Hybrid Intrusion Detection with Weighted Signature Generation over Anomalous Internet Episodes

Kai Hwang, *Fellow*, *IEEE*, Min Cai, *Member*, *IEEE*, Ying Chen, *Student Member*, *IEEE*, and Min Qin

Abstract—This paper reports the design principles and evaluation results of a new experimental *hybrid intrusion detection system* (HIDS). This hybrid system combines the advantages of low false-positive rate of signature-based *intrusion detection system* (IDS) and the ability of *anomaly detection system* (ADS) to detect novel unknown attacks. By mining anomalous traffic episodes from Internet connections, we build an ADS that detects anomalies beyond the capabilities of signature-based SNORT or Bro systems. A weighted signature generation scheme is developed to integrate ADS with SNORT by extracting signatures from anomalies detected. HIDS extracts signatures from the output of ADS and adds them into the SNORT signature database for fast and accurate intrusion detection. By testing our HIDS scheme over real-life Internet trace data mixed with 10 days of Massachusetts Institute of Technology/Lincoln Laboratory (MIT/LL) attack data set, our experimental results show a 60 percent detection rate of the HIDS, compared with 30 percent and 22 percent in using the SNORT and Bro systems, respectively. This sharp increase in detection rate is obtained with less than 3 percent false alarms. The signatures generated by ADS upgrade the SNORT performance by 33 percent. The HIDS approach proves the vitality of detecting intrusions and anomalies, simultaneously, by automated data mining and signature generation over Internet connection episodes.

Index Terms—Network security, intrusion detection systems, anomaly detection, signature generation, SNORT and Bro systems, false alarms, Internet episodes, traffic data mining.

1 Introduction

Intrusions and anomalies are two different kinds of attacks in an open network environment. An intrusion takes place when an unauthorized access of a host computer system is attempted. An anomaly is observed at the network connection level. Both attack types may compromise valuable hosts, disclose sensitive data, deny services to legitimate users, and pull down network-based computing resources [13], [14]. The *intrusion detection system* (IDS) offers intelligent protection of networked computers or distributed resources much better than using fixed-rule firewalls. Existing IDSs are built with either signature-based or anomaly-based systems [7], [18]. Signature matching is based on a misuse model, whereas anomaly detection is based on a normal use model.

The design philosophies of these two models are quite different, and they were rarely mixed up in existing IDS products from the security industry. A *signature-based* IDS like SNORT [4], [31] employs a priori knowledge of attack signatures. The signatures are manually constructed by security experts analyzing previous attacks. The collected signatures are used to match with incoming traffic to detect intrusions. These are conventional systems that detect known attacks with low false alarms. However, the signature-based IDS cannot detect unknown attacks without any precollected signatures or lack of attack classifiers [33]. Furthermore, signature matching performs well only for single-connection attacks. With the sophistication of attackers, more attacks involve multiple connections. This limits the detection range by signature matching.

On the other hand, an *anomaly-based* system uses a different philosophy. It treats any network connection violating the normal profile as an anomaly [8], [10], [16], [17]. A network anomaly is revealed if the incoming traffic pattern deviates from the normal profiles significantly. Through a data mining approach, anomaly detection discovers temporal characteristics of network traffic. This system can detect unknown attacks and handles multi-connection attacks well. However, anomaly detection may result in higher false alarms. The newly proposed HIDS is designed to solve these problems with much enhanced performance.

In this paper, we present a new *hybrid intrusion detection system* (HIDS). This system combines the positive features of both intrusion detection models to achieve higher detection accuracy, lower false alarms, and, thus, a raised level of cybertrust. Our HIDS is network-based, which should not be confused with the *host-based* IDS with the same abbreviation by other authors. An adaptive base-support threshold is applied on selected axis attributes in mining the Internet episode rules. The episode rules are used to build the HIDS, which detects not only known intrusive attacks but also anomalous connection sequences.

• *The authors are with the Internet and Grid Computing Laboratory, USC Viterbi School of Engineering, University of Southern California, 3740 McClintock Ave., EEB 212, Los Angeles, CA 90089.*
E-mail: {kaihwang, mincai, chen2, mqin}@usc.edu.

Manuscript received 24 Nov. 2004; revised 4 Dec. 2006; accepted 11 Dec. 2006; published online 2 Feb. 2007.
For information on obtaining reprints of this article, please send e-mail to: tdsc@computer.org, and reference IEEECS Log Number TDSC-0168-1104.
Digital Object Identifier no. XXX

The rest of the paper is organized as follows: Section 2 reviews related works and distinguishes the new approach from previous solutions. Section 3 introduces traffic data mining and describes the HIDS architecture. We present Internet episode rules and pruning techniques in Section 4. The weighted signature generation is specified in Section 5. We describe the HIDS simulator and attack data sets in Section 6. Experimental performance results are reported in Section 7. Finally, in Section 8, we summarize the contributions and comment on further research needed.

2 Related Works and Our Approach

In the past, data mining techniques such as using *association rules* were suggested to build IDS [1], [7], [17], [18], [19], [25], [28]. Lazarevic et al. [17] have distinguished the differences between *single-connection* and *multiconnection* attacks. Both signature-based and anomaly-based IDSs are sensitive to the attack characteristics, system training history, services provided, and underlying network conditions [21], [24]. Data mining techniques are also used to build classification models from labeled attacks [18], [19], [25].

SNORT [4], [31] and Bro [26] are two widely used IDSs that are based on the misuse model. Other attempts to solve the intrusion detection and response problem can be found in [2], [6], [20], [29]. Intrusion detection must be designed to monitor the connection features at the network, transport, and application layers [3], [12]. The MIT/LL IDS evaluation data set and reported IDS performance results were analyzed in [20], [21], [23]. We use this attack data set with mixed background traffic to test the effectiveness of HIDS.

The concept of *frequent episode rules* (FERs) was first proposed by Mannila and Toivonen [22]. Subsequently, Lee et al. [18], [19] suggested a framework to specify FERs for anomaly detection against normal traffic profiles. They developed a levelwise data mining algorithm for building ADS. Fan et al. [10] extended Lee et al.'s work to discover accurate boundaries between known attacks and unknown anomalies.

Qin and Hwang [28] refined the rule formulation procedure with an adaptive *base-support algorithm* to mine normal traffic records. Different axis attribute values apply different thresholds. Kaleton Internet [15] built a prototype system by combining the two detection systems, but they work independently without interactions. We consider close cooperation between the two subsystems. Many other researchers have studied supervised anomaly detection systems by training over attack-free traffic [1], [7], [8], [10], [17], [27], [34].

In this paper, we propose the HIDS architecture and prove its effectiveness through simulation experiments. The HIDS integrates the flexibility of ADS with the accuracy of a signature-based IDS. ADS is designed by mining FERs [18], [22], [28] over Internet connections. We developed a new *weighted signature generation* algorithm to characterize anomalous attacks and extract their signatures. The new signatures are generated from anomalies detected by ADS. This idea was inspired by earlier works on weighted association rules [30], [32]. This new approach automatically enables HIDS to detect similar anomalous attacks in the future.

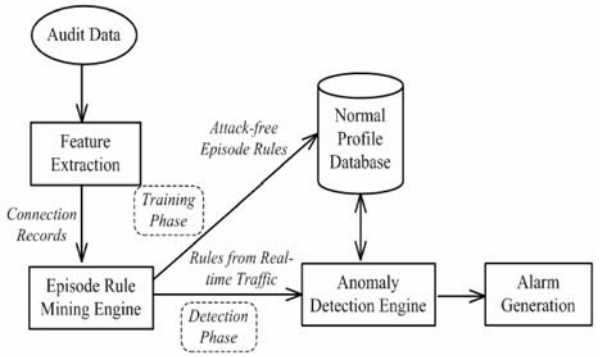


Fig. 1. Data mining scheme for network anomaly detection over Internet connection records.

3 Hybrid Intrusion Detection System (HIDS)

In this section, we first introduce the data mining concept for hybrid intrusion and anomaly detection. Then, we describe the HIDS architecture, the ADS design, and the connection features used in ADS and automated signature generation.

3.1 Traffic Data Mining for Network Anomaly Detection

Open networks face threats from both system intrusions and anomalies in Transmission Control Protocol (TCP), User Datagram Protocol (UDP), or ICMP connections. Fig. 1 shows the three major components of our network anomaly detection process. First, we apply the normal profile database and construct the anomaly detection engine. The detection engine is capable of detecting anomalous episodes that are caused by traffic anomalies. The connection records are extracted from audited Internet traffic. The concept of FER will be covered in Section 4.

The *episode rule mining engine* consists of two phases of development. The *training phase* is needed to generate the normal traffic database without attacks. Attacks may appear in the *detection phase*. The anomaly is detected once the episode rule describing the real traffic connections cannot find any match with normal connection rules in the database. With this reasoning, the network anomaly is detected by a normal-use detection model [10], [13].

We have generated an attack data set by a mixture of locally captured Internet trace files [2] and the DARPA 1999 IDS evaluation data set. The University of Southern California trace file was obtained from an ISP through the Los Nettos network in Southern California. The USC trace file does not contain packet payload. We stretched the trace file to mix with the Massachusetts Institute of Technology/Lincoln Laboratory traffic files collected. We use the toolkits by Mahoney and Chan [21] to mix Internet traffic data with the MIT/LL data set.

A drawback of the FER-based approach is caused by the fact that many attacks are triggered by a single connection and may not generate anomalous FERs. In order to solve this problem, we keep a keen interest on rare attributes of single connections. For example, connections with the same source and destination addresses are often attacks. Another problem is that a single attack may last for a long period of

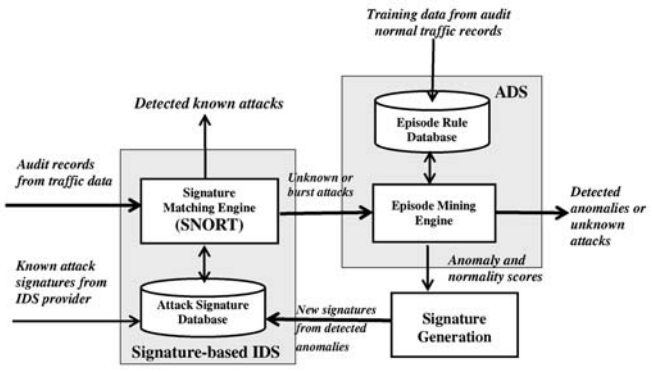


Fig. 2. A hybrid intrusion detection system built with a SNORT and an anomaly detection subsystem (ADS) through automated signature generation from Internet episodes.

time. To solve this problem, we use connection sequence numbers, instead of time stamps, to mine connections heading to the same destination.

3.2 The HIDS System Architecture

Anomaly-based systems are supposed to detect unknown attacks. These systems are often designed for offline analysis due to their expensive processing and memory overheads. Signature-based system leverages manually characterized attack signatures to detect known attacks in real-time traffic. The HIDS illustrated in Fig. 2 integrates the flexibility of ADS with the accuracy of a signature-based SNORT. The SNORT is connected in cascade with the custom-designed ADS. These two subsystems join hands to cover all traffic events initiated by both legitimate and malicious users.

By 2004, SNORT has accumulated more than 2,400 attack signatures in its database [4]. In HIDS operations, the first step is to filter out the known attack traffic by SNORT through signature matching with the database. The remaining traffic containing unknown or burst attacks is fed to the *episode-mining engine* to generate frequent episode rules with different levels of support threshold. This leveling allows the detection of some rare episodes, declared as anomalies. The frequent episodes are compared with precomputed frequent episodes from normal traffic. The episodes that do not match the normal profiles or match them with unusually high frequency are labeled as anomalous.

The anomalous episodes are used to generate signatures which capture the anomalous behavior using a weighted frequent item set mining scheme. These signatures are then added to the SNORT database for future detection of similar attacks. Unknown, burst, or multiconnection attacks are detectable by ADS. The *signature generation unit* bridges two detection subsystems in the shaded boxes. This unit characterizes the detected anomalies and extracts their signatures. We built an ADS by using the FER mining mechanisms to be described in Section 4. The new HIDS detects many novel attacks hidden in common Internet services, such as *telnet*, *http*, *ftp*, *smtp*, *e-mail*, *authentication*, and so forth. The HIDS deployment appeals particularly to protect network-based clusters of computers, resources inside internal networks (intranets), and computational Grids.

3.3 Internet Connection Features

The performance of ADS is directly affected by the features used in training and rule generation. Lee and Stolfo [19] and Lazarevic et al. [17] used connection features, temporal statistics, and content features for building IDSs. Table 1 summarized the connection and temporal statistics features

TABLE 1
Connection Features and Temporal Statistics Used in HIDS Construction

Category	Feature Name	Short Description
Connection Features	timestamp	Time when the connection begins
	duration	Duration of the connection in second
	ip_proto	IP protocol type
	src_ip	Source IP address
	dst_ip	Destination IP address
	service	Network service on the destination, e.g. http, ftp
	icmp_type	ICMP message type
	src_types	Bytes sent by the source
	dst_types	Bytes sent by the destination
	flags	SF: Both SYN and FIN packets are known, S0: Only SYN packet seen in a connection, REJ: Connection rejected by the destination
Temporal statistical features	src_count	Number of connections from the same source
	dst_count	Number of connections to the same destination
	service_count	Number of connections for the same service
	syn_dst%	% of connections with same feature and SYN errors
	service_dst%	% of connections per service to the same destination
	syn_service%	% of connections with SYN error to the same port
	ave_duration	Average duration of connections for the same service
	ave_src_bytes	Average bytes sent by the source.
	ave_dst_bytes	Average bytes sent by the destination.

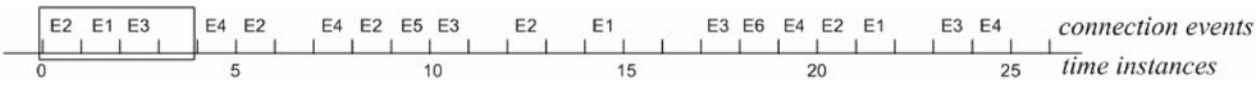


Fig. 3. Generation of a frequent episode rule by scanning a stream of Internet traffic connection events from left to right using a small scanning window.

used in our ADS to generate FERs by training and testing. Since we aim at detecting anomalies of network traffic, the content features, which are mainly extracted from system log files, are not used in this work. The connection features and temporal statistics will be used in HIDS construction.

Connection level features are extracted from raw TCPdump files. They are used in both FER and signature generation. The flags are used to signal special connection status. We listed three different flags: SF, S0, and REJ. Temporal statistics are related to connections with the same reference features. They can be used to improve the accuracy of signature generation. For example, by tracking the number of connections initiated from the same source, the *Source_Count* could be used to set the threshold when we generate new signatures. Because the episode rule generation does not take domain-specific knowledge into consideration, many ineffective or useless rules are generated. How to eliminate useless rules is a major problem in traffic data mining for anomaly detection.

4 Internet Episode Rules and Pruning Techniques

An Internet *episode* is represented by a sequence of connection events, such as TCP, UDP, ICMP, or other connections. An episode can be generated by legitimate users or malicious attackers. *Frequent episodes* mostly resulted from normal users. A *rare episode* is likely caused by intruders. Our purpose is to build an ADS that can distinguish the rare or abnormal episodes from the normal or frequent episodes automatically.

4.1 Generation of Internet Episode Rules

In Fig. 3, we show a typical stream of Internet traffic, represented by a sequence of *connection events* labeled as $E1$, $E2$, $E3$, and so forth These connection events are related to various Internet service commands such as *http*, *ftp*, *smtp*, *authentication*, and so forth. Note that some events may repeat to appear in the sequence. The time instants of these connections, in seconds, are marked below the events. A *frequent episode* is a set of connection events exceeding the occurrence threshold in a scanning window. A FER is generated out of a collection of frequent episodes. The FER is defined over episode sequences corresponding to multiple connection events in a roll.

For the first window in Fig. 3, we cover a sequence of three connection events: $E2$, $E1$, and $E3$. The event $E2$ triggers the occurrence of events $E1$ and $E2$ in a cascade. This leads to the following FER on these three events. This rule is detected within a *window size* w. The rule is backed by a *support base* s, *confidence level* c, and *minimal occurrence* f.

$$E2 \rightarrow E1, E3 \quad (s, c, w, f). \tag{1}$$

Support base s refers to the probability of event $E2$ to occur, and the confidence level c is represented by the probability of the joint connection event $(E1, E3) \bigcup E2$ to take place after event $E2$. Formally, we calculate $s = Prob.[E2]$ and $c = Prob.[(E1, E3) \bigcup E2]/Prob.[E2]$.

The *support value* s reflects the percentage of minimum occurrences of the episode rule out of the total number of connection records audited. The confidence level c is the joint probability of the minimal occurrence of the joint episodes out of the support of the left-hand side (LHS) episode. Both parameters s and c are bounded from below by a *minimum support threshold* s_o and a *minimum confidence level* c_o, respectively, where s_o and c_o are preset by experienced rule designers. The window size w is the scanning period of the window. The *minimal occurrence* f indicates the minimum number of occurrences to establish the rule in question.

For a real-life example, event $E2$ could be an authentication service requested at time zero, presented by two attributes $(service = authentication, flag = SF)$, where the *flag* is defined in Table 1. Events $E1$ and $E3$ correspond to two consecutive SMTP service requests denoted by: $(service = smtp)$ $(service = smtp)$. We obtain the following FER with a confidence level $c = 80$ percent for an authentication service followed by two SMTP services detected within a scanning window $w = 4$ s. The three joint traffic events account a support level $s = 10\,percent$ out of all possible network connections. The minimal occurrence $f = 1,200$ implies that 1,200 is the minimum number of occurrence for the rule to be generated into the rule base.

$$(service = authentication) \rightarrow (service = smtp)$$
$$(service = smtp)(0.8, 0.1, 4\text{ s}, 1, 200). \tag{2}$$

An *association rule* is aimed at finding interesting intrarelationships inside a single connection record. The FER describes the interrelationship among multiple connection records. Let T be a set of traffic connections. In general, an FER is specified by the following expression:

$$L_1, L_{2,} \ldots, L_n \rightarrow R_1, \ldots, R_m \quad (c, s, w, f), \tag{3}$$

where L_i $(1 \leq i \leq n)$ and R_j $(1 \leq j \leq m)$ are ordered connection events in a traffic record set T. We call $L_1, L_2, \ldots L_n$ the LHS *episode* and $R_1, \ldots . R_m$ the RHS (*right-hand side*) of the episode rule. The parameters (c, s, w, f) were defined above and exemplified in (2).

Normal traffic data mining excludes infrequent connection patterns. This makes the system ineffective in detecting anomalous network connections. By lowering the support threshold in the FER generation process, a large number of redundant rules would be generated. Therefore, we desire to have some rule pruning techniques (in Section 4.4) to

eliminate redundant rules and reduce the rule search space accordingly.

4.2 A Base-Support Data Mining Scheme

Most mining techniques exclude infrequent traffic patterns. This will make the IDS ineffective in detecting rare network events. If we lower the support threshold, then a large number of uninteresting patterns associated with frequent services will be discovered. We introduce a new base-support mining process to handle this problem. The process is specified in Algorithm 1. Our method is improved from the levelwise algorithm by Lee et al. [18].

Algorithm 1. Base-Support Traffic Data Mining Algorithm

1: INPUT: Base-support threshold f_0, all axis attributes and the set T of all network connections
2: OUTPUT: New FERs to add into existing rule set L
3: **for** each axis item set X in T, **do**
4: calculate support(X);
5: **end for**
6: scan T and compute $L = \{itemsetY | f(Y) \geq f0\}$;
7: **repeat**
8: generate new epsiode rule sets $E = \{e_0, e_1, \ldots, e_n\}$, where $support(e_0, e_1, \ldots e_n) \geq f_0 \times \min\{base\_sup(e_i)\}$;
9: **if** E is not empty, **then**
10: generate FERs from E with confidence e above minimum confidence e_o;
11: add the generated FERs into rule set L;
12: **end if**
13: **until** E is empty.

By using Lee"s algorithm, one iteratively lowers the minimum support value. Initially, a high minimum support value is chosen to find the episodes related to high-frequency axis attribute values. Then, the procedure iteratively lowers the support threshold by half. This links each new candidate FER with at least one new axis value. The procedure terminates when a very small threshold is reached. Let X be an *item set*. The *support base* of X denoted by $sup\_base(X)$ is the support value of the axis item set. For example, when choosing the service and flag as the axis attributes, the support base for item set $X = (\text{service} = \text{ftp}, \text{flag} = \text{S0}, \text{srchost} = 128.1.1.1, \text{destination} = 121.1.1.1)$ is defined by $sup\_base(X) = support\ (\text{service} = \text{ftp}, \text{flag} = \text{S0})$. The *base-support fraction* f for item set X is defined by:

$$f(X) = support(X) \ / \ base\_sup(X). \tag{4}$$

Similarly, the base-support fraction of an episode is defined as the percentage of the number of minimal episode occurrences to the total number of records in T, which contains the most uncommon axis attributes embedded in this episode. The *minimum support base value* of an episode $e_1, e_2, \ldots, e_n$ is denoted by $\min\{base\_sup(e_i)\}$. To generate an episode, its base-support must exceed a threshold f_o, a lower bound on $f\ (X)$ for all qualified item sets.

To construct the normal network profiles, attack-free training connection records are fed into the data mining engine. Using Lee's levelwise algorithm, it is likely for a normal service to appear in an episode rule with an extremely low support value. This is especially true when the individual connections are independent of each other.

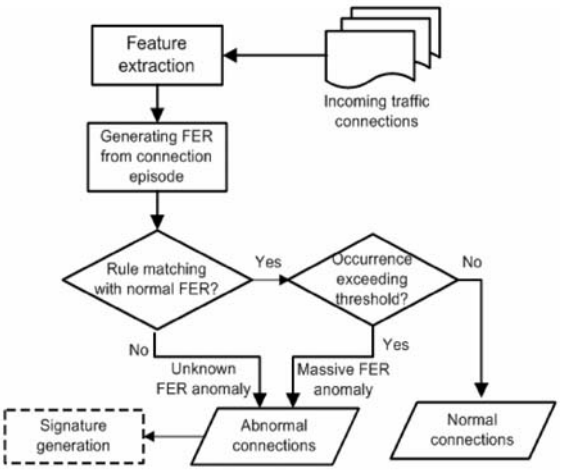


Fig. 4. Matching with frequent episode rules to detect anomalies in incoming traffic.

Our base-support algorithm solves this problem by requiring all FERs related to the common services to occur more often than others.

4.3 Episode Rule Training from Normal Traffic

Fig. 4 shows the FER generation and rule-matching process in anomaly detection based on Algorithm 1. When attacks are detected by SNORT, their *time stamps* are passed to the *packets eliminator*, and the corresponding traffic flows are deleted. The rest of the traffic is passed to the ADS. When a FER generated from the traffic does not match the normal FER database, an unknown FER anomaly is suspected. When the matched rule occurs beyond the threshold, multiple FER anomalies are suspected. The FER anomalies are confirmed by checking some error flags and temporal statistics listed in Table 1. Otherwise, the traffic connection is considered normal.

To generate FERs for normal traffic profiles, the attack-free training connection records are fed into the data mining engine. We use the audit data sets collected in weeks 1 and 3 of the 1999 MIT/LL IDS evaluation package [27]. We generated 92 FERs with the limited training time. We do not use FERs with extremely low support values. After finding FERs from each day's audit record, we simply merge them into a large rule set by removing all redundant rules based on the pruning techniques discussed in Section 4.4. Toward this end, we keep a keen interest on rare attributes of both single and multiple connections.

We use the Bro toolkit [30] to extract useful features from traffic connection records. SNORT sends its time stamp to the packet eliminator to eliminate that packet from repeated checking by the ADS. A major task is to determine the minimum occurrence of the traffic episodes. We distinguish three alert types from three possible sources: the *packet-level alerts* raised by SNORT, *unknown FER anomalies* from no-matching rules, and *massive FER anomalies* due to burst of multiple attacks. The ADS generated 187,059 traffic episode rules in 10 days.

The temporal statistics are collected from the network connection features. A *relaxation factor* α is defined by the scaling ratio of the minimal occurrence f of an FER in a

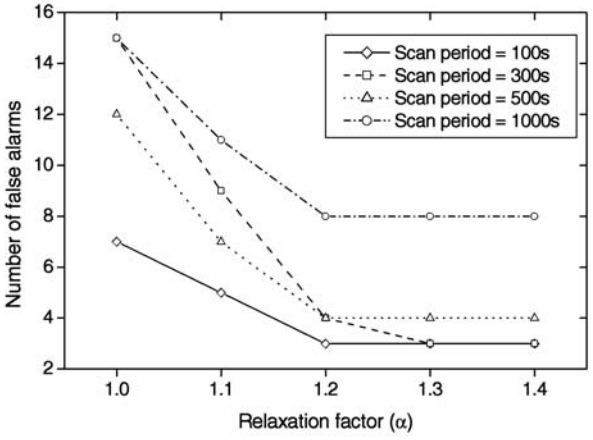


Fig. 5. Effect of relaxation factor α on false alarms in our anomaly detection system. The critical value $\alpha = 1.2$ was set for all window sizes based on our experimental findings.

window to the maximum occurrences m of the rule during the training period. This parameter decides the *threshold* value actually used to raise an alert. The factor affecting the false alarm rate is illustrated in Fig. 5. The maximum false alarm is observed when $\alpha = 1$, and we choose $\alpha \geq \mathbf{1}$. An FER is anomalous if its minimal occurrence f exceeds the maximum number m, as formally conditioned below:

$$f > \alpha\, m. \tag{5}$$

When this condition is met, we signal an anomaly alert. We need to adjust the value of α to reduce the false alarm rate. As shown in Fig. 5, the number of false alarms drops sharply as the relaxation factor increases beyond 1.2. When α exceeds a certain limit, the false alarm rate will not be reduced further. We find such a critical value in Fig. 5 at $\alpha = 1.20$ for all window sizes.

4.4 Pruning of Ineffective Episode Rules

We consider a FER *effective* if it is applicable and more frequently used in the anomaly detection process. An episode rule is *ineffective* if it is rarely used in detecting anomalies. Some FERs differ only at the LHS or at the RHS. Keeping all rules generated will enlarge the search space and thus increase the overhead. The following FER transformation laws will reduce the rule search space significantly.

4.4.1 Transposition of Episode Rules

Comparing the following two FER rules, the item set $(service = http, flag = S0)$ is implied by

$$(service = smtp, flag = SF).$$

Therefore, the second rule can be induced by the first rule. We only need to keep the first rule. The general rule of the thumb is to make the LHS as short as possible. An item set implied by an item set on the LHS should be moved to the RHS if the new rule satisfies the requirement of minimal confidence.

$$(service = smtp, flag = SF) \rightarrow (service = http, flag = S0), \\ (service = http, flag = SF). \tag{6.a}$$

$$(service = smtp, flag = SF), (service = http, flag = S0) \rightarrow \\ (service = http, flag = SF). \tag{6.b}$$

During the detection phase, we generate only one FER from a frequent episode. A large number of redundant rule comparisons could be avoided if more complex rules were removed.

4.4.2 Elimination of Redundant Episode Rules

In general, rules with shorter LHSs are more effective than rules with longer LHSs. This is because shorter rules are often much easier to compare. For example, in the following rule,

$$(service = http)(service = authentication) \rightarrow \\ (service = smtp)(0.6, 0.1). \tag{7}$$

The rule above is considered ineffective with the existence of the following rule:

$$(service = authentication) \rightarrow (service = smtp)\ (0.65, 0.1). \tag{8}$$

The authentication is related only to the *smtp* operation; the *http* does not affect the other two item sets. Therefore, $(service = http)$ can be ignored. Longer rules may introduce some redundant information. Removing them from the normal profile will reduce the false alarms.

4.4.3 Reconstruction of Episode Rules

Many FERs detected from the network traffic have some transitive patterns. Suppose we have two rules $A \rightarrow B$ and $B \rightarrow C$ in the rule set. Then, the longer rule $A \rightarrow B$, C is implied. Since we reconstruct this rule from two shorter rules, the longer rule $A \rightarrow B$, C becomes redundant. The reconstruction helps us split longer FERs into shorter ones. Rule pruning will reduce the false positive rate in an ADS. We are mainly interested in daily network traffic, like the TCP dump. For example, in the following rule,

$$(service = ftp, srcbyte = 1,000) \rightarrow (service = smtp) \\ (service = authentication) \tag{9}$$

is ineffective, because it can be reconstructed from the following two shorter rules:

$$(service = ftp, srcbyte = 1,000) \rightarrow (service = smtp) \tag{10.a}$$

$$(service = smtp) \rightarrow (service = authentication). \tag{10.b}$$

This reconstruction is more powerful if the window size is large. For smaller window sizes, the occurrence of an episode may be longer than the window size, violating the basic assumption in FER formation. This reconstruction may result in fewer false alarms. Fig. 6 shows the effects of rule pruning on the 1999 MIT/LL data sets [20]. Both data sets result in the pruning of 40-70 percent of the

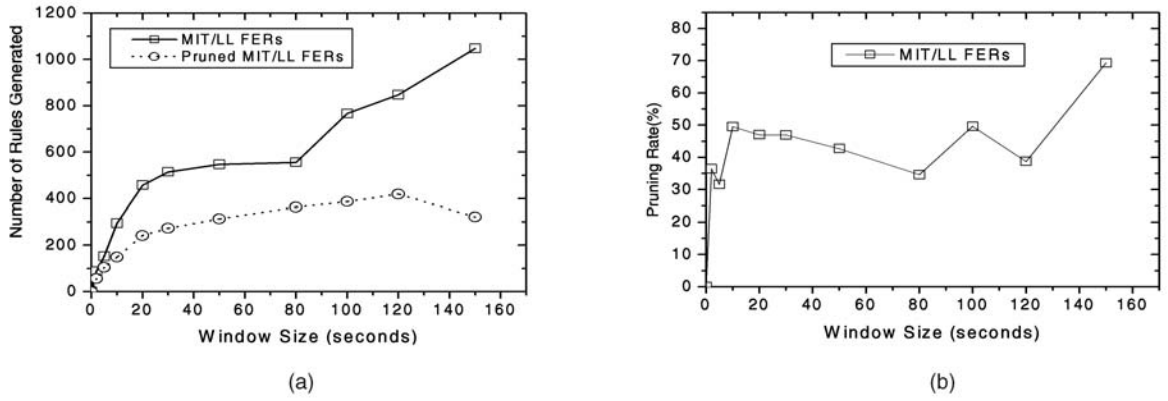


Fig. 6. Pruning of 40-70 percent ineffective episode rules from USC Internet trace mixed with MIT/LL attack data set, August 2005. (a) Episode rule growth. (b) Episode rule pruning rate.

TABLE 2
Traffic Connections and Their Anomaly and Normality Scores (*Dataset-I*)

ip_proto	src_ip	dst_ip	Service/ icmp_type	src_ bytes	dst_ count	Anomaly score	Normality score
icmp	202.77.162.213	172.16.114.50	echo_req	1480	0	0.0	1.0
tcp	172.16.112.100	172.16.114.50	http	4580	0	0.0	1.0
icmp	202.77.162.213	172.16.114.50	echo_req	1480	1	0.2	0.8
icmp	53.88.213.15	172.16.114.50	echo_reply	716	0	0.0	1.0
icmp	202.77.162.213	172.16.114.50	echo_req	1480	10	0.3	0.7
…	…	…	…	…	…	…	…
icmp	202.77.162.213	172.16.114.50	echo_req	1480	11	1.0	0.0
tcp	194.27.251.21	172.16.114.50	smtp	2795	0	0.0	1.0
icmp	202.77.162.213	172.16.114.50	echo_req	1480	10	0.9	0.1
icmp	202.77.162.213	172.16.114.50	echo_req	1480	12	1.0	0.0

redundant episode rules by application of the pruning techniques above. We generate episode rules based on connection patterns encountered. A detailed analysis of these results and a lot more examples can be found in our earlier report [28].

5 Weighted Signature Generation from Detected Anomalies

In this section, we present a weighted signature generation algorithm to characterize anomalies detected by the ADS. The signature-based SNORT imports these signatures and detects the same attacks subsequently. The traffic data set is a normal relation table consisting of N connections. A connection c_i has M attribute-value pairs $< a_j, v_{i,j} >$, where $1 \leq i \leq N$ and $1 \leq j \leq M$. The attributes are selected from the connection features and temporal statistics summarized in Table 1. Our idea extends from the work on weighted association rule mining reported in [30], [32].

The ADS assigns an *anomaly score* and a *normality score* for each connection after processing a traffic data set. The anomaly score indicates the degree of anomaly that a connection is deviated from normal traffic, whereas the normality score indicates how close a connection is related to normal traffic. The ADS assigns the anomaly and normality scores to a given connection by comparing its FERs with the normal profile. In our ADS, the sum of anomaly and normality scores is normalized to be 1. Note that the sum is not necessarily a constant in other anomaly-based detection systems.

Table 2 shows an example of traffic connections and their anomaly and normality scores. Given a set of connections, as well as their anomaly and normality scores assigned by ADS, the problem is reduced to discover the most *specific* and *discriminative* patterns of abnormal connections. A connection pattern corresponds to a subset of $< attribute, condition >$ pairs, which determine possible attribute values in the episode.

We define the anomaly or normality scores of a pattern as the sum of all anomaly or normality scores of all connections matching the pattern. We define signatures as those patterns that have high anomaly scores but relatively low normality scores. Apparently, the higher anomaly score a signature has, the more abnormal connections it would match. Similarly, the lower normal score a signature has, the less normal connections it would match and the fewer false alarms it would expect.

5.1 Weighted Signature Generation

Let $I = \{i_1, i_2, \ldots, i_d\}$ be a set of distinct items and $T = \{t_1, t_2, \ldots, t_N\}$ be a set of transactions. Each transaction t_i

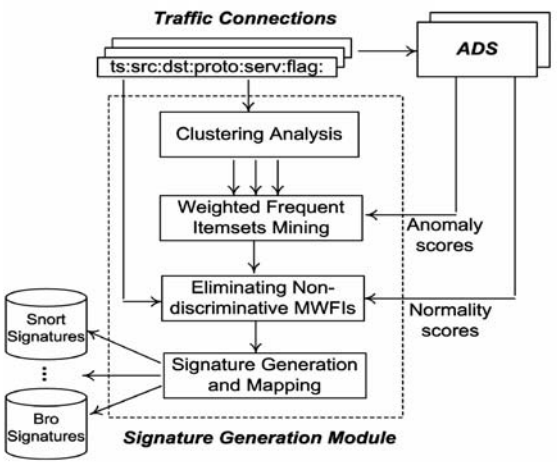


Fig. 7. Functional components in a weighted signature generation unit.

consists of a subset of items from I. An item set X is a subset of I, and its support $sup(X)$ is the fraction of connection containing X. We find the *frequent item sets* whose support exceeds a user-specified threshold (*min_sup*). We represent each connection as a transaction and each $< attribute, value >$ pair as an item. The pattern of $< attribute, condition >$ pairs are constructed from the frequent item sets.

Fig. 7 shows the functional components in the *signature generation unit* introduced in Fig. 2. First, similar abnormal connections are grouped together using a clustering analysis. Then, signatures are extracted for each group sharing some common characteristics. Attributes values of each connection are encoded into item numbers for mining the abnormal attribute-value pairs. After eliminating non-discriminative item sets, the frequent item sets are decoded into $< attribute, condition >$ pairs to form the anomaly signatures.

5.2 Cluster Analysis and Attribute Preprocessing

In traffic connections, different types of unknown attacks are often mixed together. The anomaly and normality scores assigned by ADSs are not sufficient to classify unknown attacks into different groups. It is difficult to discover accurate patterns or signatures for different types of attacks if they are mixed together. Different attacks have skewed distribution on connection volume. For example, different denial-of-sevice (DoS) attacks could have a large volume of short connections, whereas remote-to-local (R2L) or user-to-root (U2R) attacks could only have a few connections.

Table 3 illustrates the attribute discretization and encoding in a *Dataset-I*. To separate different types of attacks, we use cluster analysis to group similar attacks together and generate signatures for each attack class. We use the density-based clustering algorithm by Ester et al. [9] to obtain the partitioned cluster. Connections in low-density regions are classified as noises and thus omitted.

Since each connection is characterized by symbolic and continuous values, we discretize the attribute values such as *src_bytes* in Table 3 by the entropy method introduced by Fayyad and Irani [11]. We encode each pair $< \text{attribute}, \text{value} >$ with an item number. Each item number has six digits. The first two digits represent the attribute index. The remaining digits are discrete symbolic values. For example, the pair $< proto, tcp >$ could be encoded as $3 \times 10,000 + 2 = 030002$, where 03 is the index of attribute *prot* and 0002 is the index of the attribute value *tcp*.

5.3 Mining of Weighted Frequent Item Sets

The weighted principle is based on the fact that if a connection contains an item set X, then it also contains all

TABLE 3
The Discretization and Encoding of Attributes in Dataset-I

ID	ip_proto (1)	service/icmp_type (2)	src_bytes (3)	dst_count (4)	Anomaly score	Normality score
1	10001	20001	30148	40000	0.0	1.0
2	10002	20002	30458	40000	0.0	1.0
3	10001	20001	30148	40000	0.2	0.8
4	10001	20002	30071	40000	0.0	1.0
5	10001	20001	30148	40010	0.3	0.7
	...	...	...	...	...	...
1001	10001	20001	30148	40010	1.0	0.0
1002	10002	20003	30279	40000	0.0	1.0
1003	10001	20001	30148	40010	0.9	0.1
1004	10001	20001	30148	40010	1.0	0.0

Symbol index	ip_proto (1)	service/icmp_type (2)	src_bytes (3)	dst_count (4)
1	icmp	echo_req	continuous	continuous
2	tcp	http	-	-
3	-	echo_reply	-	-
4	-	smtp	-	-

subsets of X. Therefore, the weighted support of subsets of X must be greater than that of X. Algorithm 2 specifies a weighted a priori algorithm for generating weighted frequent item sets. The method updates the weighted support of an item set using connection weight. Connections are assigned different anomaly scores, and we find frequent item sets with respect to anomaly scores.

Our signature generation is based on a *weighted frequent item set mining* (WFIM) framework. Algorithm 2 extends from this framework to generate anomaly scores. Let w_i be the weight of connection t_i, the *weighted support* of an item set X is defined by $wsup(X) = \sum_{t_i \supseteq X, t_i \in T} w_i / \sum_{t_i \in T} w_i$. The purpose is to discover all item sets whose weighted supports are above the minimum support (*min_wsup*). We call those item sets *weighted frequent item sets* (WFIs). The anomaly score weighs a connection. The *min_wsup* helps select desired signatures.

Algorithm 2. Weighted A Priori Algorithm for Generating Signatures from Anomalies Detected

1: INPUT: A set of items I, a set of connections (that is transactions) T, weight w_t of connection t, and minimum weighted support *min_wsup*
2: OUTPUT: Weighted frequent item sets X with $wsup(X) > min\_wsup$
3: $W = \sum_{t \in T} w_t$;
4: $k = 1$;
5: $L_1 = \{i | i \in I \wedge wsup(i) > min\_wsup\}$;{ Find all weighted frequent 1 item sets }
6: **repeat**
7: $k = k + 1$;
8: $C_k = \text{apriori\_gen}(L_{k-1})$; { Generate candidate item sets }
9: **for** each connection $t \in T$, **do**
10: $C_t = \text{subset}(C_k, t)$; { Candidates contained in t }
11: **for** each candidate item set $c \in C_t$, **do**
12: $c.\text{weight} += w_t$; { Add connection weight }
13: **end for**
14: **end for**
15: $L_k = \{c \in C_k | c.\text{weight} \ / \ W \geq min\_wsup\}$;
16: **until** $L_k = \emptyset$
17: **return** $X = \cup L_k$.

We consider the all weighted connections mining process, which is equivalent to transactions in the a priori algorithm [30]. The following algorithm is specified for weighted anomaly signature generation. If the support of an item set exceeds *min_sup*, then all its subsets must be supported. This principle is used by the a priori algorithm to effectively prune candidate item sets. We add items to item sets that are sufficiently large. We follow a weighted a priori principle: If the weighted support $wsup(X)$ of an item set X exceeds *min_wsup*, all of its subsets have their weighted support exceeding *min_wsup* as well.

5.4 Signature Extraction and Mapping

For a given *min_wsup* threshold, the WFIM scheme discovers all WFIs whose weighted supports exceed the threshold. When we consider many features as discussed in Section 3.2, the number of discovered WFIs could be too large to construct the precise anomaly signatures. In particular, there are many redundant WFIs that have to be pruned before signatures can be generated. Instead of using all WFIs, we adopt a notion of *maximal weighted frequent item sets* (MWFIs) as a compact representation of all WFIs used.

TABLE 4
Mappings between Connection Attributes and SNORT Rule Keywords

Attribute Name	SNORT Rule Keyword	Short Description
ip_proto	protocol	IP protocol type
src_ip	source IP address	source IP address
dst_ip	destination IP address	destination IP address
service	destination port No.	Service type, e.g. http
icmp_type	itype	ICMP message type
src_bytes	Dsize	Packet payload size
flags	Flags	TCP flags
land	Sameip	Source and destination addresses are the same
src_count <n>	threshold: track by_src, count <n>	No. of connections to the same destination
dst_count <n>	threshold: track by_dst, count <n>	No. of connections from the same source

An MWFI is defined as a weighted frequent item set for which none of its immediate supersets has a support above the *min_wsup* threshold. Since any subset of an MWFI is also a WFI, the number of WFIs that an MWFI represents will increase exponentially with the number of items in an MWFI. For example, an MWFI with k items could represent all $2^k - 2$ WFIs. When *min_wsup* is 90 percent, there is only one MWFI for the Dataset-I, that is, {10,001, 20,001, 30,148, 40,010}. After the MWFIs of abnormal connections are discovered, we extract signatures by decoding the item numbers in MWFIs into $< attribute, condition >$ pairs. The $< attribute, condition >$ pairs represent the abstract signatures of the detected anomaly. These abstract signatures are then mapped into specific signatures of a target IDS system such as SNORT or Bro.

Table 4 shows how connection attributes are mapped into the keywords in SNORT rules. Besides the compact representation of frequent item sets, we eliminate indiscriminative ones that appear very often in normal traffic. Since the *min_sup* threshold is relatively low to discover frequent item sets in normal traffic, the number could be very large. Instead, we recalculate the weighted support of previous MWFIs by using normality scores and eliminate the ones with large support. Recall that in the MWFI for Dataset-I, the $< attribute, condition >$ pair is decoded as follows:

$$(ip\_proto = icmp), (icmp\_type = echo\_req), \\ (1,480 <= src\_bytes < 1,490), (dst\_count > 10). \quad (11)$$

The $< attribute, condition >$ pairs form an abstract signature of the ***Pod*** attack. Using the attribute mappings in Table 4, we translate the signature into a SNORT rule as follows:

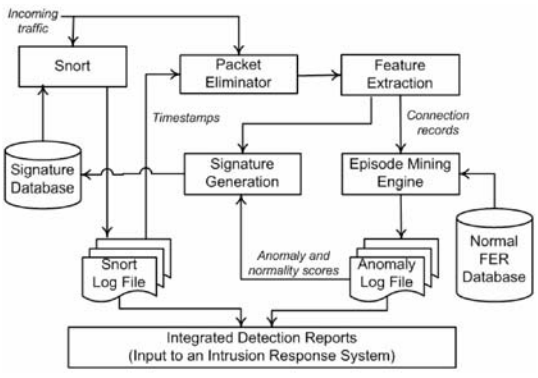


Fig. 8. The architecture of the HIDS simulator built with SNORT 2.1 and an anomaly detection subsystem bridged by a weighted signature generator.

$$alert\ icmp\$EXTERNAL\_NET\ any <> \$HOME\_NET\ any$$
$$(msg :''\ possible\ pod\ attack''; itype : 8;$$
$$dsize : 1,480 <> 1,490; threshold : type\ both, track\ by\_dst,$$
$$count\ 10\ seconds\ 1; sid : 900,001; rev : 0;). \quad (12)$$

6 HIDS Simulation and Raw Data Collection

This section introduces the HIDS simulator architecture, the attack data set used, and the Internet trace applied in our security experiments to generate the performance results.

6.1 The HIDS Simulator and Attack Data Sets

Fig. 8 shows the functional blocks in the HIDS simulator built at USC. This system was simulated on a Dell Linux server. The system consists of SNORT 2.1 for detecting known attacks, an FER-based ADS for detecting anomalies, and a *signature generator* between them. The installed SNORT 2.1 is a lightweight network IDS with more than 2,000 signatures in its database [31]. The packet eliminator removes those attacks already detected by SNORT. The eliminator applies the time stamps to eliminate packets to avoid repeated work by the ADS.

An *episode mining engine* was built in the simulator to generate the *FER database* and the *anomaly log files*. The size of the FER database is decided by training data and threshold. We applied the 1999 MIT/LL weeks 1 and 3 attack-free training data set to obtain 92 FER rules for normal traffic. This FER database is sufficient for our simulation study. The storage requirements for both signature database and the FER database are rather small in our simulator. In real-life practice, the FER database may grow to several thousand of rules.

We used the MIT/LL data set [20], [21] to evaluate the performance of our HIDS. There are, in total, more than 200 attacks in the MIT/LL data set, consisting of 58 DoS attacks, 62 R2L attacks, 31 U2R attacks, 44 Probe or port scanning attacks, and six secret or unknown attacks. Some of the attacks repeat many times at different days. These attacks are included at weeks 4 and 5 of the data sets.

Since the background traffic in the MIT/LL data set is synthetically generated, it may not represent the real traffic profile [21], [23]. We used the toolkits by Mahoney and Chan [21] to mix the collected trace at USC with the MIT/LL attack data set [20]. The USC trace was obtained from an ISP through the Los Nettos network in Southern California. After mixing the real-life background traffic, the entire traffic data set experimented with is about 1.5 times larger than the original attack data set from MIT/LL.

6.2 Simulation Parameters and Raw Data Collected

In Table 5, we summarize important system parameters and raw data collected in 10 days of simulated Internet data mining and detection experiments. We use SNORT and an ADS built at our laboratory to process the incoming traffic sequentially. In total, 35.025 million packets were processed

TABLE 5
Packet Counts, SNORT Log Files Reported, and Processing Times of SNORT and ADS in 10 Days of Internet Trace Experiments on the Simulated HIDS at USC

Day	Packet Count (M)	TCP Packets	UDP Packets	ICMP Packets	ARP Packets	Other Packets	Snort Log files	Episode Rules by ADS	Snort Time (Sec)	ADS Time (Sec)
Day 1	2.46M	1.935M	434,154	78,308	8,089	13,834	487	957	18.408	67.965
Day 2	1.95M	1.83M	19,690	1,275	28,775	86,180	830	1650	14.868	35.385
Day 3	2.64M	2.01M	599,235	9,022	8,418	13,834	971	2146	23.64	83.04
Day 4	3.525M	2.535M	977,978	1,983	10,074	13,835	3,301	2395	44.82	125.505
Day 5	2.91M	2.025M	854,757	9,768	9,601	13,834	568	1932	22.02	130.845
Day 6	3.435M	2.115M	1,277,999	14,743	8,526	13,836	903	1761	33.396	159.72
Day 7	5.1M	3.93M	1,155,477	1,840	9,441	13,839	4,043	2243	39.996	179.25
Day 8	3.12M	2.265M	841,943	1,679	7,281	13,838	1,773	2247	32.196	110.58
Day 9	4.8M	3.705M	105,9422	5,437	16,526	13,834	7,593	18103	57.504	195.93
Day 10	5.085M	4.08M	1,011,153	1,874	9,106	13,834	4,150	3789	52.428	209.895
Total	35.025M	26.43M	8.23M	125,929	115,837	210,698	24,619	37223	339.276	1298.115

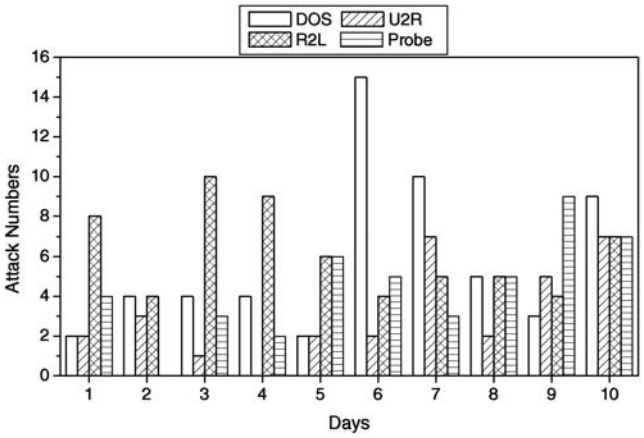


Fig. 9. Attack spectrum of the MIT/LL data set in 10 days, including four categories of DoS, U2R, R2L, and probe attacks.

by SNORT. Most traffic involve TCP, UDP, and ICMP packets, plus a smaller number of ARP and other packets. The number of incidents detected by SNORT was recorded with 24,619 alerts or log files. The total recorded time of using SNORT to scan all traffic packets was 339.276 seconds. ADS was used for 1,298.115 seconds to generate 37,223 traffic episode rules.

This implies a SNORT *processing speed* of 86,400 packets per second or an *average detection performance* of eight intrusions per second. Compared with SNORT, the ADS detection speed is relatively lower. The attack spectrum across 10 days is shown in Fig. 9. With a scanning window size of 300 seconds, ADS generated 37,223 episode rules in 1,298.115 seconds, which implies 29 rules per second on the average. Techniques to eliminate redundant episode rules reduced the FER space and, thus, the anomaly detection time accordingly.

7 Experimental Results and Performance Analysis

The detection performances of SNORT, Bro, ADS, and HIDS are reported below. ADS detects anomalous Internet connections without signatures. HIDS is designed to cope with both known and unknown attacks. Through interactive machine learning, HIDS could enhance its sensitivity to detect all kinds of intrusions or anomalies effectively.

7.1 Performance Metrics for Evaluating IDS

We use three metrics to evaluate the IDS performance, namely, the *intrusion detection rate* δ, *false alarm rate* η, and *receiver operating characteristic (ROC)*. The intrusion detection rate (denoted by δ) is formally defined by

$$\delta = d/n, \tag{13}$$

where d is the number of detected attacks, and n is the total number of actual attacks. The *false alarm rate* (denoted by η) measures the percentage of false positives among all normal traffic events. A formal definition is given by

$$\eta = p/k, \tag{14}$$

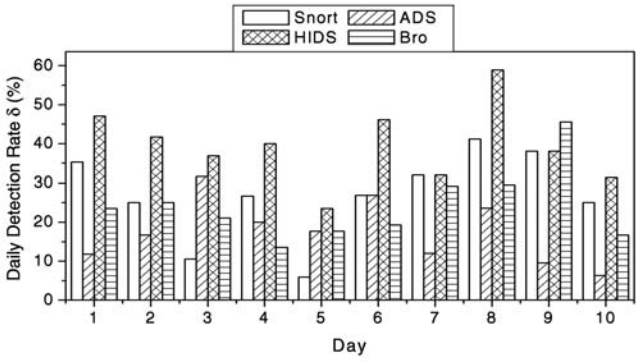


Fig. 10. Daily average detection rates of four IDSs over the 1999 MIT/LL attack data set.

where p is the total number of false positive alarms and k accounts for the total number of connection events. For TCP, SNORT generates an alarm for every connection. However, for UDP and ICMP, it generates an alarm for every packet. Therefore, we calculate m by counting all TCP connections and UDP/ICMP packets. The ROC curve evaluates the tradeoff between the intrusion detection rate and the false positive rate.

7.2 Performance Results of SNORT, Bro, ADS, and HIDS

Fig. 10 plots the daily average detection rate of using the four IDSs. It shows clearly that the HIDS achieved a detection rate between 25 and 60 percent on various days. This outperforms all three other systems significantly. ADS, Bro, and SNORT have mixed performance that varies between 5 and 40 percent in 10 days. HIDS has captured all attacks detected by either SNORT or ADS. There is a very small overlapping between the attacks detected by both SNORT and ADS jointly in the feedback loop in Fig. 2.

We plot in Fig. 11 the *average detection rate* for four attack types over 10 days of experiments using the MIT/LL attack data set. Bro performs best on U2R attacks, whereas ADS fails to detect any U2R attack with a scanning window size that equals 300 seconds. We could detect *U2R* and *data* attacks by setting a different window size. HIDS performs

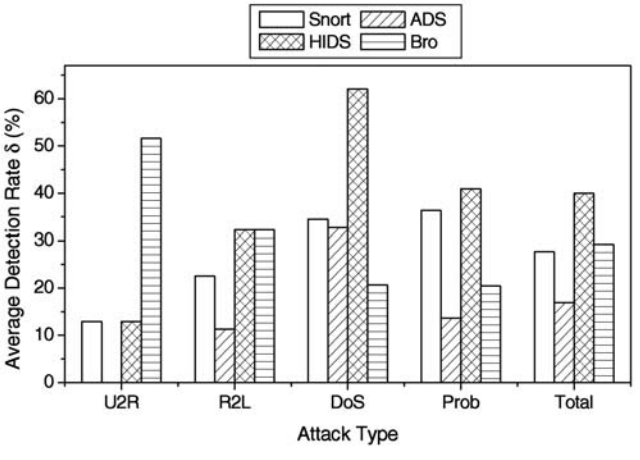


Fig. 11. Average detection rates for four attack types over 10 days.

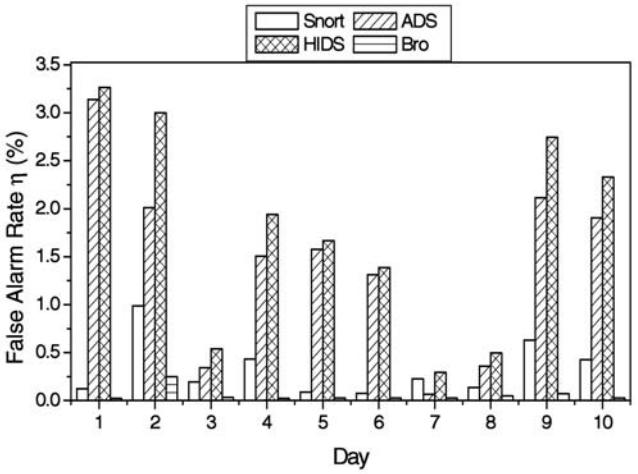


Fig. 12. Daily false alarm rates of four IDSs over 10 days using a scanning window size of 300 s.

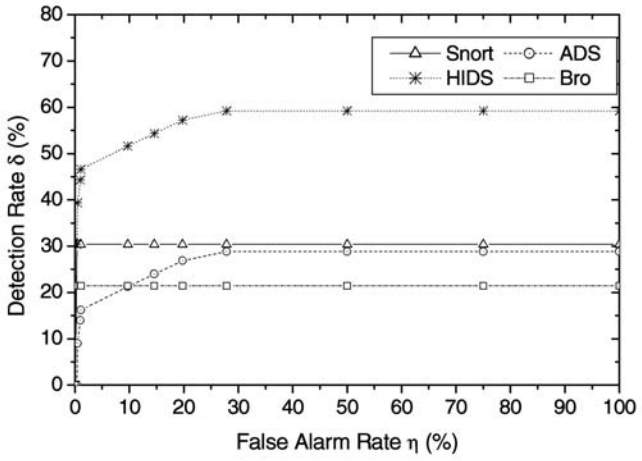


Fig. 13. ROC curves showing the variation of the average intrusion detection rate of three detection systems as the false alarm rate increases.

the best, with 60 percent detection of all DoS attacks. In total, ADS has an average of 14 percent detection rate, whereas HIDS detects 40 percent of all attacks, higher than 25 percent on SNORT and 27 percent on Bro system.

The time consumed by ADS is mainly attributed to the complexity of feature extraction, episode classification, FER generation, and matching with the FER database. Due to the sequential processing of IDS followed by ADS, the total HIDS time is the sum of the two subsystems. The ADS time dominates the total processing time. Therefore, generating new signatures reduces the total detection time.

7.3 Effects of False Alarms on IDS Performance

False alarms and intrusion detection accuracy are two related concepts. Trade-offs exist between these performance measures. Accuracy is tied to the intrusion detection success rate. We report below the effects of false alarms on the detection results in 10 days of experiments. In Fig. 12, SNORT has less than 1 percent false alarms. The recorded ADS false alarm rate was 3.2 percent in day 1 and then reduced to less than 2.2 percent in the remaining days. HIDS has a slightly higher false alarm rate less than 3.4 percent. Bro has the lowest false alarm rate, which is less than 0.2 percent. The false alarm rate increases with the increase of scanning window sizes.

SNORT contributed very little to false alarms. It is ADS that causes most false alarms. With a window of 300 seconds or less, we see 11 false alarms in 10 days. For a large window size of 7,200 seconds (2 hours), the number of false alarms increases to 17. The U2R attack has the lowest number of false alarms. The DoS and R2L attacks result in 4-6 false alarms, and the Probe attacks result in 2-4 false alarms. To avoid false alarms, one must use smaller scanning windows. Our recommendation is to limit the window size to less than 300 seconds. The ROC curves in Fig. 13 show the trade-off performance of four IDSs over all attack types.

The HIDS achieved a low 47 percent detection rate at 1 percent false alarms. However, the detection rate can be raised to 60 percent if the false alarms can be tolerated up to 30 percent. SNORT has almost a constant 30 percent detection rate with almost zero false alarms. ADS can reach a 30 percent detection rate if we can tolerate 30 percent false alarms. The enhanced HIDS performance is obviously driven by the ADS performance. As a comparison, Bro has a 22 percent detection rate with almost zero false alarms. These results confirm the claimed advantages of HIDS. The balancing point is up to the designer's choice between detection accuracy and tolerance of false alarms.

7.4 Effects of Signature Generation on SNORT Performance

To verify the weighed signature generation approach, we compare the performance of SNORT with different sets of signatures installed. We generated a set of new SNORT signatures semiautomatically by mining the weighted frequent item sets from detected anomalies. We selected the top nine signatures with largest minimum weighted supports as *generated signatures.* Fig. 14 shows the daily intrusion detection rates of SNORT with default, generated, and combined signatures installed.

The default and generated signatures are often complimentary to each other, that is, they will detect different

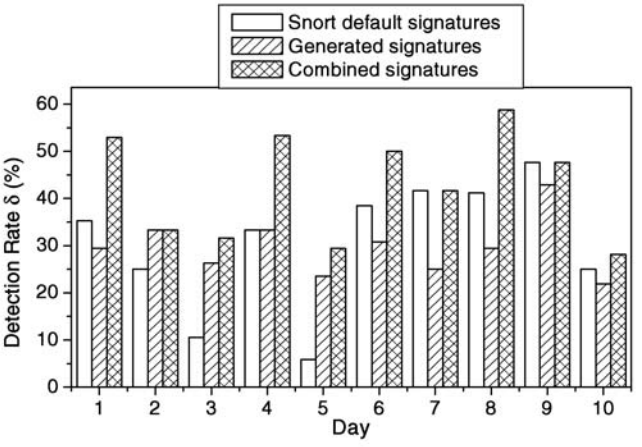


Fig. 14. Detection rates of SNORT with default, generated, and combined signatures.

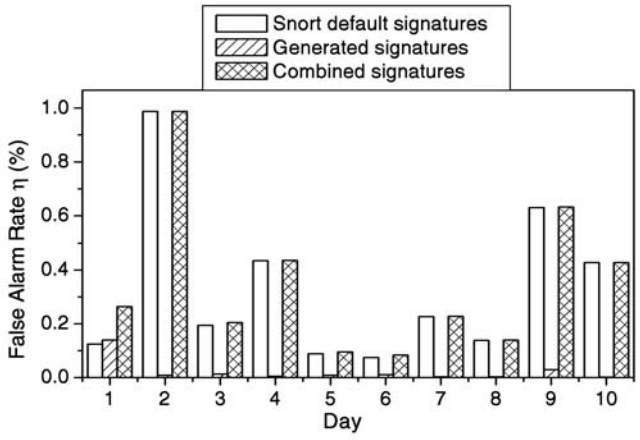


Fig. 15. False alarm rates of SNORT with default, generated, and both sets of signatures.

attacks. For example, in the first day, the default and generated signatures detected 35 and 29 percent attacks, respectively, whereas 53 percent attacks were detected when both sets of signatures were installed. On the other hand, the two sets of signatures detect overlapped attacks as well. Overall, the added signatures from ADS enhance the SNORT detection rate by 20-60 percent, except in day 9. This is a remarkable improvement with automated signature generation from the anomalies detected by ADS.

Fig. 15 shows the false alarm rate of SNORT when different sets of signatures were installed in 10 days. The generated signatures introduce very few or zero false alarms, compared with those flagged by using default signatures. The largest daily increase of false alarms is 0.03 percent in day 1. Therefore, after adding generated signatures, SNORT only encountered a less than 1 percent increase in false alarms than using the original signatures. When signature generation is not used, HIDS has a higher false alarm rate than SNORT and Bro as shown in Fig. 12.

However, in our weighted signature generation approach, only the most frequent patterns of detected anomalies are characterized as signatures. By further eliminating nondiscriminative patterns, the generated signatures are quite specific to anomalies detected. Therefore, the newly generated signatures have quite low false alarm rates. Table 6 shows the SNORT processing times under three conditions. Using only default signatures, it takes SNORT 339.276 seconds to process a total of 35.025 million packets in 10 days of trace data.

The combined processing time include both original default signatures and new signatures generated by ADS. The SNORT processing time varies in 10 days. During the first four days, the addition of generated signature even reduced the total time, because the detection rate was greatly enhanced as shown in Fig. 14. This resulted from a sooner ending of the matching process, when more signatures are used. After day 5, the processing time began to increase roughly by 15-25 percent per day. Considering all 10 days, the addition of new signatures increased the processing time to 389.808 seconds. This slight increase by 50 seconds in the overhead is totally acceptable for a 10-day trace of traffic data.

TABLE 6
SNORT Processing Times Using Default and Generated Signatures

Day	SNORT processing time (Sec)		
	SNORT Default signatures	Generated signatures	Combined signatures
Day 1	18.408	5.69	18.228
Day 2	14.868	5.38	14.868
Day 3	23.64	6.26	20.736
Day 4	44.82	14.29	43.884
Day 5	22.02	10.73	25.164
Day 6	33.396	13.88	38.796
Day 7	39.996	17.65	50.1
Day 8	32.196	13.86	38.172
Day 9	57.504	25.52	77.616
Day 10	52.428	29.32	62.244
Total	339.276	142.56	389.808

8 Conclusions and Further Research

We summarize major contributions and make some suggestions for further work on automated detection of intrusions and anomalies in an open network environment. The hybrid IDS/ADS system applies to protect any networked systems, including LAN-based clusters or intranets, large-scale computational Grids, and peer-to-peer service networks, and so forth. Summarized next are lessons learned from the HIDS construction and conclusions that can be made from the reported theories and experimental results.

1. *A new base-support data mining scheme for generating frequent episode rules.* We proposed a *base-support data mining* scheme ([Figs. 1 and 2] and Algorithm 1) to facilitate episode rule generation. Combining SNORT and our ADS, the HIDS outperforms the SNORT and Bro systems by 100 percent and 173 percent, respectively. The HIDS advantages come from using dynamic data mining threshold and automated signature generation.
2. *Anomaly detection system using Internet episode rules.* Both SNORT and ADS subsystems have low processing overhead. The integration of ADS has upgraded the SNORT detection rate by 40 percent with less than a 1 percent increase in false alarms. The enlarged signature database in SNORT pays off. Generating more signatures by ADS will further enhance the overall performance of the hybrid intrusion detection system.
3. *Pruning techniques developed to reduce the episode rule space.* Three episode rule pruning techniques are developed to reduce the frequent episode search space by 40-70 percent. Without reduction, the rule search space may escalate too large to be practical for just one day's collection of traffic records. This has significantly enhanced the ADS performance.
4. *Weighted signature generation enables hybrid intrusion detection.* Our HIDS results in a detection rate of 60 percent, which doubles the 30 percent in using

SNORT and almost triples the 22 percent in using Bro alone. To achieve an even higher detection rate, the false alarms must be maintained below 3 percent. Alerts from intrusions and anomalies detected can be correlated to result in an even smaller overhead in the detection process.

For further research, we suggest the two following issues for continued research and development effort. Both issues demand prototyping and benchmark experiments.

5. *Prototyping HIDS is needed by the research community.* The capability and accuracy of the *anomaly detection system* depend on the training data used in Internet traffic traces. We proved the concept of weighted signature generation with limited Internet trace data mixed with the MIT/LL attack data set. We suggest extending the simulated experimental work to a prototype HIDS construction. The developer can use the DETER Testbed to analyze Internet episodes without disturbing other legitimate users.
6. *Distributed HIDS will advance state of the art in collaborative intrusion detection.* Extensive benchmark experiments are needed on the DETER testbed for this purpose. Extending a centralized HIDS to a distributed one is highly recommended with strong collaboration over multiple IDS sites. This offers a logical solution to protect Grids, clusters, intranets, and so forth. Cyber trust negotiations and frequent alert information exchanges among distributed IDS sites are the key research issues yet to be solved.

ACKNOWLEDGMENTS

The funding support of this work by the US National Science Foundation ITR Grant ACI-0325409 is appreciated. The authors thank the GridSec team members for inspiring discussions during the courses of this research. In particular, we appreciate the critical comments by Dr. Ricky Kwok and the data mining experiments assisted by Ms. Hua Liu. The paper is significantly extended from the concepts and preliminary results reported in the IEEE International Symposium on Network Computing and Applications (IEEE NCA '04) and in the IEEE International Workshop on System and Network Security (SNS '05).

REFERENCES

[1] D. Barbara, J. Couto, S. Jajodia, L. Popyack, and N. Wu, "ADAM: Detecting Intrusions by Data Mining," *Proc. IEEE Workshop Information Assurance and Security,* 2001.

[2] D.J. Burroughs, L.F. Wilson, and G.V. Cybenko, "Analysis of Distributed Intrusion Detection Systems Using Bayesian Methods Performance," *Proc. IEEE Int'l Computing and Comm. Conf.,* pp. 329-334, 2002.

[3] M. Cai, K. Hwang, J. Pan, and C. Papadupolous, "WormShield: Fast Worm Signature Generation Using Distributed Fingerprint Aggregation," *IEEE Trans. Dependable and Secure Computing,* to be published.

[4] B. Casewell and J. Beale, *SNORT 2.1, Intrusion Detection,* second ed. Syngress, May 2004.

[5] W. Cohen, "Fast Effective Rule Induction," *Proc. 12th Int'l Conf. Machine Learning.* 1995.

[6] F. Cuppens and A. Miege, "Alert Correlation in a Cooperative Intrusion Detection Framework," *Proc. 2002 IEEE Symp. Security and Privacy,* pp. 187-200, 2002.

[7] L. Ertoz, E. Eilertson, A. Lazarevic, P. Tan, J. Srivastava, V. Kumar, and P. Dokas, "The MINDS—Minnesota Intrusion Detection System," *Next Generation Data Mining,* MIT Press, 2004.

[8] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A Geometric Framework for Unsupervised Anomaly Detection: Detecting Intrusions in Unlabeled Data," *Applications of Data Mining in Computer Security,* Kluwer Academic Publishers, 2002.

[9] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, "A Density-Based Algorithm for Discovering Clusters in Large Spatial Databases with Noise," *Proc. Second Int'l Conf. Knowledge Discovery and Data Mining,* 1996.

[10] W. Fan, M. Miller, S. Stolfo, W. Lee, and P. Chan, "Using Artificial Anomalies to Detect Unknown and Known Network Intrusions," *Proc. First IEEE Int'l Conf. Data Mining,* Nov. 2001.

[11] U.M. Fayyad and K.B. Irani, "Multi-Interval Discretization of Continuous-Valued Attributes from Classification Learning," *Proc. Int'l Joint Conf. Artificial Intelligence (IJCAI '93),* pp. 1022-1027, 1993.

[12] S. Floyd and V. Paxson, "Difficulties in Simulating the Internet," *IEEE/ACM Trans. Networking,* vol. 9, no. 4, pp. 392-403, Aug. 2001.

[13] K. Hwang, Y. Chen, and H. Liu, "Defending Distributed Computing Systems from Malicious Intrusions and Network Anomalies," *Proc. IEEE Workshop Security in Systems and Networks (SSN '05) held with the IEEE Int'l Parallel & Distributed Processing Symp.,* 2005.

[14] K. Hwang, Y. Kwok, S. Song, M. Cai, Y. Chen, and Y. Chen, "DHT-Based Security Infrastructure for Trusted Internet and Grid Computing," *Int'l J. Critical Infrastructures,* vol. 2, no. 4, pp. 412-433, Dec. 2006.

[15] Kaleton Internet, "Combination of Misuse and Anomaly Intrusion Detection Systems," http://www.kaleton.com, Mar. 2002.

[16] K.S. Killourhy and R.A. Maxion, "Undermining an Anomaly-Based Intrusion Detection System Using Common Exploits," *Proc. Int'l Symp. Recent Advances in Intrusion Detection (RAID '02),* pp. 54-73, Sept. 2002.

[17] A. Lazarevic, L. Ertoz, V. Kumar, A. Ozgur, and J. Srivastava, "A Comparative Study of Anomaly Detection Schemes in Network Intrusion Detection," *Proc. Third SIAM Conf. Data Mining,* 2003, http://www.users.cs.umn.edu/~kumar/papers.

[18] W. Lee, S.J. Stolfo, and K. Mok, "Adaptive Intrusion Detection: A Data Mining Approach," *Artificial Intelligence Rev.,* vol. 14, no. 6, pp. 533-567, Kluwer Academic Publishers, Dec. 2000.

[19] W. Lee and S. Stolfo, "A Framework for Constructing Features and Models for Intrusion Detection Systems," *ACM Trans. Information and System Security (TISSec),* 2000.

[20] R.P. Lippmann and J. Haines, "Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation," *Proc. Third Int'l Workshop Recent Advances in Intrusion Detection (RAID '00),* H. Debar, L. Me, and S.F. Wu, eds., pp. 162-182, 2000.

[21] M.V. Mahoney and P.K. Chan, "An Analysis of the 1999 DARPA/Lincoln Lab Evaluation Data for Network Anomaly Detection," *Proc. Int'l Symp. Recent Advances in Intrusion Detection,* pp. 220-237, Sept. 2003.

[22] H. Mannila and H. Toivonen, "Discovering Generalized Episodes Using Minimal Occurrences," *Proc. Second Int'l Conf. Knowledge Discovery and Data Mining,* Aug. 1996.

[23] J. McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Off-line Intrusion Detection System Evaluation as Performed by Lincoln Laboratory," *ACM Trans. Information and System Security,* vol. 3, no. 4, Nov. 2000.

[24] P. Ning, S. Jajodia, and X.S. Wang, "Abstraction-Based Intrusion Detection in Distributed Environments," *ACM Trans. Information and System Security,* vol. 4, no. 4, pp. 407-452, Nov. 2001.

[25] S. Noel, D. Wijesekera, and C. Youman, "Modern Intrusion Detection, Data Mining, and Degrees of Attack Guilt," *Applications of Data Mining in Computer Security,* D. Barbarà and S. Jajodia, eds., Kluwer Academic Publishers, 2002.

[26] V. Paxson, "Bro: A System for Detecting Network Intrusions in Real Time," *Proc. Seventh USENIX Security Symp.,* 1998.

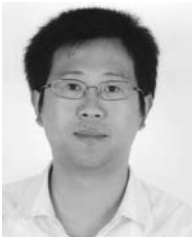
[27] P.A. Porras and P.G. Neumann, "EMERALD: Event Monitoring Enabling Responses to Anomalous Live Disturbances," *Proc. 19th Nat'l Computer Security Conf.,* pp. 353-365, Oct. 1997.

[28] M. Qin and K. Hwang, "Frequent Episode Rules for Internet Traffic Analysis and Anomaly Detection," *Proc. IEEE Network Computing and Applications (NAC '04),* Sept. 2004.

[29] D.J. Ragsdale, C.A. Carver, J. Humphries, and U. Pooch, "Adaptation Techniques for Intrusion Detection and Response Systems," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics,* pp. 2344-2349, Oct. 2000.

[30] G.D. Ramkumar, S. Ranka, and S. Tsur, "Weighted Association Rules: Model and Algorithm," *Proc. Fourth ACM Int'l Conf. Knowledge Discovery and Data Mining*, 1998.
[31] M. Roesch, "SNORT—Lightweight Intrusion Detection for Networks," *Proc. USENIX 13th Systems Administration Conf. (LISA '99)*, pp. 229-238, 1999.
[32] F. Tao, F. Murtagh, and M. Farid, "Weighted Association Rule Mining Using Weighted Support and Significance Framework," *Proc. Ninth ACM Int'l Conf. Knowledge Discovery and Data Mining (SIGKDD)*, pp. 661-666, 2003.
[33] G.B. White, E.A. Fisch, and U.W. Pooch, "Cooperating Security Managers: A Peer-Based Intrusion Detection System," *IEEE Network*, pp. 20-23, Jan. 1996.
[34] Y. Xie, H. Kim, D.R. O'Hallaron, M.K. Reiter, and H. Zhang, "Seurat: A Pointillist Approach to Anomaly Detection," *Proc. Seventh Int'l Symp. Recent Advances in Intrusion Detection (RAID '04)*.

Kai Hwang received the PhD degree from the University of California, Berkeley in 1972. He is a professor of electrical engineering and computer science and director of Internet and Grid Research Laboratory at the University of Southern California. Dr. Hwang is the founding editor in chief of the *Journal of Parallel and Distributed Computing* published by Elsevier. He is also on the editorial board of the *IEEE Transactions on Parallel and Distributed Systems*. Presently, he leads a GridSec project, which is supported by the US National Science Foundation, in developing security-binding techniques and distributed defense systems against write-once-read-many and direct denial-of-service attacks for trusted Grid, peer-to-peer, and Internet computing. An IEEE Fellow, he specializes in computer architecture, parallel processing, Internet and wireless security, P2P, Grid, cluster, and distributed computing systems. He has published more than 200 scientific papers and seven books in these areas. More information can be found at http://GridSec.usc.edu/Hwang.html.



Min Cai received the BS and MS degrees in computer science from Southeast University, China in 1998 and 2001, respectively. In 2006, he received the PhD degree in computer science from the University of Southern California. He worked on multimedia networking at IBM Research, Beijing, in 2001. His research interests include peer-to-peer and grid computing, network security, semantic Web, and Web services technologies. He is a member of the IEEE.

Ying Chen received the BS degree in computer science from Huazhong University of Science and Technology, China, in 2001 and the ME degree from Oregon Graduate Institute in 2004. She is presently pursuing the PhD degree in computer engineering at the University of Southern California. Her research interests include Internet security and distributed computing systems. She is a student member of the IEEE.

Min Qin received the BE and ME degrees in computer science from Shanghai Jiaotong University in 1999 and 2002, respectively. He is presently pursuing the PhD degree in the Computer Science Department at the University of Southern California. His current research interests include video streaming and multimedia systems.