Available online at www.sciencedirect.com
ScienceDirect
Computer Networks 51 (2007) 3448–3470
Computer Networks
www.elsevier.com/locate/comnet

An overview of anomaly detection techniques: Existing solutions and latest technological trends

Animesh Patcha *, Jung-Min Park

Bradley Department of Electrical and Computer Engineering, Virginia Polytechnic Institute and State University, Blacksburg, VA 24061, United States

Received 8 June 2006; received in revised form 8 February 2007; accepted 9 February 2007
Available online 16 February 2007

Responsible Editor: Christos Douligeris

Abstract

As advances in networking technology help to connect the distant corners of the globe and as the Internet continues to expand its influence as a medium for communications and commerce, the threat from spammers, attackers and criminal enterprises has also grown accordingly. It is the prevalence of such threats that has made intrusion detection systems—the cyberspace's equivalent to the burglar alarm—join ranks with firewalls as one of the fundamental technologies for network security. However, today's commercially available intrusion detection systems are predominantly signature-based intrusion detection systems that are designed to detect known attacks by utilizing the signatures of those attacks. Such systems require frequent rule-base updates and signature updates, and are not capable of detecting unknown attacks. In contrast, anomaly detection systems, a subset of intrusion detection systems, model the normal system/network behavior which enables them to be extremely effective in finding and foiling both known as well as unknown or "zero day" attacks. While anomaly detection systems are attractive conceptually, a host of technological problems need to be overcome before they can be widely adopted. These problems include: high false alarm rate, failure to scale to gigabit speeds, etc. In this paper, we provide a comprehensive survey of anomaly detection systems and hybrid intrusion detection systems of the recent past and present. We also discuss recent technological trends in anomaly detection and identify open problems and challenges in this area.

Keywords: Survey; Anomaly detection; Machine learning; Statistical anomaly detection; Data mining

1. Introduction

Today, the Internet along with the corporate network plays a major role in creating and advancing new business avenues. Business needs have motivated enterprises and governments across the globe to develop sophisticated, complex information networks. Such networks incorporate a diverse array of technologies, including distributed data storage systems, encryption and authentication techniques, voice and video over IP, remote and wireless access, and web services. Moreover, corporate networks

* Corresponding author. Tel.: +1 540 239 0574.
E-mail addresses: apatcha@vt.edu (A. Patcha), jungmin@vt.edu (J.-M. Park).

1389-1286/$ - see front matter
doi:10.1016/j.comnet.2007.02.001

have become more accessible; for instance, most businesses allow access to their services on their internal networks via extranets to their partners, enable customers to interact with the network through e-commerce transactions, and allow employees to tap into company systems through virtual private networks.

The aforementioned access points make today's networks more vulnerable to intrusions and attacks. Cyber-crime is no longer the prerogative of the stereotypical hacker. Joining ranks with the hackers are disgruntled employees, unethical corporations, and even terrorist organizations. With the vulnerability of present-day software and protocols combined with the increasing sophistication of attacks, it comes as no surprise that network-based attacks are on the rise [1–4]. The 2005 annual computer crime and security survey [5], jointly conducted by the Computer Security Institute and the FBI, indicated that the financial losses incurred by the respondent companies due to network attacks/intrusions were US $130 million. In another survey commissioned by VanDyke Software in 2003, some 66% of the companies stated that they perceived system penetration to be the largest threat to their enterprises. Although 86% of the respondents used firewalls, their consensus was that firewalls by themselves are not sufficient to provide adequate protection. Moreover, according to recent studies, an average of twenty to forty new vulnerabilities in commonly used networking and computer products are discovered every month. Such wide-spread vulnerabilities in software add to today's insecure computing/networking environment. This insecure environment has given rise to the ever evolving field of *intrusion detection* and *prevention*. The cyberspace's equivalent to the burglar alarm, *intrusion detection systems* complement the beleaguered firewall.

An intrusion detection system gathers and analyzes information from various areas within a computer or a network to identify possible security breaches. In other words, intrusion detection is the act of detecting actions that attempt to compromise the confidentiality, integrity or availability of a system/network. Traditionally, intrusion detection systems have been classified as a signature detection system, an anomaly detection system or a hybrid/compound detection system. A signature detection system identifies patterns of traffic or application data presumed to be malicious while anomaly detection systems compare activities against a "normal" baseline. On the other hand, a hybrid intrusion detection system combines the techniques of the two approaches. Both signature detection and anomaly detection systems have their share of advantages and drawbacks. The primary advantage of signature detection is that known attacks can be detected fairly reliably with a low false positive rate. The major drawback of the signature detection approach is that such systems typically require a signature to be defined for all of the possible attacks that an attacker may launch against a network. Anomaly detection systems have two major advantages over signature based intrusion detection systems. The first advantage that differentiates anomaly detection systems from signature detection systems is their ability to detect unknown attacks as well as "zero day" attacks. This advantage is because of the ability of anomaly detection systems to model the normal operation of a system/network and detect deviations from them. A second advantage of anomaly detection systems is that the aforementioned profiles of normal activity are customized for every system, application and/or network, and therefore making it very difficult for an attacker to know with certainty what activities it can carry out without getting detected. However, the anomaly detection approach has its share of drawbacks as well. For example, the intrinsic complexity of the system, the high percentage of false alarms and the associated difficulty of determining which specific event triggered those alarms are some of the many technical challenges that need to be addressed before anomaly detection systems can be widely adopted.

The aim of this paper is twofold. The first is to present a comprehensive survey of recent literature in the domain of anomaly detection. In doing so, we attempt to assess the ongoing work in this area as well as consolidate the existing results. While the emphasis of this paper is to survey anomaly detection techniques proposed in the last six years, we have also described some of the earlier work that is seminal to this area. For a detailed exposition of techniques proposed before 2000, the reader is directed to the survey by Axelsson [6]. Our second aim is to identify the open problems and the research challenges.

The remainder of this article is organized as follows. In Section 2, we define intrusion detection, put forth the generic architectural design of an intrusion detection system, and highlight the three main techniques for detecting intrusions/attacks in

computer networks and systems. In Section 3, we describe the premise of anomaly detection and provide detailed discussions on the various techniques used in anomaly detection. We highlight recently proposed hybrid systems in Section 4, and conclude the paper by discussing open problems and research challenges in Section 5.

2. Intrusion detection

An *intrusion detection system* is a software tool used to detect unauthorized access to a computer system or network. An intrusion detection system is capable of detecting all types of malicious network traffic and computer usage. This includes network attacks against vulnerable services, data driven attacks on applications, host-based attacks—such as privilege escalation, unauthorized logins and access to sensitive files—and malware. An intrusion detection system is a dynamic monitoring entity that complements the static monitoring abilities of a firewall. An intrusion detection system monitors traffic in a network in promiscuous mode, very much like a network sniffer. The network packets that are collected are analyzed for rule violations by a pattern recognition algorithm. When rule violations are detected, the intrusion detection system alerts the administrator.

One of the earliest work that proposed intrusion detection by identifying abnormal behavior can be attributed to Anderson [7]. In his report, Anderson presents a threat model that classifies threats as *external penetrations*, *internal penetrations*, and *misfeasance*, and uses this classification to develop a security monitoring surveillance system based on detecting anomalies in user behavior. External penetrations are defined as intrusions that are carried out by unauthorized computer system users; internal penetrations are those that are carried out by authorized users who are not authorized for the data that is compromised; and misfeasance is defined as the misuse of authorized access both to the system and to its data.

In a seminar paper, Denning [8] put forth the idea that intrusions to computers and networks could be detected by assuming that users of a computer/network would behave in a manner that enables automatic profiling. In other words, a model of the behavior of the entity being monitored could be constructed by an intrusion detection system, and subsequent behavior of the entity could be verified against the entity's model. In this model, behavior that deviates sufficiently from the norm is considered anomalous. In the paper, Denning mentioned several models that are based on statistics, Markov chains, time-series, etc.

In a much cited survey on intrusion detection systems, Axelsson [9] put forth a generalized model of a typical intrusion detection system. Fig. 1 depicts such a system where solid arrows indicate data/control flow while dotted arrows indicate a response to intrusive activity. According to Axelsson, the generic architectural model of an intrusion detection system contains the following modules:

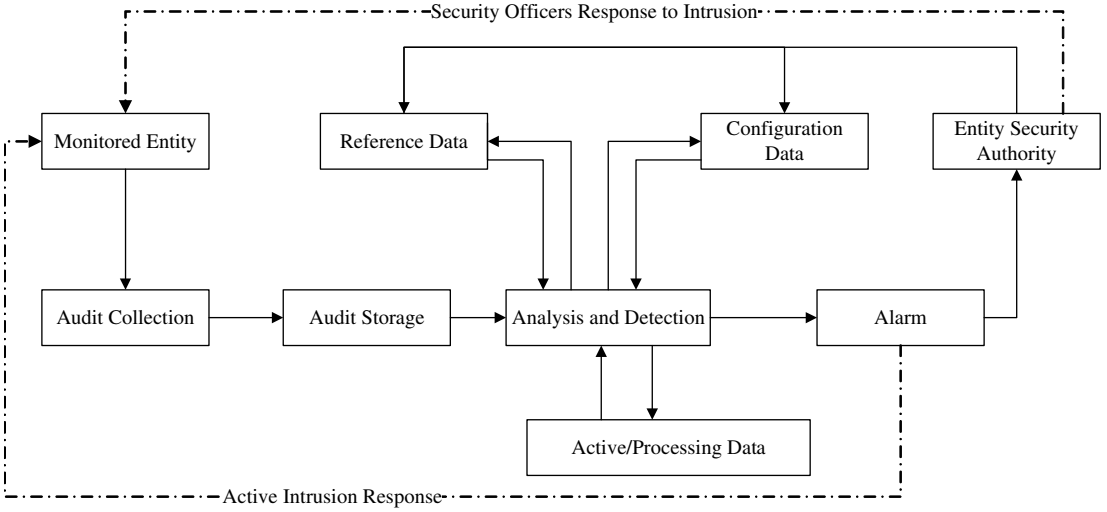


Fig. 1. Organization of a generalized intrusion detection system [9].

- *Audit data collection*: This module is used in the data collection phase. The data collected in this phase are analyzed by the intrusion detection algorithm to find traces of suspicious activity. The source of the data can be host/network activity logs, command-based logs, application-based logs, etc.
- *Audit data storage*: Typical intrusion detection systems store the audit data either indefinitely or for a sufficiently long time for later reference. The volume of data is often exceedingly large. Hence, the problem of audit data reduction is a major research issue in the design of intrusion detection systems.
- *Analysis and detection*: The processing block is the heart of an intrusion detection system. It is here that the algorithms to detect suspicious activities are implemented. Algorithms for the analysis and detection of intrusions have been traditionally classified into three broad categories: *signature* (or *misuse*) *detection*, *anomaly detection* and *hybrid* (or *compound*) *detection*.
- *Configuration data*: The configuration data are the most sensitive part of an intrusion detection system. It contains information that is pertinent to the operation of the intrusion detection system itself such as information on how and when to collect audit data, how to respond to intrusions, etc.
- *Reference data*: The reference data storage module stores information about known intrusion signatures (in the case of signature detection) or profiles of normal behavior (in the case of anomaly detection). In the latter case, the profiles are updated when new knowledge about system behavior is available.
- *Active/processing data*: The processing element must frequently store intermediate results such as information about partially fulfilled intrusion signatures.
- *Alarm*: This part of the system handles all output from the intrusion detection system. The output may be either an automated response to an intrusion or a suspicious activity alert for a system security officer.

Historically, intrusion detection research has concentrated on the analysis and detection stage of the architectural model shown in Fig. 1. As mentioned above, algorithms for the analysis and detection of intrusions/attacks are traditionally classified into the following three broad categories:

- *Signature or misuse detection* is a technique for intrusion detection that relies on a predefined set of attack signatures. By looking for specific patterns, the signature detection-based intrusion detection systems match incoming packets and/or command sequences to the signatures of known attacks. In other words, decisions are made based on the knowledge acquired from the model of the intrusive process and the observed trace that it has left in the system. Legal or illegal behavior can be defined and compared with observed behavior. Such a system tries to collect evidence of intrusive activity irrespective of the normal behavior of the system. One of the chief benefits of using signature detection is that known attacks can be detected reliably with a low false positive rate. The existence of specific attack sequences ensures that it is easy for the system administrator to determine exactly which attacks the system is currently experiencing. If the audit data in the log files do not contain the attack signature, no alarm is raised. Another benefit is that the signature detection system begins protecting the computer/network immediately upon installation. One of the biggest problems with signature detection systems is maintaining state information of signatures in which an intrusive activity spans multiple discrete events—that is, the complete attack signature spans multiple packets. Another drawback is that the signature detection system must have a signature defined for all of the possible attacks that an attacker may launch. This requires frequent signature updates to keep the signature database up-to-date.
- An *anomaly detection system* first creates a baseline profile of the normal system, network, or program activity. Thereafter, any activity that deviates from the baseline is treated as a possible intrusion. Anomaly detection systems offer several benefits. First, they have the capability to detect insider attacks. For instance, if a user or someone using a stolen account starts performing actions that are outside the normal user-profile, an anomaly detection system generates an alarm. Second, because the system is based on customized profiles, it is very difficult for an attacker to know with certainty what activity it can carry out without setting off an alarm. Third, an anomaly detection system has the ability to detect previously unknown attacks. This is due to the fact that a profile of intrusive activity is not based

on specific signatures representing known intrusive activity. An intrusive activity generates an alarm because it deviates from normal activity, not because someone configured the system to look for a specific attack signature. Anomaly detection systems, however, also suffer from several drawbacks. The first obvious drawback is that the system must go through a training period in which appropriate user profiles are created by defining "normal" traffic profiles. Moreover, creating a normal traffic profile is a challenging task. The creation of an inappropriate normal traffic profile can lead to poor performance. Maintenance of the profiles can also be time-consuming. Since, anomaly detection systems are looking for anomalous events rather than attacks, they are prone to be affected by time consuming false alarms. False alarms are classified as either being false positive or false negative. A *false positive* occurs when an IDS reports as an intrusion an event that is in fact legitimate network activity. A side affect of false positives, is that an attack or malicious activity on the network/system could go undetected because of all the previous false positives. This failure to detect an attack is termed as a *false negative* in the intrusion detection jargon. A key element of modern anomaly detection systems is the alert correlation module. However, the high percentage false alarms that are typically generated in anomaly detection systems make it very difficult to associate specific alarms with the events that triggered them. Lastly, a pitfall of anomaly detection systems is that a malicious user can train an anomaly detection system gradually to accept malicious behavior as normal.

- A *hybrid* or *compound detection system* combines both approaches. In essence, a hybrid detection system is a signature inspired intrusion detection system that makes a decision using a "hybrid model" that is based on both the normal behavior of the system and the intrusive behavior of the intruders.

3. Anomaly detection techniques

An anomaly detection approach usually consists of two phases: a *training* phase and a *testing* phase. In the former, the normal traffic profile is defined; in the latter, the learned profile is applied to new data.

3.1. Premise of anomaly detection

The central premise of anomaly detection is that intrusive activity is a subset of anomalous activity [10]. If we consider an intruder, who has no idea of the legitimate user's activity patterns, intruding into a host system, there is a strong probability that the intruder's activity will be detected as anomalous. In the ideal case, the set of anomalous activities will be the same as the set of intrusive activities. In such a case, flagging all anomalous activities as intrusive activities results in no *false positives* and no *false negatives*. However, intrusive activity does not always coincide with anomalous activity. Kumar and Stafford [10] suggested that there are four possibilities, each with a non-zero probability:

- *Intrusive but not anomalous*: These are false negatives. An intrusion detection system fails to detect this type of activity as the activity is not anomalous. These are called false negatives because the intrusion detection system falsely reports the absence of intrusions.
- *Not intrusive but anomalous*: These are false positives. In other words, the activity is not intrusive, but because it is anomalous, an intrusion detection system reports it as intrusive. These are called false positives because an intrusion detection system falsely reports intrusions.
- *Not intrusive and not anomalous*: These are true negatives; the activity is not intrusive and is not reported as intrusive.
- *Intrusive* and *anomalous*: These are true positives; the activity is intrusive and is reported as such.

When false negatives need to be minimized, thresholds that define an anomaly are set low. This results in many false positives and reduces the efficacy of automated mechanisms for intrusion detection. It creates additional burdens for the security administrator as well, who must investigate each incident and discard false positive instances.

3.2. Techniques used in anomaly detection

In this subsection, we review a number of different architectures and methods that have been proposed for anomaly detection. These include statistical anomaly detection, data-mining based methods, and machine learning based techniques.

3.2.1. Statistical anomaly detection

In statistical methods for anomaly detection, the system observes the activity of subjects and generates profiles to represent their behavior. The profile typically includes such measures as activity intensity measure, audit record distribution measure, categorical measures (the distribution of an activity over categories) and ordinal measure (such as CPU usage). Typically, two profiles are maintained for each subject: the current profile and the stored profile. As the system/network events (viz. audit log records, incoming packets, etc.) are processed, the intrusion detection system updates the current profile and periodically calculates an anomaly score (indicating the degree of irregularity for the specific event) by comparing the current profile with the stored profile using a function of abnormality of all measures within the profile. If the anomaly score is higher than a certain threshold, the intrusion detection system generates an alert.

Statistical approaches to anomaly detection have a number of advantages. Firstly, these systems, like most anomaly detection systems, do not require prior knowledge of security flaws and/or the attacks themselves. As a result, such systems have the capability of detecting ''zero day'' or the very latest attacks. In addition, statistical approaches can provide accurate notification of malicious activities that typically occur over extended periods of time and are good indicators of impending denial-of-service (DoS) attacks. A very common example of such an activity is a portscan. Typically, the distribution of portscans is highly anomalous in comparison to the usual traffic distribution. This is particularly true when a packet has unusual features (e.g., a crafted packet). With this in mind, even portscans that are distributed over a lengthy time frame will be recorded because they will be inherently anomalous.

However, statistical anomaly detection schemes also have drawbacks. Skilled attackers can train a statistical anomaly detection to accept abnormal behavior as normal. It can also be difficult to determine thresholds that balance the likelihood of false positives with the likelihood of false negatives. In addition, statistical methods need accurate statistical distributions, but, not all behaviors can be modeled using purely statistical methods. In fact, a majority of the proposed statistical anomaly detection techniques require the assumption of a quasi-stationary process, which cannot be assumed for most data processed by anomaly detection systems.

Haystack [11] is one of the earliest examples of a statistical anomaly-based intrusion detection system. It used both user and group-based anomaly detection strategies, and modeled system parameters as independent, Gaussian random variables. Haystack defined a range of values that were considered normal for each feature. If during a session, a feature fell outside the normal range, the score for the subject was raised. Assuming the features were independent, the probability distribution of the scores was calculated. An alarm was raised if the score was too large. Haystack also maintained a database of user groups and individual profiles. If a user had not previously been detected, a new user profile with minimal capabilities was created using restrictions based on the user's group membership. It was designed to detect six types of intrusions: attempted break-ins by unauthorized users, masquerade attacks, penetration of the security control system, leakage, DoS attacks and malicious use. One drawback of Haystack was that it was designed to work offline. The attempt to use statistical analyses for real-time intrusion detection systems failed, since doing so required high-performance systems. Secondly, because of its dependence on maintaining profiles, a common problem for system administrators was the determination of what attributes were good indicators of intrusive activity.

One of the earliest intrusion detection systems was developed at the Stanford Research Institute (SRI) in the early 1980's and was called the Intrusion Detection Expert System (IDES) [13,14]. IDES was a system that continuously monitored user behavior and detected suspicious events as they occurred. In IDES, intrusions could be flagged by detecting departures from established normal behavior patterns for individual users. As the analysis methodologies developed for IDES matured, scientists at SRI developed an improved version of IDES called the Next-Generation Intrusion Detection Expert System (NIDES) [15,16]. NIDES was one of the few intrusion detection systems of its generation that could operate in real time for continuous monitoring of user activity or could run in a batch mode for periodic analysis of the audit data. However, the primary mode of operation of NIDES was to run in real-time. A flow chart describing the real time operation of NIDES is shown in Fig. 2. Unlike IDES, which is an anomaly detection system, NIDES is a hybrid system that has an upgraded statistical analysis engine. In both IDES and NIDES, a profile of normal behavior based

on a selected set of variables is maintained by the statistical analysis unit. This enables the system to compare the current activity of the user/system/network with the expected values of the audited intrusion detection variables stored in the profile and then flag an anomaly if the audited activity is sufficiently far from the expected behavior. Each variable in the stored profile reflects the extent to which a particular type of behavior is similar to the profile built for it under "normal conditions". The way that this is computed is by associating each measure/variable to a corresponding random variable. The frequency distribution is built and updated over time, as more audit records are analyzed. It is computed as an exponential weighted sum with a half-life of 30 days. This implies that the half-life value makes audit records that were gathered 30 days in the past to contribute with half as much weight as recent records; those gathered 60 days in the past contribute one-quarter as much weight, and so on. The frequency distribution is kept in the form of a histogram with probabilities associated with each one of the possible ranges that the variable can take. The cumulative frequency distribution is then built by using the ordered set of bin probabilities. Using this frequency distribution, and the value of the corresponding measure for the current audit record, it is possible to compute a value that reflects how far away from the "normal" value of the measure the current value is. The actual computation in NIDES [16] renders a value that is correlated with how abnormal this measure is. Combining the values obtained for each measure and taking into consideration the correlation between measures, the unit computes an index of how far the current audit record is from the normal state. Records beyond a threshold are flagged as possible intrusions.

However the techniques used in [13–16] have several drawbacks. Firstly, the techniques are sensitive to the normality assumption. If data on a measure are not normally distributed, the techniques would yield a high false alarm rate. Secondly, the techniques are predominantly univariate in that a statistical norm profile is built for only one measure of the activities in a system. However, intrusions often affect multiple measures of activities collectively.

Statistical Packet Anomaly Detection Engine (SPADE) [17] is a statistical anomaly detection system that is available as a plug-in for SNORT [18], and is can be used for automatic detection of stealthy port scans. SPADE was one of the first papers that proposed using the concept of an anomaly score to detect port scans, instead of using the traditional approach of looking at p attempts over q seconds. In [17], the authors used a simple frequency based approach, to calculate the "*anomaly score*" of a packet. The fewer times a given packet was seen, the higher was its anomaly score. In other words, the authors define an anomaly score

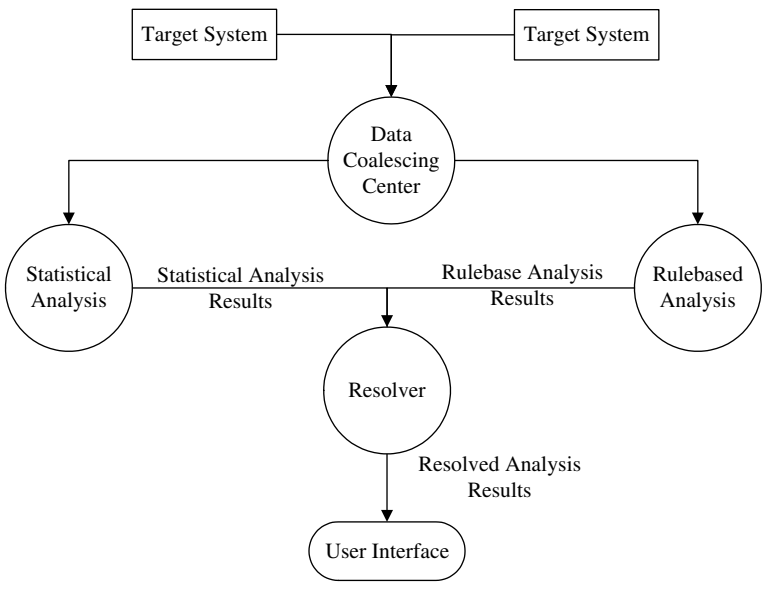


Fig. 2. Flow chart of real time operation in NIDES [12].

as the degree of strangeness based on recent past activity. Once the anomaly score crossed a threshold, the packets were forwarded to a correlation engine that was designed to detect port scans. However, the one major drawback for SPADE is that it has a very high false alarm rate. This is due to the fact that SPADE classifies all unseen packets as attacks regardless of whether they are actually intrusions or not.

Anomalies resulting from intrusions may cause deviations on multiple measures in a collective manner rather than through separate manifestations on individual measures. To overcome the latter problem, Ye et al. [19] presented a technique that used the Hotellings T^2 test[1] to analyze the audit trails of activities in an information system and detect host based intrusions. The assumption is that host based intrusions leave trails in the audit data. The advantage of using the Hotellings T^2 test is that it aids in the detection of both counter relationship anomalies as well as mean-shift anomalies. In another paper, Kruegel et al. [20] show that it is possible to find the description of a system that computes a payload byte distribution and combines this information with extracted packet header features. In this approach, the resultant ASCII characters are sorted by frequency and then aggregated into six groups. However, this approach leads to a very coarse classification of the payload.

A problem that many network/system administrators face is the problem of defining, on a global scale, what network/system/user activity can be termed as "normal". Maxion and Feather [21] characterized the normal behavior in a network by using different templates that were derived by taking the standard deviations of Ethernet load and packet count at various periods in time. An observation was declared anomalous if it exceeded the upper bound of a predefined threshold. However, Maxion et al. did not consider the non-stationary nature of network traffic which would have resulted in minor deviations in network traffic to go unnoticed.

More recently, analytical studies on anomaly detection systems were conducted. Lee and Xiang [22] used several information-theoretic measures, such as entropy and information gain, to evaluate the quality of anomaly detection methods, determine system parameters, and build models. These metrics help one to understand the fundamental properties of audit data. The highlighting features of some of the schemes surveyed in this section are presented in Table 1.

3.2.2. Machine learning based anomaly detection

Machine learning can be defined as the ability of a program and/or a system to learn and improve their performance on a certain task or group of tasks over time. Machine learning aims to answer many of the same questions as statistics or data mining. However, unlike statistical approaches which tend to focus on understanding the process that generated the data, machine learning techniques focus on building a system that improves its performance based on previous results. In other words systems that are based on the machine learning paradigm have the ability to change their execution strategy on the basis of newly acquired information.

3.2.2.1. System call based sequence analysis. One of the widely used machine learning techniques for anomaly detection involves learning the behavior of a program and recognizing significant deviations from the normal. In a seminal paper, Forrest et al. [23] established an analogy between the human immune system and intrusion detection. They did this by proposing a methodology that involved analyzing a program's system call sequences to build a normal profile. In their paper, they analyzed several UNIX based programs like *sendmail*, *lpr*, etc., and showed that correlations in fixed length sequences of system calls could be used to build a normal profile of a program. Therefore, programs that show sequences that deviated from the normal sequence profile could then be considered to be victims of an attack. The system they developed was only used off-line using previously collected data and used a quite simple table-lookup algorithm to learn the profiles of programs. Their work was extended by Hofmeyr et al. [24], where they collected a database of normal behavior for each program of interest. Once a stable database is constructed for a given program in a particular environment, the database was then used to monitor the program's behavior. The sequences of system calls formed the set of normal patterns for the database, and sequences not found in the database indicated anomalies.

[1] The Hotelling's T^2 test statistic for an observation x_i is determined as $T^2 = n(x_i - \bar{\mathbf{x}})'\mathbf{W}^{-1}(x_i - \bar{\mathbf{x}})$ where $\bar{\mathbf{x}}$ is the mean, $x_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ denote an observation of p measures on a processor system at time i and W is the sample variance. A large value of T^2 indicates a large deviation observation x_i from the in-control population.

Table 1
A summary of statistical anomaly detection systems

Reference	Highlighting feature	Methodology
Haystack [11]	It uses descriptive statistics to model user behavior Also modeled acceptable behavior for a generic user within a particular user group	Host based statistical anomaly detection
NIDES [15]	A distributed intrusion detection system that had both anomaly as well as signature detection modules	Network based statistical anomaly detection
Staniford et al. [17]	Statistical anomaly detection technique that calculates an anomaly score for each packet that it sees; it forwards the packets to a correlation engine for intrusion detection purposes when a predefined threshold was crossed	Network based statistical anomaly detection
Ye et al. [19]	It uses the Hotellings T^2 test to analyze the audit trails of activities in an computer system and detect host-based intrusions	Host based multivariate statistical anomaly detection

Another machine learning technique that has been frequently used in the domain of machine learning is the *sliding window method*. The sliding window method, a sequential learning methodology, converts the sequential learning problem into the classical learning problem. It constructs a *window classifier* h_w that maps an input window of width w into an individual output value y. Specifically, let $d = (w-1)/2$ be the "half-width" of the window. Then h_w predicts $y_{i,t}$ using the window $\langle x_{i,t-d}, x_{i,t-d+1}, \ldots, x_{i,t}, \ldots, x_{i,t+d-1}, x_{i,t+d}\rangle$. The window classifier h_w is trained by converting each sequential training example (x_i, y_i) into windows and then applying a standard machine learning algorithm. A new sequence $\mathbf{x}$ is classified by converting it to windows, applying h_w to predict each y_t and then concatenating the y_t's to form the predicted sequence $\mathbf{y}$. The obvious advantage of this sliding window method is that it permits any classical supervised learning algorithm to be applied. While, the sliding window method gives adequate performance in many applications; it does not take advantage of correlations between nearby y_t values. Specifically, the only relationships between nearby y_t values that are captured are those that are predictable from nearby x_t values. If there are correlations among the y_t values that are independent of the x_t values, then these are not captured. The sliding window method has been successfully used in a number of machine learning based anomaly detection techniques [25–27]. Warrender et al. [27] proposed a method that utilized sliding windows to create a database of normal sequences for testing against test instances. Eskin et al. [26], improved the traditional sliding window method by proposing a modeling methodology that uses dynamic length of a sliding window dependent on the context of the system-call sequence.

However, system call based approaches for host based intrusion detection system suffer from two drawbacks. Firstly, the computational overhead that is involved in monitoring every system call is very high. This high overhead leads to a performance degradation of the monitored system. The second problem is that system calls themselves are irregular by nature. This irregularity leads to high false positive rate as it becomes difficult to differentiate between normal and anomalous system calls.

3.2.2.2. Bayesian networks. A Bayesian network is a graphical model that encodes probabilistic relationships among variables of interest. When used in conjunction with statistical techniques, Bayesian networks have several advantages for data analysis [28]. Firstly, because Bayesian networks encode the interdependencies between variables, they can handle situations where data is missing. Secondly, Bayesian networks have the ability to represent causal relationships. Therefore, they can be used to predict the consequences of an action. Lastly, because Bayesian networks have both causal and probabilistic relationships, they can be used to model problems where there is a need to combine prior knowledge with data. Several researchers have adapted ideas from Bayesian statistics to create models for anomaly detection [29–31]. Valdes et al. [30] developed an anomaly detection system that employed naïve Bayesian networks[2] to perform intrusion detection on traffic bursts. Their model, which is a part of EMERALD [32], has the capability to potentially detect distributed attacks in which

[2] A naive Bayesian network is a restricted network that has only two layers and assumes complete independence between the information nodes (i.e., the random variables that can be observed and measured). These limitations result in a tree-shaped network with a single hypothesis node (root node) that has arrows pointing to a number of information nodes (child nodes). All child nodes have exactly one parent node, that is, the root node, and no other causal relationship between nodes are permitted.

each individual attack session is not suspicious enough to generate an alert. However, this scheme also has a few disadvantages. First, as pointed out in [29], the classification capability of naïve Bayesian networks is identical to a threshold based system that computes the sum of the outputs obtained from the child nodes. Secondly, because the child nodes do not interact between themselves and their output only influences the probability of the root node, incorporating additional information becomes difficult as the variables that contain the information cannot directly interact with the child nodes.

Another area, within the domain of anomaly detection, where Bayesian techniques have been frequently used is the classification and suppression of false alarms. Kruegel et al. [29] proposed a multisensor fusion approach where the outputs of different IDS sensors were aggregated to produce a single alarm. This approach is based on the assumption that any anomaly detection technique cannot classify a set of events as an intrusion with sufficient confidence. Although using Bayesian networks for intrusion detection or intruder behavior prediction can be effective in certain applications, their limitations should be considered in the actual implementation. Since the accuracy of this method is dependent on certain assumptions that are typically based on the behavioral model of the target system, deviating from those assumptions will decrease its accuracy. Selecting an inaccurate model will lead to an inaccurate detection system. Therefore, selecting an accurate model is the first step towards solving the problem. Unfortunately selecting an accurate behavioral model is not an easy task as typical systems and/or networks are complex.

3.2.2.3. Principal components analysis. Typical datasets for intrusion detection are typically very large and multidimensional. With the growth of high speed networks and distributed network based data intensive applications storing, processing, transmitting, visualizing and understanding the data is becoming more complex and expensive. To tackle the problem of high dimensional datasets, researchers have developed a dimensionality reduction technique known as principal component analysis (PCA) [33–35]. In mathematical terms, PCA is a technique where n correlated random variables are transformed into $d \leqslant n$ uncorrelated variables. The uncorrelated variables are linear combinations of the original variables and can be used to express the data in a reduced form. Typically, the first principal component of the transformation is the linear combination of the original variables with the largest variance. In other words, the first principal component is the projection on the direction in which the variance of the projection is maximized. The second principal component is the linear combination of the original variables with the second largest variance and orthogonal to the first principal component, and so on. In many data sets, the first several principal components contribute most of the variance in the original data set, so that the rest can be disregarded with minimal loss of the variance for dimension reduction of the dataset.

PCA has been widely used in the domain of image compression, pattern recognition and intrusion detection. Shyu et al. [36] proposed an anomaly detection scheme, where PCA was used as an outlier detection scheme and was applied to reduce the dimensionality of the audit data and arrive at a classifier that is a function of the principal components. They measured the Mahalanobis distance of each observation from the center of the data for anomaly detection. The Mahalanobis distance is computed based on the sum of squares of the standardized principal component scores. Shyu et al. evaluated their method over the KDD CUP99 data and have demonstrated that it exhibits better detection rate than other well known outlier based anomaly detection algorithms such as the Local Outlier Factor "LOF" approach, the Nearest Neighbor approach and the kth Nearest Neighbor approach. Other notable techniques that employ the principal component analysis methodology include the work done by Wang et al. [35], Bouzida et al. [37] and Wang et al. [38].

3.2.2.4. Markov models. Markov chains, have also been employed extensively for anomaly detection. Ye et al. [39], present an anomaly detection technique that is based on Markov chains. In their paper, system call event sequences from the recent past were studied by opening an observation window of size N. The type of audit events, $E_{t-N+1}, \ldots, E_t$ in the window at time t was examined and the sequence of states $X_{t-N+1}, \ldots, X_t$ obtained. Subsequently, the probability that the sequence of states $X_{t-N+1}, \ldots, X_t$ is normal was obtained. The larger the probability, the more likely the sequence of states results from normal activities. A sequence of states from attack activities is presumed to receive a low probability of support from the Markov chain model of the normal profile.

A hidden Markov model, another popular Markov technique, like the one shown in Fig. 3, is a statistical model where the system being modeled is assumed to be a Markov process with unknown parameters. The challenge is to determine the hidden parameters from the observable parameters. Unlike a regular Markov model, where the state transition probabilities are the only parameters and the state of the system is directly observable, in a hidden Markov model, the only visible elements are the variables of the system that are influenced by the state of the system, and the state of the system itself is hidden. A hidden Markov model's states represent some unobservable condition of the system being modeled. In each state, there is a certain probability of producing any of the observable system outputs and a separate probability indicating the likely next states. By having different output probability distributions in each of the states, and allowing the system to change states over time, the model is capable of representing non-stationary sequences.

To estimate the parameters of a hidden Markov model for modeling normal system behavior, sequences of normal events collected from normal system operation are used as training data. An expectation-maximization (EM) algorithm is used to estimate the parameters. Once a hidden Markov model has been trained, when confronted with test data, probability measures can be used as thresholds for anomaly detection. In order to use hidden Markov models for anomaly detection, three key problems need to be addressed. The first problem, also known as the *evaluation problem*, is to determine given a sequence of observations, what is the probability that the observed sequence was generated by the model. The second is the *learning problem* which involves building from the audit data a model, or a set of models, that correctly describes the observed behavior. Given a hidden Markov model and the associated observations, the third problem, also known as the *decoding problem,* involves determining the most likely set of hidden states that have led to those observations.

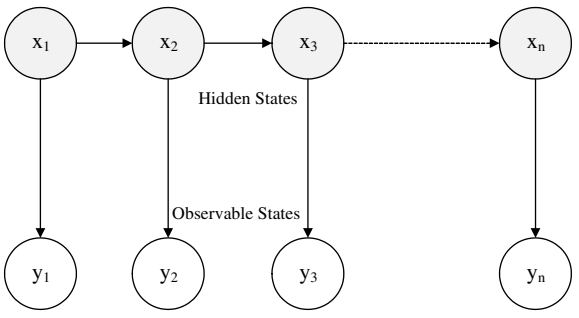


Fig. 3. Example of a hidden Markov model.

Warrender et al. [27] compare the performance of four methods viz., simple enumeration of observed sequences, comparison of relative frequencies of different sequences, a rule induction technique, and hidden Markov models at representing normal behavior accurately and recognizing intrusions in system call datasets. The authors show that while hidden Markov models outperform the other three methods, the higher performance comes at a greater computational cost. In the proposed model, the authors use an hidden Markov model with fully connected states, i.e., transitions were allowed from any state to any other state. Therefore, a process that issues S system calls will have Sstates. This implies that we will roughly have $2S^2$ values in the state transition matrix. In a computer system/network, a process typically issues a very large number of system calls. Modeling all of the processes in a computer system/network would therefore be computationally infeasible.

In another paper, Yeung et al. [40] describe the use of hidden Markov models for anomaly detection based on profiling system call sequences and shell command sequences. On training, their model computes the sample likelihood of an observed sequence using the forward or backward algorithm. A threshold on the probability, based on the minimum likelihood among all training sequences, was used to discriminate between normal and anomalous behavior. One major problem with this approach is that it lacks generalization and/or support for users who are not uniquely identified by the system under consideration.

Mahoney et al. [41–43] presented several methods that address the problem of detecting anomalies in the usage of network protocols by inspecting packet headers. The common denominator of all of them is the systematic application of learning techniques to automatically obtain profiles of normal behavior for protocols at different layers. Mahoney et al. experimented with anomaly detection over the DARPA network data [44] by range matching network packet header fields. Packet Header Anomaly Detector (PHAD) [41], LEarning Rules for Anomaly Detection (LERAD) [42] and Application Layer Anomaly Detector (ALAD) [43] use time-based models in which the probability

of an event depends on the time since it last occurred. For each attribute, they collect a set of allowed values and flag novel values as anomalous. PHAD, ALAD, and LERAD differ in the attributes that they monitor. PHAD monitors 33 attributes from the Ethernet, IP and transport layer packet headers. ALAD models incoming server TCP requests: source and destination IP addresses and ports, opening and closing TCP flags, and the list of commands (the first word on each line) in the application payload. Depending on the attribute, it builds separate models for each target host, port number (service), or host/port combination. LERAD also models TCP connections. Even though, the data set is multivariate network traffic data containing fields extracted out of the packet headers, the authors break down the multivariate problem into a set of univariate problems and sum the weighted results from range matching along each dimension. While the advantage of this approach is that it makes the technique more computationally efficient and effective at detecting network intrusions, breaking multivariate data into univariate data has significant drawbacks especially at detecting attacks. For example, in a typical SYN flood attack an indicator of the attack, is having more SYN requests than usual, but observing a lower than normal ACK rate. Because higher SYN rate or lower ACK rate alone can both happen in normal usage (when the network is busy or idle), it is the combination of higher SYN rate and lower ACK rate that signals the attack.

The one major drawback of many of the machine learning techniques, like the system call based sequence analysis approach and the hidden Markov model approach mentioned above, is that they are resource expensive. For example, an anomaly detection technique that is based on the Markov chain model is computationally expensive because it uses parametric estimation techniques based on the Bayes' algorithm for learning the normal profile of the host/network under consideration. If we consider the large amount of audit data and the relatively high frequency of events that occur in computers and networks of today, such a technique for anomaly detection is not scalable for real time operation. The highlighting features of some of the schemes surveyed in this section are presented in Table 2.

3.2.3. Data mining based anomaly detection

To eliminate the manual and ad hoc elements from the process of building an intrusion detection system, researchers are increasingly looking at using *data mining* techniques for anomaly detection [45–47]. Grossman [48] defines data mining as being "concerned with uncovering patterns, associations, changes, anomalies, and statistically significant structures and events in data". Simply put data mining is the ability to take data as input, and pull from it patterns or deviations which may not be seen easily to the naked eye. Another term sometimes used is *knowledge discovery*. Data mining can help improve the process of intrusion detection by adding a level of focus to anomaly detection. By identifying bounds for valid network activity, data mining will aid an analyst in his/her ability to distinguish attack activity from common everyday traffic on the network.

3.2.3.1. Classification-based intrusion detection. An intrusion detection system that classifies audit data as normal or anomalous based on a set of rules, patterns or other affiliated techniques can be broadly defined as a classification-based intrusion detection system. The classification process typically involves the following steps:

1. Identify class attributes and classes from training data.
2. Identify attributes for classification.
3. Learn a model using the training data.
4. Use the learned model to classify the unknown data samples.

A variety of classification techniques have been proposed in the literature. These include inductive rule generation techniques, fuzzy logic, genetic algorithms and neural networks-based techniques.

Inductive rule generation algorithms typically involve the application of a set of association rules and frequent episode patterns to classify the audit data. In this context, if a rule states that "*if event* **X** *occurs, then event* **Y** *is likely to occur*", then events **X** and **Y** can be described as sets of (variable, value)-pairs where the aim is to find the sets **X** and **Y** such that **X** "implies" **Y**. In the domain of classification, we fix **Y** and attempt to find sets of **X** which are good predictors for the right classification. While supervised classification typically only derives rules relating to a single attribute, general rule induction techniques, which are typically unsupervised in nature, derive rules relating to any or all the attributes. The advantage of using rules is that they tend to be simple and intuitive, unstructured and less rigid. As

Table 2
A summary of machine learning based anomaly detection systems

Reference	Highlighting features	Methodology
Forrest et al. [23]	It finds correlations in fixed length sequences of system calls in the UNIX operating system, and uses them to build a normal profile for anomaly detection	Sequence analysis based statistical anomaly detection
Eskin et al. [26]	It employs dynamic window sizes to improve system call modeling	System call modeling and sequence analysis
Valdes et al. [30]	It employs Bayesian inference techniques to determine if traffic bursts contain attacks. It also provides distributed attack detection capability	Bayesian networks based anomaly detection
Shyu et al. [36]	It uses principal component analysis as an outlier detection scheme to detect intrusions	Principal component analysis based anomaly detection
Yeung et al. [40]	It presents a dynamic modeling approach based on hidden Markov models and a static modeling approach based on event occurrence frequency distribution for modeling system calls. It showed that the dynamic modeling approach is better for system call datasets	Host based anomaly detection system
PHAD [41]	Examines IP headers, connections to various ports as well as packet headers of Ethernet, IP and transport layer and other transport layer packet headers	Network based anomaly detection
ALAD [43]	It detects anomalies in inbound TCP connections to well known ports on the server	Network based anomaly detection
LERAD [42]	It detects TCP stream anomalies like ALAD, but uses a learning algorithm to pick good rules from the training set, rather than using a fixed set of rules	Network based anomaly detection

the drawbacks they are difficult to maintain, and in some cases, are inadequate to represent many types of information. A number of inductive rule generation algorithms have been proposed in literature. Some of them first construct a decision tree and then extract a set of classification rules from the decision tree.[3] Other algorithms (for e.g., RIPPER [25], C4.5 [49]) directly induce rules from the data by employing a divide-and-conquer approach. A post learning stage involving either discarding (C4.5 [49]) or pruning (RIPPER [25]) some of the learnt rules is carried out to increase the classifier accuracy. RIPPER has been successfully used in a number of data mining based anomaly detection algorithms to classify incoming audit data and detect intrusions. One of the primary advantages of using RIPPER is that the generated rules are easy to use and verify. Lee et al. [45,46,50] used RIPPER to characterize sequences occurring in normal data by a smaller set of rules that capture the common elements in those sequences. During monitoring, sequences violating those rules are treated as anomalies.

Fuzzy logic techniques have been in use in the area of computer and network security since the late 1990's [51]. *Fuzzy logic* has been used for intrusion detection for two primary reasons [52]. Firstly, several quantitative parameters that are used in the context of intrusion detection, e.g., CPU usage time, connection interval, etc., can potentially be viewed as fuzzy variables. Secondly, as stated by Bridges et al. [52], the concept of security itself is fuzzy. In other words, the concept of fuzziness helps to smooth out the abrupt separation of normal behavior from abnormal behavior. That is, a given data point falling outside/inside a defined "normal interval", will be considered anomalous/normal to the same degree regardless of its distance from/within the interval. Dickerson et al. [53] developed the Fuzzy Intrusion Recognition Engine (FIRE) using fuzzy sets and fuzzy rules. FIRE uses simple data mining techniques to process the network input data and generate fuzzy sets for every observed feature. The fuzzy sets are then used to define fuzzy rules

[3] Decision trees are powerful and popular tools for classification and prediction. The attractiveness of tree-based methods is due in large part to the fact that, in contrast to neural networks, decision trees represent rules. A decision tree is a tree that has three main components: nodes, arcs, and leaves. Each node is labeled with a feature attribute which is most informative among the attributes not yet considered in the path from the root, each arc out of a node is labeled with a feature value for the node's feature and each leaf is labeled with a category or class. A decision tree can then be used to classify a data point by starting at the root of the tree and moving through it until a leaf node is reached. The leaf node would then provide the classification of the data point.

to detect individual attacks. FIRE does not establish any sort of model representing the current state of the system, but instead relies on attack specific rules for detection. Instead, FIRE creates and applies fuzzy logic rules to the audit data to classify it as normal or anomalous. Dickerson et al. found that the approach is particularly effective against port scans and probes. The primary disadvantage to this approach is the labor intensive rule generation process.

Genetic algorithms, a search technique used to find approximate solutions to optimization and search problems, have also been extensively employed in the domain of intrusion detection to differentiate normal network traffic from anomalous connections. The major advantage of genetic algorithms is their flexibility and robustness as a global search method. In addition, a genetic algorithm search converges to a solution from multiple directions and is based on probabilistic rules instead of deterministic ones. In the domain of network intrusion detection, genetic algorithms have been used in a number of ways. Some approaches [54,55] have used genetic algorithms directly to derive classification rules, while others [52,56] use genetic algorithms to select appropriate features or determine optimal parameters of related functions, while different data mining techniques are then used to acquire the rules. The earliest attempt to apply genetic algorithms to the problem of intrusion detection was done by Crosbie and Spafford [57] in 1995, when they applied multiple agent technology to detect network based anomalies. While the advantage of the approach was that it used numerous agents to monitor a variety of network based parameters, lack of intra-agent communication and a lengthy training process were some issues that were not addressed.

Neural network based intrusion detection systems have traditionally been host based systems that focus on detecting deviations in program behavior as a sign of an anomaly. In the neural network approach to intrusion detection, the neural network learns to predict the behavior of the various users and daemons in the system. The main advantage of neural networks is their tolerance to imprecise data and uncertain information and their ability to infer solutions from data without having prior knowledge of the regularities in the data. This in combination with their ability to generalize from learned data has shown made them an appropriate approach to intrusion detection. However, the neural network based solutions have several drawbacks. Firstly, they may fail to find a satisfactory solution either because of lack of sufficient data or because there is no learnable function. Secondly, neural networks can be slow and expensive to train. The lack of speed is partly because of the need to collect and analyze the training data and partly because the neural network has to manipulate the weights of the individual neurons to arrive at the correct solution. There are a few different groups advocating various approaches to using neural networks for intrusion detection. Ghosh et al. [58–60] used the feed-forward back propagation and the Elman recurrent network [61] for classifying system-call sequences. Their experimental results with the 1998 and 1999 DARPA intrusion detection evaluation dataset verified that the application of Elman networks in the domain of program-based intrusion detection provided superior results as compared to using the standard multilayer perceptron based neural network. However, training the Elman network was expensive and the number of neural networks required was large. In another paper, Ramadas et al. [62] present the Anomalous Network-Traffic Detection with Self Organizing Maps (ANDSOM). ANDSOM is the anomaly detection module for the network based intrusion detection system, called INBOUNDS, being developed at Ohio University. The ANDSOM module creates a two dimensional Self Organizing Map or SOM[4] for each network service that is being monitored. In the paper, the authors test the proposed methodology using the DNS and HTTP services. Neurons are trained with normal network traffic during the training phase to capture characteristic patterns. When real time data is fed to the trained neurons, then an anomaly is detected if the distance of the incoming traffic is more than a preset threshold.

Anomaly detection schemes also involve other data mining techniques such as support vector machines (SVM) and other types of neural network models [63,64]. Because data mining techniques are data driven and do not depend on previously observed patterns of network/system activity, some of these techniques have been very successful at detecting new kinds of attacks. However, these techniques often have a very high false positive rate. For example, as pointed out in [65], the approach

[4] A self organizing map is a method for unsupervised learning based on a grid of artificial neurons whose weights are adapted to match input vectors in a training set.

adopted by Sung and Mukkamala [66] that use a SVM technique to realize an intrusion detection system for class-specific detection is flawed because they totally ignore the relationships and dependencies between the features.

3.2.3.2. Clustering and outlier detection. Clustering is a technique for finding patterns in unlabeled data with many dimensions.[5] Clustering has attracted interest from researchers in the context of intrusion detection [67–69]. The main advantage that clustering provides is the ability to learn from and detect intrusions in the audit data, while not requiring the system administrator to provide explicit descriptions of various attack classes/types. As a result, the amount of training data that needs to be provided to the anomaly detection system is also reduced. Clustering and outlier detection are closely related. From the viewpoint of a clustering algorithm, outliers are objects not located in the clusters of a data set, and in the context of anomaly detection, they may represent intrusions/attacks. The statistics community has studied the concept of outliers quite extensively [70]. In these studies, data points are modeled using a stochastic distribution and points are determined to be outliers based on their relationship with this model. However, with increasing dimensionality, it becomes increasingly difficult to accurately estimate the multidimensional distributions of the data points [71]. Recent outlier detection algorithms [68,72,73] are based on the full dimensional distances between the points as well as the densities of local neighborhoods.

There exist at least two approaches to clustering based anomaly detection. In the first approach, the anomaly detection model is trained using unlabelled data that consists of both normal as well as attack traffic. In the second approach, the model is trained using only normal data and a profile of normal activity is created. The idea behind the first approach is that anomalous or attack data forms a small percentage of the total data. If this assumption holds, anomalies and attacks can be detected based on cluster sizes—large clusters correspond to normal data, and the rest of the data points, which are outliers, correspond to attacks.

The distances between points play an important role in clustering. The most popular distance metric is the Euclidean distance. Since each feature contributes equally to the calculation of the Euclidean distance, this distance is undesirable in many applications. This is especially true when features have very different variability or different features are measured on different scales. The effect of the features that have large scales of measurement or high variability would dominate others that have smaller scales or less variability. A distance-based approach which incorporates this measure of variability is the *Mahalanobis distance*[6] [74,75] based outlier detection scheme. In this scheme, the threshold is computed according to the most distant points from the mean of the "normal" data, and it is set to be a user defined percentage, say $m\%$, of the total number of points. In this scheme, all data points in the audit data that have distances to the mean (calculated during the training phase) greater than the threshold are detected as outliers. The Mahalanobis distance utilizes group means and variances for each variable, and the correlations and covariances between measures.

The notion of distance-based outliers was recently introduced in the study of databases [68]. According to this notion, a point, P, in a multidimensional data set is an outlier if there are less than p points from the data in the ε-neighborhood of P, where p is a user-specified constant. Ramaswamy et al. [68] described an approach that is based on computing the Euclidean distance of the kth nearest neighbor from a point O. In other words, the k-nearest neighbor algorithm classifies points by assigning them to the class that appears most frequently amongst the k nearest neighbors. Therefore, for a given point O, $d_k(O)$ denotes the Euclidean distance from the point O to its kth nearest neighbor and can be considered as the "*degree of outlierness*" of O. If one is interested in the top n outliers, this approach defines an outlier as follows: Given values for k and n, a point O is an outlier, if the distance to its kth nearest neighbor is smaller than the corresponding value for no more than $(n-1)$ other points. In other words, the top n outliers with the

[5] The number of dimensions is equivalent to the number of attributes.

[6] The basic Euclidian distance treats each variable as equally important in calculating the distance. An alternative approach is to scale the contribution of individual variables to the distance value according to the variability of each variable. This approach is illustrated by the Mahalanobis distance which is a measure of the distance between each observation in a multidimensional cloud of points and the centroid of the cloud. In other words, the Mahalanobis distance between a particular point x and the mean μ of the "normal data" is computed as: $D_M(x) = \sqrt{(x-\mu)^{\mathrm{T}}\sum^{-1}(x-\mu)}$, where $\sum$ is the covariance matrix.

maximum $D_k(O)$ values are considered as outliers. While the advantage of the k-nearest neighbors approach is that it is robust to noisy data, the approach suffers from the drawback that it is very difficult to choose an optimal value for k in practice. Several papers [76,77] have proposed using the k-nearest neighbor outlier detection algorithms for the purpose of anomaly detection.

The Minnesota Intrusion Detection System (MINDS) [78] is another network-based anomaly detection approach that utilizes data mining techniques. The MINDS anomaly detection module assigns a degree of outlierness to each data point, which is called the local outlier factor (LOF) [72]. The LOF takes into consideration the density of the neighborhood around the observation point to determine its outlierness. In this scheme, outliers are objects that tend to have high LOF values. The advantage of the LOF algorithm is its ability to detect all forms of outliers, including those that cannot be detected by the distance-based algorithms.

3.2.3.3. Association rule discovery. Association rules [79,80] are one of many data mining techniques that describe events that tend to occur together. The concept of association rules can be understood as follows: Given a database D of transactions where each transaction $T \in \mathsf{D}$ denotes a set of items in the database, an *association rule* is an implication of the form $X \Rightarrow Y$, where $X \subset \mathsf{D}$, $Y \subset \mathsf{D}$ and $X \cap Y = \emptyset$. The rule $X \Rightarrow Y$ holds in the transaction set Dwith *confidence* c if c% of transactions in X also contain Y. Two important concepts when dealing with association rules are *rule confidence* and *rule support*. The probability of rule confidence is defined as the conditional probability $P(Y \subseteq T | X \subseteq T)$ The rule $X \Rightarrow Y$ has support s in the transaction database D if s% of transactions in D contain $X \cup Y$. Association rules have been successfully used to mine audit data to find normal patterns for anomaly detection [46,50,81]. They are particularly important in the domain of anomaly detection because association rules can be used to construct a summary of anomalous connections detected by the intrusion detection system. There is evidence that suggests program executions and user activities exhibit frequent correlations among system features. These consistent behaviors can be captured in association rules.

Lee et al. [46,50] proposed an association rule-based data mining approach for anomaly detection where raw data was converted into ASCII network packet information, which in turn was converted into connection-level information. These connection level records contained connection features like service, duration, etc. Association rules were then applied to this data to create models to detect intrusions. In another paper, Barbará et al. describe Audit Data Analysis and Mining (ADAM), a real-time anomaly detection system that uses a module to classify the suspicious events into false alarms or real attacks. ADAM's training and intrusion detection phases are illustrated in Figs. 4a and 4b, respectively. ADAM was one out of seven systems tested in the 1999 DARPA evaluation [82]. It uses data mining to build a customizable profile of rules of normal behavior and then classifies attacks (by name) or declares false alarms. To discover attacks in TCPdump audit trail, ADAM uses a

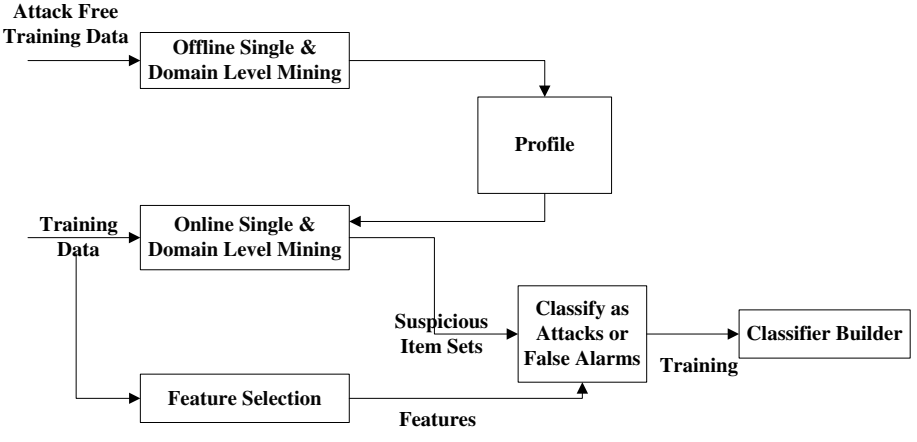


Fig. 4a. The training phase of ADAM [81].

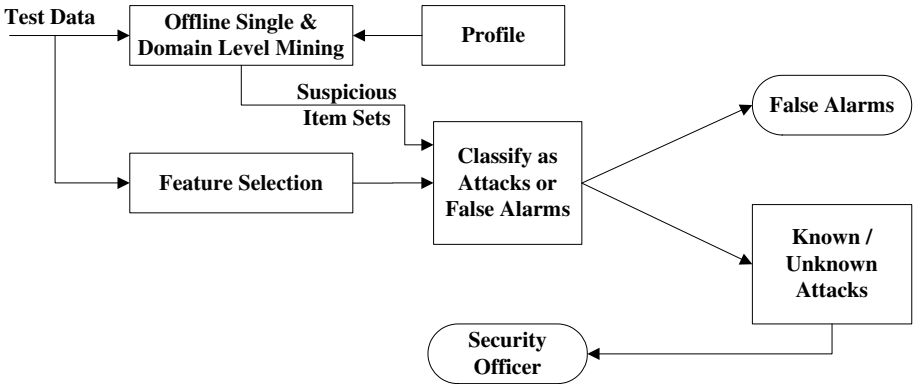


Fig. 4b. The intrusion detection phase of ADAM [81].

Table 3
A summary of data mining-based anomaly detection systems

Reference	Highlighting features	Methodology
Lee et al. [46]	It uses inductive rule generation to generate rules for important, yet infrequent events	Classification based anomaly detection
FIRE [53]	It generates fuzzy sets for every observed feature which are in turn used to define fuzzy rules to detect individual attacks	Classification based anomaly detection
ANDSOM [62]	It uses a pre-processor to summarize certain connection parameters (source and destination host and port) and then adds several values to track and classify the connection's behavior	Classification based anomaly detection
MINDS [78]	It clusters data and uses the density-based local outliers to detect intrusions	Clustering based anomaly detections
ADAM [81]	It performs anomaly detection to filter out most of the normal traffic, then it uses a classification technique to determine the exact nature of the remaining activity	Association rules and classification based anomaly detection

combination of association rules, mining and classification. During the training phase, ADAM builds a database of "normal" frequent itemsets using attack free data. Then it runs a sliding window online algorithm that finds frequent item sets in the last D connections and compares them with those stored in the normal item set repository. With the remaining item sets that have deemed suspicious, ADAM uses a classifier which has previously been trained to classify the suspicious connections as a known attack, unknown attack, or a false alarm. Association rules are used to gather necessary knowledge about the nature of the audit data. If the item set's support surpasses a threshold, then that item set is reported as suspicious. The system annotates suspicious item sets with a vector of parameters. Since the system knows where the attacks are in the training set, the corresponding suspicious item set along with their feature vectors are used to train a classifier. The trained classifier will be able to, given a suspicious item set and a vector of features, classify the item set as a known attack (and label it with the name of attack), an unknown attack, or a false alarm. The highlighting features of some of the schemes surveyed in this section are presented in Table 3.

4. Hybrid systems

It has been suggested in the literature [14,15,32,83,84] that the monitoring capability of current intrusion detection systems can be improved by taking a hybrid approach that consists of both anomaly as well as signature detection strategies. In such a hybrid system, the anomaly detection technique aids in the detection of new or unknown attacks while the signature detection technique detects known attacks. The signature detection technique will also be able to detect attacks launched by a patient attacker who attempts to change the behavior

patterns with the objective of retraining the anomaly detection module so that it will accept attack behavior as normal. Tombini et al. [83] used an approach wherein the anomaly detection technique is used to produce a list of suspicious items. The classifier module which uses a signature detection technique then classified the suspicious items into false alarms, attacks, and unknown attacks. This approach works on the premise that the anomaly detection component would have a high detection rate, since missed intrusions cannot be detected by the follow-up signature detection component. In addition, it also assumed that the signature detection component will be able to identify false alarms. While the hybrid system can still miss certain types of attacks, its reduced false alarm rate increases the likelihood of examining most of the alerts.

EMERALD [32] was developed in the late 1990's at SRI. It is an extension of the seminal work done in [8,13–15]. EMERALD is a hierarchical intrusion detection system that monitors systems at a variety of levels viz. individual host machines, domains and enterprises to form an analysis hierarchy. EMERALD uses a subscription-based communication scheme both within and between monitors. However, inter monitor subscription methodology is hierarchical and therefore limits the access to events and/or results from the layer immediately below. The system has a built-in feedback system that enables the higher layers to request more information about particular anomalies from the lower layers. To achieve a high rate of detection, the architects of EMERALD employed an ensemble of techniques like statistical analysis engines and expert systems. The single most defining feature of EMERALD is its ability to analyze system-wide, domain-wide and enterprise-wide attacks like Internet worms, DDoS attacks, etc. at the top level.

In another paper [84], Zhang et al. [85] employed the random forests algorithm in the signature detection module to detect known intrusions. Thereafter, the outlier detection provided by the random forests algorithm is utilized to detect unknown intrusions. Approaches that use signature detection and anomaly detection in parallel have also been proposed. In such systems, two sets of reports of possible intrusive activity are produced and a correlation component analyzes both sets to detect intrusions. An example of such a system is NIDES [15,16].

Lee et al. [45,86] extended the work done by them in [50] and proposed a hybrid detection scheme that utilized the Common Intrusion Detection Framework (CIDF) to automatically get audit data, build models, and distribute signatures for novel attacks to ensure that the time required to detect them is reduced. The advantage of using CIDF was that it enabled different intrusion detection and response components to interoperate and share the information and resources in a distributed environment.

Although it is true that combining multiple intrusion detection technologies into a single system can theoretically produce a much stronger intrusion detection system, the resulting hybrid systems are not always better. Different intrusion detection technologies examine system and/or network traffic and look for intrusive activity in different ways. Therefore, the major challenge to building an operational hybrid intrusion detection system is getting these different technologies to interoperate effectively and efficiently.

5. The road ahead: open challenges

In the last twenty years, intrusion detection systems have slowly evolved from host- and operating system-specific applications to distributed systems that involve a wide array of operating systems. The challenges that lie ahead for the next generation of intrusion detection systems and, more specifically, for anomaly detection systems are many. First and foremost, traditional intrusion detection systems have not adapted adequately to new networking paradigms like wireless and mobile networks nor have they scaled to meet the requirements posed by high-speed (gigabit and terabit) networks (an analysis of intrusion detection techniques for high-speed networks can be found in [87,88]). Factors like noise in the audit data, constantly changing traffic profiles, and the large amount of network traffic make it difficult to build a normal traffic profile of a network for the purpose of intrusion detection. The implication is that, short of some fundamental re-design, today's intrusion detection approaches will not be able to adequately protect tomorrow's networks against intrusions and attacks. Therefore, the design methodology of intrusion detection systems needs to closely follow the changes in system and networking technologies.

A perennial problem that prevents widespread deployment of intrusion detection systems is their inability to suppress false alarms. It has been shown in lab testing [89] that state of the art intrusion detection systems often crash under the burden of

the false alarms that they generate. When actual attacks do occur, either they are missed completely by the intrusion detection system or the traces left by the intruder and/or the traces indicating the actual attack are lost in amongst the large number of false alarms in the event logs. To make matters worse, Axelsson [90], showed that for an intrusion detection system to be effective the number of false alarms have to be really low. In his paper, Axelsson suggested that 1 false alarm in 100,000 events was the minimum requirement for an intrusion detection system to be effective. Therefore, the primary and probably the most important challenge that needs to be met is the development of effective strategies to reduce the high rate of false alarms.

Over the years, numerous techniques, models, and full-fledged intrusion detection systems have been proposed and built in the commercial and research sectors. However, there is no globally acceptable standard/metric for evaluating an intrusion detection system. Although the *Receiver Operating Characteristic* (ROC) curve has been widely used to evaluate the accuracy of intrusion detection systems and analyze the tradeoff between the false positives rate and the detection rate, evaluations based on the ROC curve are often misleading and/or incomplete [91,92]. Recently, several methods have been proposed to address this issue [90–92]. However, most, if not all, of the proposed solutions rely on parameters values (such as the cost associated with each false alarm or missed attack instance) that are difficult to obtain and are subjective to a particular network or system. As a result, such metrics may lack the objectivity required to conduct a fair evaluation of a given system. Therefore, one of the open challenges is the development of a general systematic methodology and/or a set of metrics that can be used to fairly evaluate intrusion detection systems.

There is a lack of a standard evaluation dataset that can simulate realistic network environments. While the 1998 and 1999 intrusion detection evaluations from DARPA/MIT Lincoln labs have been used to evaluate a large number of intrusion detection systems, the methodology used to generate the data as well as the data itself have been shown to be inappropriate for simulating actual network environments [93]. Therefore, there is a critical need to build a more appropriate evaluation dataset. The methodology for generating the evaluation dataset should not only simulate realistic network conditions but also be able to generate datasets that have normal traffic interlaced with anomalous traffic.

An important aspect of intrusion detection, which has also been proposed as an evaluation metric [94–96], is the ability of an intrusion detection system to defend itself from attacks. Attacks on intrusion detection systems can take several forms. As a specific example, consider an attacker sending a large volume of non-attack packets that are specially crafted to trigger many alarms within an intrusion detection system, thereby overwhelming the human operator with false positives or crashing the alert processing or display tools. Axelsson [9], in his 1998 survey of intrusion detection systems, found that a majority of the available intrusion detection systems at that time performed very poorly when it came to defending themselves from attacks. Since then, the ability of intrusion detection systems at defending themselves from attacks has improved only marginally.

Another problem that still poses a major challenge is trying to define what is "*normal*" in a network. As mentioned in [97], there is a need for the discovery of "*attack invariant*" features. An attack invariant would be a feature of the network/system that can always be verified except in the presence of an attack. Examples include traffic volume, number of connections to non standard ports, etc. In order to define such attack invariants, it is imperative that a better understanding of the nature of anomalies in the network and/or host is gained.

Traditionally, encryption has been a preferred methodology for securing data and preventing malicious users from getting access to privileged/private information. However, the widespread use of encryption implies that network administrators have a limited view of the network as traditional intrusion detection systems do not have the ability to decrypt the encrypted packets that they intercept. When an intrusion detection system intercepts an encrypted packet, it typically discards it, which results in greatly limiting the amount of traffic that it is capable of inspecting. Therefore, the challenge for security researchers is the development of security mechanisms that provide data security while not limiting the functions of intrusion detection systems.

An increasing problem in today's corporate networks is the threats posed by insiders, viz., disgruntled employees. In a survey [98] conducted by the United States Secret Service and CERT of Carnegie Mellon University, 71% of respondents out of 500 participants reported that 29% of the attacks that they experienced were caused by insiders. Respon-

dents identified current or former employees and contractors as the second greatest cyber security threat, preceded only by hackers. Configuring an intrusion detection system to detect internal attacks is very difficult. The greatest challenge lies in creating a good rule set for detecting "internal" attacks or anomalies. Different network users require different degrees of access to different services, servers, and systems for their work, thus making it extremely difficult to define and create user- or system-specific usage profiles. Although there is some existing work in this area (e.g., [99,100]), more research is needed to find practical solutions.

References

[1] E. Millard, Internet attacks increase in number, severity, in: Top Tech News, 2005.

[2] J. Phillips, Hackers' invasion of OU data raises blizzard of questions, in: The Athens News Athens, OH, 2006.

[3] C. Staff, Hackers: companies encounter rise of cyber extortion, vol. 2006, Computer Crime Research Center, 2005.

[4] M. Williams, Immense network assault takes down Yahoo, in: CNN.COM, 2000.

[5] C.S. Institute, F.B.o. Investigation, in: Proceedings of the 10th Annual Computer Crime and Security Survey 10, 2005, pp. 1–23.

[6] S. Axelsson, Intrusion Detection Systems: A Survey and Taxonomy, Chalmers University, Technical Report 99-15, March 2000.

[7] J.P. Anderson, Computer security threat monitoring and surveillance, James P Anderson Co., Fort, Washington, PA, USA, Technical Report 98-17, April 1980.

[8] D.E. Denning, An intrusion-detection model, IEEE Transactions in Software Engineering 13 (1987) 222–232.

[9] S. Axelsson, Research in intrusion-detection systems: a survey, Department of Computer Engineering, Chalmers University of Technology, Goteborg, Sweden, Technical Report 98-17, December 1998.

[10] S. Kumar, E.H. Spafford, An application of pattern matching in intrusion detection, The COAST Project, Department of Computer Sciences, Purdue University, West Lafayette, IN, USA, Technical Report CSD-TR-94-013, June 17, 1994.

[11] S.E. Smaha, Haystack: An intrusion detection system, in: Proceedings of the IEEE Fourth Aerospace Computer Security Applications Conference, Orlando, FL, 1988, pp. 37–44.

[12] D. Anderson, T. Frivold, A. Valdes, Next-generation Intrusion Detection Expert System (NIDES): A Summary, Computer Science Laboratory, SRI International, Menlo Park, CA 94025, Technical Report SRI-CSL-95-07, May 1995.

[13] D.E. Denning, P.G. Neumann, Requirements and Model for IDES—A Real-time Intrusion Detection System, Computer Science Laboratory, SRI International, Menlo Park, CA 94025-3493, Technical Report # 83F83-01-00, 1985.

[14] T.F. Lunt, A. Tamaru, F. Gilham, R. Jagannathm, C. Jalali, P.G. Neumann, H.S. Javitz, A. Valdes, T.D. Garvey, A Real-time Intrusion Detection Expert System (IDES), Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Final Technical Report, February 1992.

[15] D. Anderson, T. Frivold, A. Tamaru, A. Valdes, Next-generation intrusion detection expert system (NIDES), Software Users Manual, Beta-Update release, Computer Science Laboratory, SRI International, Menlo Park, CA, USA, Technical Report SRI-CSL-95-0, May 1994.

[16] D. Anderson, T.F. Lunt, H. Javitz, A. Tamaru, A. Valdes, Detecting Unusual Program Behavior Using the Statistical Component of the Next-generation Intrusion Detection Expert System (NIDES), Computer Science Laboratory, SRI International, Menlo Park, CA, USA SRI-CSL-95-06, May 1995.

[17] S. Staniford, J.A. Hoagland, J.M. McAlerney, Practical automated detection of stealthy portscans, Journal of Computer Security 10 (2002) 105–136.

[18] M. Roesch, Snort – lightweight intrusion detection for networks, in: Proceedings of the 13th USENIX Conference on System Administration Seattle, Washington, 1999, pp. 229–238.

[19] N. Ye, S.M. Emran, Q. Chen, S. Vilbert, Multivariate statistical analysis of audit trails for host-based intrusion detection, IEEE Transactions on Computers 51 (2002) 810–820.

[20] C. Krügel, T. Toth, E. Kirda, Service specific anomaly detection for network intrusion detection, in: Proceedings of the 2002 ACM symposium on Applied computing Madrid, Spain 2002, pp. 201–208.

[21] R.A. Maxion, F.E. Feather, A case study of Ethernet anomalies in a distributed computing environment, IEEE Transactions on Reliability 39 (1990) 433–443.

[22] W. Lee, D. Xiang, Information theoretic measures for anomaly detection, in: Proceedings of the 2001 IEEE Symposium on Security and Privacy, Washington, DC, USA, 2001, pp. 130–143.

[23] S. Forrest, S.A. Hofmeyr, A. Somayaji, T.A. Longstaff, A sense of self for unix processes, in: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, USA, 1996, pp. 120–128.

[24] S.A. Hofmeyr, S. Forrest, A. Somayaji, Intrusion detection using sequences of system calls, Journal of Computer Security 6 (1998) 151–180.

[25] W.W. Cohen, Fast effective rule induction, in: Proceedings of the 12th International Conference on Machine Learning, Tahoe City, CA, 1995, pp. 115–123.

[26] E. Eskin, S.J. Stolfo, W. Lee, Modeling system calls for intrusion detection with dynamic window sizes, in: Proceedings of the DARPA Information Survivability Conference & Exposition II, Anaheim, CA 2001, pp. 165–175.

[27] C. Warrender, S. Forrest, B. Pearlmutter, Detecting intrusions using system calls: alternative data models, in: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, USA, 1999, pp. 133–145.

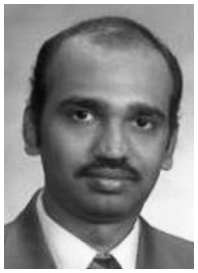
[28] D. Heckerman, A Tutorial on Learning With Bayesian Networks, Microsoft Research, Technical Report MSR-TR-95-06, March 1995.

[29] C. Kruegel, D. Mutz, W. Robertson, F. Valeur, Bayesian event classification for intrusion detection, in: Proceedings

of the 19th Annual Computer Security Applications Conference, Las Vegas, NV, 2003.
[30] A. Valdes, K. Skinner, Adaptive model-based monitoring for cyber attack detection, in: Recent Advances in Intrusion Detection Toulouse, France, 2000, pp. 80–92.
[31] N. Ye, M. Xu, S.M. Emran, Probabilistic networks with undirected links for anomaly detection, in: Proceedings of the IEEE Systems, Man, and Cybernetics Information Assurance and Security Workshop, West Point, NY, 2000.
[32] P.A. Porras, P.G. Neumann, EMERALD: event monitoring enabling responses to anomalous live disturbances, in: Proceedings of the 20th NIST-NCSC National Information Systems Security Conference, Baltimore, MD, USA, 1997, pp. 353–365.
[33] R.A. Calvo, M. Partridge, M.A. Jabri, A comparative study of principal component analysis techniques, in: Proceedings of the Ninth Australian Conference on Neural Networks, Brisbane, Qld, Australia, 1998.
[34] H. Hotelling, Analysis of a complex of statistical variables into principal components, Journal of Educational Psychology 24 (1993) 417–441, 498–520.
[35] W. Wang, R. Battiti, Identifying intrusions in computer networks with principal component analysis, in: The First International Conference on Availability, Reliability and Security, Vienna, Austria, 2006, pp. 270–279.
[36] M.-L. Shyu, S.-C. Chen, K. Sarinnapakorn, L. Chang, A novel anomaly detection scheme based on principal component classifier, in: Proceedings of the IEEE Foundations and New Directions of Data Mining Workshop, Melbourne, FL, USA, 2003, pp. 172–179.
[37] Y. Bouzida, F.e.e. Cuppens, N. Cuppens-Boulahia, S. Gombault, Efficient intrusion detection using principal component analysis, in: Proceedings of the 3ème Conférence sur la Sécurité et Architectures Réseaux (SAR), Orlando, FL, USA, 2004.
[38] W. Wang, X. Guan, X. Zhang, A novel intrusion detection method based on principle component analysis in computer security, in: Proceedings of the International Symposium on Neural Networks, Dalian, China, 2004, pp. 657–662.
[39] N. Ye, Y.Z.C.M. Borror, Robustness of the Markov-chain model for cyber-attack detection, IEEE Transactions on Reliability 53 (2004) 116–123.
[40] D.-Y. Yeung, Y. Ding, Host-based intrusion detection using dynamic and static behavioral models, Pattern Recognition 36 (2003) 229–243.
[41] M.V. Mahoney, P.K. Chan, PHAD: Packet Header Anomaly Detection for Identifying Hostile Network Traffic Department of Computer Sciences, Florida Institute of Technology, Melbourne, FL, USA, Technical Report CS-2001-4, April 2001.
[42] M.V. Mahoney, P.K. Chan, Learning Models of Network Traffic for Detecting Novel Attacks Computer Science Department, Florida Institute of Technology CS-2002-8, August 2002.
[43] M.V. Mahoney, P.K. Chan, Learning nonstationary models of normal network traffic for detecting novel attacks, in: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Canada, 2002, pp. 376–385.
[44] R. Lippmann, J.W. Haines, D.J. Fried, J. Korba, K. Das, The 1999 DARPA off-line intrusion detection evaluation, Computer Networks 34 (2000) 579–595.
[45] W. Lee, R.A. Nimbalkar, K.K. Yee, S.B. Patil, P.H. Desai, T.T. Tran, S.J. Stolfo, A data mining and CIDF based approach for detecting novel and distributed intrusions, in: Proceedings of the 3rd International Workshop on Recent Advances in Intrusion Detection (RAID 2000), Toulouse, France, 2000, pp. 49–65.
[46] W. Lee, S.J. Stolfo, Data mining approaches for intrusion detection, in: Proceedings of the 7th USENIX Security Symposium (SECURITY-98), Berkeley, CA, USA, 1998, pp. 79–94.
[47] W. Lee, S.J. Stolfo, K.W. Mok, Adaptive intrusion detection: a data mining approach, Artificial Intelligence Review 14 (2000) 533–567.
[48] R. Grossman, Data Mining: Challenges and Opportunities for Data Mining During the Next Decade, 1997.
[49] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufman, Los Altos, CA, 1993.
[50] W. Lee, S.J. Stolfo, K.W. Mok, A data mining framework for building intrusion detection models, in: Proceedings of the IEEE Symposium on Security and Privacy, Oakland, CA, 1999, pp. 120–132.
[51] H.H. Hosmer, Security is fuzzy!: applying the fuzzy logic paradigm to the multipolicy paradigm, in: Proceedings of the 1992–1993 Workshop on New Security Paradigms Little Compton, RI, United States, 1993.
[52] S.M. Bridges, R.B. Vaughn, Fuzzy data mining and genetic algorithms applied to intrusion detection, in: Proceedings of the National Information Systems Security Conference, Baltimore, MD, 2000.
[53] J.E. Dickerson, J.A. Dickerson, Fuzzy network profiling for intrusion detection, in: Proceedings of the 19th International Conference of the North American Fuzzy Information Processing Society (NAFIPS), Atlanta, GA, 2000, pp. 301–306.
[54] W. Li, Using Genetic Algorithm for Network Intrusion Detection, C.S.G. Department of Energy, 2004, pp. 1–8.
[55] M.M. Pillai, J.H.P. Eloff, H.S. Venter, An approach to implement a network intrusion detection system using genetic algorithms, in: Proceedings of the 2004 Annual Research Conference of the South African Institute of Computer Scientists and Information Technologists on IT Research in Developing Countries, Stellenbosch, Western Cape, South Africa, 2004, pp. 221–228.
[56] J. Gomez, D. Dasgupta, Evolving fuzzy classifiers for intrusion detection, in: IEEE Workshop on Information Assurance, United States Military Academy, NY, 2001.
[57] M. Crosbie, G. Spafford, Applying genetic programming to intrusion detection, in: Working Notes for the AAAI Symposium on Genetic Programming, Cambridge, MA, 1995, pp. 1–8.
[58] A.K. Ghosh, C. Michael, M. Schatz, A real-time intrusion detection system based on learning program behavior, in: Proceedings of the Third International Workshop on Recent Advances in Intrusion Detection Toulouse, France, 2000, pp. 93–109.
[59] A.K. Ghosh, A. Schwartzbard, A study in using neural networks for anomaly and misuse detection, in: Proceedings of the Eighth USENIX Security Symposium, Washington, DC, 1999, pp. 141–151.
[60] A.K. Ghosh, A. Schwartzbart, M. Schatz, Learning program behavior profiles for intrusion detection, in: Proceedings of the 1st USENIX Workshop on Intrusion

Detection and Network Monitoring, Santa Clara, CA, USA, 1999.
[61] J.L. Elman, Finding structure in time, Cognitive Science 14 (1990) 179–211.
[62] M. Ramadas, S.O.B. Tjaden, Detecting anomalous network traffic with self-organizing maps, in: Proceedings of the 6th International Symposium on Recent Advances in Intrusion Detection, Pittsburgh, PA, USA, 2003, pp. 36–54.
[63] W. Lee, S.J. Stolfo, P.K. Chan, E. Eskin, W. Fan, M. Miller, S. Hershkop, J. Zhang, Real time data mining-based intrusion detection, in: Proceedings of the Second DARPA Information Survivability Conference and Exposition, Anaheim, CA, 2001, pp. 85–100.
[64] K.M.C. Tan, R.A. Maxion, Determining the operational limits of an anomaly-based intrusion detector, IEEE Journal on Selected Areas in Communication 2 (2003) 96–110.
[65] S.T. Sarasamma, Q.A. Zhu, J. Huff, Hierarchical Kohonenen net for anomaly detection in network security, IEEE Transactions on Systems, Man and Cybernetics—PART B: Cybernetics 35 (2005) 302–312.
[66] A.H. Sung, S. Mukkamala, Identifying important features for intrusion detection using support vector machines and neural networks, in: Proceedings of the 2003 Symposium on Applications and the Internet 2003, pp. 209–216.
[67] L. Portnoy, E. Eskin, S.J. Stolfo, Intrusion detection with unlabeled data using clustering, in: Proceedings of the ACM Workshop on Data Mining Applied to Security, Philadelphia, PA, 2001.
[68] S. Ramaswamy, R. Rastogi, K. Shim, Efficient algorithms for mining outliers from large data sets, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Dallas, TX, USA, 2000, pp. 427–438.
[69] K. Sequeira, M. Zaki, ADMIT: Anomaly-based data mining for intrusions, in: Proceedings of the 8th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Edmonton, Alberta, Canada, 2002, pp. 386–395.
[70] V. Barnett, T. Lewis, Outliers in Statistical Data, Wiley, 1994.
[71] C.C. Aggarwal, P.S. Yu, Outlier detection for high dimensional data, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, 2001, pp. 37–46.
[72] M. Breunig, H.-P. Kriegel, R.T. Ng, J. Sander, LOF: identifying density-based local outliers, in: Proceedings of the ACM SIGMOD International Conference on Management of Data, Dallas, TX, 2000, pp. 93–104.
[73] E.M. Knorr, R.T. Ng, Algorithms for mining distance-based outliers in large datasets, in: Proceedings of the 24th International Conference on Very Large Data Bases, New York, NY, USA, 1998, pp. 392–403.
[74] P.C. Mahalanobis, On tests and measures of groups divergence, Journal of the Asiatic Society of Bengal 26 (1930) 541.
[75] Wikipedia, Mahalanobis Distance, vol. 2006, 2006.
[76] V. Hautamaki, I. Karkkainen, P. Franti, Outlier detection using k-nearest neighbour graph, in: Proceedings of the 17th International Conference on Pattern Recognition Los Alamitos, CA, USA, 2004, pp. 430–433.
[77] Y. Liao, V.R. Vemuri, Use of K-nearest neighbor classifier for intrusion detection, Computers & Security 21 (2002) 439–448.
[78] L. Ertöz, E. Eilertson, A. Lazarevic, P.-N. Tan, V. Kumar, J. Srivastava, P. Dokas, The MINDS - Minnesota intrusion detection system, in: Next Generation Data Mining, MIT Press, Boston, 2004.
[79] R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: Proceedings of the ACM SIGMOD Conference on Management of Data, Washington, DC, 1993, pp. 207–216.
[80] J. Hipp, U. Güntzer, G. Nakhaeizadeh, Algorithms for association rule mining - a general survey and comparison, in: Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Boston, MA, USA, 2000, pp. 58–64.
[81] D. Barbará, J. Couto, S. Jajodia, N. Wu, ADAM: a testbed for exploring the use of data mining in intrusion detection, ACM SIGMOD Record: SPECIAL ISSUE: Special section on data mining for intrusion detection and threat analysis 30 (2001) 15–24.
[82] R. Lippmann, J.W. Haines, D.J. Fried, J. Korba, K. Das, The 1999 DARPA off-line intrusion detection evaluation, Computer Networks: The International Journal of Computer and Telecommunications Networking 34 (2000) 579–595.
[83] E. Tombini, H. Debar, L. Mé, M. Ducassé, A serial combination of anomaly and misuse IDSes applied to HTTP traffic, in: Proceedings of the 20th Annual Computer Security Applications Conference, Tucson, AZ, USA, 2004.
[84] J. Zhang, M. Zulkernine, A hybrid network intrusion detection technique using random forests, in: Proceedings of the First International Conference on Availability, Reliability and Security, Vienna University of Technology, 2006, pp. 262–269.
[85] L. Breiman, Random forests, Machine Learning 45 (2001) 5–32.
[86] W.L.S.J. Stolfo, P.K. Chan, E. Eskin, W. Fan, M. Miller, S. Hershkop, J. Zhang, Real time data mining-based intrusion detection, in: Proceedings of the Second DARPA Information Survivability Conference and Exposition, Anaheim, CA, USA, 2001, pp. 85–100.
[87] C. Kruegel, F. Valeur, G. Vigna, R. Kemmerer, Stateful intrusion detection for high-speed networks, in: Proceedings of the IEEE Symposium on Security and Privacy, 2002, pp. 285– 294.
[88] A. Patcha, J.-M. Park, Detecting denial-of-service attacks with incomplete audit data, in: Proceedings of the 14th International Conference on Computer Communications and Networks, San Diego, CA, USA, 2005, pp. 263–268.
[89] D. Newman, J. Snyder, R. Thayer, Crying wolf: False alarms hide attacks, in: Network World: Network World, 2002.
[90] S. Axelsson, The base-rate fallacy and its implications for the difficulty of intrusion detection, ACM Transactions on Information and System Security 3 (2000) 186–205.
[91] JE. Gaffney, JW. Ulvila, Evaluation of intrusion detectors: a decision theory approach, in: Proceedings of the 2001 IEEE Symposium on Security and Privacy, Oakland, CA, USA, 2001, pp. 50–61.
[92] S.J. Stolfo, W. Fan, W. Lee, Cost-based modeling for fraud and intrusion detection: results from the JAM Project, in: Proceedings of the DARPA Information Survivability Conference & Exposition, 2000, pp. 130–144.

[93] J. McHugh, Testing Intrusion detection systems: a critique of the 1998 and 1999 DARPA intrusion detection, ACM Transactions on Information and System Security 3 (2000) 262–294.
[94] T. Ptacek, T. Newsham, Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection, Secure Networks Inc, 1998.
[95] U. Shankar, V. Paxson, Active mapping: resisting NIDS evasion without altering traffic, in: Proceedings of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, 2003.
[96] K.M.C. Tan, K.S. Killourhy, R.A. Maxion, Undermining an anomaly-based intrusion detection system using common exploits, in: Proceedings of the Fifth International Symposium on Recent Advances in Intrusion Detection, Zurich, Switzerland, 2002, pp. 54–73.
[97] J.M. Estevez-Tapiador, P. Garcia-Teodoro, J.E. Diaz-Verdejo, Anomaly detection methods in wired networks: a survey and taxonomy, Computer Communications 27 (2004) 1569–1584.
[98] M. Keeney, E. Kowalski, D. Cappelli, A. Moore, T. Shimeall, S. Rogers, Insider threat study: computer system sabotage in critical infrastructure sectors, U.S.S. Service and C.M.U. Software Engineering Institute, Software Engineering Institute, Carnegie Mellon University, 2005, pp. 1–45.
[99] A. Liu, C. Martin, T. Hetherington, S. Matzner, A comparison of system call feature representations for insider threat detection, in: Proceedings of the 6th Annual IEEE Systems, Man and Cybernetics (SMC) Information Assurance Workshop, West Point, NY, 2005 pp. 340–347.
[100] J.S. Park, J. Giordano, Role-based profile analysis for scalable and accurate insider-anomaly detection, in: Proceedings of the 25th IEEE International Performance, Computing, and Communications Conference, Phoenix, AZ, 2006, pp. 463–470.



Animesh Patcha is a doctoral candidate in the Bradley Department of Electrical and Computer Engineering at Virginia Tech. since August 2002. His area of research is computer and network security in wired and wireless networks. Currently, he is working on stochastic intrusion detection techniques under the expert guidance of Dr. Jung-Min Park in the Laboratory for Advanced Research in Information Assurance and Security.

Prior to coming to Virginia Tech, he received his M.S. degree in Computer Engineering from Illinois Institute of Technology, Chicago, IL in May 2002 and his B.E. in Electrical and Electronics Engineering from Birla Institute of Technology Mesra, Ranchi, India in December 1998 respectively. From January 1999 to December 2000, he was a software engineer at Zensar Technologies in Pune, India. He is currently a student member of the IEEE, SIAM, ASEE and a global member of the Internet Society.

Jung-Min Park received the B.S. and M.S. degrees both in electronic engineering from Yonsei University, Seoul, South Korea, in 1995 and 1997, respectively; and the Ph.D. degree in electrical and computer engineering from Purdue University, West Lafayette, IN, in 2003.

He is currently an Assistant Professor in the Department of Electrical and Computer Engineering at Virginia Polytechnic Institute and State University (Virginia Tech), Blacksburg, VA. From 1997 to 1998, he worked as a cellular systems engineer at Motorola Korea Inc. His current interests are in network security, applied cryptography, and cognitive radio networks. More details about his research interests and publications can be found at http://www.ece.vt.edu/faculty/park.html.

He is a member of the Institute of Electrical and Electronics Engineers (IEEE), Association for Computing Machinery (ACM), and the Korean-American Scientists and Engineers Association (KSEA). He was a recipient of a 1998 AT& T Leadership Award.