An Alert Fusion Framework for Situation Awareness of Coordinated Multistage Attacks

Sunu Mathew, Chintan Shah, Shambhu Upadhyaya
Department of Computer Science and Engineering,
State University of New York at Buffalo
Buffalo, NY 14260
Email: (smathew2, cmshah, shambhu)@cse.buffalo.edu

Abstract

Recent incidents in the cyber world strongly suggest that coordinated multistage cyber attacks are quite probable and that effective countermeasures need to be developed. Attack detection by correlation and fusion of intrusion alerts has been an active area of current research. However, most of these research efforts focus on ex post facto analysis of alert data to uncover related attacks. In this paper, we present an approach for dynamically calculating 'Scenario Credibilities' based on the state of a live intrusion alert stream. We also develop a framework for Attack Scenario representation that facilitates real-time fusion of intrusion alerts and calculation of the scenario credibility values. Our approach provides a usable mechanism for detecting, predicting and reasoning about multistage goal-oriented attacks in real time. The details of the fusion framework and a description of multistage attack detection using this framework are presented in this paper.

Keywords: Alert correlation, fusion, intrusion detection, multistage attacks, situation awareness

1 Introduction

A great deal of research activity has been witnessed recently dealing with correlation and fusion of intrusion detection system (IDS) alerts. The focus of these efforts is to provide detection capability and a better understanding of multistage attacks that may be taking place on a network. Detection of a complex multistage attack which may involve steps like using hosts as stepping stones, exploiting trust relationships between benign hosts, and data theft, require a rich set of intrusion detection sensors. These sensors have to focus on both network and host log data, and use techniques ranging from signature based techniques to user anomaly detection to detect all attack stages. Thus, the factors that have to be considered in attack scenario recognition include the volume of alerts (typically on the order of thousands a day per sensor), the kinds of sensors, the sources of the alerts (i.e. the host and subnet where the alerts are generated), and the semantics of the alerts and how the exploits they indicate are related.

Most research efforts dealing with scenario detection through alert correlation and fusion are ex post facto (they only enable the analyst to detect an attack scenario in retrospect) and are block based (they have to batch alerts over a period of time before analysis). They do not provide the user with information sufficiently quickly to be of immediate benefit in mitigating an attack. Part of the reason for this is the computation that is required to accurately construct attack scenarios, taking into account the factors mentioned earlier. However, the underlying problem remains the same − providing a quick, computationally efficient approach to fuse alerts and providing the user with a real-time scenario detection capability. In this paper, we present an approach to tackle this problem. We develop a fusion framework to handle a live alert stream, as opposed to working with blocks of alerts. The framework uses a graph based approach to represent attack scenario templates so that scenario construction from an alert stream can be done in real-time. This framework thus enables real-time situation awareness of multistage attack scenarios *while the scenarios are developing*. The contributions of this paper are:

- A graph based fusion framework for Attack Scenario construction from a live alert stream
- Description of a stream based approach to alert fusion to generate real-time situation awareness
- Preliminary results that indicate the utility of our approach

The rest of this paper is organized as follows: Section 2 gives an overview of our approach, Section 3 details the fusion framework, Section 4 presents an evaluation of the framework and Section 5 reviews some related work. Section 6 presents a discussion of the approach, and Section 7 concludes the paper and presents some directions for future work.

2 Overview of the Approach

2.1 An illustration

Figure 1 depicts a network with a demilitarized zone and a protected subnet. This kind of network setup with secure components is typical of many military installations.

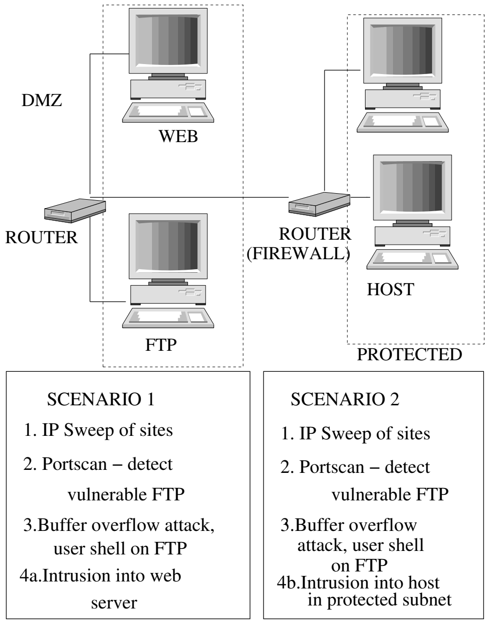

Figure 1: Event (Alert) streams corresponding to two attack scenarios

Consider the two attack scenarios that are depicted in Figure 1. The first three stages of these scenarios are identical. An intrusion alert stream at this stage could indicate either Scenario 1 or Scenario 2. The alert stream indicates the more serious Scenario 2 with the arrival of alert(s) corresponding to Stage 4b as opposed to the arrival of alerts corresponding to Stage 4a. Clearly, there is a need for a fusion framework that can detect both scenarios and differentiate the more serious one from the other from indications in the alert stream. Such a capability is useful for the overall awareness of distributed system security and to direct and guide response systems so that damage is contained and minimized. In networks with multiple sensors, the volume of alerts received is very high. In addition, the stream has alerts that are false positives and negatives, together with sensor anomalies that add 'noise'. We require to be able to detect attack scenarios kind of sevents described above while minimizing the effects of noise. The fusion framework we present in this paper allows just such a capability to be realized.

2.2 The Approach

We call our fusion framework for Scenario Recognition the **S**cenario **G**raph for **I**DS Alert **F**usion (SGIF). The traditional approaches to attack graph generation using model checking [4] use low-level system specifics like host connectivity and privileges as the basis for modeling. This suffers from the 'state explosion' problem and the graphs become unmanageable even for small networks. Our treatment of the problem is different. We use the IDS alerts as the atomic elements of our fusion framework and build a hierarchical structure that relates alerts to exploits. In our experience, the higher level of abstraction that we use results in a reduction in the size of the state space at least empirically. Figure 2 depicts the hierarchical structure of SGIF. We also give a brief description of the fusion process in this section.

2.2.1 Framework Elements

Attribute Node: An Attribute Node (also called Alert Node) represents an event in our framework. An event is usually an intrusion alert as reported by an intrusion detection sensor. This event is represented as a collection of Attribute Fields, where the Attribute Fields are elements (tags) in the newly emerging Intrusion Detection Message Exchange Format (IDMEF [17]) XML DTD (Document Type Definition). Some of the Attribute Fields are designated as Critical Fields. The values of the critical fields serve to distinguish one Attribute Node from another. An Attribute Node has a binary credibility value (1 or 'ON'and 0 or 'OFF'). Each Attribute Node with credibility 1 contributes a certain Attribute Weight to its parent Attack Node(s).

Attack Node: An Attack Node is an exploit that is represented in terms of its components (child nodes), which may be Attribute Nodes or other Attack Nodes. An Attack Node has a correlation function that defines how its child nodes are correlated. Some Attack Nodes are nodes of a higher level Scenario Graph (explained below) and are called Scenario Nodes. An Attack Node also has a dynamic Attack Node Credibility value that determines how much it contributes to its parent Attack Node(s) or Scenario Graph(s) at a particular instant of time.

Scenario Graph: We view a multistage attack scenario as a sequence of several attack stages. These stages may be exploits (e.g. Buffer overflow attack on Sendmail 8.12.1 [19]) or attack stages that may be specifically defined for the deployment environment (e.g. Access to host X in subnet $SECRET\_NET$). We view each such stage as a node (Scenario Node) of a Scenario Graph. The Scenario Graphs are goal oriented, with at least one node of a Scenario Graph being identified as a goal node. A Scenario Graph also has a credibility value (zero initially), that varies dynamically as fusion of the live intrusion alert stream proceeds.

A Scenario Graph developed based on this framework is thus a complete representation of an Attack Scenario with its constituent exploits and intrusion alerts that indicate these exploits and incorporates the various relationships that exist between these elements. Framework specifics and inter-relationships will be presented in Section 3.

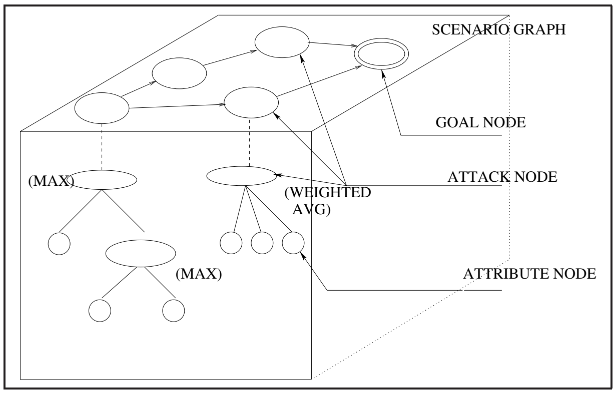

Figure 2: Hierarchical model of SGIF showing a Scenario Graph and several levels of Attack Nodes

2.2.2 Alert Fusion and Calculation of Scenario Credibility

Figure 3 depicts how alerts are processed before they are passed on to the alert fusion engine. The alerts undergo filtering, cleansing and semantic homogenization before they are converted into the IDMEF format. The fusion engine relies on a database of Scenario Graph templates developed using the framework presented in the preceding section. A graph matching approach is used to match intrusion alerts with Scenario Graph templates for the purpose of alert fusion and calculation of Scenario Graph Credibility.

Critical Field and Attribute Node Matching: Intrusion alerts arriving to the fusion engine are compared for matches of critical fields with the Attribute Nodes in the database. All Attribute Nodes that exactly match an incoming alert for all critical field values, receive a credibility value of 1 and turn $'ON'$. Each Attribute Node which is $'ON'$ contributes its predefined Attribute Node weight towards its parent Attack Node(s).

Calculation of Attack Node Credibility: Each Attack Node has one or more Attribute Nodes or other Attack Nodes as its children. The Attribute Nodes that are 'ON' at any instant of time (or Attack Nodes with non-zero credibility values) contribute weights to the parent Attack Node(s). The correlation function that is defined for the Attack Node is taken into account to determine how child node weights are used to calculate the dynamic Attack Node credibility value. Examples of correlation functions are $WEIGHTED\_AVG$ and MAX functions (explained later) and may involve factors like timing constraints (for example the maximum time interval possible between alerts).

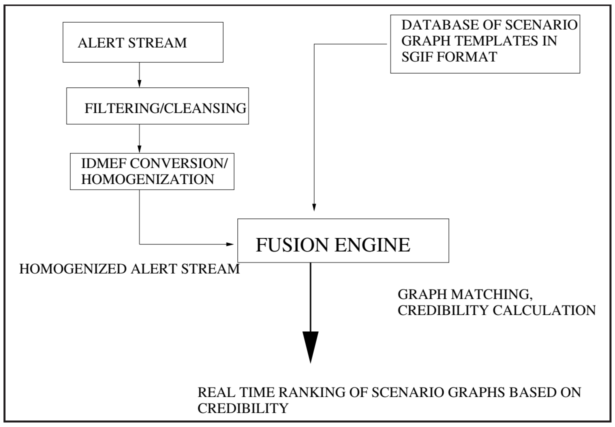

Figure 3: Overall view of real time alert fusion process

Attack Node Fusion and Scenario Graph Credibility: Scenario Graphs may have one or more Scenario Nodes with non-zero credibility values at a particular instant of time. A Scenario Graph is given a dynamic Scenario Credibility value by a fusion algorithm that takes into account the credibility values and semantics of its constituent Scenario Nodes. Various approaches and techniques from graph theory, data fusion [1] and intrusion detection can be applied to arrive at a suitable algorithm for calculating Scenario Credibility. Figure 4 depicts how our approach functions with respect to the two scenarios shown in the earlier illustration.

3 Framework Details

In this section, we give a detailed description of the SGIF framework presented earlier. We give formal definitions for the framework and its elements in order to describe their scope with respect to cyber attack representation and alert fusion. We also give some examples of exploits represented using the framework elements.

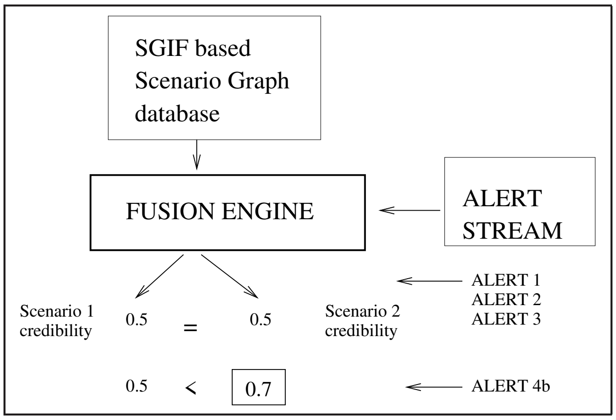

Figure 4: Scenario 2 receives a higher dynamic credibility upon arrival of alert 4b in the alert stream

```
<DetectTime ntpstamp="0xabcd">
     2004−03−31T08:12:32−01:00
</DetectTime>
<Source>
     <Node>
       <Address category="ipv4−addr">
          <address>192.0.2.200</address
       </Address>
     </Node>
</Source>
```

Figure 5: A portion of an IDMEF Alert

IDS_ID	WEB−SERVER−IDS
CREATE_TIME	2004−03−31T08:12:32−01:00
SOURCE	192.0.2.200
TARGET	128.201.35.20
TARGET.FILE	/etc/passwd
CLASSIFICATION.NAME	HOST_SYSTEM_FILE_ACCESS_REMOTE
ADDITIONALDATA.FILEOP	read

Figure 6: Attribute Node.

3.1 IDMEF XML DTD

The Intrusion Detection Message Exchange Format (IDMEF [17]), is an XML based format for the exchange of Intrusion Detection Alerts among various IDS sensors. This is still at the Internet Draft stage, but it appears that this is being increasingly accepted by IDS vendors, as plugins for well known sensors such as SNORT are starting to become available [16]. The IDMEF format represents an Alert in the form of a collection of XML fields, and this format forms an important part of the SGIF framework. Any other format representing Alerts as a group of fields could be used, but for the sake of interoperability and to be able to handle multiple sensor types from different vendors, it was felt that adopting IDMEF would be appropriate for our purpose.

3.2 Attribute Field

An *Attribute Field* is the value of an element(tag) of the IDMEF XML Document Type Definition (DTD), that is used to represent IDS alerts. An attribute field as part of an IDMEF Alert can indicate several things such as Time, Source IP address, Destination IP address, Type of attack or exploit and so on, as Figure 5 indicates. The value of an Attribute Field for a particular alert indicates part of the semantics of that alert.

Definition 3.1 An Attribute Field $A_f = (a_f, c)$ in SGIF is a tuple where A_f is an element (tag) of the IDMEF XML DTD, $a_f \in$ Domain(A_f) and c $\in$ (true, false) indicates whether the Attibute Field is *critical* or not. An IDMEF XML alert is parsed to extract the relevant values of the Attribute Fields for that alert.

Example 3.1 Consider a portion of an IDMEF alert as shown in Figure 5. This is parsed to extract an Attribute Field ‘Source.Node.Address.address’ with the value 192.0.2.200. The domain of this Attribute Field is the set of all possible Source IP addresses. This could be all IP addresses on the Internet or in the environment under consideration.

3.3 Critical Field

A *critical field* in SGIF is a an Attribute Field (above) that serves to distinguish between alerts during the graph matching phase of the fusion process. When two alerts arrive in an input stream, we attempt to match them with Attribute Node (below) templates in the SGIF database. Only those Attribute Node templates are considered for matching, whose critical fields and critical field values match those of the incoming alert.

Definition 3.2 A Critical Field A_{cf} in SGIF is an Attribute Field, with c = ’true’.

Example 3.2 Consider an alert that arrives indicating that an attempt has been made to start a user shell on a host. Suppose that in our fusion process, we are only concerned with alerts that indicate attempts to start a user shell on a certain host, with a certain IP address. Then the destination IP address can be a critical field, and only alerts that match this IP address will be considered during the fusion process.

3.4 Attribute Node

An *Attribute Node* represents a single event in the SGIF framework. This may be an alert from an Intrusion Detection sensor, Network Management System or other data source which provides input to the fusion process. Attribute Nodes in SGIF can be at varying levels of abstraction, for example, indicating the presence of a certain pattern in a TCP packet or indicating the presence of, say, a rootkit. The Attribute Node forms a template that gets matched during the fusion process and receives a dynamic credibility value. The credibility of an Attribute Node is either 1.0 ('ON' or matched) or 0.0 ('OFF' or no match). Attribute Nodes can have several Attack Nodes as its parents (e.g. some events like certain system file accesses, can be parts of the signature of several attacks or exploits). Each (*Attribute Node, Attack Node*) pair (or any (*Child Node, Parent Node*) pair with an Attack Node as the parent) has a certain *lifetime* value which indicates how long an Attribute Node will continue to be considered in the correlation of its parent Attack Node without being refreshed (i.e., without a fresh alert arriving).

Definition 3.3 An Attribute Node in SGIF is a tuple A_n = *(A,S,L)* where *A* is a set of Attribute Fields that is part of an IDMEF alert, one of which is a critical field. S $\in [0, 1]$ indicates the dynamic credibility value of an Attribute Node. *L* is a set of *(Attack Node,lifetime)* values for each of the Attack Nodes that is a parent of this Attribute Node.

Example 3.3 An Attribute Node is shown in Figure 6 to indicate that an attempt to access '/etc/passwd' has been made. Critical fields are shown in the dashed box.

3.5 Attack Node

An *Attack Node* in SGIF is a stage of a multistage attack. This may be a well known exploit or may be a user defined stage of a multistage attack (e.g., 'Possible data theft', 'Possible Phishing Attack', 'Possible Password Guess Attack'). Attack Nodes have Attribute Nodes (described above) or other Attack Nodes as their children. Other features of Attack Nodes are:

- **Child Node Weight:** Every Attribute Node or Attack Node that is a child of an Attack Node contributes a certain weight to its parent. This reflects an 'information value' approach in which is a decision is made about the relative importance of different events in givng indications about an attack stage or exploit.

- **Attack Node Type:** An Attack Node has a Type based on the kinds of child nodes it has. An *Extrinsic* Attack Node is one whose child nodes are all Attribute Nodes and thus are external events (usually intrusion detection sensor alerts). An *Intrinsic* Attack Node is one whose child nodes are all internal to the fusion process (they are all other Attack Nodes). This is usually used to represent an attack at a higher level of abstraction, in terms of component attacks which are themselves outputs of the fusion process. For example, various kinds of buffer overflow attacks possible on a host could be abstracted as a higher level Intrinsic Attack Node called 'Buffer overflow attack on Host X'. This is shown in figure 7. A *Bitrinsic* Attack Node is one whose child nodes are both Attribute Nodes and other Attack Nodes (both internal and external).

- **Attack Stage Level:** Some Attack Nodes are Scenario Nodes. Attack Nodes that are Scenario Nodes have a certain Attack Stage Level label that indicates which phase of a multistage attack they belong to. The Attack Stage Level is one of (*Reconnaisance, Intrusion, Privilege Escalation, Goal*). Here we adopt the classification in [6] for the different stages of a multistage attack.

- **Credibility Value:** Attack Nodes in SGIF are templates that receive a dynamic credibility value during the fusion process as various Attribute Nodes are matched. The credibilities are real values between 0.0 and 1.0 and the calculation of the Attack Node credibiity depends on a Correlation Function that is defined for each Attack Node. The correlation functions determine how the Attack Node credibilities are dependent on their child node weights or credibilities.

- **Correlation Function:** Several correlation functions of various complexities can be defined for Attack Nodes. We have defined two functions which we believe capture the Attack Node-Child Node relationship for a wide variety of exploits. These are described below.

  1. WEIGHTED-AVG or AND: This is basically used to represent attacks (Attack Nodes) that are detected based on signatures. The alerts or Attribute Nodes fully define the parent Attack Node. These alerts together give evidence of the Attack Node and are thus not independent of each other in this sense. For AND nodes, all the alerts have to arrive (i.e., to be matched), to reach a definite conclusion of the attack (i.e., to conclude with absolute certainty that this attack is, in fact, happening). The correlation criterion for AND Nodes of the Scenario Graph, is that all the relevant alerts arrive for maximum credibility. Absence of a few alerts puts the relevant parent node at an intermediate credibility value (the weighted average of the Attribute Nodes that receive matches).

2. MAX or OR: These represent Attack Nodes that have attributes or child nodes that give independent confirmation of an attack. Here, the weights of the child nodes independently add credibility to the parent Attack Node. The credibility value of the Attack Node here is the maximum weight or credibility value of the child nodes. The individual child nodes are correlated only in the sense that they give indications of the same attack, their contribution to the Attack Node is independent and the credibility of the Attack Node is the maximum weight among the weights of all arriving alerts (Attribute Nodes) or child Attack Nodes.

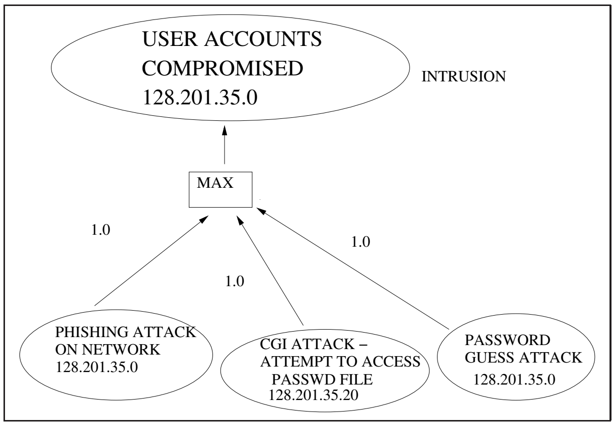

Figure 7: Attack Node

Definition 3.4 An Attack Node A_N = *(A, W,T,A_l,F,S)* is a tuple where A is the set of child nodes (Attribute Nodes or other Attack Nodes), W = (A,w) is a tuple where A is a child node and w is the contributing weight of the child node, T $\in \{Extrinsic, Intrinsic, Bitrinsic\}$ is the Attack Node Type, $A_l \in \{Reconnaisance, Intrusion, PrivilegeEscalation, Goal\}$ is the Attack Stage Level, F is the Attack Node correlation function and S $\in$ [0,1] is the Attack Node Credibility Value.

Example 3.4 An Attack Node with three other Attack Nodes as children is shown in Figure 7. The Attack Node at the top level is a Scenario Node with Attack Stage Level 'Intrusion'. The correlation function of this Scenario Node id 'MAX' and it has three other Attack Nodes at its children. thus it is an 'Intrinsic' Attack Node.

3.6 Scenario Graph

A *Scenario Node* in SGIF is a directed acyclic graph in which the nodes are Attack Nodes (Attacks Nodes which are nodes of the Scenario Graph are called Scenario Nodes). There is a link from one Scenario Node to another if the latter can follow the former in a multistage attack scenario. The links therefore represent the fact that the exploits are temporally related in the context of a multistage attack scenario. Any connected sequence of nodes is thus an attack scenario. A goal-oriented Scenario Graph is a Scenario Graph in which there an Attack Node known as the Goal Node. All connected paths lead finally to the Goal Node in such a Scenario Graph. A Scenario Graph receives a real-time Credibility Value at any instant in time depending on the state of the incoming alert stream at that instant.

Definition 3.5

A goal-oriented Scenario Graph S_G = *(N,E,S)* is a tuple where *N* is the set of Scenario Nodes, *E* is a set of directed edges (N_1,N_2) such that there is a directed edge from N_1 to N_2 if and only if N_2 can follow N_1 in a multistage attack scenario, *S* $\in$ [0,1] is a real valued Scenario Graph Credibility value.

Example 3.5

Figure 8 shows a Scenario Graph. Any path through the graph is a valid sequence of attacks or exploits. One such path is shown in bold. This Scenario Graph has the goal of access to a protected internal network.

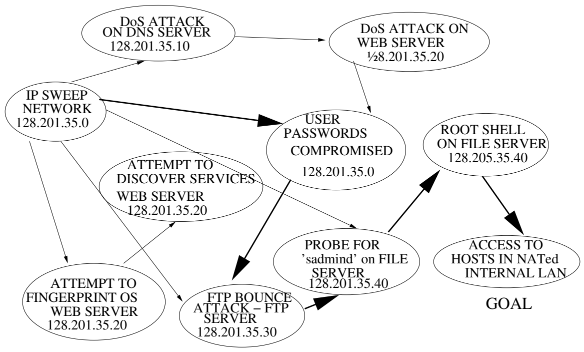

Figure 8: Scenario Graph

4 Framework Evaluation

The graph matching phase of fusion results in various Scenario Nodes of a Scenario Graph having different credibility values. These credibility values are thresholded and a visual component displays these dynamically changing credibility values as temperature changes (for example, cold Scenario Nodes and warm Scenario Nodes). The Scenario Graph itself receives a Scenario Graph Credibility value depending on the credibility values of its Scenario Nodes, and the output of the fusion process is a dynamic ranking of Scenario Graphs in the order of decreasing credibility values. The ultimate purpose of this fusion process is to provide the analyst with real-time situation awareness regarding attack scenario development. Complete details of how the fusion process works are outside the scope of this paper.

We have carried out some experiments with a prototype that uses the fusion framework presented in this paper to perform alert fusion using a stream-based approach as opposed to a block approach. We present here some preliminary results. Some of these experiments were carried out in a live environment to test the real-time nature of the system.

4.1 Experimental Setup

We developed a few Scenario Graphs manually to populate our SGIF database. These were done manually by studying multistage attacks and exploits from the open literature. Scenario Graphs were generated based on a small fragment of a test network at a federal research laboratory. These graphs were represented using elements of the SGIF framework. The test prototype prototype was designed to convert an input alert stream into the IDMEF format for homogenization and to use the predefined Scenario Graphs in its database during the fusion process.

4.2 Fusion of Input Alert Stream

We used a stream of alerts (mostly Snort based) randomly captured from the network at the research lab for which the Scenario Graphs were developed. The prototype used a stream based approach to alert fusion with a graph matching phase as described in this paper. The final Scenario Graph credibility values were calculated by a fusion algorithm that considered the credibility values of all Scenario Nodes in the relevant Scenario Graphs. Figure 9 gives a screenshot of a Scenario Graph at a point in time with an intermediate credibility value of 0.65. Scenario Nodes depicted in the figure have various credibility values. The Scenario Nodes (attack stages) that had credibility values above the threshold depicted four steps of two alternate paths that the attacker could take to reach the end goal (which was a user shell on a protected File Server). The paths are:

Path One

- Probe on DNS Server
- Fingerprint attack on DNS

Path Two

- Probe on IIS Web Server
- Attempt to gain user shell on Web Server

None of these attack stages went further along the path to the target, but these connected attack stages of alternate paths to the end goal are correlated as part of the same goal-oriented Scenario Graph.

4.3 Analysis

We can see from the figure that individual alerts in the input alert stream are seen to be correlated as part of a Scenario Graph as expected. Clearly, a stream based approach to alert correlation is useful in relating alerts as part of a goal oriented attack scenario. The fact that these seemingly unrelated alerts could be part of a coordinated attack plan could be missed by a human analyst looking through the alerts, especially when the alerts in question are widely distributed in the alert stream. Another important fact is that the output gives a *dynamic* credibility value for the Scenario Graph, that is, the credibility value at *a point in time*. The credibility of the Scenario Graph in question gets closer to its maximum value of 1.0 if all valid attack alerts happen in a short span of time. The system also gives a dynamically ranked list of candidate scenarios based on decreasing credibility values (this is not shown in the figure). This approach to alert fusion and scenario recognition is also able to deal with attack stages that are missed. The problems of false positives and false negatives are well known in the field of intrusion detection. Since the fusion algorithm takes into consideration an entire Scenario Graph, serious attack scenarios are not missed because of one or two disconnected attack stages. Thus we can see that this approach provides a usable real-time situation awareness system to a security analyst.

5 Related Work

Several papers have used the term 'Scenario Graphs' and 'Attack Graphs' ([2],[3],[5]). These are mainly defined and used in the context of network vulnerability analysis and survivability. [5] uses the term 'Scenario' to mean a sequence of states for the system being modeled and not necessarily scenarios that violate a correctness condition or have an attack oriented goal. The scenarios that violate a correctness or safety property (i.e. have nodes that do so), are called 'Safety Scenario Graphs'. The term 'Attack Graphs' is used to specify graphs that consist exclusively of attacks or exploits in the above work. 'Scenario Graph' as used in SGIF and as presented in this work is merely the top layer of an alert fusion framework with levels of 'WEIGHTED-AVG' or 'MAX' trees and Attribute Nodes. Our Scenario Graph relates exploits and is thus similar to the attack paths constructed in [18].

Several research efforts have dealt with the issue of intrusion alert correlation and fusion ([8], [9], [7], [10],

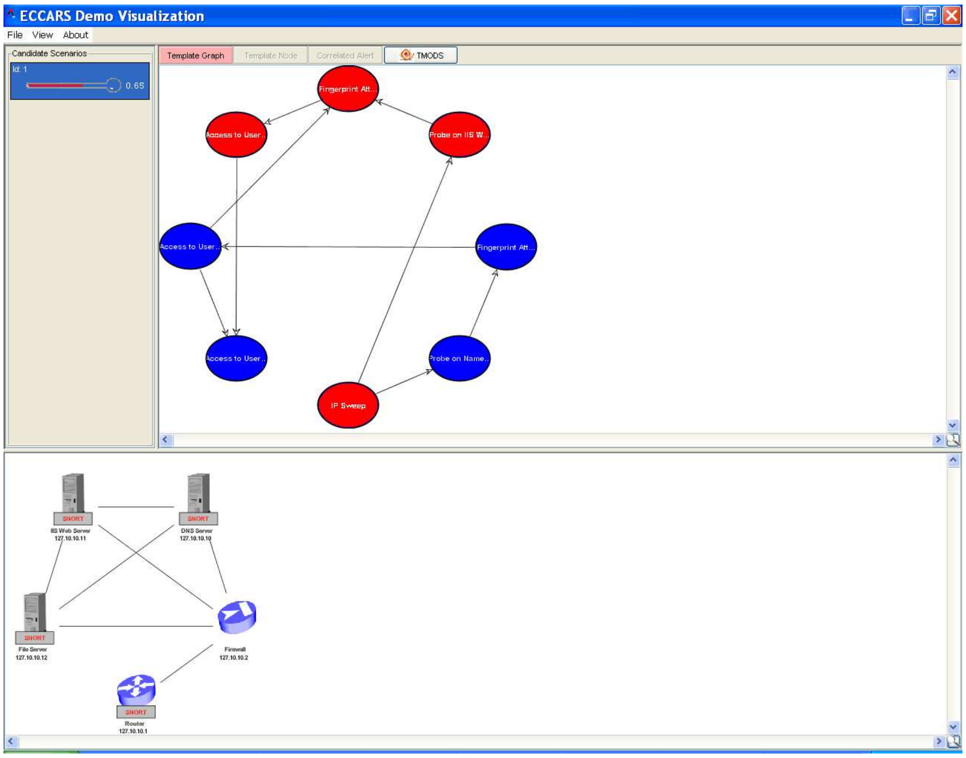

Figure 9: Fusion of input alert stream to generate real time Situation Awareness. Credibility of Scenario Graph in figure is 0.65.

[11], [14]). These follow various approaches like rule-based and probability based approaches, attempt to cluster related alerts into scenarios, and later analyze the scenarios to understand them. The difference in our approach is that we seek to calculate credibilities or probabilities of *predefined scenarios* in real-time, offering the scope for a real-time response. Our alert fusion framework uses alert fusion templates for several predefined goal-oriented scenarios and attempts to rank these scenarios in real time based on the evidence provided by the alert stream.

In [9], a real time algorithm is specified for clustering alerts into groups if they are part of the same scenario; however, complete comprehension of the scenario requires further analysis of grouped alerts. In [14], alerts are grouped together based on similarities of features, this is different from our approach and again, further analysis is required to comprehend the nature of multistage attacks, if any. The JIGSAW project [11], MIRADOR [10] and TIAA [7] all target detection of multistage attacks. However, all of these require further analysis of connected or related attack stages (e.g. the hyper-alert graphs in [7]) for comprehension of scenario specifics and prediction of attack goals. Our approach provides dynamic ranking of entire goal oriented scenarios based on their likelihood, and therefore provides a better real-time situation awareness system. Our method suffers from the drawback of not being able to detect new scenarios, but our focus is on real-time detection and we feel that offloading the task of Scenario Graph generation to offline methods is an acceptable tradeoff. Moreover, it is seen that very few multistage attack scenarios consist entirely of novel attack steps.

6 Discussion

In this section we proceed to discuss some aspects of the work we have presented in this paper.

6.1 Generation of Scenario Graphs and Framework Elements

The Scenario Graphs mentioned as part of the SGIF framework and as used by us in our experiments are connected graphs of attack stages or exploits with one final stage or exploit being the final goal. Scenario Graphs are thus goal dependent, with the graphs encapsulating all possible combinations of exploits and other attack stages that can be employed to achieve their goal. The nature of computer attacks is such that they are very specific, usually targeting certain versions of a particular operating system or software implementation, and sometimes designed to succeed only on specific hardware (for example, buffer overflow attacks). As such, generation of these Scenario Graphs is dependent on network topology, host configuration, services running (e.g. FTP) and hardware platform. Some network vulnerability and hardening research have resulted in ideas that could be of benefit in developing Scenario Graphs.[18] uses 'exploit dependencies' to construct attack paths. It also uses a 'goal-condition exploit' which is somewhat similar to our 'goal node'. A lookup table with all possible exploits and their preconditions and postconditions can be used along with network topology and host configuration information to generate the Scenario Nodes and edges of Scenario Graphs.

Hierarchical levels of Attack Nodes depend on the level of abstraction that is required at the top level of a Scenario Graph, and in the absence of standards for these levels of abstraction, this is implementation dependent. Attribute Nodes of Attack Nodes can be generated in the following way:

- Attack Nodes with the $WEIGHTED-AVG$ function are defined by their signatures (for example, a rootkit signature indicates what standard files are modified by it) and these can be used to construct Attribute Nodes.

- Attack Nodes with the *MAX* function have Attribute Nodes that independently give evidence of the parent node (for example, either access to '/etc/passwd' or '10 failed login attempts' could indicate an 'Attempt to steal or guess passwords'). These Attribute Nodes can be specified by having a Subject Matter Expert identify the events of interest.

6.2 Benefits of the SGIF Framework

We present some of the benefits of our approach:

Real time indications/warnings:
Since we use a stream based approach towards alert fusion and scenario recognition, we are able to provide the user with real time indications and warnings. Since these scenarios are constructed *apriori* and have well known goals, the user can use the situation awareness that our method provides to preempt dangerous scenarios from going to completion.

Plan Recognition/Goal Prediction:
Our method provides a means to detect attack paths with stealthy stages (or hard-to-detect stages) and can thus aid in plan recognition and goal prediction. The output of the system shows attack paths with a certain credibility. If a certain Scenario Node with an intermediate credibility value is a part of an attack path in which the other nodes have high credibility values, chances are high that this stealthy attack may, in fact, be happening. A Scenario Node with four Attribute Nodes for which only two turn 'ON' will still get an intermediate credibility value. This node will get 'noticed' when it becomes part of a high credibility attack path with other Scenario Nodes. Many cyber-attacks are intrinsically hard to detect by themselves (for example, buffer overflow attacks), but their effects or goals become more easily detected when they are defined as part of an overall attack path.

The occurrence of an attack path (subgraph), adds to the credibility values of all the Scenario Graphs that it may be a part of. These Scenario Graphs may have different goals. This can thus give a dynamically changing, real-time indication of various attack goals and paths that the attacker may have in mind.

Reasoning about multistage attacks:
Since we use a graph based approach to represent Attack Scenarios, we can take advantage of several graph based techniques that lend themselves very well to the problem domain. Various credibility metrics can be applied to calculate Scenario Graph credibilities to reflect user priorities.

- *Goal based credibility:* If the desire is to be able to detect the closeness of an adversary to achieving a certain goal, then the Scenario Graphs can be dynamically ranked on the basis of, say, the nearest Attack Node to the Goal Node that has a credibility above some threshold.

- *Credibility based on number of matches:* The number of Attack Nodes in a Scenario Graph above a certain threshold that are connected to other such Attack Nodes can be another credibility measure. This can give an indication of the coordinated nature of various attacks represented by Attack Nodes. A Scenario Graph with more connected Attack Nodes will have a greater credibility than a graph with a lesser degree of connectivity.

- *Credibility based on Attack Stage Levels:* The Attack Stage Levels mentioned earlier can be used to decide which Scenario Graph gets higher credibility. A Scenario Graph with four Attack Nodes above a certain threshold with one from each of (*Reconnaisance, Intrusion, Privilege Escalation and Goal*) can get a higher credibility than another Scenario Graph where all four nodes belong to one Attack Stage Level. This credibility metric reflects the multistage and coordinated nature of attacks.

The Security analyst would be able to use any or all of the above metrics, including combinations of metrics, as the basis to decide the credibility that Scenario Graphs would have. Efficient algorithms that exist for graphs, like Dijkstra shortest path algorithm, can be exploited to figure out the sequence of attacks that an attacker might take. It is seen that a graph based approach is inherently suited to reasoning about coordinated and multi-stage attacks as mentioned above.

6.3 Limitations

One important aspect of the fusion framework presented in this paper is that it relies on the 'weights' that each Attribute Node and Attack Node contributes to its parent Attack Nodes(s). There is no reliable way to assign these values. The same is the case with setting the lifetime values for the (Parent,Child) pairs of Attack Nodes and their children, and also the credibility thresholds for the Attack Nodes. One way to do this would be to assign initial values for these parameters with the aid of a Subject Matter Expert and then tune these values by a testing and learning process. The optimal values for these parameters would be that at which the overall Scenario Graph credibility most accurately reflects the ground truth as far as possible.

7 Conclusions and Future Work

In this paper we have presented a fusion framework that generates real time situation awareness of multistage attacks from a live alert stream. We have presented the framework in detail and have given an overview of how fusion proceeds. We have also shown some preliminary results that indicate the utility of our approach. Our future work involves working with larger datasets to tune the parameters like the weight, threshold and lifetime values. We would also like to investigate how our approach behaves when a massive alert stream involves sparse attack traffic, as also the case when attack concentration is high. We also plan to investigate how our correlation functions can be changed to improve accuracy without reducing computational efficiency and the real-time nature of the system.

References

[1] Hall, David L., Llinas, James. Handbook of Multisensor Data Fusion CRC Press. 2001

[2] Jha, Somesh, Wing, Jeannette. Survivability analysis of networked systems. In *Proceedings of the International Conference on Software Engineering, Toronto, Canada*, May 2001

[3] Jha, S, Sheyner, O., Wing, J. Two formal analyses of attack graphs. In *Proceedings of the 15th IEEE Computer Security Foundations Workshop, pages 49-64*, Canada, June 2002

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

[5] Sheyner, Oleg. Scenario Graphs and Attack Graphs:a Summary.Ph.D Thesis Summary. Carnegie Mellon University, 2004

[6] Haines, J., Ryder, D. K., Tinnel, L., Taylor, S. Validation of Sensor Alert Correlators In *IEEE Security and Privacy*, May 2001.

[7] Ning, P.,Cui, Y., Reeves, D.S., Xu, D. Techniques and Tools for Analyzing Intrusion Alerts. In *ACM Transactions on Information and System Security (TISSEC)*, Pages: 274 - 318, May 2004.

[8] Dain, O. M., Cunningham, R. K. Fusing a Heterogeneous Alert Stream into Scenarios. In *ACM Computer and Communications Security*, 2001.

[9] Dain, O. M., Cunningham, R. K. Building Scenarios from a Heterogeneous Alert Stream. In *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security, West Point, NY*, 2001.

[10] Cuppens, F. and Miege, A. Alert correlation in a cooperative Intrusion Detection Framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, Pages 187-200.

[11] Templeton, S.J.and Levitt, K. A Requires/Provides Model for Computer Attacks. In *Proceedings of the 2000 Workshop on New Security Paradigms.* ACM Press, Pages 31 - 38.

[12] Ning, P. and Xu, D. Learning Attack Strategies from Intrusion Alerts. In *Proceedings of the 10th ACM Conference on Computer and Communications Security*, Pages 200 - 209.

[13] Ning, P., Xu, D., Healey, C. G., St. Amant, R. Building Attack Scenarios through Integration of Complementary Alert Correlation Methods. In *Proceedings of the 11th Annual Network and Distributed System Security Symposium*, Pages 97 - 111.

[14] Valdes, A. and Skinner, K. Probabilistic Alert Correlation. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection*, Pages 54-68.

[15] Staniford-Chen, S., Cheung, S., Crawford, R., Dilger, M., Frank J., Hoagland J., Levitt, K., Wee, C., Yip, R., Zerkle, D. GrIDS : A Graph Based Intrusion Detection System for Large Networks. In *Proceedings of the 19th National Information Systems Security Conference*, Baltimore. Pages 54-68.

[16] SNORT Intrusion Detection Sensor. http://sourceforge.net/projects/snort-idmef

[17] The Intrusion Detection Message Exchange Format. http://www.ietf.org/internet-drafts/ draft-ietf-idwg-idmef-xml-11.txt

[18] Jajodia, S., Noel, S.,O'Berry, B. Topological Analysis of Network Attack Vulnerability. In *Managing Cyber Threats: Issues, Approaches and Challenges, Kluwer Academic Publishers, 2003.*.

[19] Sendmail buffer overflow vulnerability. http://www.cert.org/advisories/CA-2003-12.html