 3.

However, the correlation result in the GCP Scenario II is not comprehensive enough to cover the complete attack scenarios. By comparing the alert streams with the GCP document, we notice that many malicious activities in the GCP Scenario II are not detected by the IDSs. Therefore, there are some missing intrusion alerts. In our approach, we depend on alert data for correlation and scenario analysis. When there is a lack of alerts corresponding to the intermediate attack steps, we cannot construct the complete attack scenario. In practice, IDSs or other security mechanisms can miss some attack activities. We will study how to deal with the "missing" attack steps in alert analysis and scenario construction.

4.2 DEF CON 9 Capture The Flag

As another case study, we applied our algorithms on the DEF CON 9 Capture The Flag (CTF) datasets. We use Snort to analyze the network traffic and output alerts for analysis. The DEF CON 9 CTF datasets are collected on 7 subnets. However, some datasets in subnet *Eth0* are corrupted. Therefore, we do not include them in our analysis. Because there is no information available about the network topology and host configuration, we cannot fully apply our model of alert priority computation on the datasets. Therefore, we select the target alerts for correlation based on domain knowledge.

As an example, we report results of alert analysis for *subnet 4*. Snort outputs more than 378,000 raw alerts. Scanning related alerts account for 91% of the total alerts. Alert *ICMP Redirect Host* accounts for about 3% of the total and alert *MISC Tiny Fragments* accounts for 5.9% of the total. Other alerts include Buffer Overflow, DDOS, DOS, DNS, TFTP, SNMP and Web-related attacks.

Applying our alert fusion and clustering algorithms, we can reduce the redundancy of low-level alerts dramatically, in particular, scanning alerts. The number of concrete high-level hyper alerts is about 1,300. We apply the *Ljung-Box* test with the significance level $\alpha = 0.05$ to all hyper alerts, and identify 754 hyper alerts as background alerts. For convenience, we denote the following: *Host_A* : 10.255.100.250, *Host_B* : 10.255.30.201, *Host_C* : 10.255.30.202, *Host_D* : 10.255. 40.237.

We first select the alert *DDOS Shaft Zombie* targeting at *Host_A*, and apply the GCT to correlate it with other alerts. Based on the correlation results, we select a causal alert as the next correlation target alert. For example, after each GCT correlation, we select the causal alert that is oriented from *host_C* as the target alert for the next GCT correlation. Table 6 through Table 8 show the corresponding GCT correlation results with regard to the selected target alerts, i.e., *DDoS_Zombie_Host_A*, *FTP_Command_Overflow_Host_C_Src*, *FTP_CWD_Overflow_Host_C_Src*. We construct the attack scenario graph based on GCT correlation results and alert analysis.

Figure 4 shows the attack scenario targeting *Host_A* according to the network activities in subnet 4. We can see that the attackers first launch a series of port scanning, e.g., *NMAP* and *RPC_Portmap*. Then multiple *FTP Buffer Overflow* attacks are launched against the target in order to get root access. The attackers also launch some Web-related attacks against the target. There are also some other attack scenarios that our algorithms are able to find; many of them are *port scanning* followed by *Buffer Overflow* attacks.

$Alert_i$	Target Alert	GCT Index
FTP_Command_Overflow_Host_B_Src	DDOS_Shaft_Zombie	13.43
FTP_User_Overflow_Host_B_Src	DDOS_Shaft_Zombie	12.98
FTP_Command_Overflow_Host_C_Src	DDOS_Shaft_Zombie	11.43
WEB-CGI_ScriptAlias_Access	DDOS_Shaft_Zombie	11.12
TFT_GetPasswd_Host_B_Src	DDOS_Shaft_Zombie	10.88
FTP_Aix_Overflow_Host_B_Src	DDOS_Shaft_Zombie	10.83
EXPERIMENTAL_MISC_AFS_Access	DDOS_Shaft_Zombie	10.70
FTP_CWD_Overflow_Host_D_Src	DDOS_Shaft_Zombie	10.68
WEB-CGI_Wrap_Access	DDOS_Shaft_Zombie	10.54
FTP_Command_Overflow_Host_D_Src	DDOS_Shaft_Zombie	10.35
FTP_CWD_Overflow_Host_C_Src	DDOS_Shaft_Zombie	9.87
FTP_OpenBSDx86_Overflow_Host_D_Src	DDOS_Shaft_Zombie	7.86
WEB-CGI_WebDist_Access	DDOS_Shaft_Zombie	7.54

Table 6. DefCon 9: Target Alert: DDOS_Shaft_Zombie_Host_A

$Alert_i$	Target Alert	GCT Index
Scan_NMAP_TCP	FTP_Command_Overflow_Host_C_Src	11.27
ICMP_Ping_NMAP	FTP_Command_Overflow_Host_C_Src	10.93
WEB-MISC_Perl_Command	FTP_Command_Overflow_Host_C_Src	10.75
Xmas_Scan	FTP_Command_Overflow_Host_C_Src	10.23
RPC_Portmap_Request	FTP_Command_Overflow_Host_C_Src	10.17
FIN_Scan	FTP_Command_Overflow_Host_C_Src	10.13
NULL_Scan	FTP_Command_Overflow_Host_C_Src	10.11

Table 7. DefCon 9: Target Alert: FTP_Command_Overflow_Host_C_Src

$Alert_i$	Target Alert	GCT Index
Scan_NMAP_NULL	FTP_CWD_Overflow_Host_C_Src	12.72
ICMP_Ping_NMAP	FTP_CWD_Overflow_Host_C_Src	12.12
WEB-MISC_Perl_Command	FTP_CWD_Overflow_Host_C_Src	11.87
Xmas_Scan	FTP_Command_Overflow_Host_C_Src	11.63
SYN FIN_Scan	FTP_CWD_Overflow_Host_C_Src	11.27
NULL_Scan	FTP_CWD_Overflow_Host_C_Src	10.92

Table 8. DefCon 9: Target Alert: FTP_CWD_Overflow_Host_C_Src

4.3 Discussion

In our experiments, the results from the GCP show that our approach can correlate alerts of the attacks that have statistical causal relationships. The statistical causal correlations among alerts are not limited to the patterns of co-occurrences, e.g., alert *New_Client* and *Loki* in the GCP Scenario I. It also includes the case where the causal attack occurs only once, e.g., the sequential attacks in the scenario of the DEF CON 9. However, as the GCT results show, we have false causal alert candidates resulted from the GCT correlation that can result in false scenarios. One reason is that a large amount of background alerts can increase

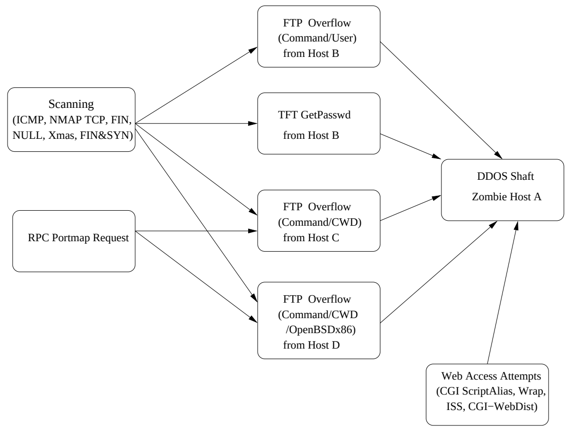

Fig. 4. DefCon 9: A scenario example of victim Host A

the false correlations. For example, we have a relatively high false causality rate in the GCP because the GCP has a lot of background alerts. Another reason is that, in our experiments, we do not have and use any training data sets. Therefore, it is different from traditional anomaly detection in which training data is used to construct the baseline that can reduce the false positive rate. In the DEF CON 9 dataset, our approach also finds some reasonable scenarios. Because of the nature of the DEF CON 9 dataset, we cannot comprehensively evaluate the success rate of our alert correlation method.

The key strength of our approach is that it can discover new alert correlations. Another advantage of our approach is that we do not require *a priori* knowledge about attack behaviors and how attacks are related when finding candidate alert correlations. In addition, our approach can also reduce the workload of security analysts in that they can focus on the causal alert candidates output by the GCT for further analysis. They do not have to assess all alerts and investigate all possible correlations. This is especially helpful when an attack is in progress and the security analysts need to figure out the attack scenarios in a timely fashion.

The time series used in our approach is based on the alert count instead of other categorical variables such as port access and pattern of TCP flag. The intuition is that if two attacks are related or have causal relationships, their occurrences should be tightly correlated because the causal attack triggers the resulting attack. Some experimental work and theoretical analysis have been presented in [1, 3, 2]. However, it is important to consider categorical variables when constructing attack scenarios. We will address this issue in our future work.

One challenge to our approach is background alert identification. Using the *Ljung-Box* test cannot cover all the background alerts. The limit of our approach is that we still need expert knowledge to further inspect the causal alert candidates resulted from GCT alert correlation when constructing attack scenarios.

Human intervention has limits in automating attack scenario constructions. In future work, we will develop new correlation algorithms, in particular, probabilistic reasoning, and will integrate other existing correlation algorithms, e.g., *prerequisite-consequence* approach, for alert correlation in order to reduce the false correlation rate and improve the accuracy of scenario analysis.

5 Related Work

Recently, there have been several proposals on alert correlation and attack scenario analysis.

Porras et al. design a "mission-impact-based" correlation system, named M-Correlator [25]. The main idea is to evaluate alerts based on security interests and attack relevance to the protected networks and hosts. Related alerts are aggregated and clustered into a consolidated incident stream. The final result of the M-Correlator is a list of rank ordered security incidents based on the relevance and priority scores, which can be further analyzed by the security analyst. This approach focuses on the incident ranking instead of attack scenario analysis. The security analyst needs to perform further correlation analysis.

Valdes and Skinner [27] develop a probabilistic-based alert correlation mechanism. The approach uses similarities of alert attributes for correlation. Measures are defined to evaluate the degree of similarity between two alerts. Alert aggregation and scenario analysis are conducted by toughening or relaxing the similarity requirement in some attribute fields. However, it is difficult for this approach to correlate alerts that do not have obvious (or predefined) similarities in their attributes.

In the approach proposed by Debar and Wespi [7], alert clustering is applied for scenario construction. Two reasoning techniques are used to specify alert relationships. Backward-reasoning looks for *duplicates* of an alert, and forward-reasoning determines if there are *consequences* of an alert. These two types of relationships between alerts are predefined in a configuration file. The main limitation of this approach is that it relies on the predefined duplicate and consequence relationships between alerts.

Goldman et al. [10] build a correlation system that produces a correlation graph, which indicates the security events that aim to compromise the same security goal, with IDS alerts as the supporting evidence of the security events. The reasoning process is based on the predefined goal-events associations. Therefore, this approach cannot discover attack scenarios if the attack strategy or objective is not known.

Some other researchers have proposed the framework of alert correlation and scenario analysis based on the pre-condition and post-condition of individual alerts [4, 5, 22]. The assumption is that when an attacker launches a scenario, prior attack steps are preparing for later ones, and therefore, the consequences of earlier attacks have a strong relationship with the prerequisites of later attacks. The correlation engine searches for alert pairs that have a consequences and prerequisites match and builds a correlation graph with such pairs. There are several limitations with this approach. First, a new attack may not be paired with any other attack because its prerequisites and consequences are not yet defined. Second, even for known attacks, it is infeasible to predefine all possible prerequisites and consequences. In fact, some relationships cannot be expressed naturally in rigid terms.

Our approach differs from prior work in that it focuses on discovering *new* and *unknown* attack strategies. Instead of depending on the prior knowledge of attack strategies or pre-defined alert pre/post-conditions, we correlate the alerts and construct attack scenarios based on statistical and temporal relationships among alerts. In this respect, our approach is analogous to *anomaly detection* techniques.

We also notice that alert correlation has been a research topic in network management for decades. There are several well-known approaches such as case-based reasoning system [19], code-book [18], and model-based reasoning systems [16, 23]. In network management system (NMS), event correlation focuses on alarms resulted from network faults, which often have fixed patterns. Whereas in security, the alerts are more diverse and unpredictable because the attackers are intelligent and can use flexible strategies. We nevertheless can borrow ideas in NMS event correlation for INFOSEC data analysis.

6 Conclusion and Future Work

In this paper, we presented an approach for correlating INFOSEC alerts and constructing attack scenarios. We developed a mechanism that aggregates and clusters raw alerts into high level hyper-alerts. Alert priority is calculated and ranked. The priority computation is conducted based on the relevance of the alert to the protected networks and systems. Alert correlation is conducted based on the Granger Causality Test, a time series-based causal analysis algorithm. Attack scenarios are analyzed by constructing a correlation graph based on the GCT results and on alert inspection. Our initial results have demonstrated the potential of our method in alert correlation and scenario analysis. Our approach can discover new attack relationships as long as the alerts of the attacks have statistical correlation. Our approach is complementary to other correlation approaches that depend on hard-coded prior knowledge for pattern matching.

We will continue to study statistical-based approaches for alert correlation, and develop algorithms to detect background alerts, develop techniques to integrate categorical variables such as patterns of TCP flags, and study how to reduce false causality rate. We will also develop other correlation algorithms, in particular, probabilistic reasoning approaches, to integrate multi-algorithms for alert correlation and scenario analysis. We will also study how to handle missing alerts of attack steps in scenario analysis. One approach may be to insert some hypothesis alerts and look for evidence to either support or degrade the hypothesis from other sensor systems. We will validate our correlation algorithms on alert streams collected in the real world.

7 Acknowledgments

This research is supported in part by a grant from DARPA (F30602-00-1-0603) and a grant from NSF (CCR-0133629). We thank João B.D. Cabrera of Scientific Systems Company for helpful discussions on time series analysis and Granger-Causality Test. We also thank Giovanni Vigna of University of California at Santa Barbara, Alfonso Valdes of SRI International and Stuart Staniford of Silicon Defense, as well as the anonymous reviewers for their valuable comments and suggestions.

References

1. J. B. D. Cabrera, L. Lewis, X. Qin, W. Lee, R. K. Prasanth, B. Ravichandran, and R. K. Mehra. Proactive detection of distributed denial of service attacks using mib traffic variables - a feasibility study. In *Proceedings of IFIP/IEEE International Symposium on Integrated Network Management (IM 2001)*, May 2001.
2. J. B. D. Cabrera and R. K. Mehra. Extracting precursor rules from time series - a classical statistical viewpoint. In *Proceedings of the Second SIAM International Conference on Data Mining*, pages 213–228, Arlington, VA, USA, April 2002.
3. J.B.D. Cabrera, L.Lewis, X. Qin, W. Lee, and R.K. Mehra. Proactive intrusion detection and distributed denial of service attacks - a case study in security management. *Journal of Network and Systems Management*, vol. 10(no. 2), June 2002.
4. S. Cheung, U. Lindqvist, and M. W. Fong. Modeling multistep cyber attacks for scenario recognition. In *Proceedings of the Third DARPA Information Survivability Conference and Exposition (DISCEX III)*, Washington, D.C., April 2003.
5. F. Cuppens and A. Miège. Alert correlation in a cooperative intrusion detection framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, pages 202–215, Oakland, CA, May 2002.
6. DAPRA Cyber Panel Program. DARPA cyber panel program grand challenge problem (GCP). http://ia.dc.teknowledge.com/CyP/GCP/, 2003.
7. H. Debar and A. Wespi. The intrusion-detection console correlation mechanism. In *4th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2001.
8. DEFCON. Def con capture the flag (ctf) contest. http://www.defcon.org. Archive accessible at http://wi2600.org/mediawhore/mirrors/shmoo/, 2000.
9. DEFCON. Def con capture the flag (ctf) contest. http://www.defcon.org. Archive accessible at http://smokeping.planetmirror.com/pub/cctf/defcon9/, 2001.
10. R.P. Goldman, W. Heimerdinger, and S. A. Harp. Information modleing for intrusion report aggregation. In *DARPA Information Survivability Conference and Exposition (DISCEX II)*, June 2001.
11. C.W.J. Granger. Investigating causal relations by econometric methods and cross-spectral methods. *Econometrica*, 34:424–428, 1969.
12. IETF Intrusion Detection Working Group. Intrusion detection message exchange format. http://www.ietf.org/internet-drafts/draft-ietf-idwg-idmef-xml-09.txt, 2002.
13. J. Haines, D. K. Ryder, L. Tinnel, and S. Taylor. Validation of sensor alert correlators. *IEEE Security & Privacy Magazine*, January/February, 2003.
14. J. Hamilton. *Time Series Analysis.* Princeton University Press, 1994.

15. A.J. Hayter. *Probability and Statistics for Engineers and Scientists.* Duxbury Press, 2002.
16. G. Jakobson and M. D. Weissman. Alarm correlation. *IEEE Network Magazine*, November 1993.
17. K. Julisch and M. Dacier. Mining intrusion detection alarms for actionable knowledge. In *The 8th ACM International Conference on Knowledge Discovery and Data Mining*, July 2002.
18. S. Kliger, S. Yemini, Y. Yemini, D. Oshie, and S. Stolfo. A coding approach to event correlations. In *Proceedings of the 6th IFIP/IEEE International Symposium on Integrated Network Management*, May 1995.
19. L. Lewis. A case-based reasoning approach to the management of faults in communication networks. In *Proceedings of the IEEE INFOCOM*, 1993.
20. G.M. Ljung and G.E.P. Box. On a measure of lack of fit in time series models. In *Biometrika 65*, pages 297–303, 1978.
21. P. Ning, Y. Cui, and D.S. Reeves. Analyzing intensive intrusion alerts via correlation. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2002.
22. P. Ning, Y. Cui, and D.S. Reeves. Constructing attack scenarios through correlation of intrusion alerts. In *9th ACM Conference on Computer and Communications Security*, November 2002.
23. Y. A. Nygate. Event correlation using rule and object based techniques. In *Proceedings of the 6th IFIP/IEEE International Symposium on Integrated Network Management*, May 1995.
24. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers, Inc, 1988.
25. P. A. Porras, M. W. Fong, and A. Valdes. A Mission-Impact-Based approach to INFOSEC alarm correlation. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2002.
26. W. Stallings. *SNMP, SNMPv2, SNMPv3, and RMON 1 and 2.* Addison-Wesley, 1999.
27. A. Valdes and K. Skinner. Probabilistic alert correlation. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2001.