Attack Plan Recognition and Prediction Using Causal Networks

Xinzhou Qin and Wenke Lee
Georgia Institute of Technology
Atlanta, GA 30332, U.S.A.
{xinzhou, wenke}@cc.gatech.edu

Abstract

Correlating and analyzing security alerts is a critical and challenging task in security management. Recently, some techniques have been proposed for security alert correlation. However, these approaches focus more on basic or low-level alert correlation. In this paper, we study how to conduct probabilistic inference to correlate and analyze attack scenarios. Specifically, we propose an approach to solving the following problems: 1) How to correlate isolated attack scenarios resulted from low-level alert correlation? 2) How to identify attacker's high-level strategies and intentions? 3) How to predict the potential attacks based on observed attack activities? We evaluate our approaches using DARPA's Grand Challenge Problem (GCP) data set. The results demonstrate the capability of our approach in correlating isolated attack scenarios, identifying attack strategies and predicting future attacks.

Keywords: Intrusion detection, alert correlation, security management, attack scenario analysis.

1. Introduction

Many security sensors and systems can be deployed to provide defense-in-depth for systems and networks. However, the large volume of security alerts makes it challenging for security operators to analyze the attack situation and take an appropriate response.

Alert correlation and analysis is a critical task in security management. Recently, several techniques and approaches have been proposed to correlate and analyze security alerts, including alert similarity measurement [30], probabilistic reasoning [16, 25], clustering algorithms [14], pre-/post-condition matching of known attacks [21, 11, 7], statistical-based analysis [24] and chronicles formalism approach [20]. All these approaches focus on the aggregation and analysis of raw security alerts, and build basic or low-level attack scenarios. However, in practice, an alert correlation system should have a hierarchical architecture. The analysis is conducted from low-level alert correlation to abstract scenario analysis at high levels. In addition, there always exist isolated attack scenarios derived from low-level alert correlation due to various reasons, e.g., IDSs miss detecting critical attacks. Therefore, in addition to the low-level correlation analysis, it is necessary to develop algorithms and tools for security analysts to further analyze and correlate attack scenarios so that they can make situation and mission assessment accurately, and take appropriate responses to minimize the damages. In addition, threat analysis and attack prediction are also helpful and important for security operators to take actions in advance to avoid potential attacks and damages.

Recognizing attack plans is one of the goals of security analysts. Plan recognition has been a research area in artificial intelligence (AI) for decades. In AI, plan recognition is a process of inferring the goals of an agent from observations of the agent's activities. Plan recognition can be characterized as *keyhole recognition* and *intended recognition* based on the role of an agent whose plan is being inferred [9]. In *keyhole recognition*, the agent is not aware that its action is being observed, i.e., the agent is only engaged in the task and does not attempt to impact the recognition process. In *intended recognition*, the agent attempts to perform actions that can aid the recognition of its plan, e.g., a language understanding system [9].

However, in attack plan recognition, traditional plan recognition techniques cannot be applied. Unlike the traditional agent that either *aids* the recognition of its plan or *does not attempt to impact* the recognition of the process, attackers can perform activities to escape detection and avoid the recognition of their attack strategies. Therefore, this type of recognition process can be categorized as *adversary recognition* that is more challenging and more uncertain in the recognition process. In addition, some assumptions of traditional plan recognition techniques are not valid anymore. First, in plan recognition, there is always an assumption that there exists a valid plan in the plan library that the agent can reach. In network security, we cannot assume that we have a complete attack plan library that in-

cludes all of the possible strategies of attackers. Therefore, we have to deal with the case that the observed attacker's activity is beyond or partially matched with our pre-defined attack plans. Second, plan recognition assumes a complete, ordered set of tasks for a plan. However, in security context, we cannot always observe all of the attacker's activities, and often can only detect incomplete attack steps due to the limitation or deployment of security sensors. Therefore, the attack plan recognition system should have the capability of dealing with partial order and unobserved activities.

In this paper, we develop a series of techniques to solve three problems. First, we consider how to correlate isolated attack scenarios derived from low-level alert correlation. Second, we address how to recognize the attacker's attack plan and intentions. Third, we discuss how to make predictions of potential attacks based on current observations and analysis. In our approach, we apply graph techniques to correlate isolated attack scenarios and identify their relationship. Based on the high-level correlation results, we further apply probabilistic reasoning technique to recognize the attack plans, evaluate the likelihood of potential attack steps and predict upcoming attacks.

The remainder of this paper is organized as follows. Section 2 discusses the related work. In Section 3, we briefly introduce two alert correlation techniques upon which we build our new techniques. Section 4 describes our correlation models and attack prediction algorithm. We report our experiment and results in Section 5. Section 6 concludes this paper and discusses some future research directions.

2. Related Work

2.1. Alert Correlation

Recently, there have been several proposed approaches to alert correlation and attack scenario analysis.

Valdes and Skinner [30] used probabilistic-based reasoning to correlate alerts by measuring and evaluating the similarities of alert attributes. Alert aggregation and scenario construction were conducted by enhancing or relaxing the similarity requirements in some attribute fields. Goldman et al. [16] built a correlation system based on Bayesian reasoning. The system predefines the relationship between mission goals and corresponding security events. Further inference and correlation rely on the predefined association.

Porras et al. designed a "mission-impact-based" correlation system [23] with focuses on the impact analysis based on the mission goals of protected networks.

Debar and Wespi [14] applied backward and forward reasoning techniques to correlate alerts with *duplicates* and *consequence* relationships, and used clustering algorithms to detect attack scenarios and situations.

Morin and Debar [20] applied *chronicles formalism* to aggregate and correlate alerts. The approach is to perform attack scenario pattern recognition based on *known* malicious event sequences. Therefore, this approach is analogous to *misuse intrusion detection*.

Ning et al. [21], Cuppens and Miège [11] and Cheung et al. [7] built alert correlation systems based on the preconditions and post-conditions of individual alerts. The correlation engine searches alert pairs that have a consequence and prerequisite matching. Further correlation graphs can be built with such alert pairs [21].

Qin and Lee [24] proposed a statistical-based alert correlation approach to identifying new alert relationship without depending on prior knowledge of attack transition patterns.

2.2. Plan Recognition

In artificial intelligence (AI), plan recognition has been an active research area. Different types of inference techniques have been applied to plan recognition, e.g., *deduction* and *abduction*. In particular, the earliest work in plan recognition was rule-based inference system [26, 31]. A milestone work of plan recognition was done by Kautz and Allen in 1986 [19]. In [19], they defined the problem of plan recognition as finding a minimal set of top-level actions, i.e., plan goals, that were sufficient to explain the observed actions. The inference was conducted by going through the rule sets. Charniak and McDemott [6] proposed that the plan recognition problem can be solved by *abduction*, or reasoning to the best explanation. Charniak and Goldman [4, 5] applied Bayesian networks to plan recognition. Carberry [3] applied Dempster-Shafer theory [28] to compute the combined support by multiple evidences to hypotheses plans. Albrecht et al. [1] proposed to construct a plan recognition inference system based on *Dynamic Belief Networks* [13]. In Dynamic Belief Networks, the influence of temporal aspects is represented by multiple nodes to indicate the status of a variable at different instances of time.

There are some challenges in applying traditional plan recognition techniques to security applications. First, traditional plan recognition techniques are usually applied in non-adversary situation. The recognition process can be either aided or non-interfered by the agent being observed. However, in the security application, the plan recognition process is an *adversary recognition* where attackers are trying to avoid or interfere with any recognition process on their intrusion activities.

Second, the assumptions used in traditional plan recognition are not valid in adversary recognition anymore. For example, in non-adversary plan recognition, a single agent and a single plan has to be determined. The observed activ-

ities are conducted by a single agent toward a single plan. Although there are some works on 'multi-agent' plan recognition, they also share this assumption. However, in attack plan recognition, it is possible that an attacker has multiple dynamic attack plans. There also exist coordinated attacks conducted by multiple attackers. In addition, in non-adversary plan recognition, there is a complete, ordered and correct set of activities. The observations available are correct and corresponding to a determined plan. Every action that is performed is observed. In adversary plan recognition, this assumption is not valid anymore.

The most related work to ours is [15] in which Geib and Goldman applied probabilistic reasoning to recognize the attacker's intentions. The approach conducts the plan recognition from raw security alerts. The plan library is defined by detailed specific attacks. This definition method has the limitation that it can increase the computation complexity of inference. In addition, it also requires a complete and ordered attack sequence (if there are missing attack steps, it inserts hypothesized attack steps in order to have a complete activity sequence) when conducting the plan recognition.

Our approach is unique in the following aspects. First, we build our plan recognition system after a low-level alert correlation step that includes alert aggregation, alert prioritization and alert correlation. The advantage of this approach is that it can reduce the computation complexity when performing the high-level attack scenario correlation and probabilistic inference. Second, we do not require a complete ordered alert sequence for inference. We have the capability of handling partial order and unobserved activity evidence sets. In practice, we cannot always observe all of the attacker's activities, and can often only detect partial order of attack steps due to the limitation or deployment of security sensors. For example, security sensors such as IDS can miss detecting intrusions and thus result in an incomplete alert stream. Third, we provide an approach to predicting potential attacks based on observed intrusion evidence.

3. Overview of Statistical and Probabilistic Reasoning Techniques for Alert Correlation

The new techniques of attack plan recognition and prediction described in this paper are built on the alert correlation approaches proposed in [24] and [25]. In this section, we briefly introduce our approaches of alert correlation.

The first processing step is *alert aggregation and clustering*. The goal if this step is to reduce the redundancy of duplicated raw alerts corresponding to same attacks output by heterogeneous security sensors. Alert aggregation is conducted based on alert attributes such as *time stamp, source IP, destination IP, port(s)*, etc. Aggregated alerts with the same attributes (except time stamps) are grouped into one cluster. A *hyper alert* is defined as a time ordered sequence of alerts that belong to the same cluster.

The next step is *alert prioritization* that prioritizes each hyper alert based on its relevance to the mission goals. 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 protected networks and hosts, as well as the severity of the corresponding attack assessed by the security analyst. When computing priority values, 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. The relevance check downgrades the impacts of some alerts unrelated to the protected domains on further correlation analysis, e.g., attackers "blindly" launch attacks against a target that has no corresponding vulnerabilities.

We apply two techniques to alert correlation. First, we use probabilistic-based reasoning method to correlate attack steps that are directly related because an earlier attack enables or positively affects the later one [25]. For example, a port scan may be followed by a buffer overflow attack on a scanned service port. We apply Bayesian-based correlation mechanism to reason and correlate attack steps based on security states of systems and networks. This approach can incorporate prior knowledge of attack transition patterns and handle uncertainty in the correlation process. Second, we apply statistical analysis to correlate attack steps that have *temporal and statistical* patterns even though they do not have obvious or direct relationship in terms of security and performance measures [24]. In particular, we apply Granger-Causality analysis [17] and some other time series analysis techniques to detect the "causal" relationship between this type of alert pairs. This approach does not require the prior knowledge of attack scenario patterns in the correlation process.

Alert correlation results in a set of correlated alerts that comprise the attack scenarios. The alert processing described earlier is usually conducted on aggregated raw alerts and reflects localized or low-level attack scenarios, e.g., a series of attacks against a department network. Low-level alert correlation can leave some correlated alert sets that are isolated from each other. Based on the attack scenarios resulted from the prior alert aggregation, alert prioritization and correlation, we further correlate isolated alert sets and conduct high-level attack plan recognition and prediction, which are the topics of this paper.

4. Models and Algorithms

In this section, we introduce our models and algorithms for correlating isolated alert sets, attack plan recognition and attack prediction.

4.1. Attack Tree Analysis

In security operations, security analysts usually predefine a set of attack plans or attack libraries that incorporate the domain knowledge of attacks or attack scenario patterns, and the knowledge of the networks and systems under protection. Attack plans or libraries are usually represented by graphs (i.e., attack graphs) that show all paths through a system that end in a state where an intruder can successfully achieve his goal. Schneier [27] described *attack tree analysis* that quantifies the security or vulnerability of a system based on the goals of the attacker. When defining the attack trees, security analysts first evaluate the vulnerabilities of the systems and networks, then pretend to be attackers and work out attack plans to achieve the intrusion goals. In this process, an attack tree is extended and branches are built to identify the different subgoals of the attacker and penetration points available to the attacker. The process continues by decomposing or expanding the means of penetration to the lowest level of intrusion, known as the leaves. An attack tree can represent each opportunity for an attack against a computer system or network. Computer systems and networks potentially contain numerous penetration points and vulnerabilities. An attack forest is defined as a consolidation of numerous attack trees [27].

Figure 1(a) shows an example of an attack tree that indicates attack methods to steal the data stored on a server and export it to the external. In the Figure 1(a), the "OR" node represents *different ways* to achieve the goals. In practice, in addition to the "OR" node, the "AND" node is also always used in an attack tree to represent *different steps* to achieve the intrusion goals.

Attack tree analysis can serve as a basis for intrusion detection, defense, response and forensic analysis. However, defining attack trees is a very challenging task. It is usually done manually and is very time consuming. Recently, Sheyner et. al [29] proposed a model checking-based technique to automatically construct attack graphs. Although it helps facilitate the task of defining attack graphs, the approach still has the limitation of scalability, in particular, when defining the attack graphs for a large network and computer systems.

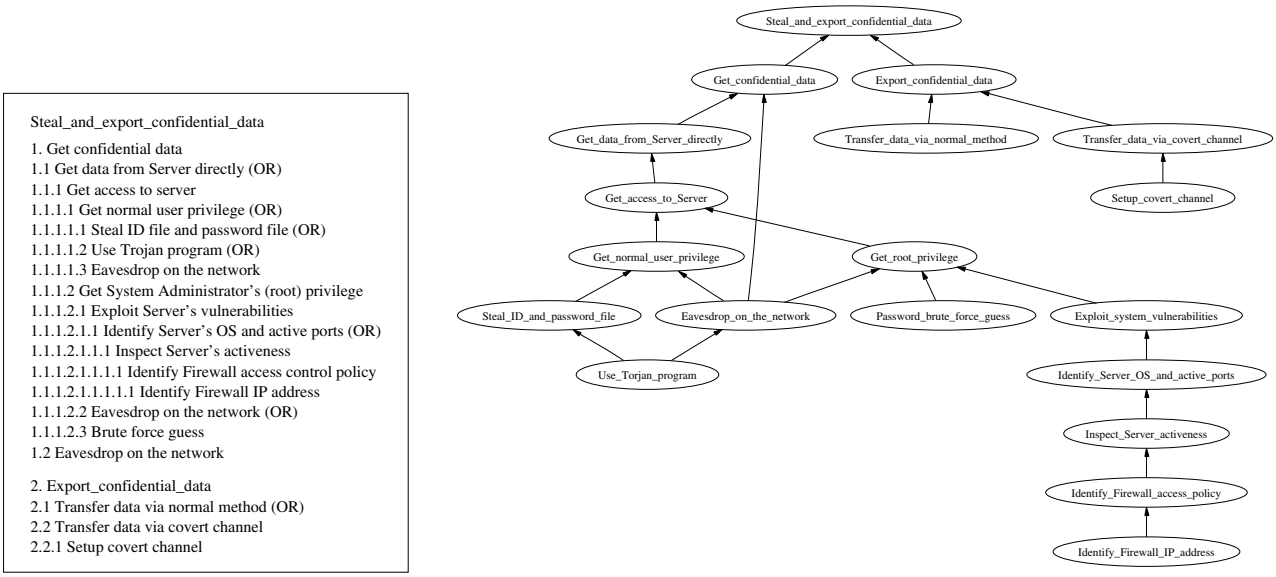
In our approach, we first use attack trees to define attack plan libraries to correlate isolated alert sets. We then convert attack trees into causal networks on which we can assign probability distribution by incorporating domain knowledge to evaluate the likelihood of attack goals and predict future attacks. Figure 1(b) shows an example of the causal network converted from the attack tree as shown in Figure 1(a). In defining attack trees, instead of using various specific attacks to define the nodes of an attack tree, we use the abstract attack class or type to represent an attack approach. For example, we use *Exploit Server Vulnerability* instead of a specific buffer overflow attack to indicate the method to break into a server to get the root access. The advantage of using attack classes to represent attack tree nodes is that it can reduce the computation complexity of probabilistic inference on the causal network that is converted from attack trees. It is well known that querying an arbitrary causal network is an NP-hard problem [10]. Therefore, in practice, a causal network is usually defined in the form of *causal polytrees* (i.e., singly-connected causal networks in which no more than two paths exist between any two nodes) so that the probabilistic reasoning can be conducted in polynomial time [22].

4.2. The Causal Network and its Parameters

A causal network (or Bayesian network) is usually represented as a directed acyclic graph (DAG) where each node represents a variable that has a certain set of states, and the directed edges represent the causal or dependent relationships among the variables. A Bayesian network consists of several parameters, i.e., prior probability of parent node's states (i.e., $P(parent\_state = i)$), a set of conditional probability tables (CPT) associated with child nodes. CPT encodes the prior knowledge between child node and its parent node. Specifically, an element of the CPT at a child node is defined by $CPT_{ij} = P(child\_state = j | parent\_state = i)$.

In our study, we build the causal networks based on attack trees and apply probabilistic inference. The root node of a causal network represents the final goal of an attack plan, non-leaf nodes represent subgoals, and leaf nodes indicate the nodes receiving evidence. We define each node of the causal network to have a binary state, i.e., 1 or 0. The value of 1 represents the goal is achieved for goal or subgoal nodes, while the value of 0 indicates the failure of the goal or subgoals. When a leaf node has a state value of 1, it indicates that the leaf node has received evidence. Otherwise, the leaf node has a value of 0.

When converting attack trees to a causal network, we can map "OR" nodes from an attack tree directly to the causal network while keeping the "OR" logical relationship. As "AND" nodes in an attack tree represent different *attack steps* to reach a goal ("OR" nodes indicate different *attack ways* to achieve an attack goal), there always exists an implicit dependent and sequential relationship between "AND" nodes in an attack tree. Therefore, we should keep such "causal" order when constructing the causal net-



(a) An example of attack tree

(b) An example of causal network

Figure 1. Attack tree and causal network

work. For example, we can define an attack tree for getting access to a server with the following attack steps which have "AND" relationship, i.e., *exploit vulnerability* AND *identify server OS* AND *identify Firewall access control policy* AND *Identify Firewall IP address*. In the causal network, we can keep the implicit sequential order among the nodes, i.e., *identify IP address*, *identify firewall access control policy*, *identify server's OS*, *exploit vulnerability* in order of the causal sequence.

When using a causal network (or Bayesian network) for inference, we need to set up two types of parameters, i.e., prior probability of parent node's states and CPT associated with each child node.

The prior probability of parent node's states (e.g., $P(parent\ node\ state = 1)$) used in the inference engine is set based on the *prior knowledge* estimation of the possibility. We used domain-specific knowledge based on prior experience and empirical studies to estimate appropriate probability values. In particular, we computed the probability of parent node's states based on historical data.

In our approach, CPT values associated with each node are adaptive to new evidence and therefore can be updated accordingly. We apply an adaptive algorithm originally proposed by [2]. The motivation of using an adaptive Bayesian network is that we want to fine-tune the parameters of the model and adapt the model to the evidence to fix the initial CPTs that may be pre-defined inappropriately. The intuition of the algorithms proposed by [2] is that we want to adapt the new model by updating CPT parameters to fit the new data cases while balancing the extent that we move away from the current model.

Specifically, we denote X as a node in a causal network, and let U be the parent node of X. X can take values of x_k, where $k = 1, ..., r$ and U has state values of u_j, where $j = 1, ..., q$. An entry of CPT of the node X can be denoted as: $\theta_{jk} = P(X = x_k|U = u_j)$. Given a set of new data cases, denoted as D, $D = y_1, ..., y_n$, and assuming there is no missing data in evidence vector of y_t, where evidence vector y_t represents the evidence at the t^{th} time, CPT updating rules can be simplified to the following [8]:

$$\theta^t_{jk} = \eta + (1-\eta)\theta^{t-1}_{jk}, \ \ for\ P(u_j|y_t) = 1 \ and\ P(x_k|y_t) = 1. \tag{1}$$

$$\theta^t_{jk} = (1-\eta)\theta^{t-1}_{jk}, \ \ for\ P(u_j|y_t) = 1 \ and\ P(x_k|y_t) = 0. \tag{2}$$

$$\theta^t_{jk} = \theta^{t-1}_{jk}, \ otherwise. \tag{3}$$

The above updating rules are interpreted as follows. Given the evidence vector y_t, if the parent node of X is observed in their j^{th} configuration, i.e., $U = u_j$, and X equals to its k^{th} value, then we regard the evidence as *supporting evidence* and we then increase the corresponding CPT value $P(X = x_k|U = u_j)$, i.e., θ_{jk}, as shown in Eq. (1). If

node X does not equal its k^{th} value but its parent node U is in the j^{th} configuration, then we regard the evidence as *non-supporting evidence* and then decrease the corresponding CPT value, i.e., θ_{jk}, as shown in Eq. (2). The learning rate η controls the rate of convergence of θ. η equaling 1 yields the fastest convergence, but also yields a larger variance. When η is smaller, the convergence is slower but eventually yields a solution to the true CPT parameter [8]. We built our inference model based on updating rules of Eq. (1) to Eq. (3).

We also need to point out that the adaptive capability of the inference model does not mean that we can ignore the accuracy of initial CPT values. If the initial values are set with a large variance to an appropriate value, it will take time for the model to converge the CPT values to the appropriate points. Therefore, this mechanism works for fine-tuning instead of changing CPT values dramatically. In practice, the initial CPT values can be computed and estimated using historical data.

4.3. Correlating Isolated Alert Sets

As discussed in Section 3, after processing raw alerts with alert aggregation, prioritization and correlation, we can reduce the large volume of raw alerts and correlate some of related alerts into different sets (or scenarios). However, it is possible that there exist some isolated correlated alert sets after the raw alert correlation due to various reasons. For example, for pattern-matching-based correlation approach, if the security sensors fail to detect some intermediate attacks in a series of coordinated attacks, the missing alerts can result in the un-match between observed alert sequences with known attack sequence patterns. The result is a set of isolated attack scenarios that belong to the same attack sequences. In addition, applying different correlation approaches together can also result in different correlation results due to the difference between correlation techniques. In such a case, it is also necessary to integrate correlation results output by different correlation engines and further correlate isolated alert sets. Third, from the security analyst's point of view, it is necessary to combine the local or low-level correlation results, investigate and assess the attack situation in order to make timely and appropriate response or prevention.

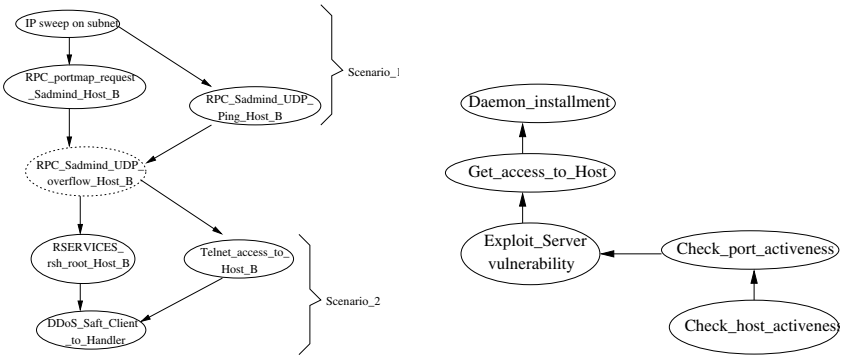
Given two individual isolated scenarios being studied, denoted as S_1 and S_2, where $S_1 = \{e_1, e_2, ..., e_i, ..., e_m\}$, $S_2 = \{e'_1, e'_2, ..., e'_i, ..., e'_n\}$ and e_i represents an alert (i.e., evidence), and given a set of attack plans, denoted as P, where $P = \{P_1, P_2, ..., P_k, ..., P_f\}$, and P_k is denoted as a specific attack plan that is represented by a causal network converted from attack trees, the problem is to find the relationship between S_1 and S_2. Algorithm 1 shows the method of correlating two isolated scenarios.

Algorithm 1 Correlation of isolated attack scenarios

Let $TPSet_1 = \{$Predecessor nodes of S_1 in $P_k\}$.
Let $TPSet_2 = \{$Predecessor nodes of S_2 in $P_k\}$.
Let P_k be an attack plan represented by a causal network, where $S_1 \in P_k$, and $S_2 \in P_k$.
Let $PSet_i = \{$Predecessor nodes of e_i in $P_k\}$, where $e_i \in S_1$.
Let $PSet'_i = \{$Predecessor nodes of e'_i in $P_k\}$, where $e'_i \in S_2$.
if $\exists e_j \in S_1$ and $e_j.attackClassNode \in PSet'_i$ and $e'_i.time < e_j.time$ and $\{e'_i.target = e_j.target$ or $e'_i.target = e_j.source\}$ **then**
  S_1 is a subgoal of S_2; S_1 and S_2 are directly related.
else if $\exists e'_j \in S_2$ and $e'_j.attackClassNode \in PSet_i$ and $e_i.time < e'_j.time$ and $\{e_i.target = e'_j.target$ or $e_i.target = e_j.source\}$ **then**
  S_2 is a subgoal of S_1; S_2 and S_1 are directly related.
else if $\{TPSet_1 \cap TPSet_2\} \neq \phi$ and they have the same target **then**
  S_1 and S_2 have indirect relationship with the same goal.
end if
Group all scenarios that are related in P_k into one evidence set.

The intuition of the correlation algorithm is to find out the relationship between two isolated attack scenarios. One relationship is that one scenario is a direct subgoal of another. This is indicated by the fact that one attack step in one scenario is a predecessor node of alerts of another scenario in the plan library. In such a case, a time constraint is applied to ensure that subgoal attack happens after the prior attacks. Another relationship is that two scenarios have an indirect relationship but have the same goal. For example, they both target at a same victim.

Figure 2(a) shows an example of how we correlate two isolated attack scenarios derived from low-level alert correlation. Assume $scenario_1$ includes alerts *IP sweep*, *RPC_Portmap_request_Sadmind_Host_B* and *RPC_Sadmind_UDP_Ping_Host_B*, and $scenario_2$ contains alerts *RSERVICES_rsh_root_Host_B*, *Telnet_access_to_Host_B* and *DDoS_Saft_Client_to_Handler*. In this case, we assume the alert *RPC_Sadmind_UDP_Overflow_Host_B* is missed that results in the isolation of $scenario_1$ and $scenario_2$. According to the corresponding attack plan as shown in Figure 2(b), attacks in $scenario_1$ have corresponding attack class nodes *Check_host_activeness* and *Check_port_activeness* in Figure 2(b). Similarly, the attacks in $scenario_2$ have corresponding abstract attack class nodes *Get_access_to_host* and *Daemon_installment* in Figure 2(b). From Figure 2(b), we can see that $scenario'_2 s$ abstract class nodes are predecessors of $scenario'_1 s$. There-



(a) Two correlated scenarios (b) The corresponding attack plan

Figure 2. An example of correlation of two isolated scenarios

fore, $scenario_2$ is actually a goal of $scenario_1$, i.e., the attacker gets ready to exploit the host vulnerability by launching attacks in $scenario_1$ so that he can access the host and install the DDoS daemon (as shown by attack steps in $scenario_2$). Although the alert corresponding to *Exploit host vulnerability* is missed, we can hypothesize the existence of such alert and apply the scenario correlation technique to correlate the two isolated scenarios, i.e., $scenario_1$ and $scenario_2$.

4.4. Probability Evaluation and Attack Prediction

We apply probabilistic inference to the causal network (or Bayesian network) to compute and evaluate the likelihood of goal and subgoals based on observed attack activities, and predict the potential upcoming attacks.

The inference process of a Bayesian network can be conducted by a series of belief propagations via message passing [22]. Specifically, each node X first receives a prior or causal support message from each of its parent nodes U, denoted as $\pi(u)$, where $\pi(u) = P(u|x)$. Each node X also receives evidence or diagnostic support message $\lambda(v)$ from each of its child nodes V, where $\lambda(v) = P(v|x)$. In runtime, when a node X is activated, it first updates the belief *Belief(x)*, i.e., the probability of X's states ($P(X = x|evidence)$), based on the evaluation values and the message $\pi(u)$ communicated with its parent nodes and $\lambda(v)$ sent by its child nodes. This is called *belief updating*. Next, the node X computes its own λ message based on λ messages received from its child nodes, and sends it to its parent nodes. This phase is *bottom-up propagation*. Finally, the node X computes its own π messages and send them to each of its child nodes. The final step is *top-down propagation*. In practice, the local belief update on node X can be executed by these three steps in any order. More detailed information can be found in [22].

Given a stream of evidence (i.e., alerts), and a causal network (i.e., attack plan) P, the inference through iterative belief updating is shown as follows:

Algorithm 2 Likelihood computation of goal and subgoal

Let P be a causal network and each node in the P has a binary state, {0,1}.
for all node $Y_i \in P$ that receives evidence e_i **do**
  Mark Y_i as an observed node with value of 1
  Let M be a collection nodes resulted from breath-first search starting from Y_i
  for all node $X \in M$ **do**
    Receive $\lambda(v)$ from all $X's$ child nodes, V.
    Receive $\pi(x)$ from all $X's$ parent nodes, U.
    Compute $\lambda_X(u)$ for all $X's$ parent nodes, U.
    Compute $\pi_v(x)$ for all $X's$ child nodes, V.
  end for
end for
for all non-leaf node $Z_i \in P$ **do**
  Compute $P(Z_i = 1|evidence)$ (i.e., the likelihood of Z_i)
end for

Algorithm 3 Potential goal(s) selection and attack prediction

for all non-leaf nodes $Z_i \in P$ **do**
  if $P(Z_i = 1|evidence)$ is the maximum or $P(Z_i = 1|evidence) > threshold$ **then**
    select Z_i as potential upcoming attack
  end if
end for

Intuitively, given a set of correlated alerts as observed evidence, we input the evidence into the causal network so that we can make inference and compute the likelihood for

each non-leaf node, denoted as Z_i, as shown in Algorithm 2. The computation result is used to infer the likelihood that a node can be the future goal(s) or future attack step, i.e., $P(Z_i = 1|evidence)$. As shown in Algorithm 3, In the final selection of possible future goal or attack steps, we can either select the node(s) that has the maximum belief value or the one(s) whose belief value is above a threshold.

5. Experiments

To evaluate the effectiveness of our alert correlation mechanisms, we applied our algorithms to one of the data sets of the Grand Challenge Problem (GCP) version 3.1 provided by DARPA's Cyber Panel program [12, 18], Scenario I.

GCP version 3.1 Scenario I contains an innovative worm attack scenario designed specifically to evaluate alert correlation techniques. In addition to the complicated attack scenarios, the GCP data sets also include many background alerts that make alert correlation and attack strategy detection more challenging. 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. Therefore, the GCP alerts are from both security systems and network management system. We applied our correlation techniques described in [24, 25] to aggregate, prioritize and correlate raw alerts and resulted in correlated alert sets.

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 select a network enclave as an example to show the process of scenario correlation and attack prediction.

Figure 3(a) shows an example of two isolated attack scenarios derived from low-level alert correlation, where *DB_FTP_Globbing_Attack* represents an buffer over flow attack against the database server, *DB_NewClient_Target* indicates an suspicious incoming connection to the database server from another server, *DB_Illegal_File_Access* represents the illegal access (write or read) to the database server, *DB_NewClient* indicates a suspicious outbound connection from database to an external host, and *Loki* means a suspicious data export via covert channel. In this case, we use the attack plan as defined in Figure 1(a). The corresponding causal network is shown in Figure 1(b). According to Figure 1(b), we can see that the alert sets {*DB_FTP_Globbing_Attack*, *DB_NewClient_Target*, *DB_Illegal_File_Access*} are corresponding to the attack steps with goals to get access to the server and get the data directly from the host, i.e., the attacker first applies buffer over flow attack against the database server, then sets up a covert channel to the host and export malicious code that is used to access to the database server to get the data. Alert sets {*DB_NewClient*, *Loki*} are attack steps that aim to set up covert channel and export confidential data to the outside. Figure 1(b) also shows that these two sets of alerts have an *indirect relationship* but the *same eventual goal* that is to steal the data from the database server and export it to the external. Therefore, applying the scenario correlation technique as described in Section 4.3, we can correlate these two scenarios as one integrated scenario, i.e., they are correlated with the same eventual goal, as shown in Figure 3(b), and group them together as one evidence set.

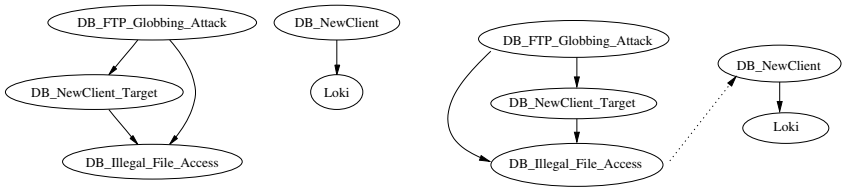
The advantage of correlating isolated scenarios is that we can accumulate more comprehensive evidence that can be used for further analysis, e.g., likelihood evaluation of each subgoal or final goal and attack prediction.

Also using database server as an example, based on the integrated evidence set, we apply probabilistic inference to the causal network as shown in Figure 1(b) to compute the likelihood of each subgoal and final goal. Table 1 shows the assessment of likelihood of some subgoals and final goal based on the evidence set. In Table 1, we show the probability result of two subgoals, i.e., *Get_confidential_data* and *Export_confidential_data*, and the final goal, i.e., *Steal_and_export_confidential_data*. We can see that probabilities of the success of subgoals and the final goal increases with the support of incoming evidence corresponding to the attack steps aimed to get data from the database server and export data to the external.

The likelihood of each node at causal network based on on-going evidence can also be used to predict the attacks. For example, after getting the evidence of *DB_FTP_Globbing_Attack*, the probability of the subgoal *Get_data_from_Server_directly* (as shown in Figure 1(b)) is increased and equals 0.67. Therefore, we expect a future attack that enables the attacker to access the file stored in the database server. For another example, when we get the evidence of *DB_NewClient*, the likelihood of *Transfer_data_via_covert_channel* (as shown in Figure 1(b)) is computed as 0.71 that means it is quite likely that we will see another attack with which the attacker can export data via the covert channel in the future. In the GCP Scenario I, the attacker did launch the attack to access the confidential data stored in database server (indicated by the alert *DB_Illegal_File_Access*) after getting the root access to the database server, and also transferred the stolen data to the external (indicated by the alert *Loki*) after setting up the covert channel (indicated by alert *DB_NewClient*).

Applying our algorithms of scenario correlation and attack prediction to the GCP data set, we can correlate some isolated attack scenarios resulted from fundamental alert correlation at low-level, and make correct prediction on the upcoming type of attack.

In our approach, one of the most important components



(a) Two isolated scenarios (b) Correlated scenarios

Figure 3. Correlation of isolated scenarios

Evidence set	$P(subgoal_1 = 1\|evidence)$	$P(subgoal_2 = 1\|evidence)$	$P(goal = 1\|evidence)$
e_1	0.58	0.55	0.56
e_1, e_2	0.58	0.71	0.63
e_1, e_2, e_3	0.78	0.71	0.74
e_1, e_2, e_3, e_4	0.78	0.81	0.77
e_1, e_2, e_3, e_4, e_5	0.78	0.85	0.81

Table 1. Likelihood evaluation of sub-goals and final goal with different evidence. Denote e_1: DB_FTP_Globbing_Attack, e_2: DB_NewClient_Target, e_3: DB_Illegal_File_Access, e_4: DB_NewClient, e_5: Loki, $subgoal_1$: Get_confidential_data, $subgoal_2$: Export_confidential_data, $goal$: Steal_and_export_confidential_data.

is the library of attack plans (defined as attack trees). It is the basis for automatically correlating isolated attack scenarios at a higher level and conducting probabilistic inference for attack prediction. It is true that there exists a limitation in this approach due to the (limited) library of attack plans. If the attack strategies are beyond the definition of attack plans, we cannot automatically correlate isolated scenarios or make an inference on the future attacks based on existing attack plan library. Such a task requires the involvement of security experts. However, we argue that, in practice, the plan library can be defined as comprehensively as possible by security experts with their knowledge of attacks and attack strategies, as well as the understanding of networks and systems under protection and the mission goals. The attack plan library can be expanded or re-defined with the new knowledge of attacks or attack scenarios. Therefore, we believe that our approach is practical and has the potential to provide security operators a way to automatically correlate isolated attack scenarios and predict future attacks based on the observed evidence and networks under protection.

6. Conclusions and Future Work

In this paper, we presented an approach to identify attack plans and predict upcoming attacks. We developed a graph-based technique to correlate isolated attack scenarios derived from low-level alert correlation based on their relationship in attack plans. We conducted probabilistic inference to evaluate the likelihood of attack goal(s) and predict potential upcoming attacks based on causal network converted from attack trees.

There are still some challenges in attack plan recognition. First, the current approach is based on predefined attack plans built on security experts' knowledge and understanding of networks and systems under protection. If attackers' activities are beyond the predefined scope of attack plans, we have to face the challenge of handling and identifying new attack scenarios. Another challenge is how to distinguish the deceptive plan and the real goal of the attackers. That is, we need to develop a mechanism to identify and avoid the misleading of attackers. Finally, we also need to consider how to effectively distinguish the attacks conducted by a single attacker and a group of collaborated attackers. We will study these challenges in our future work.

7. Acknowledgments

This work is supported in part by NSF grants CCR-0133629 and CCR-0208655 and Army Research Office contract DAAD19-01-1-0610. The contents of this work are solely the responsibility of the authors and do not necessar-

ily represent the official views of NSF and the U.S. Army.

References

[1] D. Albrecht and A. Nicholson. Bayesian models for keyhole plan recognition in an adventure game. *User Modeling and User-Adapted Interaction*, pages 5–47, 1998.

[2] E. Bauer, D. Koller, and Y. Singer. Update rules for parameter estimation in Bayesian networks. In *Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 3–13, Providence, RI, August 1997.

[3] S. Carberry. Incorporating default inferences into plan recognition. In *Proceedings of the Eighth National Conference on Artifici al Intelligence*, pages 471–478, Boston, Massachusetts, 1990.

[4] E. Charniak and R. P. Goldman. A probabilistic model of plan recognition. In *Proceedings of the Ninth National Conference on Artifici al Intelligence*, pages 160–165, Anaheim, California, 1991.

[5] E. Charniak and R. P. Goldman. A bayesian model of plan recognition. *Artificial Intelligence*, 64(1):53–79, Novemeber 1993.

[6] E. Charniak and D. McDemott. *Introduction to Artificial Intelligence*. Addison Wesley, 1985.

[7] 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.

[8] I. Cohen, A. Bronstein, and F. G. Cozman. Online learning of bayesian network parameters. *Hewlett Packard Laboratories Technical Report, HPL-2001-55(R.1)*, June 2001.

[9] P. R. Cohen, C. R. Perrault, and J. F. Allen. Beyond question answering. In W. Lehnert and M. Ringle, editors, *Strategies for Natural Language Processing*, pages 245–274. 1981.

[10] G. F. Cooper. Probabilistic inference using belief networks is np-hard. Technical Report KSL-87-27, Stanford University, 1988.

[11] 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.

[12] DAPRA Cyber Panel Program. DARPA cyber panel program grand challenge problem (GCP). http://www.grandchallengeproblem.net/, 2003.

[13] T. Dean and T. Wellman. *Planning and Control*. Morgan Kaufmann, 1991.

[14] H. Debar and A. Wespi. The intrusion-detection console correlation mechanism. In *4th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2001.

[15] C. W. Geib and R. P. Goldman. Plan recognition in intrusion detection system. In *DARPA Information Survivability Conference and Exposition (DISCEX II)*, June 2001.

[16] 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.

[17] C. W. J. Granger. Investigating causal relations by econometric methods and cross-spectral methods. *Econometrica*, 34:424–428, 1969.

[18] J. Haines, D. K. Ryder, L. Tinnel, and S. Taylor. Validation of sensor alert correlators. *IEEE Security & Privacy Magazine*, January/February, 2003.

[19] H. Kautz and J. F. Allen. Generalized plan recognition. In *Proceedings of the Fifth National Conference on Artificia l Intelligence*, pages 32–38, September 1986.

[20] B. Morin and H. Debar. Correlation of intrusion symptoms: an application of chronicles. In *Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003)*, Pittsburgh, PA, September 2003.

[21] 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.

[22] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, Inc, 1988.

[23] 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.

[24] X. Qin and W. Lee. Statistical causality analysis of INFOSEC alert data. In *Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection (RAID 2003)*, Pittsburgh, PA, September 2003.

[25] X. Qin and W. Lee. Discovering novel attack strategies from INFOSEC alerts. In *Proceedings of the 9th European Symposium on Research in Computer Security*, Sophia Antipolis, France, September 2004.

[26] C. Schmidt, N. Sridharan, and J. Goodson. The plan recognition problem: an intersection of psychology and artificial intelligence. *Artificial Intelligence*, 11:45–83, 1978.

[27] B. Schneier. *Secrets and Lies: Digital Security in a Networked World*. John Wiley & Sons, August 2000.

[28] G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

[29] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. M. Wing. Automated generation and analysis of attack graphs. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Oakland, CA, May 2002.

[30] A. Valdes and K. Skinner. Probabilistic alert correlation. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID)*, October 2001.

[31] R. Wilensky. *Planning and Understanding*. Addison Wesley, 1983.