A SURVEY ON ALERT AGGREGATION IN INTRUSION DETECTION

Dr.M.Ramesh kumar[1], Dr. W. R. Salem Jeyaseelan[2], Dr. N. Rajesh Kumar[3]

[1], Associate Professor, Department of Computer Science and Engineering,

[2], Associate Professor, Department of Information Technology,

[3], Associate Professor, Department of Electronics and Communication Engineering,

[1], VSB College of Engineering Technical Campus, Coimbatore, Tamilnadu, India.

[2], Karpagam College of Engineering (Autonomous), Coimbatore, Tamilnadu, India.

[3], KPR Institute of Engineering and Technology Coimbatore, Tamilnadu, India.

E-mail: maestro.ramesh@gmail.com, salemjeyaseelan@kce.ac.in, rajeshkumar.n@kpriet.ac.in

ABSTRACT

Alert aggregation is an important subtask of intrusion detection. The goal is to identify and to cluster different alerts produced by low-level intrusion detection systems, firewalls, etc. belonging to a specific attack instance which has been initiated by an attacker at a certain point in time. Thus, meta-alerts can be generated for the clusters that contain all the relevant information whereas the amount of data (i.e., alerts) can be reduced substantially. Meta-alerts may then be the basis for reporting to security experts or for communication within a distributed intrusion detection system. We propose a novel technique for online alert aggregation which is based on a dynamic, probabilistic model of the current attack situation. Basically, it can be regarded as a data stream version of a maximum likelihood approach for the estimation of the model parameters. We demonstrate that the number of missing meta-alerts is extremely low. In addition, meta-alerts are generated with a delay of typically only a few seconds after observing the first alert belonging to a new attack instance.

Index Terms—*Intrusion detection, alert correlation, alert reduction, correlation data sets.*

1. INTRODUCTION

Intrusion detection system (IDS) is a device or a software application that monitors the entire network and system activities for detecting malicious activities or policy violations and provides reports to the management. IDS are besides other protective measures such as virtual private networks, authentication mechanisms, or encryption techniques very important to guarantee information security. They help to defend against the various threats to which networks and hosts are exposed to by detecting the actions of attackers or attack tools in a network or host-based manner with misuse or anomaly detection techniques At present, most IDS are quite reliable in detecting suspicious actions by evaluating TCP/IP connections or log files, for instance. Once AN ID finds a suspicious action, it immediately creates an alert which contains.

Information about the source, target, and estimated type of the attack (e.g., SQL injection, buffer overflow, or denial of service). As the intrusive actions caused by a single attack instance— which is the occurrence of an attack of a particular type that has been launched by a specific attacker at a certain point in time—are often spread over many network connections or log file entries, a single attack instance often results in hundreds or even thousands of alerts. IDS usually focus on detecting attack types, but not on distinguishing between different attack instances. In addition, even low rates of false alerts could easily result in a high total number of false alerts if thousands of network packets or log file entries are inspected. The most general types of alerts are shown below,

TRUE POSITIVE ALERT	**FALSE POSITIVE ALERT**
A Legitimate attack which triggers an IDS to produce alert or alarm	An event signaling an IDS to produce an alarm when no attack has been taken place.
TRUE NEGATIVE ALERT	**FALSE NEGATIVE ALERT**
When no attack has been taken place and no alarm is raised	A failure of an IDS to detect an actual attack.

Table 1.1: General Types of Alerts

As a consequence, the IDS create many alerts at a low level of abstraction. It is extremely difficult for a human security expert to inspect this flood of alerts, and decisions that follow from single alerts might be wrong with a relatively high probability. In our opinion, a "perfect" IDS should be situation-aware [2] in the sense that at any point in time it should "know" what is going on in its environment regarding attack instances (of various types) and attackers. In this paper, we make an important step toward this goal by introducing and evaluating a new technique for alert aggregation. Alerts may originate from low-level IDS such as those mentioned above, from firewalls (FW), etc. Alerts that belong to one attack instance must be clustered together and meta-alerts must be generated for these clusters.

The main goal is to reduce the amount of alerts substantially without losing any important information which is necessary to identify ongoing attack instances. We want to have no missing met alerts, but in turn we accept false or redundant meta-alerts to a certain degree. This problem is not new, but current solutions are typically based on a quite simple sorting of alerts, e.g., according to their source, destination, and attack type. Under real conditions such as the presence of classification errors of the low-level IDS (e.g., false alerts), uncertainty with respect to the source of the attack due to spoofed IP addresses, or wrongly adjusted time windows, for instance, such an approach fails quiteoften.

2. SYSTEM A NALYSIS

Most existing IDS are optimized to detect attacks with high accuracy, However they still have following disadvantages, Our proposed system Online Intrusion Alert Aggregation with Generative Data Stream Modeling is a generative modeling approach using probabilistic methods. Assuming that attack instances can be regarded as random processes “producing” alerts, we aim at modeling these processes using approximate maximum likelihood parameter estimation techniques. Thus, the beginning as well as the completion of attack instances can be detected.

2.1 ADVANTAGES

It is a data stream approach, i.e., each observed alert is processed only a few times. Thus, it can be applied online and under harsh timing constraints. In the proposed scheme of Online Intrusion Alert Aggregation with Generative Data Stream Modeling, we extend our idea of sending Intrusion alerts to the mobile. This makes the process easier and comfortable.

Online Intrusion Alert Aggregation with Generative Data Stream Modeling does not degrade system performance as individual layers are independent and are trained with only a small number of features, thereby, resulting in an efficient system.

Online Intrusion Alert Aggregation with Generative Data Stream Modeling is easily customizable and the number of layers can be adjusted depending upon the requirements of the target network. Our framework is not restrictive in using a single method to detect attacks. Different methods can be seamlessly integrated in our framework to build effective intrusion detectors.

Our framework has the advantage that the type of attack can be inferred directly from the layer at which it is detected. As a result, specific intrusion response mechanisms can be activated for different attacks

3. FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

3.1 ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

3.2 TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.3 SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of thesystem.

4. SYSTEM DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

What data should be given as input?

How the data should be arranged orcoded?

The dialog to guide the operating personnel in providing input.

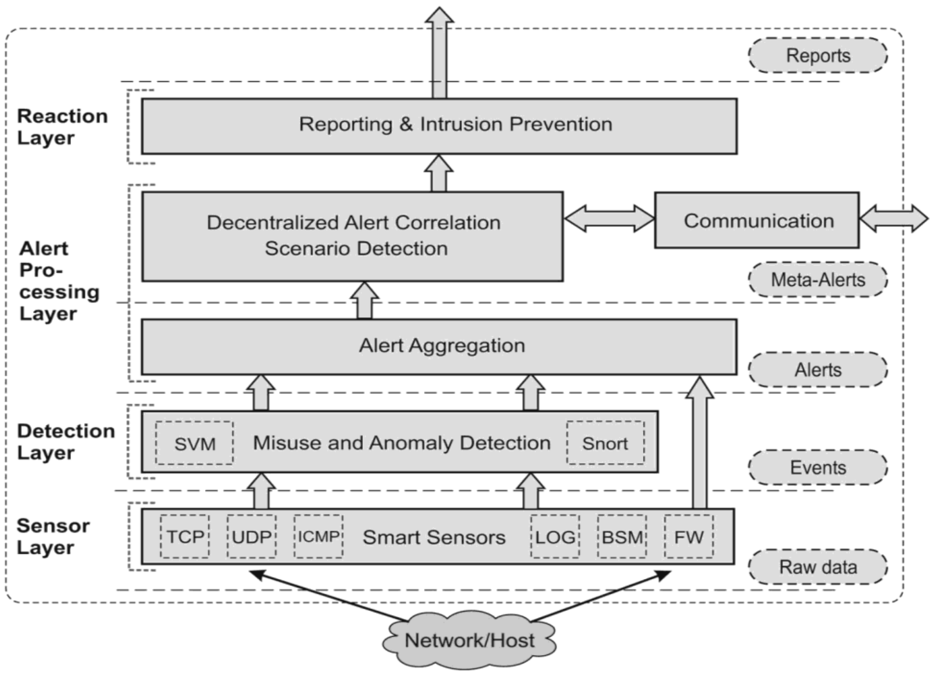
Methods for preparing input validations and steps to follow when error occur.

Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

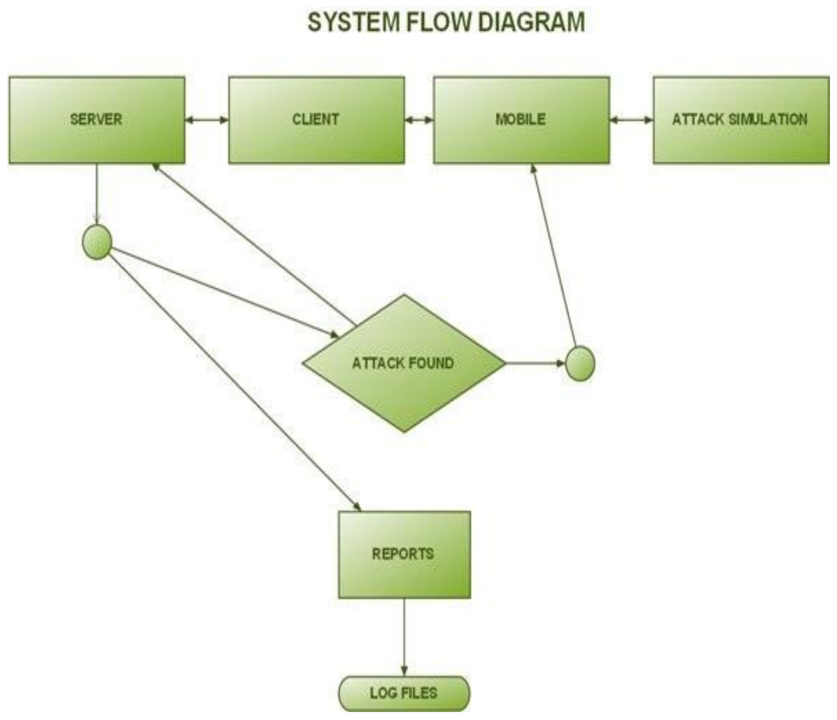
A quality output is one, which meets the requirements of the end user and presents the Information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.



Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

- ✓ Select methods for presenting information.
- ✓ Create document, report, or other formats that contain information produced by the system.

- ✓ The output form of an information system should accomplish one or more of the following objectives.
- ✓ Convey information about past activities, current status or projections of the Future.
- ✓ Signal important events, opportunities, problems, or warnings.
- ✓ Trigger an action.
- ✓ Confirm an action.
- ✓ System flow diagram



5. MODULES DESIGN

There are five Modules in

Client

Server

DARPA DataSet

Mobile

Attack Simulation

Server module is the main module for this project. This module acts as the Intrusion Detection System. This module consists of four layers viz. sensor layer (which detects the user/client etc.), Detection layer, alert processing layer and reaction layer. In addition there is also Message Log, where all the alerts and messages are stored for the references. Client module is developed for testing the Intrusion Detection System. In this module the client can enter only with a valid user name and password. If an intruder enters with any guessing passwords then the alert is given to the Server and the intruder is also blocked. Even if the valid user enters the correct user name and password, the user can use only for minimum number of times. For example even if the valid user makes the login for repeated number of times, the client will be blocked and the alert is sent to the admin. In the process level intrusion, each client would have given a specific process only. For example, a client may have given permission only for P1process. If the client tries to make more than these processes the client will be blocked and the alert is given by the Intrusion Detection System. In this client module the client can be able to send data. Here, when ever data is sent Intrusion Detection System checks for the file. If the size of the file is large then it is restricted or else the data is sent.

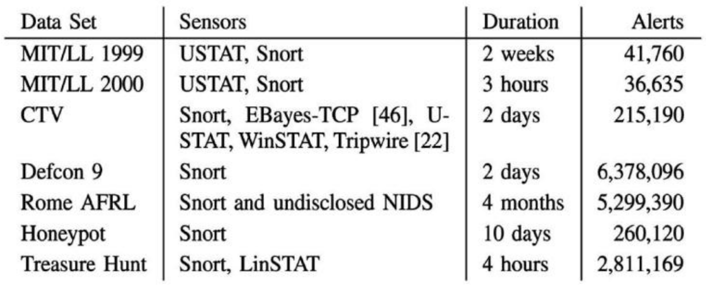
6. CORRELATION COMPONENTS AND ANALYSIS RESULTS

Alert correlation is a multicomponent process that receives as input a stream of alerts from multiple intrusion detection systems. In each component of the process, alerts are merged into high-level intrusion reports or tagged as non-relevant if they do not represent successful attacks. Finally, the alerts are prioritized according to the site's security policy, and eventually the results are reported.

Meta-Alerts

When two or more related alerts are merged as part of the alert correlation process, the result is called a meta-alert. The process of "merging" alerts refers to the action of subsuming a set of related alerts as a single meta-alert at a higher level of abstraction. Thus, a meta-alert is an intrusion report at a higher level of abstraction that comprises the information contained in all alerts that

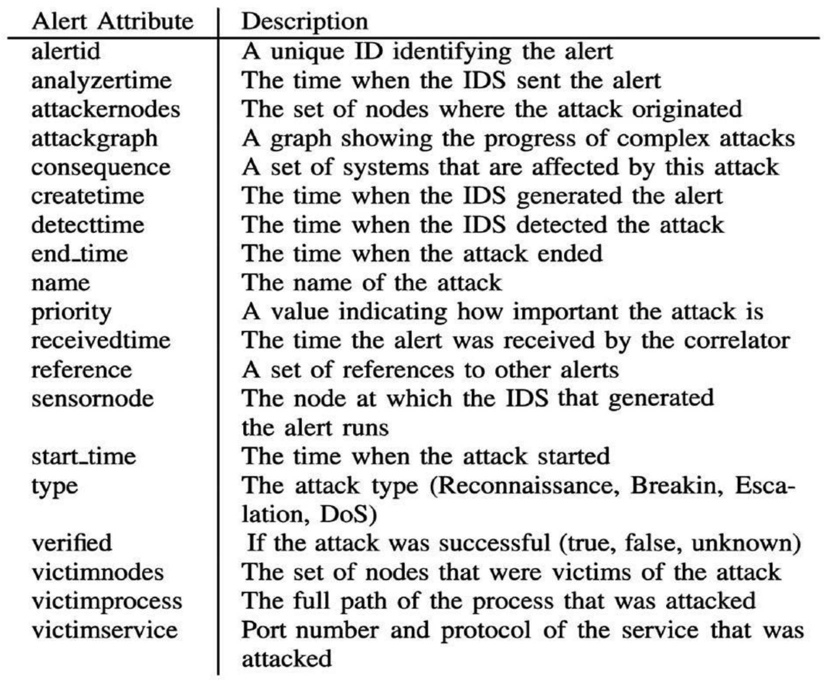
Data Set Summary Information



Data Set	Sensors	Duration	Alerts
MIT/LL 1999	USTAT, Snort	2 weeks	41,760
MIT/LL 2000	USTAT, Snort	3 hours	36,635
CTV	Snort, EBayes-TCP [46], U-STAT, WinSTAT, Tripwire [22]	2 days	215,190
Defcon 9	Snort	2 days	6,378,096
Rome AFRL	Snort and undisclosed NIDS	4 months	5,299,390
Honeypot	Snort	10 days	260,120
Treasure Hunt	Snort, LinSTAT	4 hours	2,811,169

The pseudocode for the normalization process is as follows:

```
global normalization_db, alertname_db
normalize(raw_alert)
{
alert    new alert
alert.alertid get_unique_id()
alert.name
get alertname from alertname_db using (raw_alert.name, raw_alert.sensortype) as key
mappings get all m:mapping from
normalization_db where m.sensor =
raw_alert.sensor
for each m:mapping in mappings:
alert_attr m.alert_attr
raw_attr m.raw_attr
alert.alert_attr raw_alert.raw_attr
pass alert to next correlation component
}
```



Alert Attribute	Description
alertid	A unique ID identifying the alert
analyzertime	The time when the IDS sent the alert
attackernodes	The set of nodes where the attack originated
attackgraph	A graph showing the progress of complex attacks
consequence	A set of systems that are affected by this attack
createtime	The time when the IDS generated the alert
detecttime	The time when the IDS detected the attack
end_time	The time when the attack ended
name	The name of the attack
priority	A value indicating how important the attack is
receivedtime	The time the alert was received by the correlator
reference	A set of references to other alerts
sensornode	The node at which the IDS that generated the alert runs
start_time	The time when the attack started
type	The attack type (Reconnaissance, Breakin, Escalation, DoS)
verified	If the attack was successful (true, false, unknown)
victimnodes	The set of nodes that were victims of the attack
victimprocess	The full path of the process that was attacked
victimservice	Port number and protocol of the service that was attacked

7. DARPA Dataset

This module is integrated in the Server module. This is an offline type of testing the intrusions. In this module, the DARPA Data Set is used to check the technique of the Online Intrusion Alert Aggregation with Generative Data Stream Modeling. The DARPA data set is downloaded and separated according to each layers. So we test the instance of DARPA Dataset using the open file dialog box. Whenever the dataset is chosen based on the conditions specified the Intrusion Detection System works. Mobile This module is developed using J2ME. The traditional system uses the message log for storing the alerts. In this system, the system admin or user can get the alerts in their mobile. Whenever alert message received in the message log of the server, the mobile too receives the alert message.

7.1 Attack Simulation

In this module, the attack simulation is made for our self to test the system. Attacks are classified and made to simulate here. Whenever an attack is launched the Intrusion Detection System must be capable of detecting it. So our system will also be capable of detecting such attacks. For example if an IP trace attack is launched, the Intrusion Detection System must detect it and must kill or block the process.

7.2 ALGORITHM FOR THE PROPOSED IDS

Step 1: Select the 'n' layers needed for the whole IDS.

Step 2: Build Sensor Layer to detect Network and Host Systems.

Step 4: Classify various types of alerts. (For example alert for System level intrusion or process Level intrusion)

Step 5: Code the system for detecting various types of attacks and alerts for respective attacks.

Step 6: Integrate the system with Mobile device to get alerts.

Step 7: Specify each type of alert on which category it falls, so that user can

Step 8: Build Reaction layer with various options so that administrator/user can have various options to select or react on any type of Intrusion.

Step 9: Test the system using Attack Simulation module, by sending attacks.

Step 10: Build a log file, so that all the reports generated can be saved for future references.

8.CONCLUSION

A novel technique for online alert aggregation is implemented to address the problem of accuracy and efficiency of Intrusion Detection System. The Developed and the presented architecture which was tested with the misuse based anomaly detection technique were successful. We also proposed a misuse based anomaly detection algorithm for our system. As our contribution, we make the system more efficient in identify the intrusion alerts and also we extend this work by sending the Alerts as Message to the Network Administrator who governs the Network or Intrusion Detection System. Most of the present existing Intrusion Detection System does not have a generalized framework. Our proposed architecture is similar to layered approach, so according to the network environment, the network administrator can add or remove the layers. If a new updated version of detection comes in future, then it will be very easy to add the layer with our proposed system. We also tested our system by launching various attacks to the system, and we found how the system detects and reacts according to the developed IDS. As a future work, this work can be extended as not only to detect attacks and also to prevent attacks. As mentioned earlier, our proposed system allows adding new layers, the prevention layer functionality layer can also be added with our system, as a future work.

REFERENCES

[1] Alexander Hofmann and Bernhard Sick, “Online Intrusion Alert Aggregation with Generative Data Stream Modeling”, IEEE Transactions on Dependable and Secure Computing, Vol. 8, No. 2, March – April 2011.

[2] A. Allen, “Intrusion Detection Systems: Perspective,” Technical Report DPRO- 95367, Gartner, Inc., 2003.

[3] S. Axelsson, “Intrusion Detection Systems: A Survey and Taxonomy,” Technical Report 99-15, Dept. of Computer Eng., Chalmers Univ. of Technology, 2000.

[4] M. Bishop, Pattern Recognition and Machine Learning. Springer, 2006.

[5] M.R. Endsley, “Theoretical Underpinnings of Situation Awareness: A Critical Review,” Situation Awareness Analysis andMeasurement, M.R. Endsley and D.J. Garland, eds., chapter 1, pp. 3-32, Lawrence Erlbaum Assoc., 2000.

[6] M.R. Henzinger, P. Raghavan, and S. Rajagopalan, Computing on Data Streams. Am. Math. Soc., 1999.

[7] Kapil Kumar Gupta, Baikunth Nath and Ramamohanarao Kotagiri, “Layered Approach using Conditional Random Fields for Intrusion Detection”, IEEE Transactions on Dependable and Secure Computing, Vol.7, No.1, January-March 2010.

[8] F. Valeur, G. Vigna, C. Kru¨ gel, and R.A. Kemmerer, “AComprehensive Approach to Intrusion Detection Alert Correlation,” IEEE Trans. Dependable and Secure Computing, vol. 1, no. 3, pp. 146-169, July-Sept. 2004.

[9] R. Durst, T. Champion, B. Witten, E. Miller, and L. Spagnuolo, “Addendum to Testing and Evaluating Computer Intrusion Detection Systems,” Comm. ACM, vol. 42, no. 9, p. 15, Sept. 1999.

[10] R. Durst, T. Champion, B. Witten, E. Miller, and L. Spagnuolo, “Testing and Evaluating Computer Intrusion Detection Systems,” Comm. ACM, vol. 42, no. 7, pp. 53-61, July 1999.

[11] S.T. Eckmann, G. Vigna, and R.A. Kemmerer, “STATL: An Attack Language for State-Based Intrusion Detection,” J. Computer Security, vol. 10, nos. 1-2, pp. 71-104, 2002.

[12] A.K. Ghosh, J. Wanken, and F. Charron, “Detecting Anomalous and Unknown Intrusions against Programs,” Proc. Ann. Computer Security Application Conf. (ACSAC ’98), pp. 259-267, Dec. 1998.

[13] R. Gula, “Correlating IDS Alerts with Vulnerability Information,” technical report, Tenable Network Security, Dec. 2002.

[14] J. Haines, D.K. Ryder, L. Tinnel, and S. Taylor, “Validation of Sensor Alert Correlators,” IEEE Security and Privacy Magazine, vol. 1, no. 1, pp. 46-56, Jan./Feb. 2003.

[15] L.T. Heberlein, G.V. Dias, K.N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, “A Network Security Monitor,” Proc. IEEE Symp. Research in Security and Privacy, pp. 296-304, May 1990.

[16] K. Ilgun, “USTAT: A Real-Time Intrusion Detection System for UNIX,” Proc. IEEE Symp. Research on Security and Privacy, May 1993.

[17] ISS, Realsecure, http://www.iss.net/, 2004.

[18] H.S. Javitz and A. Valdes, “The NIDES Statistical Component Description and Justification,” technical report, SRI Int’l, Mar. 1994.

[19] K. Kendall, “A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems,” master’s thesis, MIT, June 1999.