Characterizing Flow-Level Traffic Behavior with Entropy Spaces for Anomaly Detection

Chapter · January 2013

DOI: 10.1201/b14574-15

CITATION
1

READS
83

4 authors:

Pablo Velarde Alvarado
Universidad Autónoma de Nayarit
27 PUBLICATIONS **43** CITATIONS
SEE PROFILE

Cesar Vargas-Rosales
Tecnológico de Monterrey
100 PUBLICATIONS **431** CITATIONS
SEE PROFILE

Homero Toral-Cruz
University of Quintana Roo
40 PUBLICATIONS **119** CITATIONS
SEE PROFILE

Julio Ramirez-Pacheco
Center for Research and Advanced Studies of the National Polytechnic Institute
46 PUBLICATIONS **98** CITATIONS
SEE PROFILE

Chapter 11

Characterizing Flow-Level Traffic Behavior with Entropy Spaces for Anomaly Detection

Pablo Velarde-Alvarado, Cesar Vargas-Rosales, Homero Toral-Cruz, Julio Ramirez-Pacheco, and Raul Hernandez-Aquino

Contents

11.1 Introduction

Cyber-attacks carried out directly against networking infrastructure are becoming more and more prevalent. Both the number and the complexity of attacks have increased dramatically in the last years. At the same time, the surges of network security threats have the potential to significantly impede productivity, disrupt business and operations, and result in information and economic losses.

Typical perimeter defenses, such as firewalls, conventional network intrusion detection systems (NIDS), application proxies, and virtual private network servers, have become important components of a security infrastructure. However, they do not provide a sufficient level of protection against sophisticated attacks. Besides, the network has a dynamic behavior for which more elaborate and complicated security procedures need to be derived; this is emphasized when different network segments work with different protocols and services. In addition, when a new service is introduced, or the number of users changes, network dynamics is altered, and this produces modifications to which the security tasks need to adapt in order to protect the network. To overcome these limitations, one promising approach makes use of entropy to obtain knowledge of the structure and composition of traffic, which is summarized by behavioral traffic profiles. This approach is being proposed as a good candidate in traffic characterization for the development of a new generation of NIDS. Some of today's major challenges in NIDS design are to achieve sensitivity in the detection of sophisticated attacks, to successfully obtain an early detection, and to minimize false-positive and false-negative rates, among others.

In this chapter, we present methodologies based on information theory and present entropy spaces and pattern recognition (PR) techniques for decision-making processes in order to detect anomalies in traffic traces. We introduce the entropy space technique together with the excess point methodology, and both help us characterize special features by using probability density function (PDF) estimations. In some cases, a Gaussian distribution is used to show that it is close in the description of distances to a central reference point in the entropy spaces that allows us to classify anomalies for different time slots in a traffic trace.

11.2 Background

In this section, we review the basic concepts of intrusion detection systems (IDSs), entropy, principal component analysis (PCA), and PR that are relevant to the approach we are proposing.

11.2.1 Intrusion Detection Systems

The attacks launched against hosts or networks are identified as intrusions. An intrusion to a host or network is defined in [1] as an unauthorized attempt or achievement to access, alter, render unavailable, or destroy information on a system or the system itself. Individuals regarded as intruders include both those without the proper authorization for the use of resources and those who abuse their authority (insiders) [2]. Heady et al. [3] also define intrusion as any set of actions that attempt to compromise the confidentiality, integrity, or availability of a computational resource. Throughout this chapter, we will use the terms intrusion and attack interchangeably. The definitions for intrusion detection and IDS used in this chapter are as follows: Intrusion detection is the problem of identifying an attempt to compromise the confidentiality, integrity, or availability of a computational resource or information. IDS is a computer system (possibly a

combination of hardware and software) that automates the intrusion detection process. Note that intrusion detection is generally decoupled from the response to the intrusion. Typically, an IDS only alerts the information technology (IT) manager when an attack or a potential attack condition is recognized. The IT manager then takes necessary and appropriate actions to minimize (preferably avoid) any actual damage. Basically, IDSs can be classified into two types: host IDS (HIDS) and NIDS. A HIDS operates on individual devices or hosts in the system to monitor and analyze all the incoming and outgoing traffic on the device only; such systems are not addressed in this chapter. On the other hand, NIDS sensors are typically connected to a network through the use of switched port analyzer connections or taps placed at strategic locations having the ability to monitor all traffic on a network, and its position allows them to detect attacks on any host in the network they protect. When intrusive activity occurs, the NIDS generates an alarm to let the network administrator know that the network is possibly under attack. A passive NIDS is not proactive in that it does not take any action by itself to prevent an intrusion from happening once detected. A reactive system, also known as an intrusion prevention system (IPS), autoresponds to threats, for example, reconfigures access control list on routers and firewalls dynamically, closes network connections, kills processes, and resets transmission control protocol (TCP) connections. Intrusion prevention technology is considered by some to be a logical extension of intrusion detection technology [4,5]. NIDSs play a fundamental role in any enterprise's multilayered defense-in-depth strategy as they constitute, besides firewalls and cryptography, the third security layer. NIDSs can be subdivided into two categories with respect to the implemented detection technique, namely, misuse-based NIDS, also sometimes referred to as signature-based NIDS (S-NIDS), and behavior-based NIDS, also known as anomaly-based NIDS (A-NIDS). S-NIDSs are relying on pattern matching techniques; they monitor packets and compare with preconfigured and predetermined attack patterns known as signatures [6]. According to [7], these signatures can be classified into two main categories based on the information they examine to detect subversive incidents: content-based signatures and context-based signatures. Content-based signatures are triggered by data contained in packet payloads, and context-based signatures are triggered by data contained in packet headers. These signatures are based on known vulnerabilities such as those published by Common Vulnerabilities and Exposures [8]. In terms of advantages, these systems are very accurate at detecting known attacks and tend to produce few false positives (i.e., incorrectly flagging an event as malicious when it is legitimate). There are two important drawbacks to signature-based IDS solutions. First, there will be a lag between the new threat discovered and the signature being applied in NIDS for detecting the threat. During this lag time, the NIDS will be unable to identify the threat. Second, on large-scale networks, where the amount of data passing through the network is extensive, S-NIDSs have performance and scalability problems. Most widely deployed NIDS are signature based; an example is Snort [9], an open-source NIDS or IPS capable of performing real-time traffic analysis to detect possible attacks, instances of buffer overflows, stealth port scans, etc. With respect to anomaly-based systems, they must create a profile that describes the normal behavior of certain traffic features based on information gathered during a training period. These profiles are then used as a baseline to define normal activity. If any network activity deviates too far from this baseline, for example, when an attack occurs, then the activity generates an alarm. Anything that does not fall within the boundaries of this normal use is flagged as an anomaly that may be related to a possible intrusion attempt. Anomaly is a pattern in the data that does not conform to the expected behavior, also referred to as outliers, exceptions, peculiarities, and surprises [10]. Anomaly-based systems have the advantage over signature-based systems in the ability to detect never-seen attacks (zero-day attacks) from the moment they appear. Nevertheless, if the profile is not defined carefully, there will be

Should this be changed to 'their' (referring to the NIDS sensors)?

lots of false alarms, and the detection system will suffer from degraded performance. Another limitation that is well known in the research community is the difficulty of obtaining enough high-quality training data. Traditional anomaly-based systems use a volumetric approach to detect anomalies by monitoring the amount of traffic in the network. However, volumetric intrusion detection has become ineffective because the behavior patterns of the attacks have evolved; these attacks attempt to avoid detection by low-rate infection techniques. This issue has been addressed with the introduction of solutions based on robust traffic features that are sensible to low-rate attacks [11]. The traffic features are extracted from the packet headers, payload, or a combination of both. Lakhina et al. [12] applied an information-theoretic analysis on feature distributions of IP addresses and service ports using entropy to quantify the changes in those traffic distributions. Some examples of anomaly-based systems include NETAD [13], LERAD [14], and PAYL [15].

Please provide the spelled-out form of these acronyms.

Network anomaly detection has been approached using various techniques such as artificial intelligence [16], machine learning [17], statistical signal processing [18], and information theory [19]. A malicious activity has behavior patterns that can be distinguished from a legitimate one. For example, scanning is a typical process that an intruder often uses to gather knowledge from the target. These probes are atypical of a legitimate user and alter the natural diversity of network traffic. Information theory metrics have been used to detect specific types of malicious traffic such as port scanning attacks, distributed denial-of-service (DDoS) attacks, and network worms.

Please check orphan section head 11.2.1.1.

11.2.1.1 Data Preparation for Traffic Profiling

Supervised detection builds specifications of normal behavior from training data, and it is clear that the effectiveness of the detection critically relies on the quality and completeness of the training data. Therefore, one requirement imposed on a training data set is that it should be attack-free, that is, it should not contain malicious activity that would induce the resulting models to consider malicious behavior as normal. The cleaning process of the training data can be performed manually; however, this time-consuming alternative may be prohibitive for a large amount of raw traffic data. In addition, this cleaning process has to be performed periodically because the system needs to be updated regularly to adapt to network changes. Manual inspection can be assisted by an S-NIDS, which can preprocess the training set to discover known attacks (e.g., web scanners and old exploits).

11.2.2 Entropy

Entropy has been used by various disciplines, one of which is physics. Mathematically, entropy measures the concentration of a PDF, so that the entropy is large when the PDF is not concentrated. The entropy therefore has a similar role as the variance. However, in contrast to the variance, the entropy is completely independent of the expected value and of any norm structure on the sample space [20]. Commonly, it is necessary to describe events in terms of its complexity, disorder, randomness, diversity, or uncertainty; it is in these cases where the entropy becomes a fundamental tool to quantify these types of properties in the data. Shannon [21], in a well-known publication of 1948, proposed a mathematical tool to measure the entropy and it was the origin of information theory. In information theory, entropy is a function that measures the information content of a data set or, equivalently, measures the uncertainty of a random variable.

Consider a discrete random variable X that takes values from an alphabet $\chi = \{x_1, \cdots, x_M\}$ with cardinality M, where a realization is denoted by x_k. Let $p_x(x_k)$ denote the discrete PDF or

probability mass function of X so that $p_x(x_k)$ is the probability that realization x_k occurs. Shannon entropy for a random variable X is defined by its expected information content:

$$H(X) = E\left[I_X\right] = -\sum_{k=1}^{M} p_X\left(x_k\right)\log_2 p_X\left(x_k\right), \tag{11.1}$$

where $I_X = -\log_2 p_x(x_k)$ is the information content of a particular realization x_k. Because we use the base 2 logarithm, entropy has units of bits (natural logarithms give nats, and base 10 logs give bans). In this chapter, base 2 logarithms are used and $0 \log_2 0$ is defined to be 0. The entropy attains its minimum value $H = 0$, when all the realizations in a sample space are the same, that is, the state is pure, which is when $p_X(x_k) = 1$ for some $k = 1, \ldots, M$. On the other hand, the maximum value $H = \log_2 M$ is related to realizations that are equally probable corresponding to a totally random state, that is, $p_X(x_k) = 1/M$, $\forall k$.

Probability distributions $p_X(x_k) = p_k$ can be approximated by relative frequencies from a training data set of size N. These are the naive empirical-frequency estimates of the probabilities denoted by $\hat{p}_k = n_k / N$, where n_k counts the number of times each x_k appears in the data set. Replacing the probabilities p_k by $\hat{p}_k$ in Equation 11.1, the Shannon entropy of a data set can be estimated as follows:

$$\hat{H}(X) = -\sum_{k=1}^{M} \hat{p}_k \log_2 \hat{p}_k = -\sum_{k=1}^{M} \frac{n_k}{N} \log_2 \left(\frac{n_k}{N}\right). \tag{11.2}$$

The entropy estimator underestimates the actual entropy for small data sets [22], but the data sets used in traffic analysis are large enough to make a small bias that vanishes in the limit.

Entropy has been extensively studied for anomaly detection and prevention [12,23–25]. Researchers have suggested the use of entropy as a succinct means of summarizing traffic distributions for different applications, in particular, in anomaly detection and in fine-grained traffic analysis and classification. With respect to anomaly detection, the use of entropy for tracking changes in traffic distributions provides two significant benefits. First, the use of entropy can increase the sensitivity of detection to uncover anomalous incidents that may not manifest as volume anomalies. Second, using such traffic features provides additional diagnostic information into the nature of the anomalous incidents (e.g., making distinction among worms, DDoS attacks, and scans), which is not available from just volume-based anomaly detection [26]. When an attack takes place, it is likely to produce some kind of unusual effect on the network traffic (illegitimate traffic); on the other hand, if the data flow is licit, there will be no unusual effect on the traffic (legitimate traffic). Entropy measures allow identifying changes in distance and divergence between the probability densities related to these two types of traffic. Entropy measure can be used to detect such anomalies in the traffic monitored; for example, in Figures 11.1 and 11.2, entropy for the traffic features of IP source address (srcIP), IP destination address (dstIP), source port (srcPrt), and destination port (dstPrt) is shown. These entropy values are obtained from the packets being monitored as a function of time. In both figures, the Blaster worm attack was introduced, and those entropy values are the result of the Blaster propagation. This analysis is at the packet level, that is, a promiscuous method where the traffic features are extracted from each and every packet that traverses the router where monitoring takes place. The monitoring is carried out at fixed time slot with durations of 60 s each, or each time slot corresponds to 1 min of traffic

monitored. In Figure 11.1, one can see that, at the first 70 min, the entropy values vary within normal or typical levels and have normal randomness; after the 70th minute, when the Blaster worm was introduced, the entropy levels start incrementing significantly in a very short time for the srcIP and the dstIP features. During the Blaster worm propagation experiment, there can be seen two points in the 38 min that last such attack. First, the entropy value according to the specific traffic feature is stabilized to a high or low value with almost no randomness in such values; second, the entropy for the features corresponding to the sources (either port or address) and those of the destinations has a negative linear dependency that is different in sign from that of normal traffic. These effects are also caused by the worm dynamics, that is, variation in srcIP is small, but a high variation of dstIP is attempting to infect computers.

Figure 11.2 shows the entropy values for the four features of the traffic that correspond to the time slots 1 through 10 of Figure 11.1. This close view of Figure 11.1 permits the identification of another interesting property of the entropy-based traffic analysis that is not seen easily. In time slots 3 and 4, an anomalous behavior can be detected in the features corresponding to the srcIP and the dstIP because there is negative linear dependence; that is, when one increases in value, the other decreases in value and vice versa. A closer view at the traffic packets in such time slots allowed seeing that there was a scan attack directed toward the ports of the proxy server monitoring the network. The interesting point of this scan attack is that there was an attack that is within normal entropy levels and that the entropy levels should not be considered the only procedure to detect anomalies.

Figure 11.3 shows traffic as a function of time. Traffic is measured in units of packets per time slot, that is, number of packets per minute. From the point of view of traffic volume, it can be

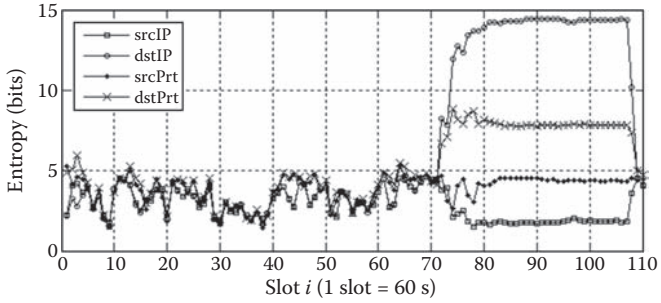

Figure 11.1 Entropy for the traffic features for a Blaster worm attack.

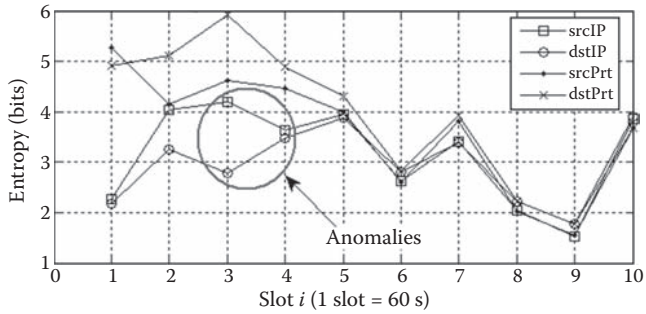

Figure 11.2 Entropy for the traffic features: a close view to slots 1 through 10 of Figure 11.1.

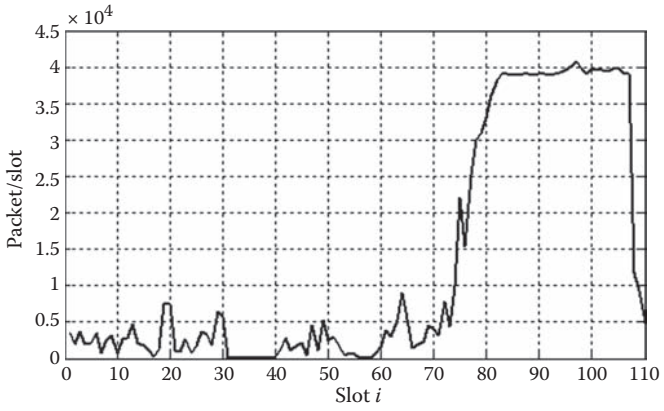

Figure 11.3 Traffic volume for the network with the Blaster attack experiments.

seen that the Blaster worm attack can be detected, but the attack detected by the negative linear dependency of the entropy values, as shown in Figure 11.2, cannot be detected (time slots 3 and 4) because the traffic volume in such time slots is within normal levels.

11.2.3 Pattern Recognition

Network intrusion detection (NID) is essentially a PR problem in which network traffic patterns are classified as either normal or abnormal. This section presents a brief introduction to the basic concepts of PR. For a more extensive treatment, see, for example, [27–29].

PR is the scientific discipline of machine learning (or artificial intelligence) that aims at classifying data (patterns) into a number of categories or classes [30]. To understand how PR works, it is needed to describe some important concepts.

Pattern: It is a representation of a given object, observation, or data item. Examples of patterns are measurements of a voltage signal in electrocardiogram, a pixel in a two-dimensional (2D) image, fingerprints of different persons, and a cluster of traffic flows.
Features: It is a set of data or attributes obtained from measurements of patterns. These features are useful for their further characterization.
Feature selection: It is a process to determine which set of features is the most appropriate to describe a set of patterns according to their relevance. Feature selection reduces dimensionality by selecting a subset of original variables.
Feature extraction: It is a process to reduce the amount of redundant data given in a set of features; such a set should be transformed to obtain its reduced characteristic representation. This is done by dimensionality reduction applying (linear or nonlinear) projection of p-dimensional vector onto d-dimensional vector ($d < p$).
Feature vector: It is the reduced characteristic representation of a given set of features. It stores the relevant data of such features, and it is helpful for its further classification. For its easy management, these relevant data are represented in a vector form as shown in Equation 11.3:

$$x = [x_1, x_2, \cdots, x_p]^T \tag{11.3}$$

where ^{T} denotes transposition and x_i are measurements of the features of an object. Each of the feature vectors identifies uniquely a single pattern.

Feature space: It is a given space where each pattern is a point in the space. Each feature vector is represented as a coordinate on such space. Thus, each axis represents each feature, and the dimension p should be equal to the number of used features.

Class label or class: A class can be viewed as a source of patterns whose distribution in the feature space is governed by a PDF specific to the class. For example, $\{w_1, w_2,\cdots,w_c\}$ represents the finite set of c classes of nature.

Training set is a set of patterns already having been classified.

A PR system is an automatic system that aims at classifying the input pattern into a specific class. It proceeds into two successive tasks: (1) the analysis (or description) that extracts the characteristics from the pattern being studied and (2) the classification (or recognition) that enables us to recognize an object (or a pattern) by using some characteristics derived from the first task [30].

A PR system's main task is to classify patterns in a set of classes, where these classes are defined by a human who designs a system (supervised learning) or are learnt and assigned by using pattern similarity (unsupervised learning).

In this work, in particular, a statistical PR approach is implemented. By using this approach, each pattern is represented by a set of features or measurements in a p-space. An adequate feature selection allows establishing disjoint regions. Thus, each class may be distinguished accurately as a different class. Such disjoint regions are obtained by using training sets, and each disjoint region represents each class. By using the statistical approach, each decision region is determined by the PDF of the patterns belonging to each class.

11.2.4 Principal Component Analysis

PCA is an eigenvector method designed to model linear variation in high-dimensional data. PCA performs dimensionality reduction by projecting the original p-dimensional data onto the d-dimensional linear subspace ($d < p$) spanned by performing a covariance analysis between factors [31,32]. The goal of PCA is to reduce the dimensionality of the data while retaining as much as possible the variation present in the original data set. In pattern classification tasks, significant performance improvements can be achieved by a mapping between the original feature space to a lower-dimensional feature space according to some optimality criteria. In our proposed architecture, classification of traffic patterns for intrusion detection is performed through the features measured or extracted from flow clusters; this information is passed to a stage of feature selection implemented by PCA. Based on the above, it is possible to build more effective data analyses on the reduced-dimensional space: classification, clustering, and PR.

PCA problem can be described as follows: let $\boldsymbol{X} \in \mathbb{R}^{p\times N}$ be the original data of N observations on p correlated variables, that is, $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2,\cdots,\boldsymbol{x}_p]$, where $\boldsymbol{x}_i = [x_{1i}, x_{2i},\cdots,x_{pi}]^{T}$, $i = 1, 2, \ldots, N$, should be transformed into $\boldsymbol{Y} = \left[\boldsymbol{y}_1, \boldsymbol{y}_2,\cdots,\boldsymbol{y}_p\right] \in \mathbb{R}^{p\times N}$ of p uncorrelated variables and organized in decreasing order according to its variance. The covariance matrix of the random vector $\boldsymbol{X}$ is formally defined by $\text{cov}[\boldsymbol{X}] = \sum_X = E\left[(\boldsymbol{X}-\boldsymbol{\mu})(\boldsymbol{X}-\boldsymbol{\mu})^T\right] \in \mathbb{R}^{p\times p}$, where $\boldsymbol{\mu} = E[\boldsymbol{X}]$. Let $\left\{\boldsymbol{v}_i \in \mathbb{R}^{p\times 1} : i = 1,2,\cdots,p\right\}$ denote the eigenvectors of $\sum_X$, and let $\left\{\lambda_i \in \mathbb{R} : i = 1,2,\cdots,p\right\}$ denote the corresponding eigenvalues arranged in descending order. Furthermore, let $\boldsymbol{V} = [\boldsymbol{v}_1, \boldsymbol{v}_2,\cdots,\boldsymbol{v}_p]$ be the $p \times p$ orthogonal matrix constructed from the eigenvectors and $\Lambda = \text{diag}\{\lambda_1, \lambda_2,\cdots,\lambda_p\}$ be the $p \times p$ diagonal matrix constructed from the eigenvalues. The principal components are the

Please check if the edit made here is OK.

eigenvectors with the d largest eigenvalues, that is, $\left[\sum_X \boldsymbol{v}_i = \lambda_i \boldsymbol{v}_i\right] \in \mathbb{R}^{p\times d}$ for $i = 1, 2, \cdots, d$. Finally, $\boldsymbol{Y}$ is given by the linear combination of optimally weighted variables $[x_{1i}, x_{2i},\cdots,x_{pi}]$:

$$Y = \sum_{i=1}^{N}\left(x_{1i}v_1 + x_{2i}v_2 + \cdots + x_{pi}v_p\right) = VX \tag{11.4}$$

11.3 Method of Entropy Spaces

In order to make use of the method of entropy spaces (MES), one needs to generate as input the flow-level network traffic and perform an abstraction of the data to obtain a three-dimensional (3D) representation by point cloud data. The coordinates of each point in the 3D space generated represent the naive entropy estimates of three features of a cluster of flows for a given cluster key. Such generated 3D spaces under typical traffic are used to define a behavior profile of network traffic.

The input that needs to be generated for the application of MES starts with the definition of a trace χ. A traffic trace is obtained where packets are captured from the network where anomaly detection wants to be applied, and then such a trace is divided into m nonoverlapping traffic slots with maximum time duration of t_d seconds, which is chosen empirically according to sensitivity analysis. In any slot i, K_i flows of packets are generated. The flows to be generated are defined under a five-tuple and interflow gap of 60 s. A five-tuple is a stream of packets having the same four flow fields and protocol, that is, the same source and destination IP addresses, same source and destination port numbers, and same protocol number [33]. The four flow r-fields are labeled as follows: $r = 1$ for source IP address (srcIP), $r = 2$ for destination IP (dstIP), $r = 3$ for source port (srcPrt), and $r = 4$ for destination port (dstPrt). After that, the set of flows are clustered for each slot i; in other words, one needs to fix an r-field, now named cluster key (CK-r). For a given i-slot, each cluster is formed by containing flows under the same r-field but leaving free the rest of the r-fields. As an example, for CK-$r = 1$ or CK srcIP, each cluster is formed with all flows that possess the same source IP address regardless of the value of the rest of the r-fields ($r = 2, 3, 4$). These fields are denoted as free dimensions. It is clear in this example that the complete number of flow clusters depends on the cardinality of the alphabet $\left|\mathbb{A}_i^{r=1}\right|$, where $\mathbb{A}_i^{r=1}$ is the set of IP source addresses shown in the traffic trace within time slot i.

Taking the aforementioned example, entropy estimation for each j-cluster flow that belongs to the CK srcIP is represented as a 3D Euclidean point, denoted as $(\hat{H}_{\text{srcPrt}}, \hat{H}_{\text{dstPrt}}, \hat{H}_{\text{dstIP}})j$, where $\hat{H}_{\text{srcPrt}}$, $\hat{H}_{\text{dstPrt}}$, and $\hat{H}_{\text{dstIP}}$ are the respective entropy estimates of the free dimensions of the cluster flows calculated using Equation 11.1 or 11.2 and $j = 1,2,\cdots,\mathbb{A}_i^{r=1}$. The number of points generated in the 3D space for each slot i depends on the number of packets captured in such slot and hence on the cardinality of the set $\left|\mathbb{A}_i^{r=1}\right|$. Thus, the set of data points in a slot i is given by $(\hat{H}_{\text{srcPrt}}, \hat{H}_{\text{dstPrt}}, \hat{H}_{\text{dstIP}})_1$, $(\hat{H}_{\text{srcPrt}}, \hat{H}_{\text{dstPrt}}, \hat{H}_{\text{dstIP}})_2$, …, $\left(\hat{H}_{\text{srcPrt}}, \hat{H}_{\text{dstPrt}}, \hat{H}_{\text{dstIP}}\right)\left|\mathbb{A}_i^{r=1}\right|$. One can repeat this procedure of obtaining the entropy points of the free dimensions for each time slot m of the traffic trace to generate the entropy space of CK srcIP. Afterward, one can plot those points using a scatter plot to form a point cloud data. A similar procedure is repeated in order to obtain three entropy spaces of the three remaining CKs. In this chapter, the analysis will be focused on the study of the CK srcIP.

Figure 11.1 shows the entropy spaces of the CK srcIP for three traffic traces. The traces were captured from a campus LAN. Figure 11.4a shows typical TCP traffic of 8 h during typical

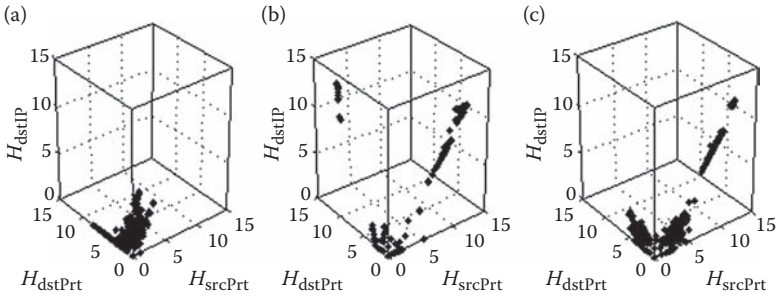

Figure 11.4 Entropy spaces for CK srcIP. (a) Typical traffic; (b) Blaster worm propagation; (c) Sasser worm propagation.

working hours; similar patterns were obtained for the rest of the weekdays. Figure 11.4b and c shows the behavior of Blaster and Sasser worms during 30 min of traffic capture, respectively. Figure 11.4a shows that, although typical traffic tends to be concentrated, the behaviors of Blaster and Sasser traffic tend to increase their entropy value and spread the points on the entropy space. In addition, there are patterns of positive correlation between the entropies of the free dimensions, specifically H_{dstIP} versus H_{srcPrt} for Blaster worm and H_{dstIP} versus H_{dstPrt} for Sasser worm.

In Figures 11.1 and 11.2, the negative linear dependency was discussed because it has sensitivity to some kind of attacks such as the scan attack. This anomalous behavior is observed and captured clearly through the use of the entropy spaces as it can be seen in Figure 11.5, with the point pattern organization in such spaces that is structured differently from that of a normal traffic entropy space. It can also be seen in Figure 11.5 that the values of entropy in such spaces are large. The entropy space shown in Figure 11.5 corresponds to a different data set from those of the Blaster and Sasser attacks shown in previous figures. The data set is from the CAIDA project and contains only the traffic generated by the propagation of the Witty worm that was captured in the backbones of the Internet. In Figure 11.5b, such linear dependency can be seen through the correlation coefficients of the srcPrt and dstIP features. We found that entropy estimates of the cluster flows exhibit linear relationships that can be seen through the scatter plots and that are dependent on the network activity. This linearized behavior has been detected in other traffic data sets, confirming its sensibility to different attacks. Hence, a detection tool based on the correlation

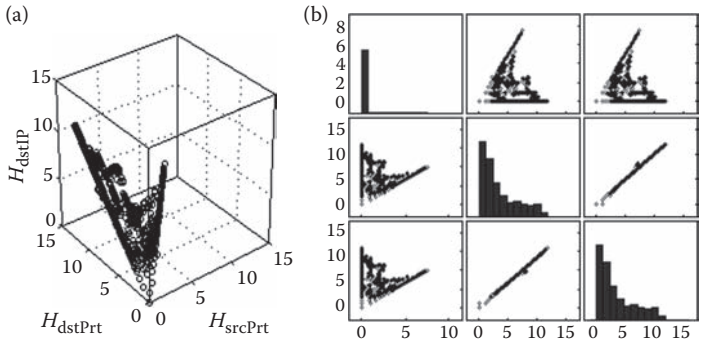

Figure 11.5 Entropy space for CK srcIP. (a) Witty worm traffic; (b) plot matrix for Witty worm propagation.

coefficients and the covariances can be applied to the traffic data sets at the time-slot level and detect such trends.

Once one has the entropy spaces, one can see that these are represented by the vector $X^p \in \mathbb{R}^3$. Because the amount of information is large, a dimension reduction procedure needs to be applied. In this case, PCA, as introduced in Section 11.2, is applied to this vector to reduce the dimensionality of it and to generate a new vector z-score $Z^r \in \mathbb{R}^k, k \leq 3$. To obtain accurate analysis estimation, tools such as kernel density estimation (KDE) are useful. This analysis was applied on data points at the slot level on the PCA-1 by using a Gaussian kernel of 200 points, with a bandwidth of $h = 1.06\sigma J^{1/5}$, the Silverman's criteria, where J is the number of observations and σ is the standard deviation of the set of observations. KDE shows that the traffic slots have Gaussian bimodality behavior in its PDF. Each mode was labeled as principal mode and far mode, respectively, as shown in Figure 11.6.

For PCA-1, the empirical obtained mean values were among 4.3 (positive far mode) and –4.2 (negative far mode) units of PCA-1. Negative far mode was the most frequent. In the case of the studied attacks, far mode presented anomalous values on slots with anomalous traffic. For instance, in the case of Blaster worm, the three first slots show values of –9, –11, and –13 PCA-1 units, respectively (Figure 11.7a). These values were classified as an anomaly

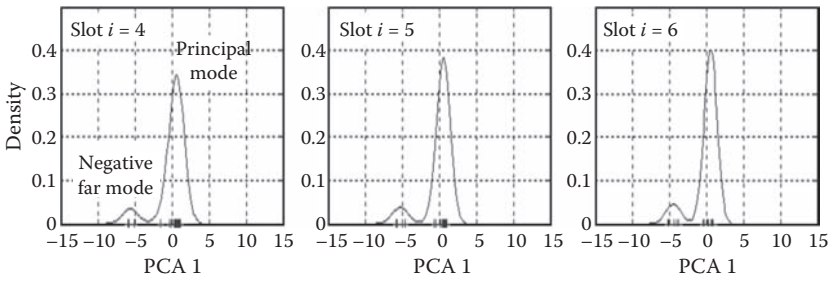

Figure 11.6 Density estimation of PCA-1 presents bimodal behavior on traffic slots.

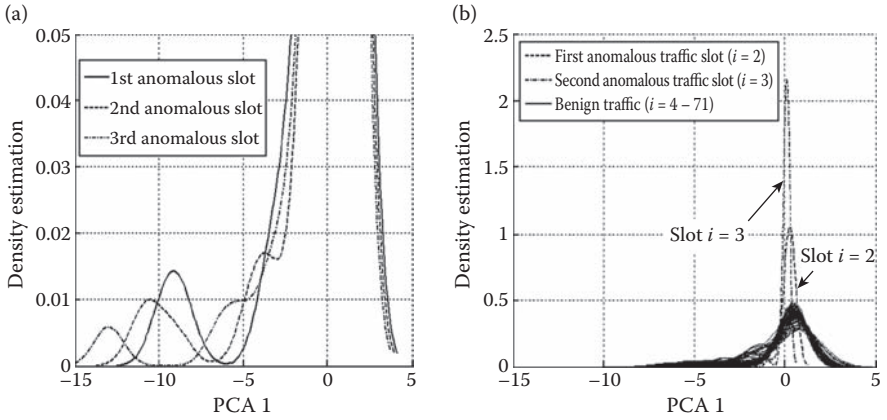

Figure 11.7 Cluster flow anomalies. (a) First three anomalous slots related to the Blaster worm traffic. (b) Two anomalous slots related to port scan attack.

because each value is lower than the mean and the threshold value (–4.2). Thus, this anomalous behavior can be detected easily in an early stage. A second feature that was used because of its sensitive behavior is the standard deviation of the principal mode. The standard deviation empirically obtained in typical conditions was 1.5 units. Then, the standard deviation of the principal mode was taken to show anomalous behavior. For instance, in the case of Sasser worm, the standard deviation of the principal mode on anomalous slots decreased to 0.4 unit in average. In other tests, this time detecting a port scanning attack, the standard deviation shows 0.7 and 0.4 unit on two anomalous traffic slots. In this case, Figure 11.7b shows an outlier-like density in slots 2 and 3.

Nevertheless, KDE only provides the density distribution shape but not its parameters. Therefore, a technique to extract these parameters (mean of the far mode and standard deviation of the principal mode) is needed. It is important because these parameters can be utilized in A-NIDS. The Gaussian mixture model (GMM) is the approach used to extract the parameters of the distribution [34]. A GMM implementation can be found in Statistic Toolbox of MATLAB®. `gmdistribution.fit` forms groups of data and then fits them on a GMM of $\mathbb{K}$ components. This function performs the maximum likelihood estimation of the GMM by using the expectation maximization algorithm. The algorithm returns the parameters $\theta_k = \left[\pi_k, \mu_k, \sigma_k\right]_{k=1}^{\mathbb{K}}$, where $\mathbb{K} = 3$ is the number of modes of the estimated PDF under KDE for PCA-1, π_k are the proportions of the mixture, and μ_k and σ_k are the mean and standard deviation, respectively, that are related to the required information about principal and far modes.

In the PCA-1 for CK srcIP analysis, we obtain the mixture vector θ_k. From θ_k, principal and far modes can be identified on an *i* slot. The principal mode corresponds to *k* mixture, which has max (π_k) (maximum proportion). The far mode corresponds to mixture with max ($|\mu_k|$) (maximum mean value).

According to these empirical results, feature selection chooses the far mode mean value, from now on denoted as r_1, and the standard deviation of the principal mode, denoted as r_2, as the most appropriate set of features. The behavior of the aforementioned features allows one to establish that they are good parameters to identify anomalous and typical traffic. Experiments described throughout this document support the aforementioned assumption.

Once the results of the PCA are processed by using KDE, one needs to do a traffic analysis under typical conditions, which generally shows that the behavior of feature r_1 follows a trimodal distribution, with more prevalent trends to negative values with respect to positive values and a residual mode centered at zero, as shown in Figure 11.8a. The model that describes this behavior was first observed by KDE; subsequently, its distribution parameters were obtained by GMM.

Such a probabilistic model for feature r_1 based on GMM is as follows:

$$f_{\mathrm{GM}}\left(r_1;\theta_k\right) = \sum_{k=1}^{3} \pi_k \mathcal{N}\left(r_1 \middle| \mu_k, \sigma_k^2\right) = \sum_{k=1}^{3} \frac{\pi_k}{\sqrt{2\pi}\sigma_k} e^{-\frac{\left(r_1-\mu_k\right)^2}{2\sigma_k^2}} \tag{11.5}$$

where μ_k and σ_k are the mean and standard deviation of the *k*th component, respectively, and π_k are the mixture proportions. The three components form a three-component vector $\theta_k = \left[\pi_k, \mu_k, \sigma_k\right]_{k=1}^{3}$. $\mathcal{N}\left(r_1 \middle| \mu_k, \sigma_k^2\right)$ represents a Gaussian multivariate distribution. The fit values of this mixture are shown in Table 11.1.

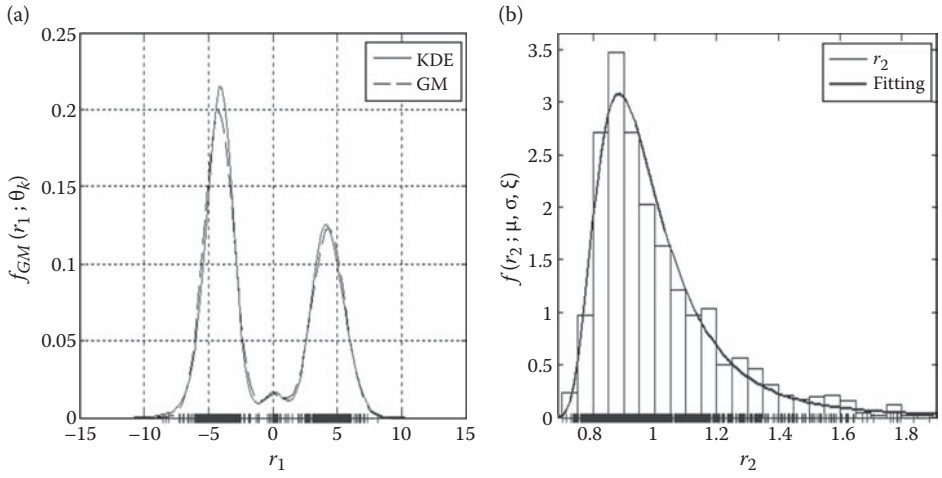

Figure 11.8 Fitting distribution to selected features in cluster flows. (a) Fitting feature by using GMM. (b) Fitting curve of feature by using a GEV distribution.

Table 11.1 Fit Parameters for Mixture

Parameter	*k = 1*	*k = 2*	*k = 3*
Mixing (π_K)	0.3946	0.5740	0.0314
Mean (μ_k)	4.3239	–4.1988	0.0980
Variance $\left(\sigma_k^2\right)$	1.6378	1.3208	0.7546

In the case of feature r_2, analysis on the set of typical traffic traces showed that their behavior followed a generalized extreme value (GEV) distribution, with the following profile behavior model:

$$f\left(r_2;\mu,\sigma,\xi\right)=\frac{1}{\sigma}\left[1+\xi*\left(\frac{r_2-\mu}{\sigma}\right)\right]^{-1/\xi-1}*\exp\left\{-\left[1+\xi*\left(\frac{r_2-\mu}{\sigma}\right)\right]^{-1/\xi}\right\} \tag{11.6}$$

for $1 + \xi^*(x_2 - \mu/\sigma) > 0$, where $\mu \in \mathbb{R}$ is the localization parameter, $\sigma > 0$ is the scale parameter, and $\xi \in \mathbb{R}$ is the shape parameter.

Fitting parameters for Equation 11.6 that describe the behavior of r_2 are $\xi = 0.2228$, $\sigma = 0.1221$, and $\mu = 0.9079$. The obtained shape with these values is shown in Figure 11.8b.

11.3.1 Excess Point Method

In this section, we describe the method of excess points, where the 3D entropy spaces created are used to determine the presence of anomalies by detecting the region where such points in the 3D space are

located. This method is an alternative method to that just described. It starts with the generation of the entropy spaces as described in the previous section for the application of MES. The excess point method consists of finding a centroid for the benign traffic and then characterizing it by the distances of the normal traffic points to that benign centroid. After the normal traffic characterization, the anomaly traffic is detected by taking the anomaly points' distances to the benign centroid previously found that exceed a maximum distance found for the benign traffic (excess points). A PDF can then be found using KDE for both the benign and the anomalous traffic distances to the benign centroid with which one can obtain a decision region that will determine if a given window can be declared as having a normal or an anomalous behavior. The steps that describe the excess point method are the following:

- One must choose a number of days D in which Internet traces in the network segment to be analyzed will be gathered to have reliable information about the normal behavior or the network. There is a need to make a compromise in this matter, as having lots of information (many days of collected data) will have the effect of finding a more accurate overall behavior of the network, but it will be time consuming and of heavy processing to analyze the traffic. Then again, having few training days for the data results in a less accurate modeling of the network but will consume less memory, which helps in processing speed. The proposed period to train the data is 1 to 2 weeks.
- A window time of t_w seconds is declared. This period will be the basic period used to capture packets in every trace. Just as in the number of days, there must be a compromise in choosing the window size, as it must be sufficiently large to gather enough information about the trace, and it should be sufficiently small to have a faster response when an anomaly appears in the network segment being analyzed. The data captured in this time window will constitute the actual training data for the IDS. In this method, a window size of 60 s was used.
- Traffic from the D days is captured in time windows, each one with duration of t_w seconds. A total of Q time windows containing traffic over the D days will be obtained. Then, a data mining process must be applied to extract the information about the packet headers in each window time. Although there are other parameters that could be exploited to have a different analysis, in this system, the features that will be used will be those regarding computer information (IPs) and service information (ports).
- Using the method described in this section, the entropy spaces must be found for each CK and for every window obtained in the D days from which the data were gathered, such as those depicted in Figure 11.9 for $D = 5$ days.
- The next step consists of finding the benign or reference centroid $c^r = (x_{cn}, y_{cn}, z_{cn})$ from the captured data. The benign centroid constitutes the representative point that helps in differentiating a benign from a malicious traffic. To obtain the centroid of the captured data, a method based on "bins" b_j is proposed.
- The method divides the whole entropy space H_i^r of each cluster key into m smaller cubes (bins) of side length s entropy units (according to the logarithm base, the units may be different). The value of s must be chosen between certain range, as it must be small enough to preserve the behavioral characteristics of the traces, and it must be big enough to reach an appropriate centroid. Once a side length s is chosen, the number of bins in which each entropy space will be divided to is given by

$$m = \left(\left\lceil \frac{\max\{x_c\}}{s} \right\rceil \right) \cdot \left(\left\lceil \frac{\max\{y_c\}}{s} \right\rceil \right) \cdot \left(\left\lceil \frac{\max\{x_c\}}{s} \right\rceil \right). \tag{11.7}$$

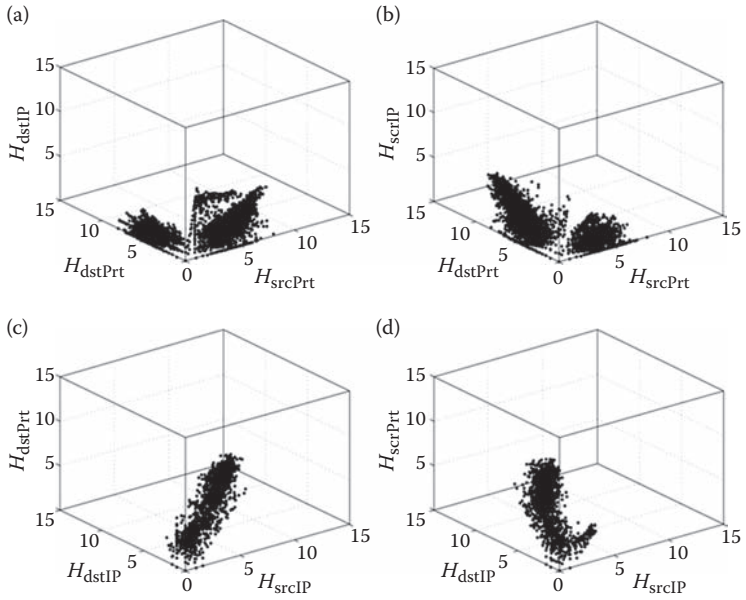

Figure 11.9 Entropy spaces for the four cluster keys. (a) Source IP entropy space; (b) destination IP entropy space; (c) source TCP port entropy space; (d) destination TCP port entropy space.

- After choosing s, an average is obtained from all the points that fall into the volume of each cubic bin found. The average of each bin will function as a representative point for that bin, that is,

$$\left(x_b, y_b, z_b\right) = \frac{1}{J}\sum_{u=1}^{J}\left(x_u, y_u, z_u\right), \quad b = 1,2,\ldots,m, \tag{11.8}$$

  where J is the total number of points that fall within the volume.
- Once the representative points for all the bins in which the space is divided (m) have been obtained, an average of all these representative points must be found, and it will become the centroid $\left(c_i^r\right)$ for each ith window:

$$c_i^r = \left(x_i, y_i, z_i\right) = \frac{1}{m}\sum_{b=1}^{m}\left(x_b, y_b, z_b\right), \quad i = 1,2,\ldots,Q. \tag{11.9}$$

Finally, the benign centroid is found by calculating an average of all the windows' Q centroids:

$$c^r = \left(x_{cn}, y_{cn}, z_{cn}\right) = \frac{1}{Q}\sum_{i=1}^{Q} c_i^r. \tag{11.10}$$

The centroids must be found for each cluster key. In Figure 11.10a and b, it can be seen that the entropy space resulting from a Blaster worm attack is easily distinguished from the benign traffic entropy space as there are many excess points.

The reason for using this procedure (the use of bins) to find the benign centroid follows the fact that the normal behavior of the traffic inside the network analyzed reflects many entropy points that are very near the space borders. Having that many points gathered into specific areas has the effect of pulling the centroid near that area, and as a result, the centroid found will not be a good representation of the whole extension of the entropy points. With this procedure, a centroid closer to the middle point of the range of the space points in every dimension can be found.

- Once the benign centroids are found for each cluster key, the Euclidian distances δ_p of all the points $p = 1, 2, \ldots, P$, in each of the entropy spaces to the benign centroid can be found. The set of benign points' distances to the benign centroid $D_b = \{\delta_{b1}, \delta_{b2}, \ldots, \delta_{bP}\}$ is then obtained.
- The next step in the excess point method consists of obtaining the anomalous traffic distances from the anomalous points to the benign centroid. The entropy points are obtained for traffic traces known to be anomalous using the same method used for the benign traffic. Then, the malicious distances set $D_m = \{\delta_{m1}, \delta_{m2}, \ldots, \delta_{mT}\}$ corresponding to the T anomaly points distances to the benign centroid.
- Having found the distance sets for the benign and anomalous traffic entropy points, we use elements of Bayes decision theory to provide a Bayesian classifier with which a traffic trace can be classified as having normal or unusual behavior. The rules of the Bayes decision theory, explained in more detail in [29–31], are then used.
- In this methodology, there are two classes defined as C_i with $i = 1, 2$, corresponding to benign and anomalous classes, respectively. With the event of an input entropy point (x_p, y_p, z_p), its distance to the benign centroid x and the a priori probabilities of the incoming point $p(x)$ and $p(C_i)$, the Bayes classifier rule is applied as follows:

$$\text{if } P(C_1|x) > P(C_2|x), \quad x \rightarrow C_1, \tag{11.11}$$

$$\text{if } P(C_1|x) < P(C_2|x), \quad x \rightarrow C_2. \tag{11.12}$$

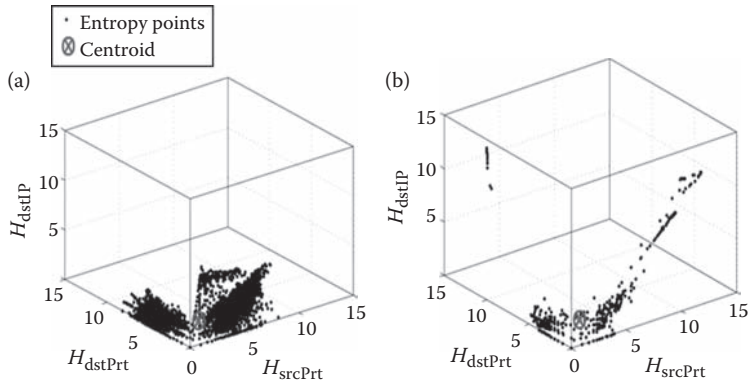

Figure 11.10 Entropy space identification for Blaster worm attack. (a) Benign traffic entropy space for srcIP; (b) Blaster worm attack traffic entropy space for srcIP.

By using Bayes rule, we can obtain the a posteriori probabilities and decisions as follows:

$$\text{if } P(x|C_1)\ P(C_1) > P(x|C_2)\ P(C_2), \quad x \to C_1, \tag{11.13}$$

$$\text{if } P(x|C_1)\ P(C_1) < P(x|C_2)\ P(C_2), \quad x \to C_2. \tag{11.14}$$

Then, probabilities $P(C_i|x)$ can be associated with PDFs obtained for the benign and anomalous entropy point distances. The excess points, as stated in the initial description of the method, are those points from the malicious distances set that exceed a maximum benign distance threshold t_b. The PDF of the benign traffic distances is obtained. The method used for the PDF estimation was the kernel smoothing algorithm. After applying the kernel smoothing algorithm, the PDF describing the distances from the benign points to the centroid is found. Figure 11.11a shows a histogram of the distances of the normal points to the srcIP benign centroid. Figure 11.11b shows a PDF found with the kernel estimation for the destination IP and also a normal PDF with the mean and variance of the distance set. Then, for the anomaly traffic, the excess points distance set is found by taking all the distances from the malicious set that exceed the maximum benign distance threshold. The threshold is obtained by the likelihood ratio method, where the total number of malicious points whose distance exceeds the maximum benign distance is divided by the total number of malicious points. An empirical analysis of training data sets helps in determining the value this likelihood ratio must have.

The excess point methodology can be summarized as follows:

- Captured traces are gathered just as in the training period in windows of t_w seconds. For every window, there will be a captured trace Y.
- Information about the trace packet headers' features is retrieved.
- The entropy spaces for the current window are obtained for every cluster key using the same methodology described.
- A new set of distances from each of the entropy space points to the benign centroid is found for every cluster key.

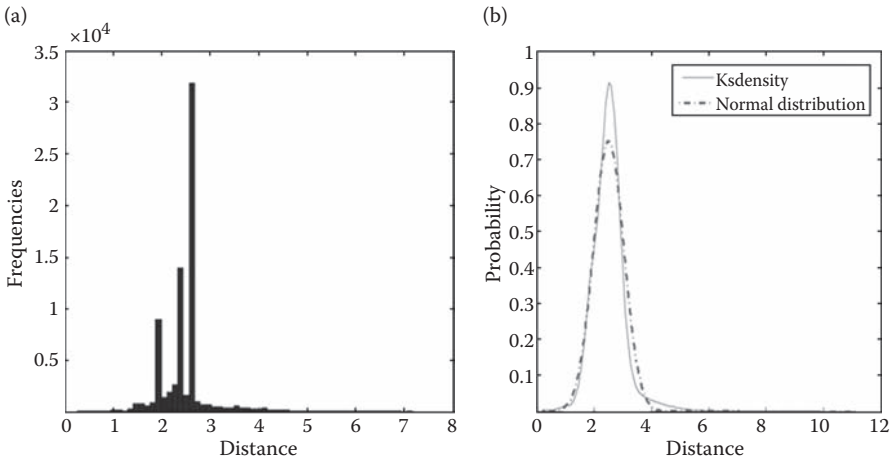

Figure 11.11 PDF estimation for srcIP benign traffic. (a) Histogram of the distances to the benign centroid for srcIP; (b) PDF for the distances to the benign centroid for srcIP.

- Every point in the set Y can be classified as benign or anomalous by using the Bayes decision rule considering with the PDFs found in the training stage.

11.4 Architecture for A-NIDS Based on MES

In this section, we describe the overall design of our anomaly-based NID architecture. Also, some test results are shown where Blaster and Sasser attacks are detected using the methodology described in Section 11.3. This framework (Figure 11.12) implements behavioral profiles based on entropy spaces. This architecture can be divided into two layers. The first layer does training functions based on the selected features r_1 and r_2, and its respective parameters of the behavior profiles are obtained by using probabilistic techniques from transformed data by PCA, whose models are specified in Equations 11.5 and 11.6. The second layer performs actual measurements on the cluster flows in an i traffic slot. This procedure converts cluster flows to a 2D point in the feature space. If the feature measurements are deviated from the typical traffic profile, then this entire traffic slot is labeled anomalous. Each block that forms the framework for the training layer is described as follows:

- *Preprocessing:* In this stage, each typical traffic trace is sanitized. Such process is done to remove duplicated or miscaptured packets. The trace is filtered according to the protocol (TCP, user datagram protocol, and Internet control message protocol).
- *Flow generator:* On each training traffic slot, flows are generated according to the five-tuple definition.

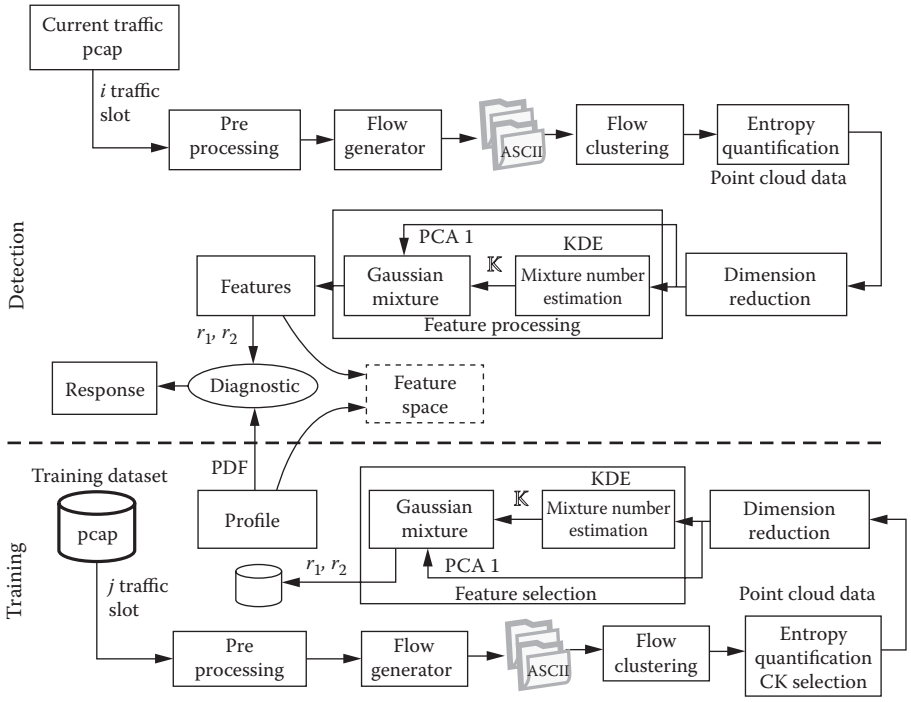

Figure 11.12 Block diagram architecture of the A-NIDS proposal.

- *Flow clustering:* For a given CK-*r*, the generated flows are clustered. The number of cluster flows will depend on the cardinality of the alphabet of the fixed *r*-field, that is, $\mathbb{A}_i^r$.
- *Entropy quantification:* For each cluster flow, the naive entropy of the free dimensions is estimated. The result is represented as a point cloud in a 3D space.
- *Dimension reduction:* The set of entropy points in a traffic slot is transformed by using PCA. The first principal component behavior has properties that capture the changes in traffic patterns caused by illegitimate activities.
- *Feature selection:* The two selected features of the first principal component are obtained by a process that identifies the $\mathbb{K}$ number of mixtures in the KDE, and their respective density parameters are estimated using GMM.
- *Profile:* The analysis of the training data set under this procedure generates new data with the values of the features; these data are fitted with the models described in Equations 11.5 and 11.6. With this, the typical behavior of the traffic is characterized.

In the detection layer, the process is similar to that of the training layer; in other words, it does perform the functions as in the training layer from preprocessing up to the feature selection, and the only difference is that a profile is not generated. Instead of this, from each traffic slot being monitored, and after passing the aforementioned listed stages, r_1 and r_2 features are extracted and compared to the reference profile provided by the training layer. In case a difference is found between such features and the obtained profile, that is, upon finding feature deviations, an anomaly is flagged to the network administrator so that it can be further analyzed and decisions can be done regarding possible intrusions. The excess point method can be incorporated within the A-NIDS architecture as an alternate or a backup diagnostic tool that provides the response in the detection layer. The use of excess point methodology can help reduce information processing because the point cloud data are the ones used to perform such methodology avoiding the use of dimension reduction.

Figure 11.13a shows the two types of traffic, the normal and anomalous traffic, produced by Blaster. The former is grouped into three different regions, and the latter is grouped only into two regions. It can help to apply supervised classification techniques by dividing their corresponding

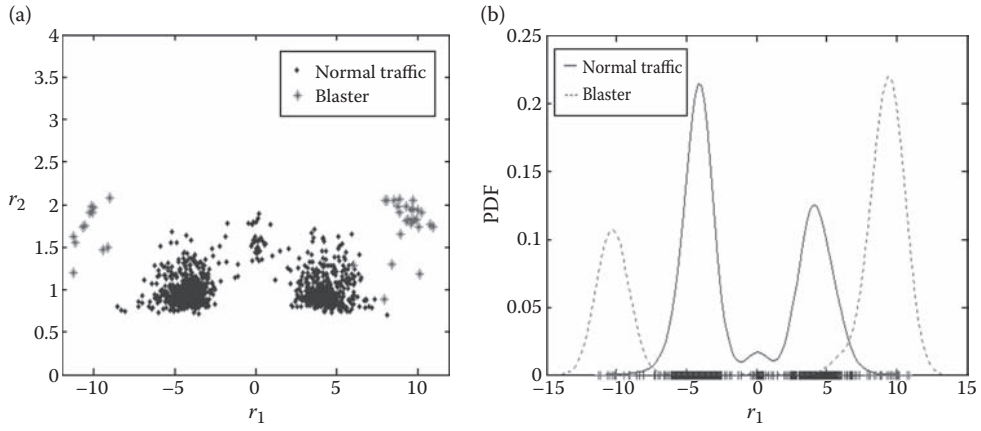

Figure 11.13 Comparison between typical and anomalous traffic slots caused by Blaster worm. (a) Feature space. (b) PDFs.

classes. In Figure 11.13b, traffic distribution is shown for the feature r_1 on both cases of normal and anomalous traffic produced by Blaster worm.

11.4.1 Results of Tests

In this section, we present results that show how, with the help of the PCA methodology, we can reduce the dimensionality of the entropy spaces and still detect anomalies based on a new 2D feature space. We also show that the characterization of the points in such feature space can be characterized by a mixture of Gaussian PDFs.

Figure 11.13a shows the points in the feature space after applying the PCA for the two types of traffic: normal and anomalous. Each time slot produces several of the points shown. The points that are not concentrated in the 2D graph are those points produced by the Blaster worm attack. The normal traffic is grouped into three different regions, and the anomalous traffic is grouped only into two regions. Both regions are clearly separated from one another; it helps to apply supervised classification techniques by dividing their corresponding classes.

As seen in Figure 11.13b, the model for r_1 shows a marked difference in the properties of traffic caused by the Blaster worm with respect to typical traffic. The extreme points (vertices) of the convex hull of the data points in the feature space contain the class of normal traffic. Then, a classifier can be designed in terms of the data instance that belongs to the outside of the boundaries of the class of typical traffic.

In the case of the Sasser worm, traffic slots are grouped in one region as shown in Figure 11.14a. In Figure 11.14b, it shows the difference between normal traffic and anomalous traffic produced by the Sasser worm as a decrease in the variance in the first principal component, which is captured by feature r_2.

11.4.2 Excess Point Results

The benign centroid for the excess point method was found using the bins method with a bin's side length of $s = 1.5$ bits, which showed good results for the centroid's location in the entropy space. The likelihood ratios for the Blaster and Sasser worms where excess points were found are presented in Table 11.2, where a value of zero means that there were no excess points found for a given cluster key.

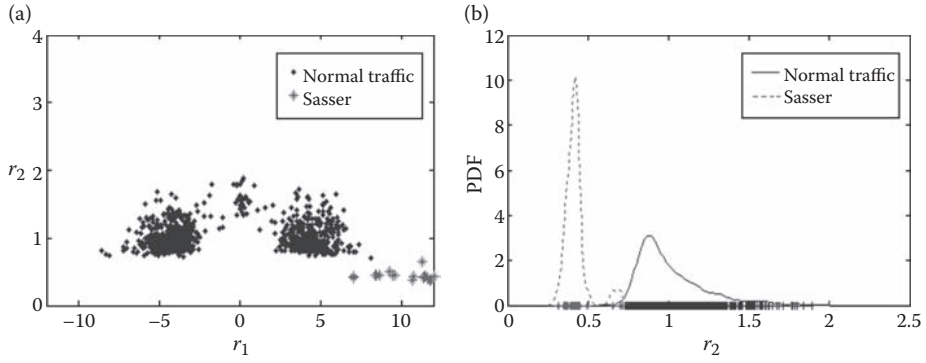

Figure 11.14 Comparison between typical and anomalous traffic slots caused by Sasser worm. (a) Feature space. (b) Probability densities.

Table 11.2 Likelihood Ratios of Blaster and Sasser Worms

Attack	*srcIP*	*srcPrt*	*dstIP*	*dstPrt*
Blaster	0.01	0.00046	0	0.000911
Sasser	0.0034	0.000325	0.0018	0.000454

In Figure 11.15, the results of the PDFs found for all the parameters of a Blaster worm attack are shown, whereas, in Figure 11.16, the results for all the parameters of a Sasser worm attack are shown. As seen in Figure 11.15, for three of the four cluster keys (having dstIP as an exception, i.e., there were no excess points for this parameter), there is a clear detection of a Blaster attack, as the PDFs are easily distinguished. In Figure 11.16, for the four cluster keys, there is a clear distinction between both PDFs; therefore, the attack is detected. The PDF fit found with the kernel smoothing method could be replaced by a normal distribution for both benign and anomalous traffic to simplify the analysis, as has been long time studied, and the probabilities of error are well established for that type of distribution. The normal distribution can be used given that the goal of effectively identifying a distinction between two classes of traffic behaviors is accomplished.

The excess point method performs well if attacks have the same characteristics as those of the Blaster and Sasser worms, where IP source and destination addresses produce high entropy variability in the entropy spaces. The reason for which the methodology of the excess points method was inefficient for other types of attacks is because some of them exhibit a behavior that derives into entropy points very near the benign centroid; therefore, the amount of anomaly points that

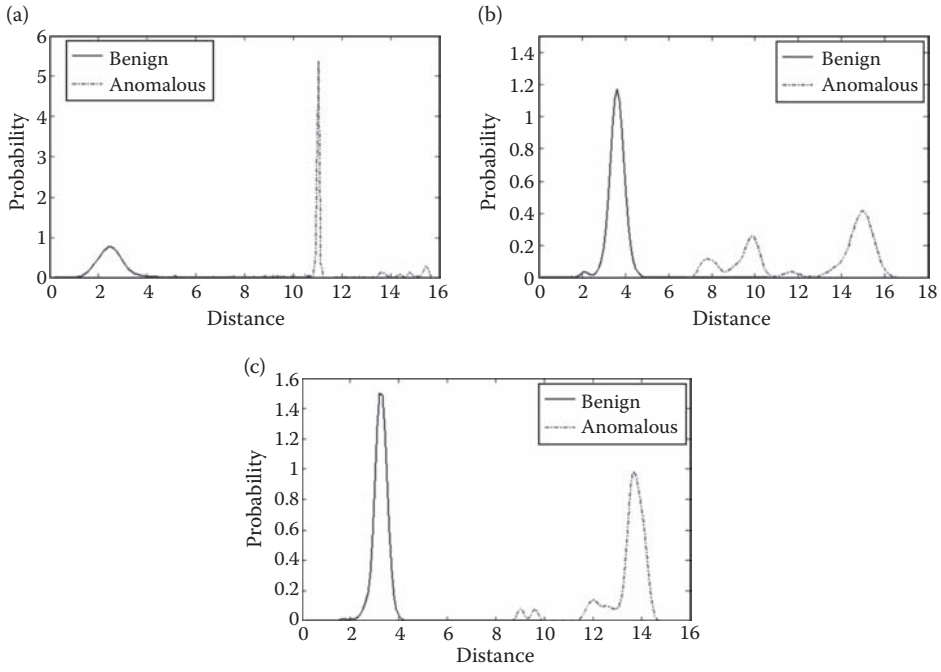

Figure 11.15 Probability comparison for a Blaster worm attack for the excess point method. (a) PDF comparison for CK srcIP; (b) PDF comparison for CK srcPrt; (c) PDF comparison for CK dstPrt.

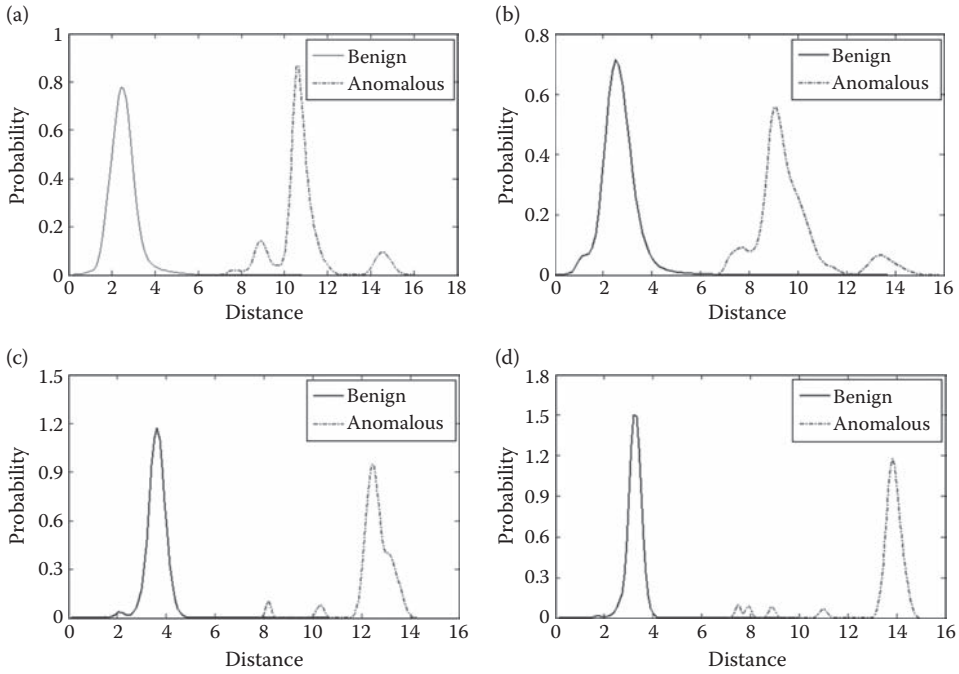

Figure 11.16 Probability comparison for a Sasser worm attack for the excess point method. (a) PDF comparison for CK srcIP; (b) PDF comparison for CK dstIP; (c) PDF comparison for CK srcPrt; (d) PDF comparison for CK dstPrt.

are located outside the boundary provided by the benign behavior (i.e., the excess points) is very small or zero in some cases, and the profiling for that traffic becomes impossible with this method. For other attacks, the centroid obtained from the traffic data sets can be close to the benign centroid obtained by the training data set, confirming the need to use a combination of detection tools in parallel to make a decision about the anomaly.

11.5 Experimental Platform, Data Set, and Tools

The worm propagation has been tested in a class C IP network subdivided into four subnets. There are 100 hosts running Windows XP SP2 mainly. Two routers connect the subnets with 10 Ethernet switches and 18 IEEE 802.11b/g wireless access points. The data rate of the core network is 100 Mbps. A sector of the network is left vulnerable on purpose, with 10 not patched Windows XP stations. In the experiments, Blaster and Sasser worms were released in the vulnerable sector.

The data set was collected by a network sniffer tool based on libpcap library used by tcpdump. This data set contains traces corresponding to a 6-day period of standard traffic in a user's typical work hours and one standard traffic trace combined with anomalous traffic of the day of the attacks. All traces were sanitized to remove spurious data using plab, a platform for packet capture and analysis. Traces were split in segments using tracesplit, which is a tool that belongs to Libtrace. The traffic files in ASCII format suitable for MATLAB processing were created with ipsumdump. Entropy estimation was implemented by a Perl script, which uses hash functions proving efficiency and speed in the analysis of large data sets.

Please check if the spelling of this word is correct.

11.6 Conclusions

We have proposed an architecture for anomaly detection based on information theory concepts such as the entropy. We have also shown that the methodology proposed is successful by detecting different attacks using different methodologies. The proposed method has been tested under load on real-time network traffic data and deploying real attacks. Empirical results have shown that the methods introduced in this chapter are able to detect traffic anomalies. Techniques such as entropy spaces, excess point method, PR, and fitting models were used. Their use allows identifying sensible features (mean of the far mode and standard deviation of the principal mode) that help in detecting anomalous traffic in specific slots and comparing them with normal traffic. We also introduced the sensibility of the entropy to attacks of different characteristics, where negative linear dependency captured the anomalies; this was shown by the use of correlation coefficients between the four features that were analyzed from the packets in the data sets. By using these results, a framework architecture is proposed to analyze network traffic anomalies. Deviations from a given reference obtained from typical traffic are considered anomalous. Future work will consider scenarios with high volume of traffic, the evaluation of more attacks, and adding a preprocessing stage by using an S-NIDS like Snort. The preprocessing implies filtering well-known attacks to obtain features that represent normal traffic in a better way.

11.7 Future Research

The results presented in this chapter direct us to consider several actions to pursue as future research areas related to the area of anomaly detection in networks. For example, regarding the entropy spaces, we can see that these need to be defined formally through the use of vector spaces, and once having this base, the entropy space can be considered as composed of two regions: the benign and anomaly regions. Maximum likelihood and maximum a posteriori probability theories from information and communication theory could be applied in order to have a decision process.

Please check if the edit made here is correct.

Another area of interest would be the anomaly detection through the characterization of the traffic interarrivals of packets. From the networking point of view, traffic characteristics such as heavy tails, self-similarity, long-range dependence, and alpha stability are changed when an anomaly is present, and metrics from information theory can be used to track the amount of change and determine the presence or absence of anomalies.

Another important point of view is that of adaptability and dynamism that the references must have in order to cope with the changing characteristics of the benign traffic because applications and new users are added to the network on a daily basis. Some of the results, although iterative as introduced in this book, can be seen as a product of a set of discrete-time sequences that need to be processed by digital filters to produce a response sensible to anomalies. These filters must adapt to the network conditions and need to be changing with time by dynamic procedures that instruct how parameters must be set at different times.

Please check if the edit made here is correct.

Acknowledgments

We would like to thank the sponsorship of CONACyT through the project Investigación Básica SEP-CONACyT CB-2011-01 "Modelos y algoritmos basados en Entropía para la Detección y Prevención de Intrusiones de Red." This chapter and the results presented here would not have been possible without such support.

References

1. Canavan, J. E., *Fundamentals of Network Security*, Artech House, 2001, 194.
2. Mukherjee, B., Heberlein, T. L. and Levitt, K. N., Network intrusion detection, *IEEE Network*, 8 (3), 26–41, May/June 1994.
3. Heady, R., Luger, G., Maccabe, A. and Servilla, M., The Architecture of a Network Level Intrusion Detection System, Technical Report CS90-20, Department of Computer Science, University of New Mexico, August 1990.
4. Furnell, S. M., Katsikas, S., Lopez, J. and Patel, A., *Securing Information and Communications Systems: Principles, Technologies, and Applications (Information Security & Privacy)*, Artech House, 2008, 166.
5. Solomon, G. and Chapple, M., *Information Security Illuminated*, Jones & Bartlett Learning, 2005, 327.
6. Whitman, M. E. and Mattord, H. J., *Principles of Information Security*, Fourth Edition, Course Technology, Cengage Learning, 2012, 305.
7. Malik, S., *Network Security Principles and Practices*, Cisco Press, 2002, 420.
8. Common Vulnerabilities and Exposures. CVE is a dictionary of publicly known information security vulnerabilities and exposures. Available at http://cve.mitre.org/.
9. Snort: The De Facto Standard for Intrusion Detection and Prevention. Available at http://www.sourcefire.com/security-technologies/open-source/snort.
10. Chandola, V., Banerjee, A. and Kumar, V., Anomaly detection: A survey, *ACM Computing Surveys*, 41 (3), 2009, article 15.
11. Thatte, G., Mitra, U. and Heidemann, J., Detection of low-rate attacks in computer networks, in Proceedings of the 11th IEEE Global Internet Symposium, Phoenix, AZ, USA, IEEE, April 2008, 1–6.
12. Lakhina, A., Crovella, M. and Diot, C., Mining anomalies using traffic feature distributions, in ACM SIGCOMM, 2005, 217–228.
13. Mahoney, M., Network traffic anomaly detection based on packet bytes, in Proceedings of the ACM SIGSAC, 2003.
14. Mahoney, M. and Chan, P. K., Learning nonstationary models of normal network traffic for detecting novel attacks, in Proceedings of the SIGKDD, 2002.
15. Wang, K. and Stolfo, S., Anomalous payload-based network intrusion detection, *Recent Advances in Intrusion Detection. Lecture Notes in Computer Science*, Jonsson, E., Valdes, A. and Almgren, M. (Springer Eds.), 2004, 203–222.
16. Karim, A., Computational intelligence for network intrusion detection: Recent contributions, in CIS 2005, Part I, LNAI 3801, Hao, Y. et al. (Eds.), Springer-Verlag, Berlin, 2005, 170–175.
17. Tsai, C., Hsu, Y., Lin, C., and Lin, W., Intrusion detection by machine learning: A review, *Expert Systems with Applications, Elsevier*, 36 (10), 2009, 11994–12000.
18. Thottan, M. and Ji, C., Anomaly detection in IP networks, *IEEE Transactions on Signal Processing*, 51 (5), 2003, 2191–2204.
19. Celenk, M., Conley, T., Willis, J. and Graham, J., Predictive network detection and visualization, *IEEE Transactions on Information Forensics and Security*, 5 (2), 2010, 287–299.
20. Toft, J., Minimization under entropy conditions, with applications in lower bound problems, *Journal of Mathematical Physics*, 45 (8), August 2004.
21. Shannon, C., A mathematical theory of communication. *Bell System Technical Journal*, 27, 1948, 379–423 and 623–656.
22. Paninski, L., Estimation of entropy and mutual information, *Neural Computation*, 15, 2003, 1191–1253.
23. Gu, Y., McCallum, A., and Towsley, D., Detecting anomalies in network traffic using maximum entropy estimation, in Proceedings of the 5th ACM SIGCOMM Conference on Internet Measurement (IMC '05), ACM, New York, 2005, 1–6.
24. Wagner, A. and Plattner, B., Entropy based worm and anomaly detection in fast IP networks, in Proceedings of the 14th IEEE International Workshop on Enabling Tech.: Infrastructure for Collaborative Enterprise, 2005, 172–177.
25. Xu, K., Zhang, Z. and Bhattacharyya, S., Internet traffic behavior profiling for network security monitoring. *Transactions on Networking, IEEE/ACM*, 16 (3), 2008, 1241–1252.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

26. Lall, A., Sekar, V., Ogihara, M., Xu, J. and Zhangz, H., Data streaming algorithms for estimating entropy of network traffic, in International Conference on Measurement and Modeling of Computer Systems, Saint Malo, France, 2006.
27. Marques de Sa, J.P., *Pattern Recognition: Concepts, Methods and Application*, Springer, 2001.
28. Bishop, C. M., *Pattern Recognition and Machine Learning*, Springer Science+Business Media, LLC, 2006.
29. Duda, R. O., Hart, P.E. and Stork, D. G., *Pattern Classification*, 2nd edition, Wiley, New York, 2001.
30. Kpalma, K. and Ronsin, J., An overview of advances of pattern recognition systems in computer vision. *Vision Systems: Segmentation and Pattern Recognition*, Obinata, G. and Dutta, A. (Eds.), I-Tech, Vienna, Austria, June 2007, 546.
31. He, X., Yan, S., Hu, Y., Niyogi, P. and Jiang Zhang, H., Face recognition using Laplacianfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27 (3), 2005, 328–340.
32. Härdle, W. and Hlávka, Z., *Multivariate Statistics: Exercises and Solutions*, Springer Science+Business Media, LLC, 2007, ISBN 978-0-387-70784-6.
33. Barakat, C., Thiran, P., Iannaccone G., Diot, C. and Owezarski, P., Modeling Internet backbone traffic at the flow level. *IEEE Transactions on Signal Processing, Special Issue on Networking*, 51 (8), August 2003.
34. McLachlan, G. J. and Peel, D., *Finite Mixture Models*, Wiley Interscience Publication, 2000.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.

Please provide publisher location.