Effective Discovery of Attacks Using Entropy of Packet Dynamics

Chan-Kyu Han and Hyoung-Kee Choi, Sungkyunkwan University

Abstract

Network-based attacks are so devastating that they have become major threats to network security. Early yet accurate warning of these attacks is critical for both operators and end users. However, neither speed nor accuracy is easy to achieve because both require effective extraction and interpretation of anomalous patterns from overwhelmingly massive, noisy network traffic. The intrusion detection system presented here is designed to assist in diagnosing and identifying network attacks. This IDS is based on the notion of packet dynamics, rather than packet content, as a way to cope with the increasing complexity of attacks. We employ a concept of entropy to measure time-variant packet dynamics and, further, to extrapolate this entropy to detect network attacks. The entropy of network traffic should vary abruptly once the distinct patterns of packet dynamics embedded in attacks appear. The proposed classifier is evaluated by comparing independent statistics derived from five well-known attacks. Our classifier detects those five attacks with high accuracy and does so in a timely manner.

We have routinely witnessed a range of unusual events in a network. Some of these network-based anomalies are malicious and become major threats to network security. These threats have led to a steady need for development of countermeasures. An intrusion detection system (IDS) identifies malicious anomalies and helps protect a network. Thus, such systems have become an indispensable component of computer networks. Two requirements can summarize the most desirable attributes of an IDS:

- *Responsiveness* — Real-time responsiveness is of supreme concern in an IDS due to imminence of attacks. To achieve this goal in real time, the runtime efficiency of an IDS must be high.
- *Effectiveness* — An IDS must be able to detect a range of anomalies with diverse structures and generate a maximum of true positives and a minimum of false positives.

These two requirements are difficult to satisfy at the same time. A very responsive IDS [1, 2] adopts a relatively simple detection algorithm and suffices in a situation in which real-time alerts are essential. However, such systems are not known for their accuracy.[1] In comparison, a highly effective IDS [3–5] employs relatively complex algorithms that are highly accurate and not subject to false alarms. This type of system tends to take a lot of time before concluding that an attack is underway. In certain situations this relatively slow reaction keeps such a system from being a best choice.

Our goal in this article is to take significant steps toward a system that satisfies both requirements of responsiveness and effectiveness. Furthermore, we want to develop a method for visual inspection of the process of monitoring network traffic. In principle, network traffic contains a wealth of information about normal and abnormal traffic behavior. The recognition of anomalies in the time domain is difficult because they are buried within the other traffic. We seek to transform the time domain into a two-dimensional coordinate. This new coordinate is designed to distinguish anomalies from the mass of network-wide traffic. Lastly, the effectiveness of an IDS relies on an optimum threshold. This threshold constitutes the boundary between detection and false alarms. In general, a globally accepted threshold value does not exist. Such a value should be determined by a network operator and depends on management policy. Our goal in this case is to help operators select the best-fit threshold value.

Our work begins with the observation that entropy varies abruptly when anomalies agitate the system [4, 6]. For instance, the results of port scanning increase the entropy of the destination port, and the infected host would observe a decrease in the entropy of the source Internet Protocol (IP). We find that entropy is a particularly effective metric for determining normal or abnormal system status and distribution. The central question is how to effectively measure entropy by observing the exchange of packets between computer networks. The energy exchange in thermodynamics is analogous to packet dynamics in computer networks [7, 8]. Researchers have concluded that the effect of energy exchange can be measured using entropy. We adapted the entropy computation to the measurement of packet dynamics in a computer network.

We believe that a necessary first step in this adaptation is to understand the packet dynamics of network-wide traffic. For instance, a denial of service (DoS) attack and flash crowds cause destination hosts to concentrate the distribution of traffic on the victim. Network scanning has a dispersed dis-

[1] *Throughout the article, the term* accuracy *implies high true positives and low false positives, unless specified.*

tribution for destination hosts and a bottleneck distribution for destination services. This bottleneck distribution is concentrated on the vulnerable ports. Concentration and dispersion are, respectively, two patterns of packet dynamics frequently perceived in a DoS attack and network scanning. This understanding of packet dynamics is a valuable reference for designing a smart classifier of our own.

We evaluated the proposed algorithm by using five well-known attacks and comparing its results with those of three other popular algorithms. Our results show that our algorithm is the most effective at detection of the four algorithms and is only 3 percent less efficient than the most efficient of the other algorithms. Our objective in this article is not to deliver a fully automatic anomaly diagnostic system. Instead, we seek to demonstrate the utility of new primitives and techniques that a future system could exploit to diagnose attacks.

Related Work

An anomaly-based (also called behavior-based) IDS compares the observed activities of the system with its normal profile and reports as intrusions any divergences from this normal profile. Current approaches to intrusion detection fall into two types, *volume-based* and *feature-based*. Volume-based detection uses deviations in overall traffic volume (e.g., bandwidth, the number of flows) to determine anomalies. This approach detects attacks by identifying abrupt changes in traffic volume and has been successful in isolating large changes in traffic [1, 9].

A question has been raised, however, of whether traffic volume alone is sufficient to identify sophisticated low-rate attacks and to distinguish innocent traffic of unusually large volume from large volumes with malicious intent [4, 10, 11]. As a complementary (not a substitute) metric for traffic volume, a traffic feature-based IDS has been proposed. A feature is a descriptive statistic that can be calculated from one or more packets in traffic such as mean packet length or distribution of IP addresses. It is critical for an IDS to select best-fit features for protection purposes, but it also should minimize the number of features so as to perform effectively [10, 12].

Network anomalies would change the unique normal behavior of a system. This change can be perceived by distributional modifications of the parameters in either traffic volume or traffic features. Much research has been directed toward developing a method to track changes in system condition so as to detect anomalies.

Time-series analyses detect anomalies in traffic volume by exploiting temporal patterns in time-series traffic data. These techniques model an underlying normal profile based on periodic observations of traffic and signal an alarm if a current observation deviates from the normal profile by a certain threshold degree. The exponential weighted moving average (EWMA) and Holt-Winters forecasting, examined in [1], operate in real time, but are prone to be biased along with outlying values. Signal processing techniques such as Fourier transform and Wavelet analysis have been adapted to detect a broad range of volume-based anomalies [2, 13]. This improvement comes as the result of one extra transformation of time-series data into a new coordinate that yields better visibility of stealth attacks. However, this signal processing approach does not depart from the time-series analysis in any radical way.

Entropy in information theory is an especially excellent tool for measurement of the distributional change in system condition [14]. Entropy provides useful descriptions of the long-term behavior of random processes. The key idea is that once abnormal traffic contaminates long-term behavior, the entropy value of the system should immediately reflect this contamination. The authors of [15] presented an entropy-based detection method that mainly targets fast-scanning worms but can be extended to other massive network events. This detection method takes advantage of fluctuations in the entropy values of flow-related metrics. Noteworthy research in [4] diagnoses network-wide anomalies by separating network traffic into normal and anomalous components based on the entropy values of traffic features. A coordinate transformation method, Principal Component Analysis (PCA) [3], is used in the separation. However, this approach operates in a postmortem fashion because of its complex calculation of the traffic matrix [15]. In [16] the authors also use entropy to summarize traffic feature distributions with a goal of classifying and profiling traffic on the backbone of the Internet. Another behavioral IDS was developed in [5] that detects anomalies by comparing the current network traffic against a baseline distribution. The maximum entropy [5, 17] technique provides a flexible approach for estimating the baseline distribution, and relative entropy [16] is used to compare the empirical distribution with the baseline distribution so as to relate outcomes to an anomaly.

The authors of [7] provided a view of a network conversation exchange for a real-time monitoring system. Their algorithm is similar to the proposed algorithm in its use of entropy and in building a physical model to monitor packet exchanges. Our work complements this earlier work by providing another fully designed model to identify a variety of anomalous behaviors, including attacks.

The performance of previous methods of intrusion detection was greatly influenced by the parameter settings used to model normal traffic. As a promising alternative, an unsupervised method was explored; this unsupervised approach alleviated the bias or the failure in training of normal traffic profile by dispensing with such a baseline [9, 11].

Intrusion Detection Using Entropy

We deliberately modeled the entropy computation to the measurement of packet dynamics in a computer network. Well-known classic thermodynamics theory is used to gain an understanding of packet dynamics and further detection of nefarious incidents.

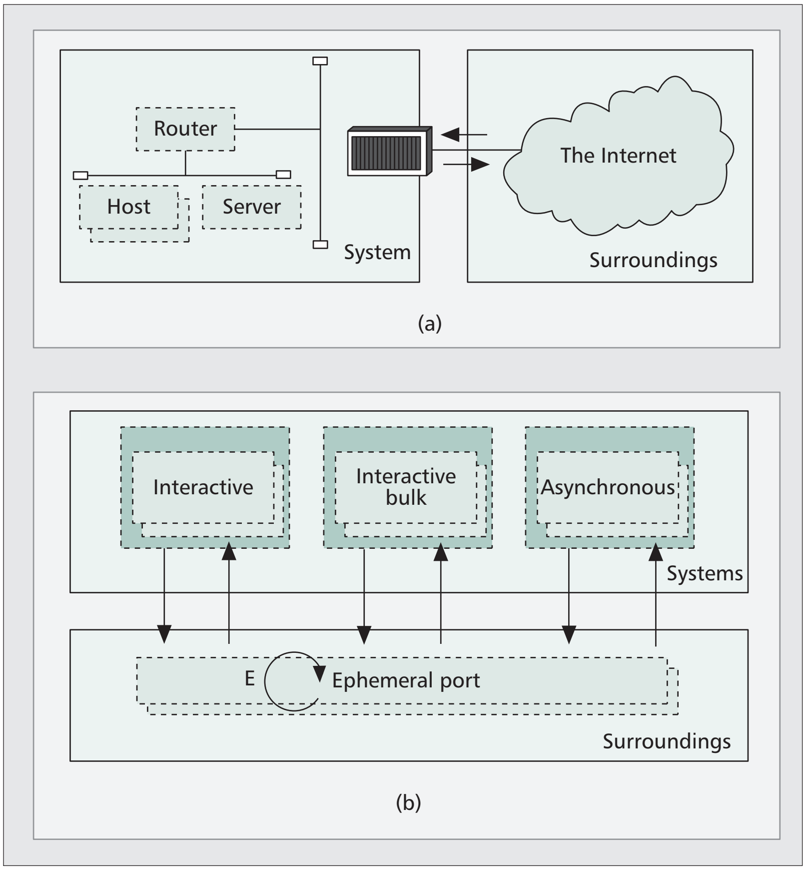
Thermodynamics Theory

Space in thermodynamics theory [18] consists of *systems*. Systems exchange energy constantly with neighboring systems until they reach an overall state of space in equilibrium. A sudden change of state because of abnormal events in the space leads to an abrupt increase of its entropy. This abrupt change of entropy in the face of an abnormal event is a central idea in this article. If one can measure the entropy of space, it is possible to diagnose abnormal and irregular events even though these events are not immediately recognizable by visual inspection.

We attempted to apply a thermodynamics model to a computer network and to build our own model of the network. Two obvious questions about this new model are how to reflect the system and its surroundings on the computer network, and how to measure entropy. The answers are later in this section.

Designation of a System and its Surroundings using Packet Dynamics

Determining the number of spaces in a model and dividing a space into a system and its surroundings depends perhaps on the design goals. This brings us to a design choice of significant implications.



■ Figure 1. *Designation of system(s) and its surroundings in a computer network. Arrows indicate packet flows between (a) inside the network (system) and the Internet (surroundings) in IP address space; (b) ports on three group services (systems) and ephemeral ports (surroundings) in Port space.*

We designated the IP address and the port number as the two spaces in our computer network model. The IP address space is meaningful for tracking the origin and destination of attacks. We are interested in diagnosing an attack that originates from the Internet and has a victim in the protected network as its destination. As shown in Fig. 1a, our model determines that the inside network (i.e., the protected network) is the system and the rest of the Internet is its surroundings.

The Port space is helpful in determining the services targeted by attackers. If our interest was to find the exact service being exploited, we would need 65,536 systems in the model to monitor all 65,536 ports independently, a task that is almost impossible to handle efficiently in the model. A better design in this situation would be to group services into meaningful categories according to our interest and purpose. We used packet dynamics in applications as a basic metric for dividing services into groups. As a result, we have chosen three systems for the Port space: They are Interactive, Interactive Bulk, and Asynchronous, as shown in Fig. 1b. The first service group, Interactive, includes applications in a client-server model in which a single request packet generates a single response packet. Telnet, Secure Shell (SSH), and X-windows are typical applications in this group. Multiple packets are generated as a response to a single request packet in the Interactive Bulk service. Typical applications include the File Transfer Protocol (FTP), Hypertext Transfer Protocol (HTTP), and Simple Mail Transfer Protocol (SMTP). A multimedia streaming service is appropriately included in this group. The Asynchronous group is a bit different from the previous two service groups because applications in this group operate in the peer-to-peer (P2P) model. Popular P2P applications can be included in this service group in which two peers exchange packets asynchronously and bidirectionally. Ports associated with all other services and ephemeral ports make up the surroundings in the Port space (Fig. 1b).

Detecting Attacks using Entropy

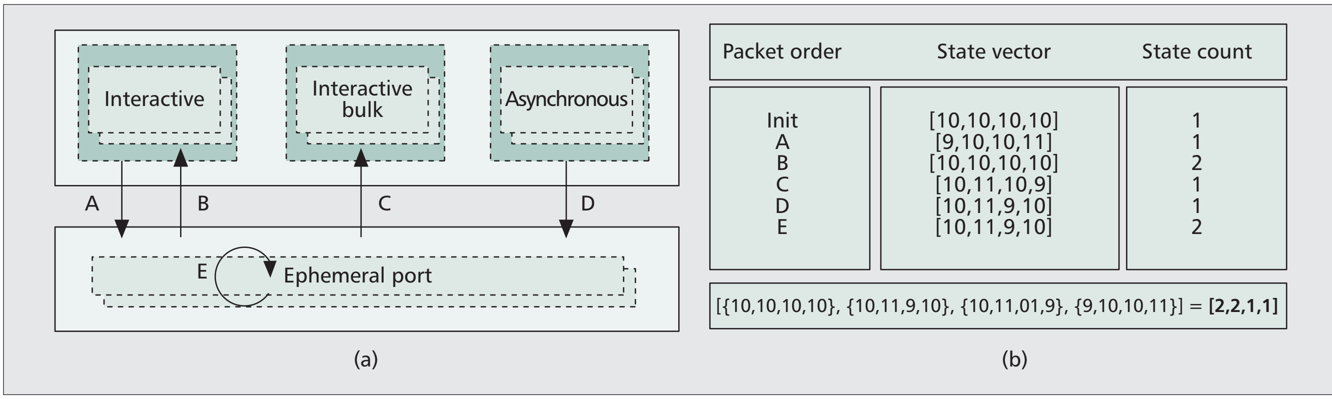
In a network-wide view of the model, the imbalance of network loads causes numerous single-link traffic data to flow continuously between the system and its surroundings. The flow of traffic distributes this imbalance throughout the network. At a certain point, the overall loads within the system and its surroundings reach a point at which they are very nearly balanced. This state is called dynamic equilibrium, and reaching it changes the entropy of the space to quite a low frequency. When an attack is suddenly introduced into the network, the load in one system increases sharply, disrupting this dynamic equilibrium. As a consequence, the entropy of the space fluctuates, and we regard the discontinuity of the entropy in time as a clue to diagnosing illegitimate activities in the network.

A certain number of *tokens* is assigned to each system and its surroundings in the space. The *state vector* (SV) and the *state count* (SC) are two state variables used in the model to record the number of tokens in each system and its surroundings. The SV is a pair of the number of tokens in the system and its surroundings. The SC is a counter of the distinct state vectors. Braces and brackets, respectively, are used to represent the state vector and the group of state counts available at any given time.

As a packet flows between a system and its surroundings, one token from the source system is given to the destination surroundings and vice versa. This movement of tokens updates or creates the state vector and increases the corresponding state count by one. Figure 2 illustrates management of the state vector and calculation of the state count in the Port space. Assume that three systems and one surrounding are given 10 tokens, respectively, in the beginning. At this time, the state vector and the corresponding state count should be $\{10,10,10,10\}_{sv}$ and one. Five packets, A to E, flow in Fig. 2a. Packet A moves from the Interactive group to the surroundings. Immediately after this movement, the state vector changes to $\{9,10,10,11\}_{sv}$ and its corresponding state count is one. Second packet B moves from the surroundings to the Interactive group. The state vector is $\{10,10,10,10\}_{sv}$ again. The state count for $\{10,10,10,10\}_{sv}$ is updated to two. The six state vectors are summarized in Fig. 2b. In the end, there are four distinct state vectors in the illustration. State counts of $\{10,10,10,10\}_{sv}$ and $\{10,11,9,10\}_{sv}$ are two, and state counts of $\{10,11,10,9\}_{sv}$ and $\{9,10,10,11\}_{sv}$ are one. The four state counts corresponding to the four distinct state vectors are succinctly represented by $[2,2,1,1]_{sc}$.

Equation 1 measures the entropy (e_t) in a given time (t). d_i is the state count for i-th state vector. m_t is the number of distinct state vectors in a given time. Equation 2 calculates p_i, which is a relative frequency of d_i in a given time.

$$e_t = -\sum_{i=1}^{m_t} p_i \log p_i \tag{1}$$



■ Figure 2. *Illustration: a) An arrow indicates a packet flow. Five packets, A through E, flow in the Port space. Packet E is self-returning. b) Six state vectors, the corresponding state counts, and a calculation of the state count.*

$$p_i = d_i \Big/ \sum_{i=1}^{m_t} d_i \tag{2}$$

$$\begin{aligned} e_t &= -\sum_{i=1}^{4} p_i \log p_i \\ &= -\left\{\left(2\times\frac{1}{3}\log\frac{1}{3}\right)+\left(2\times\frac{1}{6}\log\frac{1}{6}\right)\right\} \approx 0.5774 \end{aligned} \tag{3}$$

In the illustration shown in Fig. 2, $[d_1, d_2, d_3, d_4]_{sc}$ is $[2, 2, 1, 1]_{sc}$ and $[p_1, p_2, p_3, p_4]_{sc}$ is $[1/3, 1/3, 1/6, 1/6]_{sc}$. The number of distinct state vectors (m_t) is four. Equation 3 shows the calculation of the entropy in this space.

According to Eq. 1, the entropy increases when either the number of distinct state vectors (m_t) increases or the variance of the state count ($V[d_i]$) decreases. Concentration and dispersion patterns in the packet dynamics should increase m_t and decrease $V[d_i]$; that is, the number of packets flowing one way is relatively far greater than the number flowing the other way. This one-way packet movement would create a number of distinct state vectors with the same number of state counts. As a consequence, the numbers of distinct state vectors increases, but the variance of the state count decreases. Bogus requests do not generate immediate responses in general because of silent targets and blockages at the firewall. When a large number of bogus packets flow in the same direction in the network, the packet dynamics in this situation are quite similar to one-way packet movement and as a result, entropy should also increase.

When a large volume of packets flow in the network as a result of network instability or unintentional applications, this rather random packet dynamic does not make a discernible change in either $V[d_i]$ or m_t but only increases the average of d_i. In this case, entropy does not increase. Furthermore, because it has minimal influence, a failure in individual link traffic does not change the entropy.

Accuracy Validation Using an Entropy Graph

We implemented the proposed algorithm using Perl and ran it on real traffic traces available on the Internet. We used four traces containing five malicious attacks: they are the Code Red Worm, Witty Worm, Slammer Worm, DoS, and distributed DoS (DDoS) attacks. Table 1 summarizes these four traces and five attacks.

A desirable feature of a trace used to evaluate a given IDS is that the attacks that trace contains cannot be discerned by visual inspection because the attacks are buried within regular traffic. The MIT Defense Advanced Research Projects Agency (DARPA) trace is such trace. However, the other three traces contain only worms. Our solution was to create three new traces by combining individual worms with regular traffic after meshing the content of the traces. These new traces contain regular traffic volumes about three times larger than the worm attacks in terms of bandwidth and number of packets.

In a given period of one second, we measured the state vectors and the state counts in the IP address and Port space. We calculated the two entropy values for the two spaces at the end of the period and plotted these two entropy values as a pair as shown in Fig. 3. This figure is called the entropy coordinate graph, shortened to entropy graph. The x-axis and the y-axis in Fig. 3 are the entropy values, respectively, of the IP address space and the Port space. Figure 3 shows five entropy graphs. Each entropy graph contains 40 points in total, 20 consecutive points each from the normal and abnormal periods.

In the first two graphs related to the DoS attacks, the entropy values from both spaces increase linearly as soon as the abnormal traffic is introduced into the network. The center points move, respectively, from (3.2, 1.8) to (11.4, 10.9) in the DDoS attack and from (2.2, 2.3) to (5.4, 5.5) in the DoS attack. The rationale behind this abrupt increase in entropy is as follows: DDoS attacks accompany a large number of incoming packets from various sources of IP addresses. These incoming packets are destined for a few vulnerable service ports on a targeted IP address. This concentration of packets would increase the numbers of distinct state vectors in the IP address space as well as in the Port space, and the entropy increases similarly.

Figures 3c through 3e show, respectively, the entropy graphs for Witty Worm, Code Red Worm, and Slammer Worm. The distances between the center points in the three worms are, respectively, 1.7, 1.4, and 1.3. The entropy increases because of the dispersion of packets in the IP address space (i.e., from attacker to target) and the concentration of packets in the Port space (toward vulnerable service ports).

The center points in Figs. 3a and 3b move in about a 45° line. The 45° movement implies that the IP and Port spaces contribute equally to increasing the entropy. The angles of the center points' movement in Figs. 3d and 3e are less than 45°, implying that the IP address space has more control over the entropy. The reason for this inequality is that vulnerable service ports (i.e., 80 for Code Red and 1434 for Slammer) are well-known, and these services belong to Interactive Bulk. We found a large volume of traffic associated with these two port numbers in the normal traffic. Unique patterns of packet dynamics in normal and abnormal traffic cancel each other on these ports.

Data sets	Attacks	Period	Data volume	Year collected
SONY MAWI	Slammer	7m	66 MB	2003
CAIDA	Witty	9m	100 MB	2001
NLANR	Code Red	9m	96 MB	2001
MIT DARPA	DoS	23h 50m	382 MB	1998–1999
MIT DARPA	DDoS	2h 14m	117 MB	2000

■ Table 1. *Five attacks in four traces. These attacks are used to evaluate performance of the proposed algorithm and the other three algorithms.*

	EWMA	Holt-Winters	Proposed	PCA
DoS	0.76	0.78	0.85	0.78
DDoS	0.83	0.85	0.97	0.96
Witty Worm	0.69	0.77	0.98	0.78
Slammer Worm	0.68	0.73	0.87	0.73
Code Red Worm	0.66	0.64	0.78	0.76

■ Table 2. *AUC value of four classifiers for five attacks.*

Performance Evaluation Using a Receiver Operating Characteristic Curve

The proposed scheme is described only for the IP address space. Extension to the Port space should be straightforward.

Algorithm and Metrics

The proposed scheme calculates an entropy value for the IP address space in a given time. An additional state variable records recent k entropy values. Let us denote $e_t(k)$ as representative of the recent k entropy values after calculation of the n-th entropy value. If the difference between $e_t(k) - e_{t-1}(k)$ is greater than the given threshold T, then the proposed scheme is supposed to trigger an alarm, thus signaling a possible attack. The key question to be solved is how to select the representative values of k and T in a manner that is appropriate for an effective and responsive classifier. At least three existing solutions were considered as means to provide the best selection of a representative value: a moving average, a moving median, and a statistical hypothesis test such as a student's t-test. A hypothesis test was excluded because of the relatively high computational cost. We chose the moving median because the median is more immune than a mean value to fluctuating noise. In this way the proposed scheme is quite deliberate in its capability to report real signs of intrusion and to reject false alarms.

In the following discussion we use the k value of 1 for the proposed scheme unless specified. The effects of k on effectiveness and responsiveness, and T values on the system will be discussed later in this section.

Receiver Operating Characteristic Curve

Based on the base-rate fallacy [19], a true positive alone is insufficient for a discussion of accuracy. The following example illustrates why in evaluating accuracy one must consider both true positives (TPs) and false positives (FPs), and balance these two extremes. Assume there are 70 positive instances out of 100 instances. Alice's classifier is so liberal that it predicts all instances will be positive. The true positive rate (TPR) is calculated as 1 (70/70). If we consider only the TPR, her classifier is 100 percent accurate. However, her classifier has predicted as positive 30 instances that are actually negative; these are FPs. The false positive rate (FPR) can compensate for this fallacy. The FPR is calculated as 1 (30/30), meaning that the algorithm is 100 percent inaccurate.

A receiver operating characteristic (ROC) curve is a graphical plot concerning the balance of the TPR versus the FPR because threshold values vary [20]. ROC space is a two-dimensional unit [0,1] in which the TPR is plotted on the y-axis and the FPR is plotted on the x-axis. One point in the ROC curve is drawn from a paired TPR and FPR with one threshold value. Multiple points can be calculated by varying threshold values. These points make up a concave curve, namely the ROC curve.

For a meaningful comparison, it is necessary to normalize the thresholds used in different classifiers so that they have the same absolute value. The ROC makes it meaningful to compare classifiers by modulating to 1 a distance between the upper and lower limits of thresholds. By doing so, the lower limit of threshold is located at (1,1) in the ROC space, and the upper limit is located at (0,0). ROC curves from different classifiers may have a different number of points on the curve. However, all ROC curves must start at (1,1) and end at (0,0). As the threshold increases from the lower to the upper limits, a decision rule tends to change from liberal to conservative.

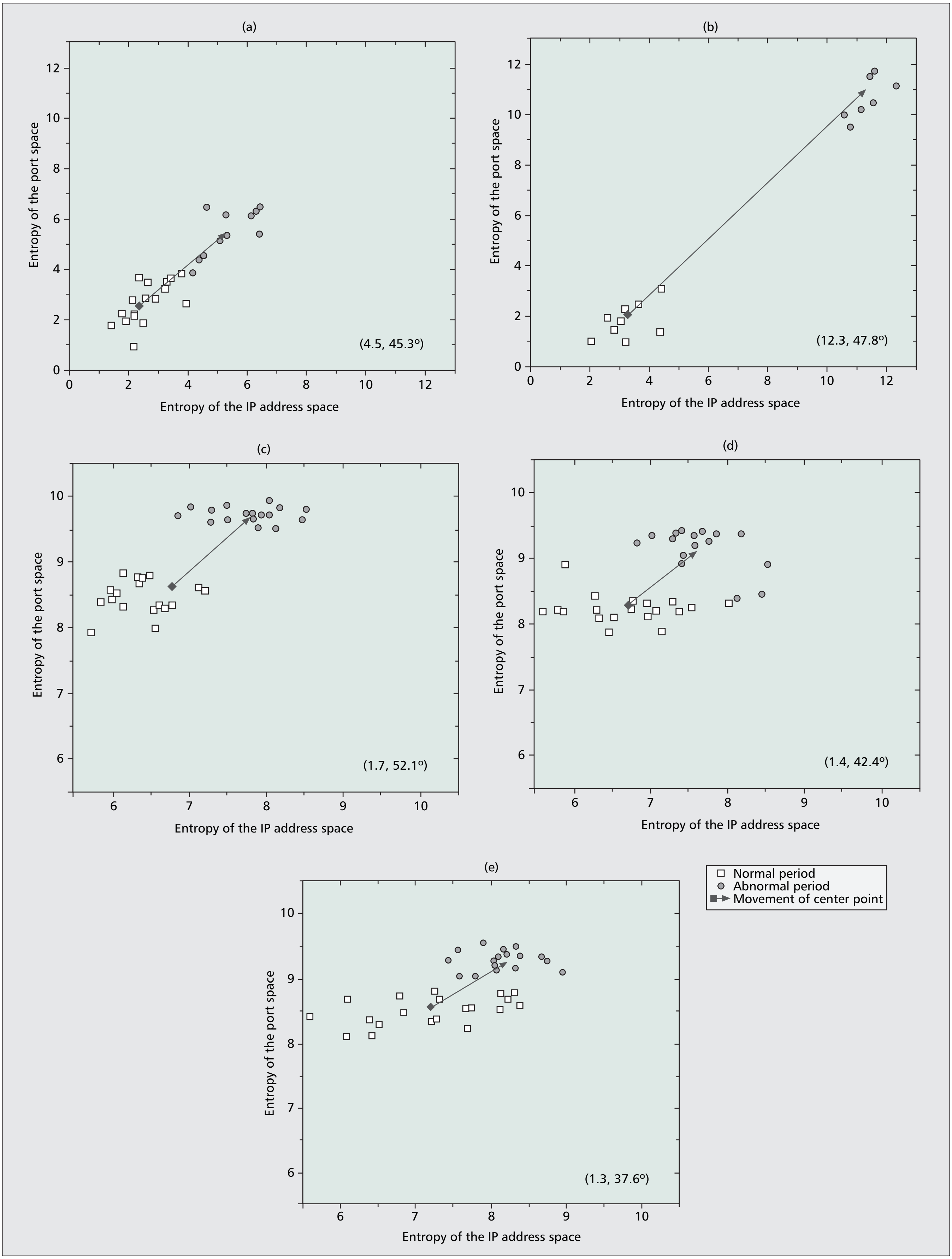
A number of popular classifiers, including ours, use a threshold to predict an instance as positive or negative. Finding an optimum threshold operating point poses a challenge because the threshold value can influence the accuracy of classifiers. However, in this article we do not suggest an optimum threshold for the proposed algorithm because this value is absolutely a designer's choice, and a system administrator should decide the threshold empirically based on administrative policy. We suggest only how a designer or system administrator can find an optimum threshold operating point on the ROC curve. Points that represent the optimum thresholds may lie on the upper concave hull of the ROC curve near (0,1). This is because the point at (0,1) implies perfect classification. It is acceptable to choose a point closest to (0,1) for the optimum threshold. Choosing that point could also maximize the difference between the TPR and the FPR.

Comparison of Effectiveness

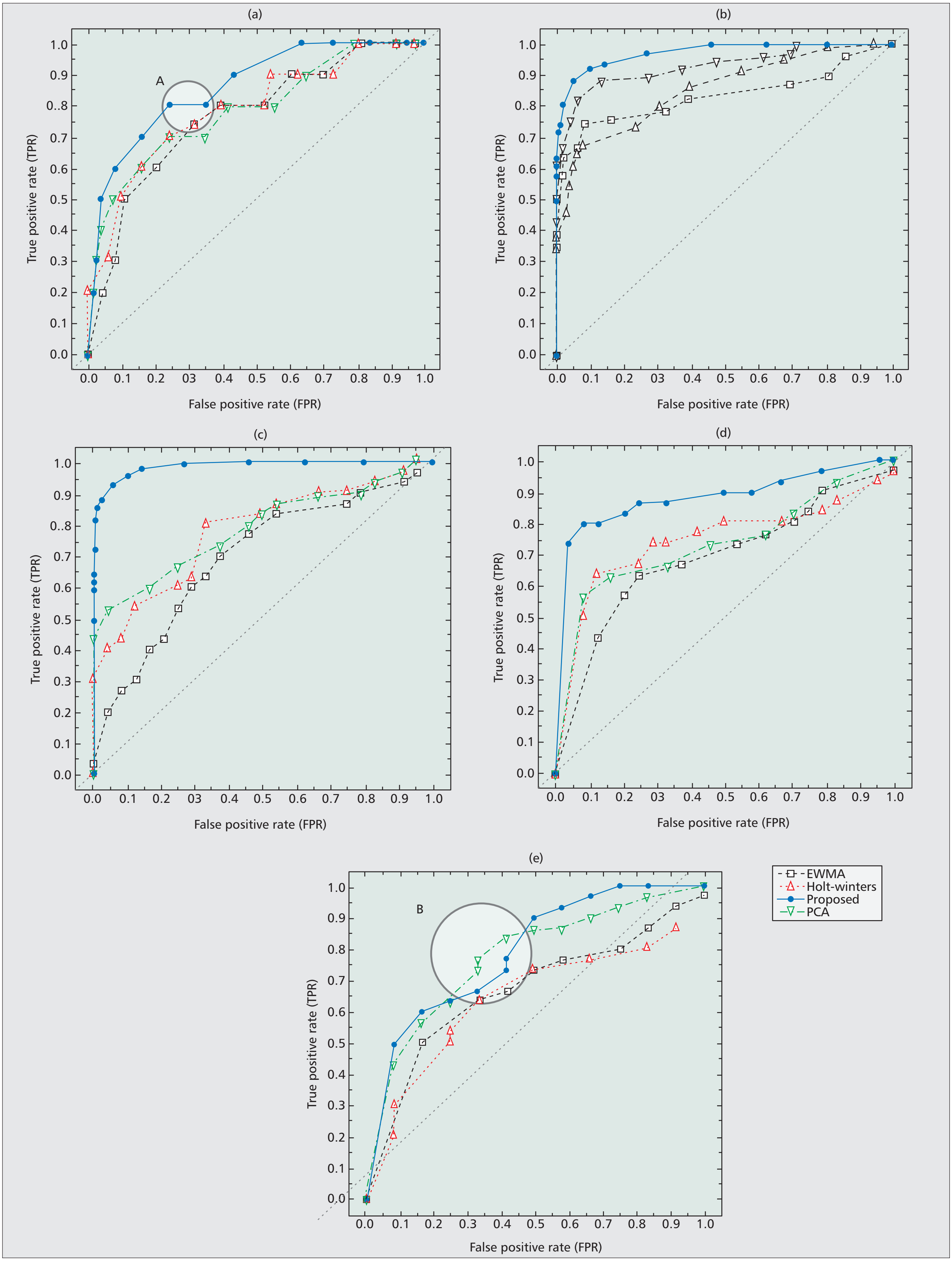
Figure 4 displays an evaluation of the accuracy of the proposed scheme under five attack scenarios. Three well-known systems were selected for use in comparing accuracy: PCA [3], EWMA, and Holt-Winters. Each figure shows four ROC curves for the four detection systems.

An accurate classifier tends to draw the ROC curve toward the upper left corner in the ROC space. The broader an integral area of the ROC curve is drawn, the more accurate a corresponding algorithm is. Let us denote the integral area of the ROC curve as the area under curve (AUC). Table 2 shows the AUCs of the four IDS with respect to the five attacks.

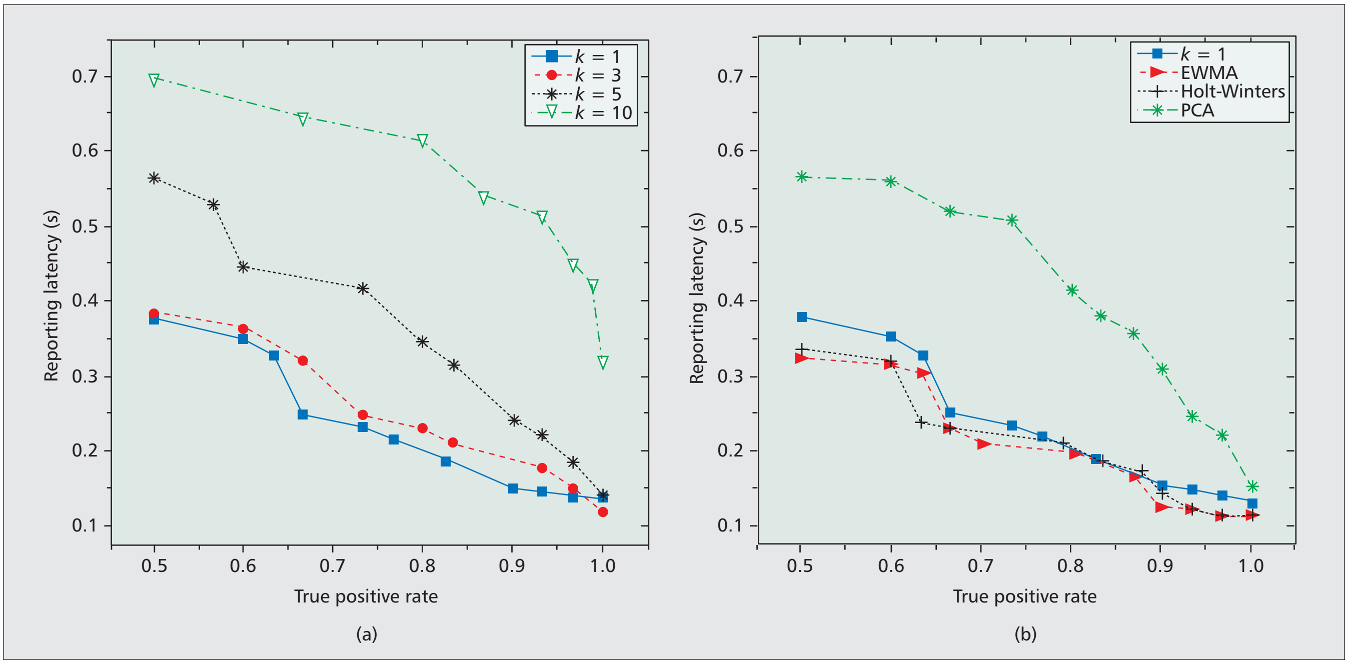
Figures 4a and 4b, respectively, show the ROC curves of the DoS and DDoS attacks. The AUC of proposed algorithm in the DoS attack is 0.85. This is approximately 10 percent greater than the AUC of PCA. In the DDoS attack, our proposed algorithm has a 0.97 AUC, and Holt-Winters has an AUC value of less than 0.85. It is interesting to note the circled area (A) in Fig. 4a, where the ROC curve is rather flat.



■ Figure 3. *Five entropy graphs for five attacks. An arrow in the graph indicates the movement of a center point. The two values in the lower right corner of the graphs are, respectively, the distance and the angle of the center points' movement. a) Denial of service; b) distributed denial of service; c) Witty Worm; d) Code Red Worm; e) Slammer Worm.*



■ Figure 4. *Five ROC spaces for five attacks. ROC curves of four classifiers are drawn in each ROC space; points on the diagonal line in the ROC space are based on a random decision. a) Denial of service; b) distributed denial of service; c) Witty Worm; d) Slammer Worm; e) Code Red Worm.*



■ Figure 5. *Performance comparison in responsiveness. The x-axis is set to a TPR to normalize the different thresholds in four IDS to the same TPR values: a) comparison of reporting latency for different four different k values; b) comparison of reporting latency for four IDSs.*

As the curve moves from right to left, the threshold value increases, simply meaning that the decision rule becomes more conservative. In general, as the rule becomes more conservative, both the TPR and the FPR tend to decrease. However, the TPR remains the same in this case, whereas the FPR decreases from 0.35 to 0.24. This is because changes in the threshold value are not large enough to change the TPR. If this flat curve appears between two points that are candidates for the optimum threshold, one should pick the point closest to the left.

Figures 4c and 4d, respectively, show the ROC curves of the Witty Worm and Slammer Worm. The AUC values of our algorithm in these attacks are 0.98 and 0.87, respectively. However, the rest of the classifiers have AUC values of less than 0.8. Figure 4e shows the ROC curve of the Code Red Worm. The AUC of the proposed algorithm is slightly greater than the one for PCA. The circled area (*B*) in Fig. 4e indicates that in the range between the 0.63 TPR and the 0.85 TPR, PCA performs better than the proposed algorithm.

We also measured the degree to which, if any, different *k* values affected the effectiveness of a classifier. We concluded that classifier effectiveness bore little relationship to the number of recent *k* values. We reached this conclusion because the greater the number of recent *k* values included, the less inclined the classifier was to raise an alarm. By reacting this way, the classifier can simultaneously lower its FPR by rejecting many false positives and its TPR by being reluctant to accept true positives. The outcome from these two rates of lowering FPR and TPR is difficult to predict, and determination of a trend in the *k* value is similarly elusive.

Comparison of Responsiveness

In order to compare responsiveness, we measured an average of the reporting latencies in which the reporting latency is the time difference between the onset of an attack and the onset of detection. Figure 5a shows the reporting latency for four different *k* values. We set the TPR for the *x*-axis with the same rationale as in the ROC curve. An interesting observation in Fig. 5a is that it takes longer to raise alarms as the *k* value increases and the TPR decreases. This is because a conservative IDS takes longer to raise an alarm, and an IDS tends to be more conservative as the *k* value increases and the TPR decreases. Figure 5b compares the reporting latencies of the four detection systems with respect to the TPR. EWMA is the most agile in signaling alarms. PCA takes slightly longer than 35 percent of the response time required by the proposed algorithm at a TPR of 0.9. According to Fig. 5b, our proposed classifier takes on average about 3 percent longer to respond than EWMA, confirming that the two algorithms are comparable in terms of responsiveness.

Conclusion

Detecting network anomalies is an ill defined problem, and most systems available for their detection do not combine effectiveness and responsiveness. They tend to do well in one or the other quality, but not both. We initiated our research in an effort to determine where a detection system could be designed that would satisfy both qualities at the same time. The basic idea is that anomalous traffic is different from benign traffic in a way that can be distinguished by patterns in packet dynamics. To detect malicious attacks, we measured time-variant entropy values in packet dynamics by adapting thermodynamics theory. The experimental results demonstrated that even with small rates of anomalous traffic, our intelligent classifier significantly improved the accuracy of intrusion detection. As a tutorial, this article provides a comprehensive survey and discussion of anomaly-based detection of a network attack. This article also serves as a tutorial introduction to ROC graphs and as a practical guide for using them in research. Future work will include an analysis using a variety of the attacks available today. Furthermore, it is interesting to see how the proposed algorithm performs in comparison with commercialized products used in real networks.

References

[1] J. D. Brutlag, "Aberrant Behavior Detection in Time Series for Network Monitoring," *USENIX LISA*, Dec. 2000.

[2] S. S. Kim and A. L. N. Reddy, "Statistical Techniques for Detecting Traffic Anomalies Through Packet Header Data," *IEEE/ACM Trans. Net.*, vol. 16, no. 3, June 2008.

[3] A. Lakhina, M. Crovella, and C. Diot, "Diagnosing Network-Wide Traffic Anomalies," *ACM SIGCOMM*, Aug. 2004.

[4] A. Lakhina, M. Crovella, and C. Diot, "Mining Anomalies Using Traffic Feature Distributions," *ACM SIGCOMM*, Aug. 2005.
[5] Y. Gu, A. McCallum, and D. Towsley, "Detecting Anomalies in Network Traffic Using Maximum Entropy Estimation," *ACM SIGCOMM IMC,* Oct. 2005.
[6] G. Androulidakis, V. Chatzigiannakis, and S. Papavassiliou, "Network Anomaly Detection and Classification via Opportunistic Sampling," *IEEE Network*, vol. 23, no. 1, Jan./Feb. 2009.
[7] J. Zachary, J. McEachen, and D. Ettlich, "Conversation Exchange Dynamics for Real-Time Network Monitoring and Anomaly Detection," *IEEE Int'l. Info. Assurance Wksp.*, Apr. 2004.
[8] M. Burgess, "Thermal, Nonequilibrium Phase Space for Networked Computers," *American Physical Society*, vol. G2, no. 2, Aug. 2000.
[9] Y. Zhang *et al.*, "Network Anomography," ACM SIGCOMM IMC, Oct. 2005.
[10] G. Nychis *et al.*, "An Empirical Evaluation of Entropy-Based Traffic Anomaly Detection," *ACM SIGCOMM IMC*, Oct. 2008.
[11] H. J. Kim, J. C. Na, and J. S. Jang, "Network Traffic Anomaly Detection Based on Ratio and Volume Analysis," *Int'l. J. Comp. Science and Net. Security*, vol. 6, no. 5B, May 2006.
[12] N. Ye *et al.*, "Probabilistic Techniques for Intrusion Detection Based on Computer Audit Data," *IEEE Trans. Sys. Man, and Cybernetics Part A: Sys. And Humans*, vol. 31, no. 4, July 2001.
[13] P. Barford *et al.*, "A Signal Analysis of Network Traffic Anomalies," *ACM SIGCOMM IMC*, Nov. 2002.
[14] W. Lee and D. Xiang, "Information-Theoretic Measures for Anomaly Detection," *IEEE Symp. Security and Privacy*, Mar. 2001.
[15] A. Wagner and B. Plattner, "Entropy Based Worm and Anomaly Detection in Fast IP Networks," *IEEE WETICE*, June 2005.
[16] K. Xu, Z.-L. Zhang, and S. Bhattacharyya, "Profiling Internet Backbone Traffic: Behavior Models and Applications," *ACM SIGCOMM*, Aug. 2005.
[17] N. G. Duffield *et al.*, "Entropy of ATM Traffic Streams: A Tool for Estimating QoS Parameters," *IEEE JSAC*, vol. 13, no. 6, Aug. 1995.
[18] Y. A. Cengel and M. A. Boles, *Thermodynamics: An Engineering Approach*, 5th ed., McGraw-Hill Science, 2005.
[19] S. Axelsson, "The Base-Rate Fallacy and the Difficulty of Intrusion Detection," *ACM Trans. Info. and Sys. Security*, vol. 3, no. 3, Aug. 2000.
[20] T. Fawcett, "ROC Graphs: Notes and Practical Considerations for Researchers," Tech. Rep. HPL-2003-4; http://www.hpl.hp.com/techreports/2003/HPL-2003-4.html

Additional Reading

[1] K. Yoshida, "Entropy Based Intrusion Detection," *IEEE PACRIM*, Aug. 2003.

Biographies

CHAN-KYU HAN (hedwig@ece.skku.ac.kr) received her B.S. and M.S. degree in computer engineering from Sungkyunkwan University in 2006 and 2008, respectively. She is currently working toward her Ph.D. degree in mobile systems engineering at the same university. Her research interests include wireless networking, mobile computing, and network security.

HYOUNG-KEE CHOI (hkchoi@ece.skku.ac.kr) received a Ph.D. degree in electrical and computer engineering from Georgia Institute of Technology in 2001. He is an assistant professor and a director of the Education Center for Mobile Communications in Sungkyunkwan University, Korea. He joined Lancope in 2001 and remained until 2004, where he guided and contributed to research in Internet security. His research interests span network security and Internet traffic modeling. He serves as an Associate Editor for *ACM Transactions on Internet Technology*.