Network Forensics Analysis with Evidence Graphs (Demo Proposal)

Wei Wang and Thomas E. Daniels
Department of Electrical and Computer Engineering
Iowa State University
Ames, Iowa 50010
Email: {weiwang,daniels}@iastate.edu

***Abstract*— We develop a prototype network forensics analysis tool that integrates presentation, manipulation and automated reasoning of intrusion evidence. We propose the evidence graph as a novel graph model to facilitate the presentation and manipulation of intrusion evidence. For automated evidence analysis, we develop a hierarchical reasoning framework that includes local reasoning and global reasoning. In local reasoning, we apply Rule-based Fuzzy Cognitive Maps (RBFCM) to model the state evolution of suspicious hosts. In global reasoning, we aim to identify group of strongly correlated hosts in the attack and derive their relationships in the attack scenario. Our analysis mechanism effectively integrates analyst feedbacks into the automated reasoning process. Experimental results demonstrate the potential of our proposed techniques.**

I. Introduction

Our work is motivated by the requirements of network forensics analysis. Current practices in network forensics analysis are to manually examine logs, a time-consuming and error prone process [12]. To effectively assist the forensics analyst, we argue that network forensics analysis mechanisms should meet the following essential needs:

- Short response times: Large volume of irrelevant information and increasingly complex attack strategies make manual analysis impossible in a timely manner. Automated evidence analysis would produce an immediate impact on law enforcement's ability to reduce response times.
- Friendly interface: Intrusion evidence and analysis results should be presented in an intuitive approach. The ad-hoc nature of cyber attacks indicates that expert opinion and out-of-band information must be efficiently integrated into the automated reasoning process.

In view of these requirements, we develop a prototype network forensics analysis mechanism that integrates novel techniques for evidence presentation, interaction and automated reasoning. We propose the evidence graph model as an intuitive approach to present and manipulate intrusion evidence. Based on the evidence graph, we develop a hierarchical reasoning framework for automated evidence analysis.

Our evidence analysis mechanism focus on identifying the group of hosts involved in the attack and determining the roles of each host in the group, which would help to answer questions like:

- How likely is a specific host relevant to the attack?
- What is the role the host played in the attack?
- How strongly are two hosts M and N connected in the attack ?

In summary, our prototype tool has the following features:

1) A flexible pre-processing mechanism to reduce redundancy in intrusion alerts;
2) A novel graph model that facilitates effective presentation and interaction with intrusion evidence;
3) A hierarchical reasoning framework for automated inference of attack group identification and scenario reconstruction.

We have submitted a full paper containing this information to ACSAC'05 [21].

II. Network Forensics Analysis Mechanism

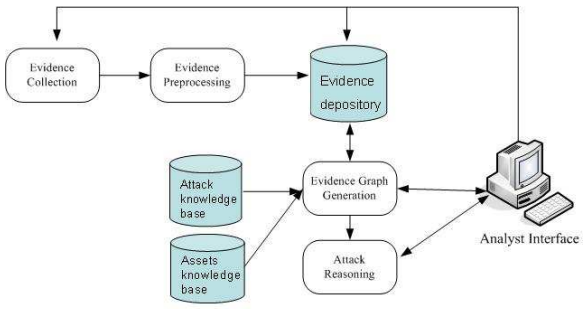

Fig. 1. Network Forensics Analysis Mechanism

Figure 1 shows the architecture of our network forensics analysis tool. Functionalities of each component is briefly described as follows:

1) *Evidence collection module* collects intrusion evidence from networks and hosts under investigation.
2) *Evidence preprocessing module* parses certain types of evidence like intrusion alerts into desired format and aggregates low level evidence into high level ones to reduce redundancy.
3) *Attack knowledge base* provides prior knowledge of known exploits.
4) *Assets knowledge base* provides prior knowledge of the networks and hosts under investigation.

5) *Evidence graph generation module* generates a graph representation of evidence in the depository based on our evidence graph model.
6) *Attack reasoning module* performs automated reasoning based on the evidence graph.
7) *Analyst interface module* provides visualization of evidence graph and reasoning results to the analyst and passes analyst feedbacks to the graph generation and reasoning module.

In the initial phase, the intrusion evidence collected are preprocessed and stored into the evidence depository. Next, the graph generation module constructs the evidence graph with evidences retrieved from the depository. Following that, the reasoning module performs automated inference based on the evidence graph and present results to the analyst. Through the interface module, the analyst could provide expert opinions and out-of-band information in two approaches: (1) directly edit the evidence graph (2) send queries to retrieve specific evidence. The reasoning process is then performed on the updated evidence graph for improved results.

A. Sources of Intrusion Evidence

Evidences for network forensics investigation can be classified into two categories: primary evidence and secondary evidence. Primary evidence refer to information that directly indicate attacks or security policy violations. For example, alerts from IDS and system integrity monitors are primary evidence. Secondary evidence refer to information that does not directly represent attacks but could provide complementary information for investigation. For example, raw network flow logs and host configurations are secondary evidence. Secondary evidence comes from extensive sources and in a much higher volume. Generally, primary evidence is the starting point of forensic investigation and provides the basis for searches towards secondary evidence. Querying the secondary evidence usually has two objectives: to discover hidden suspicious events and to evaluate the trustworthyness of primary evidence. In our current prototype, we use network IDS alerts as the primary evidence; raw network flow logs and host logs are used as secondary evidence.

B. Evidence Preprocessing

We apply the *Leader-Follower* model to aggregate raw IDS alerts into high level hyper alerts. The flexible alert aggregation algorithm merges raw alerts based on similarity of attributes with self-extending time window. Details of the *Leader-Follower* algorithm is described in [21].

C. Evidence Graph Model

We defined the evidence graph model to present the observed intrusion evidences as well as perform automated inference. An evidence graph is a quadruple G= (N, E, S, R), where N is the set of nodes, E is the set of directed edges, S is the set of labels that indicate the status of nodes and R is the set of labels that indicate the attributes of edges. In the evidence graph, each node represents a host of forensic investigation interest and each edge represents an observed intrusion evidence. Functionalities of our evidence graph model are:

1) The evidence graph provides the analyst an intuitive visualization of observed evidence;
2) The evidence graph provides a convenient interface for the analyst to interact with the evidence and add expert feedback.
3) The evidence graph provides the basis for automated reasoning procedure.

Each node in the evidence graph is characterized by the following labels:

1) Host: The suspicious host of interest.
2) States: States of the node is defined by a set of fuzzy variables S=$\{$*Attacker,Victim,Stepping Stone,Affiliated*$\}$.

The states are inferred through our local reasoning process via Rule-Based Fuzzy Cognitive Maps (RBFCM). We argue that it is essential to keep track of the evolution of host states because it provide context for evaluating evidence and helps to display the advancing stages of an attack to the forensic analyst.

Each edge in the evidence graph is represented by the following labels:

1) General attributes: The set of general attributes of an edge depends on the specific type of intrusion evidence. For network IDS alerts, we define the set of attributes as source/ target IP address, time stamp and classification. Time stamp of the edge is an interval [start time, end time].
2) Weight: Weight is a fuzzy value $w \in [0, 1]$ that represents the impact of evidence on the target system.
3) Host Importance: Host importance $h \in [0, 1]$ is an optional parameter to relate importance of evidence with certain hosts.
4) Relevancy: Relevancy is a fuzzy value $r \in [0, 1]$ that represents the belief that the attack indicated by the evidence would successfully achieve expected impact on the target host.

We calculate priority score for an edge to indicate the overall importance of the evidence in the attack context. The priority score $p(e)$ of an edge e is calculated as the product of its weight, relevancy and host importance.

D. Hierarchical Reasoning Framework

Based on the evidence graph, we develop a hierarchical reasoning framework of two levels: local reasoning and global reasoning.

1) Local Reasoning: The objective of local reasoning is to infer the state evolution of a host from observed evidence. In the current prototype,we develop causal inference via Rule-Based Fuzzy Cognitive Maps (RBFCM) to model the states of nodes. A RBFCM is essentially a standard rule based fuzzy system plus feedback and mechanisms to deal with causal relations [2]. As shown in figure 2, a RBFCM consists of fuzzy concepts and fuzzy rule bases. In our context, concepts are the

defined states {Attacker, Victim, Stepping Stone, Affiliated}. Fuzzy rule bases consist of "IF...Then..." fuzzy rules that define how each concept is affected by other concepts and inputs. The fuzzy rules are designed from expert knowledge.

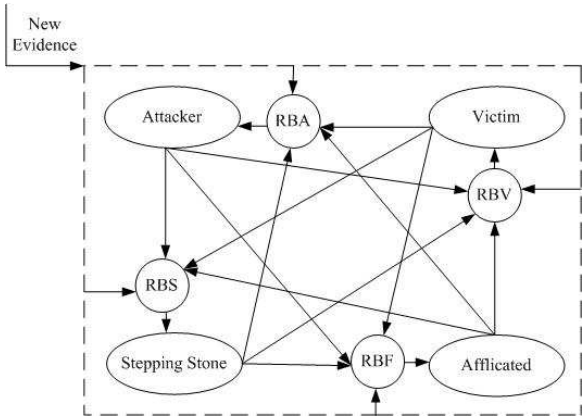

Fig. 2. RBFCM model for local reasoning

In the RBFCM shown in figure 2, fuzzy rules are used to map multiple inputs (current value of states and new evidence) to the output(updated value of states). States of a node in the evidence graph is updated in an incremental manner.

2) Global Reasoning: The objective of global reasoning is to identify a set of highly correlated hosts that belongs to the attack scenario of interest and derive their relationships by refining the local role estimates to properly fit into the scenario context. Based on the evidence graph, we approach the global reasoning task as a group detection problem, which is to discover potential members of an attack group given the intrusion evidence observed. The attack group detection procedure works in two different phases: (1) creating new attack groups by selecting seed for the group and (2) extending existing groups by discovering more hidden members. In the following we present our approach in these two tasks.

1) Seed Selection: The first phase of group detection is to select certain node as seed of the attack group. In our prototype tool, there two approaches for seed generation: (1) select seed based on states and context of nodes; (2) select seed based on certain graph metrics.
2) Group Expansion: The attack group expansion procedure is based on the intuitive notion that members of the same attack group should be strongly connected with each other. Based on the evidence graph,we compute the "distance" between two nodes and larger score indicates stronger correlation between two nodes in the attack. The distance between two nodes in the evidence graph is evaluated as the reciprocal of aggregated priority scores between them.

   In the group expansion process, we first identify all external neighbors of current seed members as the list of candidate nodes. In the second step, a ranked list is formed based on the distance between each candidate node to current group members; Finally, the ordered list is cut at a predefined threshold and nodes within the distance threshold are added to the group as new seed members of the group. If no candidate nodes is within the distance threshold, the group expansion procedure terminates.

III. Related Work

Our work extends current work in several areas into a flexible network forensics analysis mechanism.

Intrusion Detection Systems: Intrusion detection techniques have been widely studied since the early 1980's. Generally they are classified into two categories: anomaly detection and misuse detection [9]. IDS' are an important source of evidence for forensics analysis. However we cannot solely rely on IDS' as they only catch known attacks or unusual behavior. Also, the high volume and low quality of IDS alerts makes it difficult for forensics investigators to identify a clear picture of the attack. We incorporate aggregated IDS alerts into our evidence graph model and evaluate their effects in reasoning process.

Attribution Techniques: Attack attribution techniques aim to locate the true origin of attack flows. IP spoofing and stepping stone connections are two common techniques attackers use to conceal their origin [7]. Therefore, attribution techniques generally fall into two classes: stepping stone detection [22]–[24] and IP traceback [1], [8], [12], [19]. Attribution methods can be integrated into our analysis model as evidence sources on query. In future work, we will incorporate results of stepping stone detection and IP traceback into our local and global reasoning process.

Alert Correlation: As intrusion alerts only reflect elementary steps in an attack, alert correlation methods aim at reconstructing the attack scenario by linking alerts that satisfy certain relationships together. Past work on alert correlation include attribute similarity based [3], [5], [13], [14], [20], predefined scenario based [6], [10], [11], [15], pre/post condition based [4], [17], [18] and methods based on multiple information sources [16]. We extend a simple and flexible attribute-based alert aggregation mechanism derived from [20] in our evidence pre-processing module. Pre/post condition and predefined scenario based methods can be leveraged to calculate correlation scores in our global reasoning process. Our evidence graph model provides an intuitive view to correlate intrusion alerts with secondary evidence.

IV. Experimental Results

In our current prototype, we use Snort as the network IDS sensor to generate intrusion alerts and use TCPDUMP to collect raw network traffic in the testbed. Evidence collected are stored into a MySQL database. We implemented a set of Perl scripts to aggregate intrusion alerts, extract flow information and automatically integrate prior knowledge in reasoning process. We develop an application based on LEDA library to manipulate evidence graphs and reasoning results. In the following example, we illustrate our analysis procedure step by step and show results of local and global reasoning in detail.

A. Scenario Setup

A simple multi-stage attack scenario is implemented in our testbed. The attack group involved in the scenario consists of five hosts in separate subnets. Roles of hosts in the attack group is shown in table I. In addition to two stepping stones, the attacker also uses one third party host that has public ftp service as the relay for attack tools and stolen data.

TABLE I
ROLE CONFIGURATION OF HOSTS

Attacker	192.168.21.3
Stepping Stone 1	192.168.25.3
Stepping Stone 2	192.168.22.4
Victim	192.168.23.4
FTP Relay	192.168.24.4

The attack scenario includes the following steps:

1) Samba remote buffer overflow attack against stepping stone 1 from attacker. The attack is successful and a shell of stepping stone 1 is obtained.
2) Download attack tools from ftp relay to stepping stone 1 and start an Netcat backdoor on stepping stone 1. Establish connection from attacker to stepping stone 1 through the backdoor.
3) Windows DCOM remote buffer overflow attack against stepping stone 2 from stepping stone 1. The attack is successful and a shell of stepping stone 2 is obtained.
4) Download attack tools from ftp relay to stepping stone 2 and starts a backdoor on stepping stone 2. Establish backdoor connection from stepping stone 1 to stepping stone 2.
5) Frontpage Server 2000 buffer overflow attack against the victim from stepping stone 2. The attack is successful and a shell of stepping stone 2 is obtained.
6) Download backdoor program from ftp relay to victim and starts backdoor on victim.
7) Transfer data from the victim to the ftp relay and close backdoor connections.

B. Evidence Preprocessing

Throughout the attack process, Snort reported a huge number of 7501 alerts. With our aggregation procedure, the alerts information is highly condensed. The result is 4 hyper-alerts related to port scan activity and 17 hyper-alerts which represent possible exploits.

C. Constructing Evidence Graph

Based on the primary evidence of intrusion alerts, our graph generation module generated the initial evidence graph shown in figure 3. The number attached with each edge denotes the sequence of corresponding evidences in time order. Note that there's no exploit activities involved with the ftp relay, therefore it does not show up in the initial evidence graph.

To get a clearer view, the analyst can apply filter conditions to remove irrelevant evidences. On the other hand, the analyst can specify queries for secondary evidence to provide a more

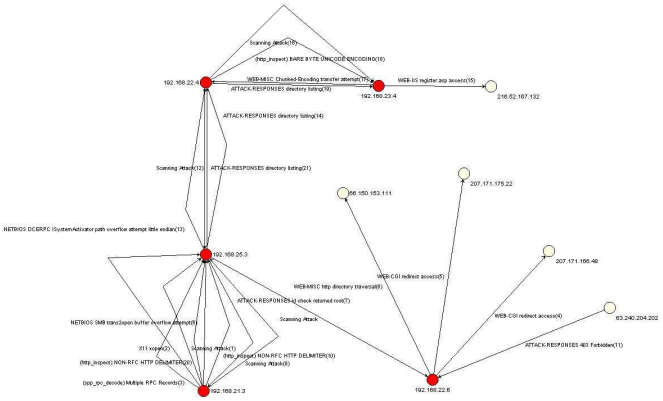

Fig. 3. Evidence Graph from primary evidence

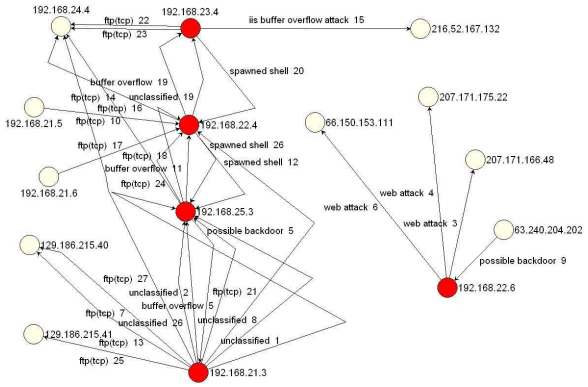

Fig. 4. Enriched Evidence Graph with secondary evidence

comprehensive view of what's going on in the network. For example, based on figure 3, we can ignore all port scan attacks and find out all file transfer connections to hosts in the evidence graph during a specified time window and get an updated evidence graph in figure 4. We notice that several potentially suspicious hosts include the ftp relay are brought up in the updated evidence graph. Specific alerts in figure 3 are replaced by abstractions from the knowledge base to offer a high level picture in figure 4.

D. Local Reasoning

In the first step, the analyst would examine the states of hosts from local reasoning process. Based on the RBFCM-based local reasoning procedure, states of nodes in the evidence graph are inferred and shown in table II. The hosts that have "Attacker" state activated are highlighted in figure 3 and figure 4.

E. Global Reasoning

First, we use degree of a node as the metrics to generate the initial seed for attack group. From table II we can see that host 192.168.22.4 has the highest degree, thus we choose it as the initial seed for attack group.

In the next step, we expand the group by evaluating the distance between candidate nodes to seeds of the group. Distance between neighbor pairs are shown in table III. Starting

TABLE II

LOCAL REASONING RESULTS

Host	degree	AT	VI	SS	AF
192.168.22.4	12	0.85	0.85	0.87	0.84
192.168.25.3	12	0.85	0.80	0.94	0.84
192.168.21.3	11	0.80	0	0	0.84
192.168.23.4	6	0.69	0.85	0	0.82
192.168.24.4	5	0	0	0	0.84
192.168.22.6	4	0.85	0.50	0	0
129.186.215.40	2	0	0	0	0.81
129.186.215.41	1	0	0	0	0.69
192.168.21.5	1	0	0	0	0.70
192.168.21.6	1	0	0	0	0.70
207.171.166.48	1	0	0.67	0	0
207.171.175.22	1	0	0.67	0	0
66.150.153.111	1	0	0.67	0	0
216.52.167.132	1	0	0.69	0	0
63.240.204.202	1	0	0.71	0	0

from the seed, the attack group is expanded incrementally. With distance threshold as 1, the result attack group is shown in figure 5. The highlighted oval node indicates the initial seed of the attack group and the highlighted square nodes are members of the attack group identified through the group expansion process. Each edge in figure 5 are labelled with distance between its source and target.

TABLE III

DISTANCE BETWEEN PAIR OF NODES

Host 1	Host 2	Distance
192.168.25.3	192.168.22.4	0.23
192.168.21.3	192.168.25.3	0.24
192.168.23.4	192.168.22.4	0.43
192.168.23.4	192.168.24.4	0.59
192.168.21.3	129.186.215.40	0.67
63.240.204.202	192.168.22.6	1.11
192.168.21.3	192.168.22.4	1.18
192.168.21.6	192.168.22.4	1.19
192.168.22.4	192.168.24.4	1.19
192.168.21.5	192.168.22.4	1.19
192.168.21.3	192.168.24.4	1.25
192.168.23.4	216.52.167.132	1.25
192.168.21.3	129.186.215.41	1.27
192.168.25.3	192.168.24.4	1.27
192.168.22.6	207.171.175.22	1.43
192.168.22.6	207.171.166.48	1.43
192.168.22.6	66.150.153.111	1.43

By observing the states of members in the attack group we can see that host 192.168.21.3 has only *Attacker* state activated. Hosts 192.168.25.3,192.168.22.4 both have *Stepping Stone* state activated. Host 192.168.23.4 has both *Attacker* and *Victim* state activated but not *Stepping Stone*. Further examination of the activation time of states clearly suggest that host 192.168.21.3 is the initial start point of attack and hosts 192.168.25.3,192.168.22.4 are used as stepping stones. Both 129.186.215.40 and 192.168.24.4 are affiliated with attacker and stepping stones, which suggest that more investigation is needed to find out whether they are truly related to the attack. Intuitively from the graph we can see that 192.168.24.4 seems more suspicious than 129.186.215.40 because it is affiliated with more members in the attack group. We can also

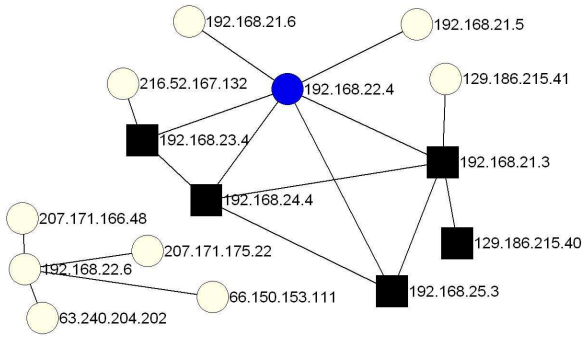

Fig. 5. Initial Seed and Attack Group for the scenario

observe that although 192.168.22.6 is labelled as an *Attacker* in local reasoning process, it is unrelated to the main attack group. Thus we can regard it as a background attacker when investigating the main attack group.

The results show that with sufficient evidence, our automated analysis mechanism is effective in discovering attack group and high-level scenario of the attack.

V. SUMMARY

We develop a prototype network forensics analysis tool that integrates automated reasoning with convenient presentation and interaction with intrusion evidence. This work is only the starting point of our efforts towards network forensics analysis. We expect to interact with forensics experts in the demonstration and evaluate our techniques with more realistic experiments.

REFERENCES

[1] Steven M. Bellovin. Internet draft: ICMP traceback messages. Available at http://www.ietf.org/internet-drafts/draft-bellovin-itrace-00.txt, March 2000.

[2] J. P. Carvalho and J. A. B. Tome. Rule Based Fuzzy Cognitive Maps: Fuzzy Causal Relations. In *Proceedings of the 8th International Fuzzy Systems Association World Congress(IFSA99)*, Taiwan, 1999.

[3] F. Cuppens. Managing alerts in a multi-intrusion detecttion environment. In *Proceedings of the 17th Annual Computer Security Applications Conference(ACSAC)*, 2001.

[4] F. Cuppens and A. Miege. Alert Correlation in a Cooperative Intrusion Detection Framework. In *Proceedings of the 2002 IEEE Symposium on Security and Privacy*, May 2002.

[5] O. Dain and R. Cunningham. Building scenarios from a heterogeneous alert stream. In *Proceedings of the 2001 IEEE workshop on Information Assurance and Security*, pages 231–235, 2001.

[6] O. Dain and R. Cunningham. Fusing a heterogeneous alert stream into scenarios. In *Proceedings of the 2001 ACM workshop on Data Mining for Security Applications*, pages 231–235, 2001.

[7] Thomas E. Daniels. *Reference Models for the Concealment and Observation of Origin Identity in Store-and-Forward Networks*. PhD thesis, Purdue University, West Lafayette,Indiana, 2002.

[8] Drew Dean, Matt Franklin, and Adam Stubblefield. An algebraic approach to ip traceback. In *Proceedings of 2001 Network and Distributed Systems Security Symposium*, pages 3–12, San Diego, California, February 2001.

[9] H. Debar, M. Dacer, and A. Wespi. A revised taxonomy for intrusion-detection systems. In *IBM Research Report*, 1999.

[10] Herve Debar and Andreas Wespi. Aggregation and Correlation of Intrusion-Detection Alerts. In *Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection(RAID)*, October 2001.

[11] S. Eckmann, G. Vigna, and R. Kemmerer. Statl: An attack language for state-based intrusion detection. Dept. of Computer Science, University of California, Santa Barbara., 2000.

[12] Institute for Security Technology Studies. Law enforcement tools and technologies for investigating cyber attacks: Gap analysis report. http://www.ists.dartmouth.edu, February 2004.

[13] K. Julisch. Mining alarm clusters to improve alarm handling efficiency. In *Proceedings of the 17th Annual Computer Security Applications Conference(ACSAC)*, pages 12–21, 2001.

[14] K. Julisch. Clustering intrusion detection alarms to support root cause analysis. In *ACM Transactions on Information and System Security*, pages 443–471, Nov 2003.

[15] 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'03)*, 2003.

[16] Benjamin Morin, Ludovic Me, Herve Debar, and Mireille Ducasse. M2D2: A Formal Data Model for IDS Alert Correlation. In *Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection*, pages 115–137, 2002.

[17] P. Ning, Y. Cui, and D. S. Reeves. Constructing attack scenarios through correlation of intrusion alerts. In *9th ACM Conference on Computer and Communicaitons Security*, November 2002.

[18] P. Ning and D. Xu. Learning attack stratagies from intrusion alerts. In *Proceedings of the 10th ACM Conference on Computer and Communicaitons Security*, 200-209, 2003.

[19] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer. Single-packet ip traceback. *IEEE/ACM Trans. Netw.*, 10(6):721–734, 2002.

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

[21] Wei Wang and Thomas E. Daniels. Building evidence graphs for network forensics analysis. Submitted to 21st Annual Computer Security Applications Conference(ACSAC'05).

[22] X. Wang and D. S. Reeves. Robust correlation of encrypted attack traffic through stepping stones by manipulation of interpacket delays. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS 2003)*, Washington DC, USA, Oct. 2003.

[23] K. Yoda and H. Etoh. Finding a connection chain for tracing intruders. In *Proceedings of the 6th European Symposium on Research in Computer Security (ESORICS 2000)*, Toulouse, France, Oct. 2000.

[24] Y. Zhang and V. Paxson. Detecting stepping stones. In *Proceedings of the 9th USENIX Security Symposium*, pages 171–184, Denver, USA, Aug. 2000.