Zhang B, Yang JH, Wu JP *et al.* Diagnosing traffic anomalies using a two-phase model. JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY 27(2): 313–327 Mar. 2012. DOI 10.1007/s11390-012-1225-0

Diagnosing Traffic Anomalies Using a Two-Phase Model

Bin Zhang (张 宾), Jia-Hai Yang (杨家海), *Member*, *CCF*, *ACM*, *IEEE*
Jian-Ping Wu (吴建平), *Fellow*, *IEEE*, *Member*, *CCF*, *ACM*, and Ying-Wu Zhu (朱应武)

Network Research Center, Tsinghua University, Beijing 100084, China
Tsinghua National Laboratory for Information Science and Technology (TNList)
Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China

E-mail: zhang_bin163@163.com; {yang, jianping}@cernet.edu.cn; zhuyw06@gmail.com

Received July 11, 2011; revised December 30, 2011.

Abstract Network traffic anomalies are unusual changes in a network, so diagnosing anomalies is important for network management. Feature-based anomaly detection models (ab)normal network traffic behavior by analyzing packet header features. PCA-subspace method (Principal Component Analysis) has been verified as an efficient feature-based way in network-wide anomaly detection. Despite the powerful ability of PCA-subspace method for network-wide traffic detection, it cannot be effectively used for detection on a single link. In this paper, different from most works focusing on detection on flow-level traffic, based on observations of six traffic features for packet-level traffic, we propose a new approach B6-SVM to detect anomalies for packet-level traffic on a single link. The basic idea of B6-SVM is to diagnose anomalies in a multi-dimensional view of traffic features using Support Vector Machine (SVM). Through two-phase classification, B6-SVM can detect anomalies with high detection rate and low false alarm rate. The test results demonstrate the effectiveness and potential of our technique in diagnosing anomalies. Further, compared to previous feature-based anomaly detection approaches, B6-SVM provides a framework to automatically identify possible anomalous types. The framework of B6-SVM is generic and therefore, we expect the derived insights will be helpful for similar future research efforts.

Keywords anomaly detection, entropy, support vector machine, classification, traffic feature

1 Introduction

Anomalies such as network scans, worms, DDoS attacks, can cause performance degradation of network devices and end hosts, consume network resources, and lead to security issues concerning all Internet users. Thus, detecting anomalies has become an important issue for the network. Traditional approaches to anomaly detection use attack signatures also called misuse detection that can identify attacks with known patterns. Although signature-based detection finds most known attacks, it fails to identify new attacks and anomalies that have not appeared before and do not have known signatures.

With new signature increasing, signature-based detection is lagging behind the creation of malicious threats[1], which makes newer antivirus technologies and techniques, such as behavior-based detection, increasingly important. Behavior-based anomaly detection techniques model the normal behavior of network traffic and identify anomalies as deviations from the normal behavior. A lot of volume-based detection techniques were proposed that monitor the aggregate or per-link traffic load of a network, to detect anomalies that trigger significant traffic volume changes. However, not all network incidents result in big traffic volume shifts. Low traffic rate attacks produce limited change in traffic load and, therefore, go undetected with volume-based detection systems.

Feature-based anomaly detection methods seek to address the limitations of volume-based systems by examining a range of network traffic features, instead of relying only on traffic volume. Lakhina *et al.*[2] showed that despite their diversity, most traffic anomalies share common characteristics: they lead to a change in distributional aspects of packet header fields (i.e., source and destination addresses and ports, so called traffic features). Lakhina *et al.* proposed an entropy-based PCA-subspace method using traffic features. This technique has received great attention and inspired a lot of

Regular Paper

This work is supported by the National Basic Research 973 Program of China under Grant No. 2009CB320505, the National Science and Technology Supporting Plan of China under Grant No. 2008BAH37B05, the National Natural Science Foundation of China under Grant No. 61170211, the Ph.D. Programs Foundation of Ministry of Education of China under Grant No. 20110002110056, and the National High Technology Research and Development 863 Program of China under Grant Nos. 2008AA01A303 and 2009AA01Z251.

related researches[3-7].

PCA (Principal Component Analysis) is a dimensionality-reduction technique operating on a traffic matrix that returns a compact representation of a multi-dimensional dataset by reducing traffic data to a lower dimensional subspace. The traffic matrix is usually formed by OD (Origin-Destination) flow traffic volume/feature from network-wide traffic. Although PCA-subspace method can be effectively used for network-wide anomaly detection, Daniela *et al.*[4] showed the poor detection ability when using PCA-subspace method for a single link traffic detection.

The reason that PCA can be well suited for network-wide traffic is based on such an observation: an anomaly of an OD flow will propagate on all links of the OD flow traversed, i.e., specific traffic feature for all links will change in the same way caused by an anomaly. Hence, applying PCA on all these links will make the anomaly stand out. However, applying PCA for all traffic features of a single link is not appropriate since some anomalies will cause some traffic feature values to increase and other feature values to decrease at the same time. The results of PCA may make them cancel out each other.

Most work nowadays focuses on flow-level data. At least five minutes delay is needed even for the online detection methods, so anomaly detection methods based on flow-level data are mostly used for the warning/alerting to the network manager and hard to be used for the next generation intrusion detection system design. An ideal IDS (Intrusion Detection System), besides warning, will identify the anomaly packet in real time. Hence, exploring detection methods based on packet-level data is still needed. Our work in this paper mainly focuses on anomaly detection and identification for the packet-level traffic data.

At the same time most work just triggers an alarm for an anomaly. However, once an alarm is raised, a root cause analysis needs to be performed in order to tackle anomalies. Root cause analysis is normally left to network operators, who use their intuition and knowledge to analyze traffic trace where the anomaly was flagged in search of events that can explain the anomaly. This manual process is time-consuming and error-prone. In a large network with hundreds of links, the number of events that trigger alarms may easily overwhelm the network operations center. Under such a condition, the operator is very likely to ignore alarms or even not to deploy the detection system in the first place. In our work, by a two-phase process, B6-SVM shows potential in identifying possible types for known anomalies automatically.

In this paper, different from most work focusing on flow-level traffic, we focus on the packet-level traffic on a single link. We view traffic features from six dimensions (source and destination addresses and ports, packet size, flow) and pinpoint anomalies in a multi-dimensional view using SVM (support vector machine), whose core idea is to transform the traffic anomaly detection issue to an SVM-based classification decision issue. Our work differs from previous SVM-based anomaly detection techniques by modeling the values of 6-dimensional traffic features entropy, and further by two-phase classification, we can identify anomaly types automatically during detection. We call our detection method B6-SVM. Our contributions are three fold: 1) we propose an entropy-based SVM method used for a single link packet-level traffic realtime anomaly detection; 2) we propose two new traffic features (packet size, flow) to assist detection which can detect some anomalies undetected only by four traffic features; 3) we propose a framework for identifying known anomalies automatically by a two-phase SVM classification.

Our work begins with the observation of the diurnal pattern of traffic volume and feature entropy, which shows the volume and entropy change with time and there is no fixed base model for normal traffic. However, packet-level traffic feature entropy values for a short interval are relatively constant and we can detect anomalies based on the model in that interval. We analyze traffic measurements from two famous academic and research networks: China Educational & Research NETwork (CERNET) and the Tsinghua University campus NETwork (TUNET). We find B6-SVM is an effective way to detect and identify a wide range of important anomalies. The power of B6-SVM is shown by 1) the successful detection and identification of anomalies injected into the traffic; 2) the discovery of new anomalies that we had not anticipated, and low false alarm rate in detecting anomalies. We believe our methods are practical. Our objective is to initially test the proposed B6-SVM framework using offline analysis of large datasets, and to subsequently deploy a real-time classification system in the future.

The rest of the paper is organized as follows. In Section 2, we introduce our experimental data. In Section 3, we show the diurnal pattern of traffic volume and feature entropy and introduce entropy based on packets unit series. In Section 4, we elaborate on the utility of traffic feature distributions for diagnosing anomalies. In Section 5, we describe our anomaly diagnosis methodology using SVM. In Section 6, we manually inject previously identified anomalies into our traffic to demonstrate the sensitivity of our method. We also show the ability to find new anomalies with B6-SVM. Finally, we survey related work in Section 7 and conclude this paper in Section 8.

2 Data

2.1 Network Environment

The network environment where the traffic measurement used for the analysis is composed of TUNET and core nodes of CERNET. CERNET is the largest and first nation-wide education and research computer network in China, and also one of China's major backbone networks. CERNET is the sole fixed network access for students in Chinese university and college campuses. More than 1 500 research and education institutions, and 20 million users among 31 provinces have connected to CERNET, constructing a four-level hierarchy: campus network, province network, regional network and national backbone. Five external links connect the CERNET from Beijing to Hong Kong and other countries, and the international links aggregate bandwidth is over 11 Gbps.

TUNET is the biggest campus network in China and also one of the biggest campus networks in the world, which has more than 400 sub-nets and connects more than 50 000 computers. TUNET owns more than two class B global IPv4 addresses since April 4, 2005. TUNET is designed as 3-layer topology: core layer (C), distribution layer (D) and access layer (A), where the core layer is composed of six high-end routers that are configured with 10 Gbps interfaces, two of which are connected to CERNET with 1 Gbps link. The traffic collection environment is shown in Fig.1.

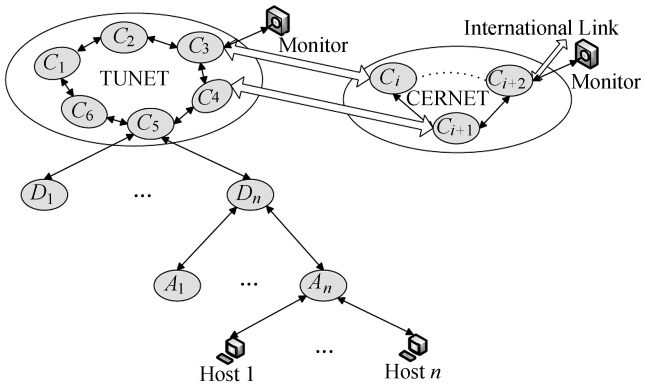


Fig.1. Traffic measurement environment.

2.2 Datasets

In our paper, we use packet (header) traces collected by two monitoring devices. One of the monitors is set on one link from TUNET to CERNET (1 Gbps). The other one is set on one CERNET international link (from China to USA — 2 Gbps). All traces are fully captured without sampling for accurate analysis. Each trace file is a collection of raw packet data which is of 40 B length. Each raw packet data's format is as below:

```
unsigned int timestamp_high;
unsigned int timestamp_low;
unsigned short mac_port;
unsigned short pkt_length;
unsigned short eth_proto;
unsigned char pkt_data[40];
```

The 40 B `pkt_data` consists of IP and TCP header. We collected one week of traffic by the two monitors for the period from July 4, 2007 to July 10, 2007.

3 Packets Unit Entropy

3.1 Why Choose Packets Unit Entropy?

Shannon introduced information entropy to capture the degree of concentration or dispersal of a distribution of a sample. We start with an empirical process $X = \{n_i;\, i = 1, \ldots, N\}$, meaning that the feature i occurs n_i times in this sample. Then the sample's entropy is defined as:

$$H(X) = -\sum_{i=1}^{N} \left(\frac{n_i}{S}\right) \log\left(\frac{n_i}{S}\right), \tag{1}$$

where $S = \sum_{i=1}^{N} n_i$ is the total number of observations of X. The value of entropy lies in the range $(0, \log N)$. The entropy value takes on the value 0 when the distribution is maximally concentrated, i.e., there is just one observable feature ($N = 1$). Sample entropy value takes on the value $\log N$ when the distribution is maximally dispersed, i.e., $n_1 = n_2 = \cdots = n_N$.

Hence, entropy can be computed on a sample of consecutive packets. Since our traces are fine-grained packet-level data instead of coarse-grained flow-level data, instead of computing entropy by an aggregated time-bin traffic as [2-7] (for example, every 5 minutes), we compute each entropy of traffic features using the same methodology as [8] which sets a sliding window of a fixed packet number W. The window size W is a tunable parameter that controls how smoothing of short-term fluctuations the detector will do. Increasing W will reduce the variation in entropy and may reduce false positives rate resulting from brief and presumably insignificant anomalies. However, W should be kept small enough in order that attacks can be detected timely. Based on the recommendation of [8] and our experiments, we use a window size of 10 000 packets. That is, $W = S = 10\,000$. We compute the first S packets feature entropy for a traffic trace and then move to the next S packets. We define the consecutive S packets as a packet unit.

The trick is that S is fixed using packets unit entropy, while S is variant using fixed interval entropy.

When S is fixed, the entropy values with the same S value can indicate the real degree of dispersal or concentration accurately, but if S is variant, entropy values with different S values may have some deviations for indicating the real degree of dispersal or concentration — for example, if there are 10^4 packets for an interval and 10^5 packets for the next interval, the maximum entropy values are $\log 10^4$ and $\log 10^5$ respectively — we cannot say the larger one is more dispersed than another due to different S values. Sometimes smaller entropy value may be comparatively more dispersed than the bigger one due to much smaller S value. Hence the packets unit entropy is much more accurate and proper to capture the degree of dispersal or concentration of a distribution of a sample. For the flow-level traffic it is difficult to compute entropy using the same methodology as packet-level data, hence [9] uses $H(X)/\log(N)$ to compute the normalized entropy (between zero and one).

3.2 There Is No Fixed Base Model

From observations of one week datasets from CERNET and TUNET, we find the diurnal pattern of traffic viewing from both volume and feature entropy (we use source IP entropy to illustrate). Fig.2 illustrates the traffic pattern of CERNET traffic datasets on July 7, 2007. Fig.2(a) shows packets number per minute in a day, Fig.2(b) shows fixed time interval entropy values per millisecond in a day, and Fig.2(c) shows packets unit series entropy values per 10 000 packets in a day.

From Fig.2 we find that the volume and entropy have similar patterns in a day, i.e., entropy tends to increase when volume for a fixed interval increases, which means that anomalies showing unusual traffic volumes will also sometimes show unusual entropy values. Thus some anomalies detected on the basis of traffic volume are also detected on the basis of entropy changes. Another important observation is that entropy changes with time, and there is no fixed model relying on short period entropy values for normal traffic. A normal entropy value for one interval maybe indicates as being abnormal for another interval. Baseline model relying on short period must be periodically retrained to capture evolving trends in the underlying data for detecting anomalies.

But for a short consecutive interval the normal traffic entropy value is comparatively steady especially for packets unit entropy (from Fig.2 we can see packets unit entropy is the most steady one). Hence, we can train a model for a fixed interval (in minute scale) to identify anomalies as deviations from it at that interval, and re-train the base model for the next interval.

4 Feature Distributions

Lakhina *et al.*[2] analyzed traffic feature distributions for various anomalies in their work. We here do not intend to repeat their work. While Lakhina *et al.*[2] focus on four traffic features, we extend traffic features to six dimensions, i.e., source IP addresses (srcIp), destination IP addresses (dstIp), source port (srcPort), destination port (dstPort), packet size (psize) and the flow (here a flow is defined as a symbol of the 5 tuples instead of flow size). The reason is that anomalies will also cause the change of packet size and flow distributions of normal traffic and we can identify them using these changes. They can help us identify some mixed attacks which may not be detected only by using other four dimensions[2]. We will illustrate further in Section 6. Notice here flow distributions is not the same as flow length distributions for an interval. For example, if all 1-packet flows in an interval, flow distribution is maximally dispersed and flow length distribution is maximally concentrated.

Fig.3 illustrates an example of how feature distributions change as the result of a traffic anomaly — in this case, a DDoS attack occurs in traffic. Two extended traffic features are illustrated: packet size in the upper half of the figure, and flow in the lower half of the figure. Each plot shows a distribution of features found in a

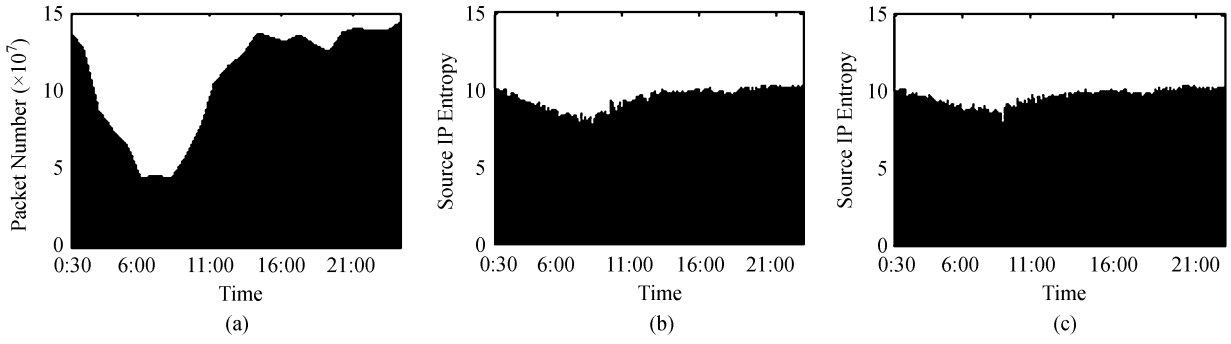


Fig.2. Volume and entropy diurnal pattern. (a) Time series volume. (b) Time series entropy. (c) Packets unit series entropy.

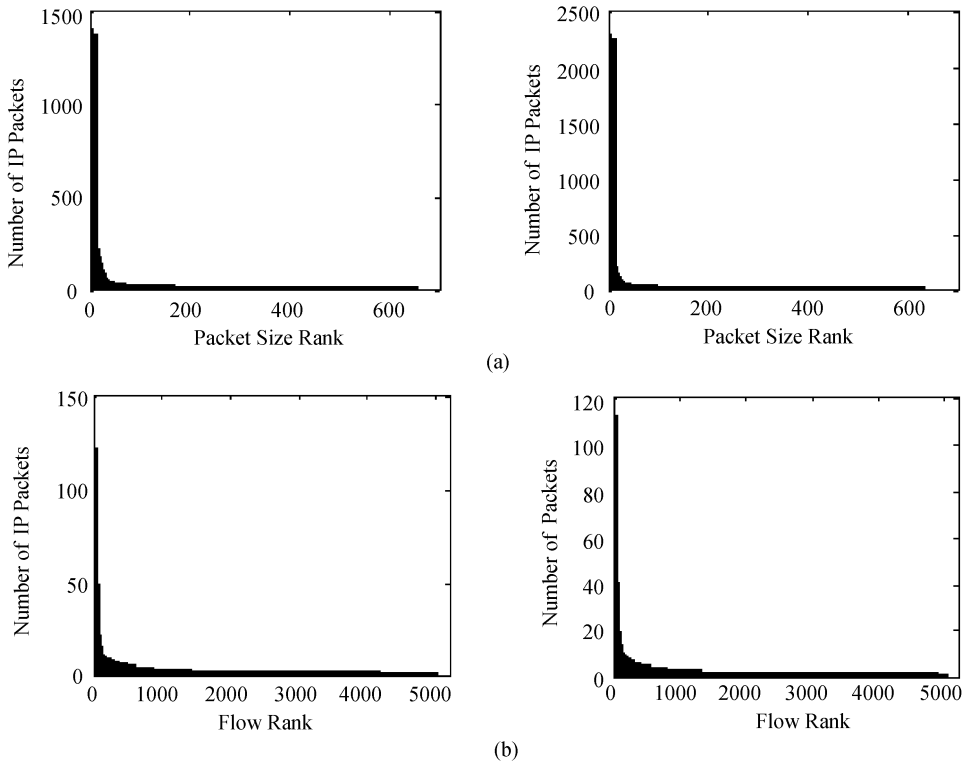


Fig.3. Distribution changes induced by DDoS. (a) Concentrated packet size. (b) Dispersed flow.

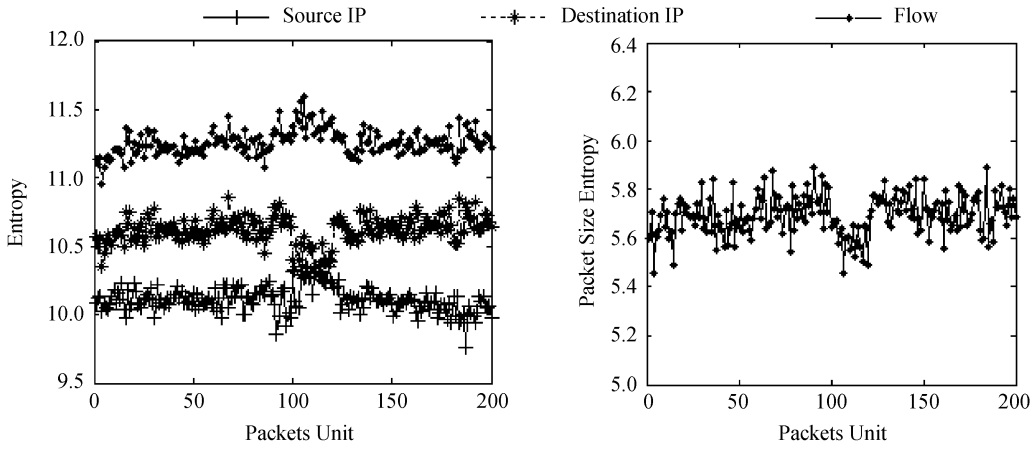


Fig.4. Features entropy changes induced by DDoS attack.

packets unit (10 000 consecutive packets). Distributions are plotted as histograms over the set of features in decreasing rank order. On the left in each case is the distribution during a normal period, and on the right is the distribution during a period with DDoS attack. From Fig.3(a), we can see the distribution is much more concentrated during the anomaly than during normal conditions for packet size. The reverse effect occurs with respect to flow. From Fig.3(b), we can see the flow distribution becomes more dispersed during DDoS attack. The reason is that DDoS attack leads to many 1-packet flows and also many fixed length (40 B) packets, which leads to flow distribution dispersion and packet size distribution concentration.

At the same time DDoS attack will make srcIp disperse and dstIp concentrate. Fig.4 shows the 4-dimensional entropy change of 200 consecutive packet units during DDoS attack (packets unit #100∼#120). We can see clearly that entropy values deviate from normal level during anomalous time. Many other anomalies, such as worms, alpha flows, flash crowd, scan, will also partially effect the six features' entropy values, which

is shown in Table 1. Thus, we can identify anomalies as deviations from the normal feature entropy values for an interval viewed from the six dimensions.

Table 1. Qualitative Effects on Feature Distributions by Various Anomalies

Anomaly	H_{srcIp}	H_{dstIp}	$H_{scrPort}$	$H_{dstPort}$	H_{flow}	H_{psize}
Alpha Flows	−	−	o	o	−	o
DDoS	+	−	o	o	+	−
DoS	−	−	o	o	−	−
Flash Crowd	o	−	+	−	+	o
Port Scan	o	−	o	+	+	−
Network Scan	o	+	o	−	+	−
Point to Multipoint	−	+	−	+	+	o
Worms	o	+	o	−	+	o

Note: '+' indicates entropy value increase, '−' indicates entropy value decrease, 'o' indicates entropy value relatively constant (maybe a little increase or a little decrease).

5 Diagnosis Methodology

Our anomaly diagnosis methodology leverages these observations about six features entropy to detect and classify anomalies. We introduce SVM classification method and show how it can be used to detect anomalies across multiple traffic features. Classification is used to learn a model/classifier from a set of labeled data instances (training) and then, classify a test instance into one of the classes using the learned model (testing). Classification-based anomaly detection techniques operate in a two-phase fashion: 1) the training phase trains a classifier using the available labeled training data; 2) the testing phase classifies a test instance as normal or anomalous, using the classifier.

5.1 SVM

SVM is one of the most actively developed classification method in data mining and machine learning. SVM provides salient properties such as the margin maximization and nonlinear classification via "kernel tricks", and is proven to be effective in many real-world applications[10-11]. SVM classifier is a machine learning approach based on the structural risk theory introduced by Vapnik in [10]. The use of SVM has showed encouraging results, achieving a higher classification accuracy compared with other machine learning techniques on very high volumes of backbone traffic traces[12]. [12] concluded that SVM is the best choice for the tradeoff between classification accuracy and computational performance.

We summarize the construction of an SVM classifier as follows. We consider a set of labeled samples represented by $(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_n, y_n)$, where $\boldsymbol{x}_i \in \Re^d$ denotes a n-dimensional vector and $y_i \in \{-1, +1\}$ is the label associated to it. SVM training process produces a linear decision boundary (optimal hyperplane) which can separate two classes (+1 and −1). It is formulated by minimizing the training error while maximizing the separating margin as illustrated in Fig.5. The optimization is usually solved through the Lagrange dual, which can be reformulated as:

$$\max\Big(\frac{1}{2}\sum_{i}^{n} a_i - \frac{1}{2}\sum_{i,j}^{n} a_i a_j y_i y_j \boldsymbol{x}_i \boldsymbol{x}_j\Big)$$

$$\text{subject to} \quad 0 \leqslant a_i \leqslant C, \sum_{i}^{n} a_i y_i = 0,$$

where $(a_i)_{i\in n}$ are lagrangian multipliers computed during the optimization for every training sample and C is a tradeoff parameter between margin and error. This process chooses a fraction of training samples $\boldsymbol{x}_i$ that have $a_i > 0$, and these samples are called support vectors, which are used to define the decision boundary. This formulation works only for linearly separable classes.

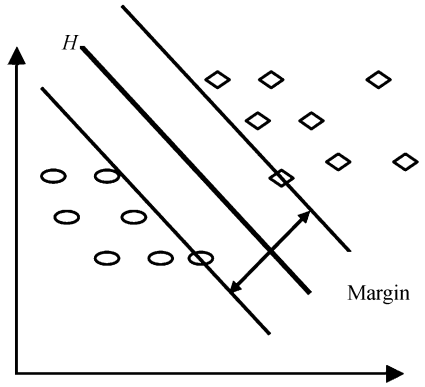


Fig.5. Optimal separating hyperplane.

However, since not all classification problems can be solved by a linear classifier for real data, an extension is needed to non-linear decision surfaces. To solve the problem, the dot product $(\boldsymbol{x}_i \cdot \boldsymbol{x}_j)$ in the linear algorithm is replaced for a non-linear kernel function $K(\cdot)$, where $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \phi(\boldsymbol{x}_j)$ and ϕ is a feature mapping function to a high-dimensional space ϕ: $\Re^d \mapsto H$. Such a replacement is called "kernel trick", which enables the linear algorithm to map the data from the original space $\Re^d$ to some different space H called feature space. Non-linear SVMs can be generated in the feature space, since linear operations in the feature space are equivalent to non-linear operation in input space. Finally, the decision function derived by SVM classifier for training samples $\boldsymbol{x}_i$, a test sample $\boldsymbol{x}$, and a bias term b can be computed as follows for a two-class problem:

$$\text{sign}\,(f(\boldsymbol{x})), f(\boldsymbol{x}) = \sum_{i}^{n} a_i y_i K(\boldsymbol{x}_i, \boldsymbol{x}) + b.$$

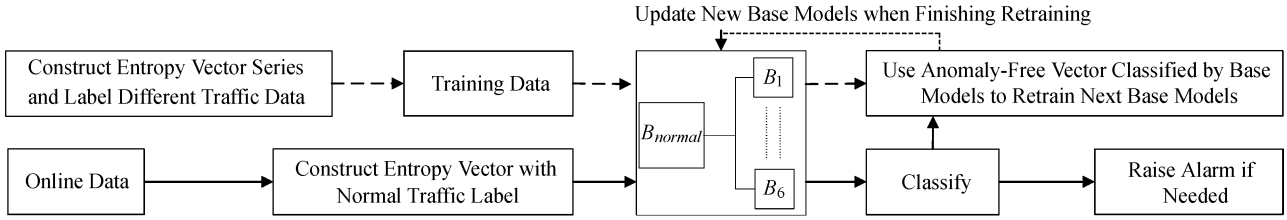


Fig.6. Roadmap of anomaly detection framework using B6-SVM. The dashed lines illustrate the training processes. The solid lines illustrate the detection processes.

One-class SVM is an unsupervised classification proposed by Scholkopf *et al.*[13] for estimating the support of a high-dimensional distribution. One-class SVM algorithm first maps input data into a high dimensional feature space via a kernel function and treats the "origin" as the only example from other classes. It then iteratively finds the maximal margin hyperplane that best separates the training data from the "origin". Solving one-class SVM problem is equivalent to solving the dual quadratic programming (QP) problem:

$$\min\Big(\frac{1}{2}\sum_{i,j}^{n} a_i a_j K(\boldsymbol{x}_i, \boldsymbol{x}_j)\Big)$$

$$\text{subject to} \qquad 0 \leqslant a_i \leqslant \frac{1}{\upsilon n},\ \sum_{i}^{n} a_i = 1,$$

where a_i is a Lagrange multiplier, and υ is a parameter that controls the tradeoff between the distance of the hyperplane from the origin and maximizing the number of data points contained by the hyperplane. We find $\upsilon = 0.2$ is a good tradeoff during many experiments for different υ values. After solving for a_i, we can use a decision function to classify data. The decision function is: $f(\boldsymbol{x}) = \text{sign}(\sum_i a_i K(\boldsymbol{x}_i, \boldsymbol{x}) - \rho)$, and the offset ρ can be recovered by $\rho = \sum_j a_j K(\boldsymbol{x}_j, \boldsymbol{x}_i)$.

Multi-class SVM is an extension from two-class SVM. Multi-class SVM can be used in either one-against-all or one-against-one fashion. Each class is trained separately against the union of all other classes for the one-against-all technique. Applying the trained SVMs on a test data point $(\boldsymbol{x}_{ij}, \boldsymbol{y}_{ij})$ yields a vector of prediction scores $(g_1, g_2, \ldots, g_c)_{ij}$, where c is the number of classes. For one-against-one technique, each class is trained separately against each other class. Applying the trained SVMs to test data yields a vector of prediction scores $(g_1, g_2, \ldots, g_b)_{ij}$ where $b = c(c-1)/2$.

5.2 B6-SVM Construction via SVM

For accurate detection, we use SVM in a multi-dimensional way. The main idea of B6-SVM is to identify anomalies as deviations from the normal traffic behavior viewed from six dimensions as illustrated in Fig.6. In training process, we first need to construct entropy vector series from raw traffic data. An entropy vector $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{i6})$ is constructed by calculating the 6-dimensional traffic features' entropy value of 10 000 consecutive packets, i.e., $(H_{srcIp}, H_{dstIp}, H_{scrPort}, H_{dstPort}, H_{flow}, H_{psize})$. We get base models from training n samples of the 6-dimensional entropy vectors. As we illustrated in Subsection 5.1, the training entropy vector series must have a label to indicate which class it belongs to. Let $\boldsymbol{X}$ be a set of input sequences (entropy vectors) and let Y be the corresponding set of sequences of labels. The data $(Y, \boldsymbol{X})$ consist of n samples of entropy vector $(y_i, \boldsymbol{x}_i) = (y_i, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6})$, $i = 1, \ldots, n$. The training samples number n is a tunable parameter that controls training time and predicts precision. In practice, we found $n = 200$ is a good tradeoff.

We construct seven base models from training data. We call these models $B_{normal}, B_1, \ldots, B_6$ which are shown in Table 2. B_{normal} trained from anomaly-free traffic is used for one-class classification and is labeled

Table 2. Qualitative Effects on Feature Distributions by Various Anomalies

Model	Entropy Vector Series
B_{normal}	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots\rangle$
B_1	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1} - \alpha, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(1, x_{i1} + \alpha, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots\rangle$
B_2	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1}, x_{i2} - \alpha, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(1, x_{i1}, x_{i2} + \alpha, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots\rangle$
B_3	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1}, x_{i2}, x_{i3} - \alpha, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(1, x_{i1}, x_{i2}, x_{i3} + \alpha, x_{i4}, x_{i5}, x_{i6}), \ldots\rangle$
B_4	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1}, x_{i2}, x_{i3}, x_{i4} - \alpha, x_{i5}, x_{i6}), \ldots,$ $(1, x_{i1}, x_{i2}, x_{i3}, x_{i4} + \alpha, x_{i5}, x_{i6}), \ldots\rangle$
B_5	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5} - \alpha, x_{i6}), \ldots,$ $(1, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5} + \alpha, x_{i6}), \ldots\rangle$
B_6	$\langle(0, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6}), \ldots,$ $(-1, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6} - \alpha), \ldots,$ $(1, x_{i1}, x_{i2}, x_{i3}, x_{i4}, x_{i5}, x_{i6} + \alpha), \ldots\rangle$

Note: '+' indicates entropy value increase, '−' indicates entropy value decrease, '0' indicates value relatively constant (maybe a little increase or a little decrease).

with one label 0. $B_1, \ldots, B_6$ are trained by the combination of anomaly-free traffic (label 0), low entropy (label -1) and high entropy (label 1) traffic in one dimension respectively. We do not simply use one-dimensional feature (for example, $\langle(0, x_{i3}), (-1, x_{i3} - \alpha), (1, x_{i3} + \alpha)\rangle$ for B_3) to train $B_1, \ldots, B_6$ because the 6-dimensional features are not independent from each other, which we can see from Fig.4. Hence, we cannot simply diagnose anomalies by absolute high or low entropy values in one dimension.

The parameter α in Table 2 controls the tradeoff between detection rate and false alarm rate. Too low value will generate much more false alarms although the detection rate is high; on the contrary, too high value will miss many anomalies though the false alarm rate is low. In practice, we found $\alpha = 0.6$ for $B_1 \sim B_4$ and $\alpha = 0.4$ for B_5 and B_6 is a good trade-off between detection rate and false alarms.

B_{normal} is used for one-class classification. One-class classification based anomaly detection techniques assume that all training instances have only one-class label. Such techniques learn a discriminative boundary around the normal instances using a one-class classification algorithm. Any test instance that does not fall within the learned boundary is declared as anomalous.

$B_1, \ldots, B_6$ are used for three-class classification. Previous work[14] has indicated that the one-against-one approach yields slightly more accurate results and faster SVM training. Further, training time of any of $B_1, \ldots, B_6$ is only 2 instead of 3. For example, $\langle(-1, x_{i1}, x_{i2}, x_{i3} - 0.6, x_{i4}, x_{i5}, x_{i6}), (1, x_{i1}, x_{i2}, x_{i3} + 0.6, x_{i4}, x_{i5}, x_{i6})\rangle$ need not be trained for B_3. Although one-against-one training is conducted twice, each time only the data points in two classes are involved. Besides, the one-against-one training can be done in parallel.

5.3 Kernel Function Choices

The feature vectors need not be computed explicitly when using kernel functions, which can greatly improve computational efficiency since we can directly compute the kernel values and operate on their images. Some common kernels are polynomial, linear and Gaussian radial basis function (RBF) kernels.

Generally speaking, RBF kernel ($K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-\gamma\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2)$, with the hyper parameter $\gamma > 0$) is a first choice because it is effective and robust for a wide range of applications[15]. RBF kernel non-linearly maps samples into a higher dimensional space. Hence, unlike the linear kernel, RBF can handle the case when the relation between attributes and class labels is non-linear. Further, the number of hyper parameters can influence the complexity of model selection, RBF kernel has less hyper parameters and has fewer numerical difficulties[10].

The marginal Gaussian distributions of traffic data is also an important reason to choose Gaussian RBF kernel. The Gaussian RBF kernel will map data from the original input space to some different feature space. The outliers whose marginal distributions violate Gaussian will be easily classified in the new feature space. Actually we tried all kinds of kernel functions in our experiment and found Gaussian RBF kernel to be the best choice in classification.

5.4 Parameter Optimization

An efficient model selection is needed for generating highly performance SVM classifiers capable of dealing with continuous updates of training data. The model selection consists of two main phases: the searching phase and test phase. The searching phase needs to solve an optimization whose goal is to find optimal values for the SVM hyper-parameters (C and γ) with respect to selection criterion. γ affects the width of the Gaussian functions of the RBF kernel and C is a penalty parameter for classification errors. The criterion is an objective function $\mathcal{F}$ evaluated over a training dataset $\mathcal{D}$ in terms of the cross-validation error ϵ. Our model parameter selection problem takes on the following form $\min(\varepsilon((C, \gamma), \mathcal{D}))$. The test phase is used to the production and evaluation on a test set of the final SVM-model created, based on the optimal hyper-parameters set found in the searching phase.

A common way of searching phase is to divide the dataset into two parts. One part is considered unknown. Prediction accuracy obtained from the unknown set can reflect the performance classifying an independent dataset more precisely. The procedure is known as cross-validation. In "v-fold" cross-validation, we can separate the training set into v subsets with equal size. One subset can be tested using the classifier trained on the remaining $v - 1$ subsets. We choose "5-fold" cross-validation in practice. Each instance of the whole training set is predicted once, hence the cross-validation accuracy is the percentage of data that are correctly classified. Further, the cross-validation can prevent the overfitting problem[15].

We use "grid-search" on γ and C for cross-validation. Various pairs of (C, γ) values are tested and the pair with the best cross-validation accuracy is chosen. Since a complete grid-search construction may still be time-consuming, we use a "coarse grid" first. A "finer grid" search on the region can be conducted after identifying a better region on the grid. We try exponentially growing sequences of γ and C to find optimal parameters. For every new training dataset $\mathcal{D}$, we need a cross-validation process to get an optimal (C, γ) values

meeting $\min(\varepsilon((C,\gamma),\mathcal{D}))$.

5.5 Detection and Identification Using B6-SVM

In testing process we first construct a packets unit entropy vector from the online data (Fig.6), then we predict the vector by B_{normal}. Any deviations from B_{normal} will be further tested by $B_1 \sim B_6$ in parallel. We can check the output value of $B_1 \sim B_6$ to identify the probable attacks. Hence our model can not only detect anomalies but also identify the anomalous types roughly by the classification results. Notice that we need to retrain base models using the coming 200 normal data points tested by current base models for the next time interval. Once the retraining process is over, we need to update the base models.

```
01:  For i = 1:6
02:    c_i = classification result of B_i;
03:  C = ⟨c_1, c_2, c_3, c_4, c_5, c_6⟩
04:  If C = ⟨0, 0, 0, 0, 0, 0⟩ Then
05:    {output normal; exit; }
06:  Match C with all row vertors in Table 1;
07:  If matches exist Then
08:    {output matched anomaly types;
09:    raise yellow alarm;}
10:  Else
11:    {raise red alarm to indicate new anomaly;}
```

Fig.7. Anomaly type identification algorithm.

Fig.7 describes how to identify anomaly types based on classification results of $B_1 \sim B_6$. Let $C = \langle c_1, c_2, c_3, c_4, c_5, c_6\rangle$ indicate the classification results. $c_i \in \{-1, 1, 0\}$, $c_i = -1$ matches "$-$" in Table 1, $c_i = 1$ matches "$+$" in Table 1 and $c_i = 0$ matches any of $\{-, +, \mathrm{o}\}$ in Table 1. We reduce false alarms by classifying the deviations detected by B_{normal} as normal for the output results $C = \langle 0, 0, 0, 0, 0, 0\rangle$. For $C \neq \langle 0, 0, 0, 0, 0, 0\rangle$, we match C using Table 1 following the below rules: 1) if only one row in Table 1 is matched, we identify corresponding anomaly, for example, if $C = \langle 0, -1, 0, 1, 0, 0\rangle$, it will be a port scan; 2) if more than one row in Table 1 are matched, we identify corresponding mixed anomalies, for example, if $C = \langle 0, -1, 0, 0, 1, 0\rangle$, it may be a port scan and flash crowd mixed attack; 3) if none row is matched, it may be new anomalies, for example, $C = \langle 1, -1, -1, 0, 0, 0\rangle$.

5.6 Computational Complexity

The quadratic problem must be solved and the support vectors must be chosen when training with SVM. There are two lower bounds on the computational cost of solving the SVM QP problem for arbitrary matrices $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \phi(\boldsymbol{x}_i) \cdot \boldsymbol{\phi}(x_j)$. One is proportional to R^3 which R is the number of support vectors, the other is proportional to nV where n is number of samples and V is the number of support vectors[10].

Hence the training complexity of SVMs is highly dependent on the size of a dataset. SVM training time is typically super-linear in the number of training samples, so learning a smaller training set for the same data is a net win. If we train features for every packet it will need $n = 200 \times 10\,000$ samples. Since one sample was constructed by 10 000 packets for our models, there are only 200 samples needed for training on entropy values. Hence, training on entropy values instead of directly on packet features will decrease the training time dramatically. The computational complexity of the testing process is proportional to the number of support vectors and the features which is usually small.

On a dual 2.80 GHz CPUs and 2 GB memory 32-bit Intel x86 architecture, training and creating the machine using 200 data points requires time less than 30 seconds. Hence, the online data will be tested by the base model trained from the traffic a few seconds before. From our observations we note that all features entropy values keep relatively steady in minute scale. We train next base models online using the coming 200 normal data points tested by current base models. We update models for new time series detection once new models are generated. Once trained, the prediction of latency to an arbitrary data point takes time about two microseconds, i.e., B6-SVM can detect about 0.5 million packets per second. The statics on our datasets shows that average packet size is usually large than 500 B. Thus, B6-SVM can test about $5\times10^5\times500\times8 = 2\times10^9$ bits per second, i.e., B6-SVM can be implemented on 2 Gbps link for realtime detection theoretically.

6 Evaluation

In this section, we evaluate the B6-SVM using the datasets introduced in Section 2. As an implementation of SVMs, we use the LIBSVM package[16]. LIBSVM is an integrated software for SVM classification, regression and distribution estimation. LIBSVM supports multi-class and one-class classification. The whole classification process includes two processes: the training process and the predicting process. During SVM training, the goal is to learn each class based on each element (data item or data point) and its corresponding label in the training set, by maximizing the separation between data points with same labels (the same class) and other data points. Many studies[10-12] have shown that SVMs tend to obtain superior results, compared with other classifiers, for predicting individual labels. The advantage of SVMs stems from its ability to use high-dimensional feature spaces via kernels and from theoretical guarantees on generalization ability.

6.1 Methodology

To evaluate detection rate and identification rate, we consider generating synthetic anomalies — normal traffic mixed with known anomalies. We select a random 200 consecutive packet units traffic from our data as our background traffic, which is assumed anomaly-free. We manually generate three anomalies with the method similar to [9] and superimpose the anomalies into our traffic respectively (we can get rid of anomalous traffic through manual analysis). The first is a port scan. The second is a classical DDoS attack and the third is a worm.

The port scan is generated by a number of attack sources scanning a fixed host's all ports sequentially, and each attack source scans using a short packet with 56-packet size. The DDoS attack is generated by a single destination address receiving traffic from a large amount of sources. Each attack source generates packet using a fixed packet size of 40 B and a single flow per packet. The worm is generated by some attack sources scanning all hosts' 80 and 8080 ports in a network for vulnerable hosts.

For evaluating our methods on varying anomaly intensities, we thin the background traffic trace by selecting one out of every N packets, then extract the anomaly and inject it into the background traffic trace. We inject the anomaly in turn into the background trace. After each injection, we apply the B6-SVM method to determine whether the injected anomaly can be detected. This allows us to compute a detection rate based on packets unit entropy.

The experimental results of [4] has hinted that PCA-subspace method is poor for a single link traffic detection, we will compare the detection rate of B6-SVM and PCA-subspace method under injecting external anomalies. To compare the detection rate of the PCA-subspace method with B6-SVM for a single link data under the same condition, we use the same training data of B6-SVM for the PCA training, and the similar way that used in [4] for single link anomaly detection. That is, we constitute a 200×6 matrix for PCA training, which denotes the packet unit series of all features. Thus, each column i denotes the packet unit series of the i-th feature and each row j represents an instance of all the features at the packet unit j. We use the PCA subspace Mark Coates[3] shared for us to evaluate. We set the confidence limit $1 - \alpha = 99.5\%$ for the PCA method. The residual is obtained by the difference between the original data and the data mapped onto the first four principal axes, which capture 96% of whole energy. We also choose the same test dataset for B6-SVM and PCA subspace method.

6.2 Results

The resulting detection and identification rates from injecting anomalies of different intensity are shown in Fig.8. From the plots we note that, compared with B6-SVM, PCA-subspace method is relatively poor for a single link traffic anomaly detection. The detection rate is very low for DDoS and port scan because these two attacks both cause two features entropy to increase and two to decrease. They will cancel out each other in the dimension reduction process of PCA. The detection result of worms is a little better because worms will cause two features entropy to increase and one to decrease. However, increasing or decreasing entropy values of different features from the normal level will cause anomaly to stand out for B6-SVM since SVM transfers the input space to a higher dimensional space to make samples linearly separable in new space by a kernel trick. Entropy changes in the opposite direction will not cancel out each other for B6-SVM. We do not test the detection rate for DoS and Alpha flow for B6-SVM

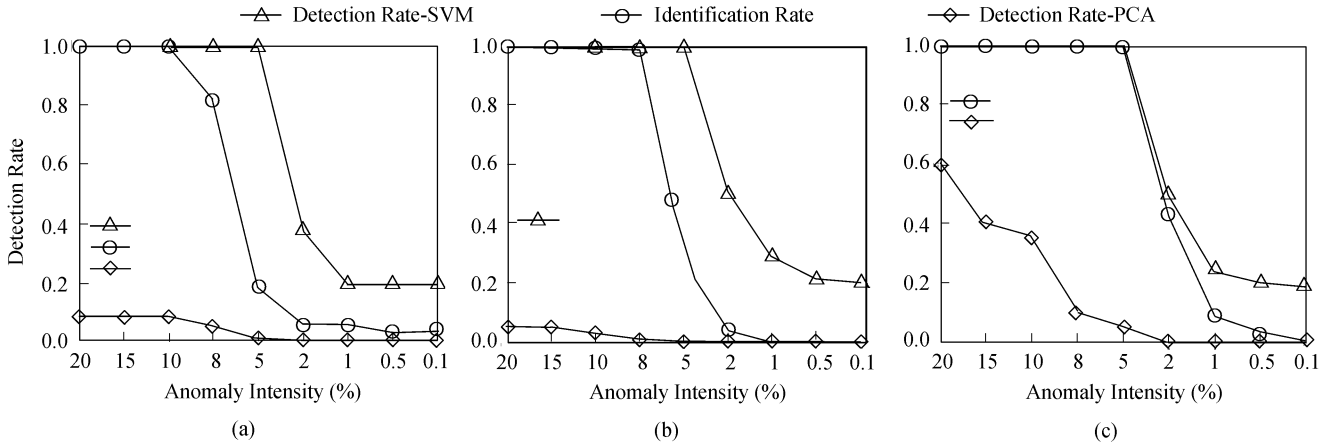


Fig.8. Detection and identification results from injecting different anomalies. (a) Port scan. (b) DDoS attack. (c) Worms.

and PCA-subspace method, however, we think they both will have a good performance for the two attacks. The reason is that DoS and Alpha flow both make the features entropy change in the same direction.

This figure also sheds light on a number of aspects of detection and identification rate of B6-SVM. First, all anomalies are easily detected when they occur at a high volume. All attacks can be detected when they comprise at least 5% of the traffic on average. Second, all known anomalies are also easily identified by B6-SVM when they occur at high volume. All port scans are identified when they comprise at least 10% of the traffic on average. All DDoS attacks are identified when they comprise at least 15% of the traffic on average. And all worm scans are identified when they comprise at least 5% of the traffic on average. Third, the identification rate is a little lower than the detection rate for same anomaly intensity. Not all anomalies detected by B_{normal} will be identified by $B_1 \sim B_6$ successfully when these anomalies are in a low intensity. The results are encouraging the use of B6-SVM for anomaly detection.

Table 3 shows all anomalies found in our datasets by B6-SVM. Totally 763 alarms are triggered for CERNET data and 611 alarms are raised for TUNET data. We manually generate mixed attacks which are composed of half port scans and half worms attacks, and we inject them into part of our traffic data with 8% intensity. They are all successfully identified by the output results of $B_1 \sim B_6$: $\boldsymbol{x} = \langle 0, 0, 0, 0, 1, -1 \rangle$. None of them can be detected by only using four traffic features (srcIp, dstIp, srcPort, dstPort)[2]. For the DoS attack and Alpha flow, we can only detect them if just using four traffic features (srcIp, dstIp, srcPort, dstPort), but the specific types cannot be identified because they both cause similar entropy changes for the four features. However, we can identify them by different changes of packet size entropy. Hence, adding flow and packet size features not only can help detect more anomalies but also assist us to identify more anomaly types compared with only using four traffic features.

Table 3. Anomalies Found in Datasets

Anomaly	CERNET	TUNET
DDoS	153	67
DoS	45	82
Alpha Flow	137	32
Port Scan	122	131
Worms	189	218
Mixed Attack	35	19
Unknown	37	28
False Alarm	45	34
Total	763	611

Altogether 37 unknown anomalies in CERNET and 28 unknown anomalies in TUNET are detected by B6-SVM. Among them 26 unknown anomalies in CERNET and 18 unknown anomalies in TUNET are detected by the output results of $B_1 \sim B_6$: $\boldsymbol{x} = \langle -1, -1, -1, -1, -1, 0 \rangle$. Through manual analysis, we find IP addresses which belong to the same subnet disappear suddenly for that anomalous period, which may indicate a switch failure at that moment. The rest 11 unknown anomalies in CERNET and 10 unknown anomalies in TUNET that cannot be identified from their patterns may be some new anomalies that we do not know yet or may be false positives. Besides, there are 45 and 34 alarms in CERNET and TUNET data triggered by the concentration of IP addresses. Through manual analysis, we find a fixed IP address emerges too many times in these packet series, and we further identify the IP address as a proxy server or NAT box. We consider these alarms arousing from middleboxes as false alarms.

Table 3 sheds light on false alarm rate. The table shows that in one week of data, only 45 anomalies in CERNET and 34 anomalies in TUNET were false alarms. This is the minimum value, because some anomalies in the unknown category might be considered false alarms if their nature were completely understood. If we take them as false alarms, there are $45 + 11 = 56$ false alarms in CERNET and $34 + 10 = 44$ false alarms in TUNET data. The false alarm rate is in the order of 7.3% for CERNET data and 7.2% for TUNET data of all detections.

The detection rate and false alarm rate are encouraging using B6-SVM. Besides, B6-SVM can automatically identify the possible known anomaly types in the detecting process, which is fallible and annoying for manual analysis.

7 Related Work

Anomaly detection is becoming a hot research topic in recent years. Some studies have been restricted to point-solutions for specific types of anomalies, e.g., port scans[17], worms[18-19], spam[20], DoS attacks[21-22], and flash crowds[23]. Some studies analyze overall traffic volume behavior, e.g., by proposing edge detection[24-25], wavelet-based signal analysis[26], or forecasting techniques[27]. They flag peaks and shifts in volumes as suspicious events on the assumption that anomalies are reflected as significant changes in traffic volumes. Traffic volumes, however, comprise natural bursts and dips that are due to legitimate applications (e.g., distributed computing, update rollouts, backups), and therefore these methods are likely to generate many false alarms.

Zhang *et al.*[28] introduced a general framework that aims to identify anomalies from network-wide link load traffic data. These studies are successful in identifying

anomalies that result in (network-wide) traffic volume deviations. However, they are not so effective in detecting stealthy attacks, such as low-rate port scanning, that do not result in notable traffic volume changes.

Feature-based anomaly detection methods seek to address the limitations of volume-based methods by examining a range of traffic features, instead of relying solely on the traffic volume. Commonly used traffic features are IP header fields. Feature-based anomaly detection methods base the observation that traffic features exhibit regular patterns under normal conditions, which may be violated by anomalies.

Gu *et al.*[29] used a single composite feature distribution to characterize network traffic and computed a parametric model of the distribution using training traffic data. The network traffic is compared to the constructed base model to identify anomalies. The authors assume that their training dataset does not contain any anomalies. Wagner *et al.*[30] studied the compressibility of different IP header fields in traffic traces. They found that the compressibility of traffic features changes drastically during well-known worm outbreaks. Ringberg *et al.*[31] introduced Web-Class, an online repository of anomaly-labeled traffic traces that researchers may use for evaluating anomaly detection techniques. Soule *et al.*[32] studied the network traffic anomalies observed in two adjacent backbone networks. They found that large-scale anomalies can leave substantially different footprints due to differences in the traffic collection infrastructure of two networks. Brauckhoff *et al.*[33] described how traffic sampling can influence the accuracy of anomaly detection systems. They showed that although sampling can influence volume-based anomaly detection metrics, it does not affect the distribution of traffic features significantly.

Scherrer *et al.*[34] introduced a long-range dependent non-Gaussian model of network traffic, and proposed an anomaly detection method that identifies significant changes in the estimated parameters of the model. Andreas *et al.*[35] proposed a histogram-based traffic anomaly detection by directly modeling the detailed characteristics of histograms, which can identify coarse-grained changes between distributions. More recently, Fernando *et al.*[36] found that "when many flows are multiplexed on a non-saturated link, their volume changes over short timescales tend to cancel each other out, making the average change across flows close to zero. This equilibrium property can be violated by many traffic anomalies." Based on this observation, they proposed a novel anomaly detection method called ASTUTE (A Short-Timescale Uncorrelated-Traffic Equilibrium).

Closer to our approach, the pioneer work by Lakhina *et al.*[2,37] introduced PCA-subspace method to identify network wide anomalies using feature distributions. The proposed anomaly detection scheme uses PCA to identify an orthogonal basis along which the measurement data exhibit the highest variance. The principal components with high variance model the normal behavior of a network traffic, whereas the remaining components of small variance can be used to identify anomalies. This technique has received a large amount of attention, and inspired related research[3-7].

Ringberg *et al.*[38] performed a study on the sensitivities of the PCA method. They illustrated how the PCA method can be sensitive to the number of principal components used to describe the normal subspace, which can limit PCA's effectiveness if not properly configured. They also showed that outages can pollute the normal subspace, a kind of perturbation to the subspace that is not adversarial. [4] shows that the sensitivities observed in [38] come from PCA's inability to capture temporal correlations. They proposes to replace PCA by a Karhunen-Loeve expansion. [7] proposes a robust defense against a malicious adversary and demonstrate its effectiveness. [5] extends the PCA-subspace method to identify the IP flow(s) that are responsible for the anomaly. [3, 6] focus on PCA-subspace method scalability and improve the spatial and temporal complexity of PCA.

Different from these studies, we focus on packet-level traffic of a single link. We find that PCA-subspace method is not fit for this case. We extend four traffic features into six to assist anomaly detection. We propose B6-SVM to detect anomaly under this condition. SVM has been widely used for classification in many areas and has been showed an efficient classification method[10-11]. [12] compares all kinds of classification methods for network traffic classification and showed SVM consistently achieved the highest accuracy, achieving a 98% of classification accuracy on very high volumes of the backbone traffic traces. At the same time, researchers have successfully applied SVM to distinguish normal from abnormal traffic[14,39-40]. Different from these studies, we propose an entropy-based SVM detection method, which dramatically decrease the training samples number for the same training dataset. Besides, our method can both detect and identify anomalies by a two-phase process.

Few studies have tried to address the problem of automating root cause analysis of traffic anomaly detectors. Lakhina *et al.*[2] proposed a clustering method of entropy residuals to classify the anomalies found by PCA anomaly detector. Li *et al.*[5] combined PCA with traffic sketches to develop "Defeat", a detector that can also identify the flows involved in the anomalies.

[41] introduces an anomaly extraction technique using "frequent itemset mining" based on histogram detection method[35]. [42] also proposes frequent itemset mining to extract anomalies based on NetRflex (PCA-subspace). [43] proposes a general root cause analysis method for all detectors called URCA (Unsupervised Root Cause Analysis) by classifying anomalies using metrics of the traffic they impact. A little different from these studies, our work is an attempt to identify anomalies during detection process instead of analyzing the results of detectors.

8 Conclusions

General network anomaly detection is an ambitious goal. In this paper, we show that although PCA is powerful for world wide traffic analysis, it is difficult to be applied for single link traffic. We propose a new feature-based anomaly detection approach — B6-SVM, which is based on modeling the characteristics of six different traffic features for diagnosing anomalies. Compared with previous feature-based anomaly detection techniques, we focus on packet-level traffic anomaly detection for a single link and propose an entropy-based SVM detection method. Our work not only extends four traffic features to six features to improve the detection capability but also proposes a framework to identify anomalies during detection by a two-phase SVM classification.

Our work provides a framework to use feature entropy to identify different anomalies. The presented methodology is generic. Table 1 can be extended with new features' entropy that might become informative for revealing new anomalies, and can be extended with new anomalies identified by B6-SVM if we can confirm their entropy changing characteristics for the designated dimensions. Our test results demonstrate B6-SVM's effectiveness in diagnosing anomalies. B6-SVM can be deployed on 2 Gbps links for packet-level real-time anomaly detection.

Our method also has some limitations. For example, in the initial training process, B6-SVM needs to learn the behavior of normal traffic. We assume that the training datasets are anomaly-free, which is harsh for the real traffic. All our following up testing and retraining processes are based on the initial models. Hence, B6-SVM is very sensitive to the initial traffic we used for training. Although we can get anomaly-free traffic through visual observations from traffic, it is harsh for randomly chosen traffic in real conditions. Our future work will focus on diminishing possible anomalies in the initial training of the base models since we have taken steps for the retraining process (we only train anomaly-free samples identified by previous base models). Possible methods may include getting rid of some heavy-hitters by setting some thresholds for different features or using some clustering methods.

References

[1] http://www.symantec.com/.

[2] Lakhina A, Crovella M, Diot C. Mining anomalies using traffic feature distributions. In *Proc. ACM SIGCOMM*, Philadelphia, USA, Aug. 22-26, 2005, pp.217-228.

[3] Ahmed T, Coates M, Lakhina A. Multivariate online anomaly detection using kernel recursive least squares. In *Proc. IEEE INFOCOM*, Anchorage, Alaska, USA, May 6-12, 2007, pp.625-633.

[4] Brauckhoff D, Salamatian K, May M. Applying PCA for traffic anomaly detection: Problems and solutions. In *Proc. INFOCOM*, Rio de Janeiro, Brazil, Apr. 19-25, 2009, pp.2866-2870.

[5] Li X, Bian F, Crovella M, Diot C, Govindan R, Iannaccone G, Lakhina A. Detection and identification of network anomalies using sketch subspaces. In *Proc. IMC*, Rio de Janeiro, Brazil, Oct. 25-27, 2006, pp.147-152.

[6] Liu Y, Zhang L, Guan Y. Sketch-based streaming PCA algorithm for network-wide traffic anomaly detection. In *Proc. the 30th International Conference on Distributed Computing Systems*, Genova, Italy, Jun. 21-25, 2010, pp.807-816.

[7] Rubinstein B I P, Nelson B, Huang L *et al.* Antidote: Understanding and defending against poisoning of anomaly detectors. In *Proc. the 9th Internet Measurement Conference*, Chicago, USA, Nov. 4-6, 2009, pp.1-14.

[8] Feinstein L, Schnackenberg D, Balupari R, Kindred D. Statistical approaches to DDos attack detection and response. In *Proc. DARPA Information Survivability Conference and Exposition* (*DISCEX*), Washington DC, USA, Apr. 22-24, 2003, pp.303-314.

[9] Nychis G, Sekar V, Andersen D G, Kim H, Zhang H. An empirical evaluation of entropy-based traffic anomaly detection. In *Proc. the 8th IMC*, Vouliagmeni, Greece, Oct. 20-22, 2008, pp.151-156.

[10] Vapnik V. The Nature of Statistical Learning Theory. New York: Springer, 1995.

[11] Burges C J C. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 1998, 2(2): 121-167.

[12] Kim H, Claffy K, Fomenkov M *et al.* Internet traffic classification demystified: Myths, caveats, and the best practices. In *Proc. ACM CoNEXT*, Madrid, Spain, Dec. 9-12, 2008, Article No.11.

[13] Scholkopf B, Platt J C, Shawe-Taylor J C *et al.* Estimating the support of a high-dimensional distribution. *Neural Computation*, 2001, 13(7): 1443-1471.

[14] Lin C H, Liu J C, Ho C H. Anomaly detection using LibSVM training tools. In *Proc. International Conference on Information Security and Assurance*, Busan, Korea, Apr. 24-26, 2008, pp.166-171.

[15] Keerthi S S, Lin C. Asymptotic behaviors of support vector machines with Gaussian kernel. *Neural Computation*, 2003, 15(7): 1667-1689.

[16] Chang C C, Lin C J. LIBSVM: A library for support vector machines, 2010, http://www.csie.ntu.edu.tw/∼cjlin/libsvm/.

[17] Jung J, Paxson V, Berger A, Balakrishnan H. Fast portscan detection using sequential hypothesis testing. In *Proc. IEEE Symposium on Security and Privacy*, Berkeley, CA, USA, May 9-12, 2004, pp.211-225.

[18] Li Z, Wang L, Chen Y, Fu Z. Network-based and attack-resilient length signature generation for zero-day polymorphic worms. In *Proc. the 15th IEEE International Conference on*

Network Protocols (*ICNP*), Beijing, China, Oct. 16-19, 2007, pp.164-173.

[19] Liu Z, Shu G, Li N, Lee D. Defending against instant messaging worms. In *Proc. GLOBECOM*, San Francisco, USA, Nov. 27-Dec. 1, 2006.

[20] Zhong Z, Ramaswamy L, Li K. ALPACAS: A large-scale privacy-aware collaborative anti-spam system. In *Proc. IEEE INFOCOM*, Phoenix, USA, Apr. 13-18, 2008, pp.556-564.

[21] Luo X, Chang R. On a new class of pulsing denial-of-service attacks and the defense. In *Proc. Network and Distributed System Security Symposium*, San Diego, California, USA, Feb. 2005.

[22] Ning P, Liu A, Du W. Mitigating DoS attacks against broadcast authentication in wireless sensor networks. *ACM Transactions on Sensor Networks*, 2008, 4(1): 1-31.

[23] Jung J, Krishnamurthy B, Rabinovich M. Flash crowds and denial of service attacks: Characterization and implications for CDNs and Web sites. In *Proc. the 11th WWW*, Honolulu, Hawaii, USA, May 7-11, 2002, pp.293-304.

[24] Krishnamurthy B, Sen S, Zhang Y, Chen Y. Sketch-based change detection: Methods, evaluation, and applications. In *Proc. the 3rd ACM IMC*, Miami, Florida, USA, Oct. 27-29, 2003, pp.234-247.

[25] Won Y J, Choi M J, Hong J W K, Kim M S, Hwang H, Lee J H, Lee S G. Fault detection and diagnosis in IP-base mission critical industrial process control networks. *IEEE Communications Magazine*, 2008, 46(5): 172-180.

[26] Barford P, Kline J, Plonka D, Ron A. A signal analysis of network traffic anomalies. In *Proc. the 2nd ACM SIGCOMM Internet Measurement Workshop*, Marseille, France, Nov. 6-8, 2002, pp.71-82.

[27] Brutlag J D. Aberrant behavior detection in time series for network monitoring. In *Proc. the 14th Systems Administration Conference*, New Orleans, Dec. 3-8, 2000, pp.139-146.

[28] Zhang Y, Ge Z, Greenberg A, Roughan M. Network anomography. In *Proc. the 5th ACM SIGCOMM Internet Measurement Conference*, Berkeley, CA, USA, Oct. 19-21, 2005, pp.317-330.

[29] Gu Y, McCallum A, Towsley D. Detecting anomalies in network traffic using maximum entropy estimation. In *Proc. Internet Measurement Conference*, Berkeley, CA, USA, Oct. 19-21, 2005, pp.45-50.

[30] Wagner A, Plattner B. Entropy based worm and anomaly detection in fast IP networks. In *Proc. the 14th IEEE International Workshops Enabling Technologies: Infrastructure Collaborative Enterprise*, Washington DC, USA, June 13-15, 2005, pp.172-177.

[31] Ringberg H, Soule A, Rexford J. Webclass: Adding rigor to manual labeling of traffic anomalies. *SIGCOMM Comput. Commun. Rev.*, 2008, 38(1): 35-38.

[32] Soule A, Larsen H, Silveira F, Rexford J, Diot C. Detectability of traffic anomalies in two adjacent networks. In *Proc. the 8th Int. Conf. Passive and Active Network Measurement*, Louvain-la-neuve, Belgium, Apr. 5-6, 2007, pp.22-31.

[33] Brauckhoff D, Tellenbach B, Wagner A, May M, Lakhina A. Impact of packet sampling on anomaly detection metrics. In *Proc. the 6th ACM SIGCOMM Conference on Internet Measurement*, ACM Press, Oct. 25-27, 2006, pp.159-164.

[34] Scherrer A, Larrieu N, Owezarski P, Borgnat P, Abry P. Non-Gaussian and long memory statistical characterizations for Internet traffic with anomalies. *IEEE/ACM Trans. Dependable and Secure Computing*, 2007, 4(1): 56-70.

[35] Kind A, Stoecklin M P, Dimitropoulos X. Histogram-based traffic anomaly detection. *IEEE Transactions on Network and Service Management*, 2009, 6(2): 110-121.

[36] Silveira F, Diot C, Taft N, Govindan R. Astute: Detecting a different class of traffic anomalies. In *Proc. SIGCOMM*, New-Delhi, India, Aug. 30-Sept. 3, 2010, pp.267-278.

[37] Lakhina A, Crovella M, Diot C. Diagnosing network-wide traffic anomalies. In *Proc. SIGCOMM*, Portland, OR, USA, Aug. 30-Sept. 3, 2004, pp.219-230.

[38] Ringberg H, Soule A, Rexford J, Diot C. Sensitivity of PCA for traffic anomaly detection. In *Proc. ACM SIGMETRICS International Conf. Measurement and Modeling of Computer Systems*, San Diego, CA, Jun. 12-16, 2007, pp.109-120.

[39] Ma J, Perkins S. Online novelty detection on temporal sequences. In *Proc. the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington DC, USA, Aug. 24-27, 2003, pp.613-618.

[40] Li K, Teng G. Unsupervised SVM based on p-kernels for anomaly detection. In *Proc. Innovative Computing, Information and Control*, Beijing, China, Aug. 30-Sept. 1, 2006, pp.59-62.

[41] Brauckhoff D, Dimitropoulos X, Wagner A, Salamatian K. Anomaly extraction in backbone networks using association rules. In *Proc. the 9th IMC*, Chicago, Illinois, USA, Nov. 4-6, 2009, pp.28-34.

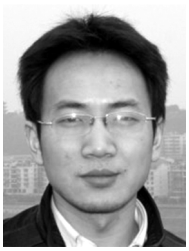
[42] Paredes-Oliva I, Dimitropoulos X, Molina M, Barlet-Ros P, Brauckhoff D. Automating root-cause analysis of network anomalies using frequent itemset mining. In *Proc. SIGCOMM* (*Poster*), New Delhi, India, Aug. 30-Sep. 3, 2010, pp.467-468.

[43] Silveira F, Diot C. URCA: Pulling out anomalies by their root causes. In *Proc. the 29th INFOCOM*, San Diego, USA, Mar. 14-19, 2010, pp.722-730.

Bin Zhang is a Ph.D. candidate in Department of Computer Science and Technology, Tsinghua University. He received the Bachelor's degree in computer software from Zhengzhou University, China in 1998 and the Master's degree in network information security from Shanghai Jiaotong University, China, in 2005. During his Ph.D. career, he has published more than 10 papers in refereed international conferences (NOMS, IM, LCN, IWQoS, APNOMS, etc) and journals (the Computer Journal, Journal of Software). His current research interests focus on traffic measurement and analysis, traffic modeling and network traffic anomaly detection.

Jia-Hai Yang received his M.S. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 1992 and 2003, respectively. He is now a full professor of Tsinghua University. He has published more than 100 papers in refereed international conferences and journals, and two books on network management and Internet measurement. His research interests include Internet architecture and its protocols, IP routing technology, network measurement, network management, etc. He also serves as Technical Program Committee (TPC) member for several international conferences. He is a member of CCF, ACM, and IEEE.

Jian-Ping Wu received the M.S. and Ph.D. degrees in computer science from Tsinghua University, Beijing, China, in 1997. He is now a full professor with the Computer Science Department, Tsinghua University. In the research areas of the network architecture, high-performance routing and switching, protocol testing, and formal methods, he has published more than 300 technical papers in academic journals and proceedings of international conferences, including IEEE/ACM Transactions on Networking, IEEE Transactions on Multimedia, IEEE ICNP, IEEE INFOCOM, etc. Prof. Wu has organized and chaired/co-chaired several international conferences, workshops, and an IETF meetings, including ICNP 2007, ICDCS 2008, etc. He also serves as TPC member for many international conferences. Professor Wu is a fellow of IEEE and a member of CCF and ACM.



Ying-Wu Zhu received the Bachelor's degree in information from Wuhan University of Technology, China in 2000. He received his Master's degree from Department of Computer Science and Technology, Tsinghua University, China in 2010. When he was a student in Tsinghua University, his interest lies in network traffic security.