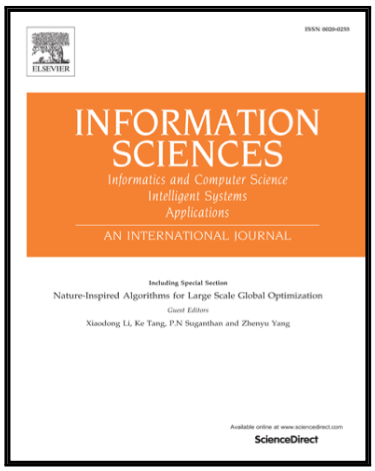


Abstracting Massive Data for Lightweight Intrusion Detection in Computer Networks

Wei Wang, Jiqiang Liu, Georgios Pitsilis, Xiangliang Zhang

PII: S0020-0255(16)31238-5
DOI: 10.1016/j.ins.2016.10.023
Reference: INS 12582

To appear in: *Information Sciences*

Received date: 14 March 2016
Revised date: 3 October 2016
Accepted date: 9 October 2016

Please cite this article as: Wei Wang, Jiqiang Liu, Georgios Pitsilis, Xiangliang Zhang, Abstracting Massive Data for Lightweight Intrusion Detection in Computer Networks, *Information Sciences* (2016), doi: 10.1016/j.ins.2016.10.023

Abstracting Massive Data for Lightweight Intrusion Detection in Computer Networks

Wei Wang [a], Jiqiang Liu [a], Georgios Pitsilis [b], Xiangliang Zhang [c]

[a] *School of Computer and Information Technology, Beijing Jiaotong University*
No.3 Shangyuancun, 100044 Beijing, China

[b] *Computer Science Research, Athens, Greece*

[c] *Division of Computer, Electrical and Mathematical Sciences & Engineering*
King Abdullah University of Science and Technology (KAUST), Saudi Arabia

Abstract

Anomaly intrusion detection in big data environments calls for lightweight models that are able to achieve real-time performance during detection. Abstracting audit data provides a solution to improve the efficiency of data processing in intrusion detection. Data abstraction refers to abstract or extract the most relevant information from the massive dataset. In this work, we propose three strategies of data abstraction, namely, exemplar extraction, attribute selection and attribute abstraction. We first propose an effective method called exemplar extraction to extract representative subsets from the original massive data prior to building the detection models. Two clustering algorithms, Affinity Propagation (AP) and traditional k-means, are employed to find the exemplars from the audit data. K-Nearest Neighbor (k-NN), Principal Component Analysis (PCA) and one-class Support Vector Machine (SVM) are used for the detection. We then employ another two strategies, attribute selection and attribute extraction, to abstract audit data for anomaly intrusion detection. Two http streams collected from a real computing environment as well as the KDD'99 benchmark data set are used to validate these three strategies of data abstraction. The comprehensive experimental results show that while all the three strategies improve the detection efficiency, the AP-based exemplar extraction achieves the best performance of data abstraction.

Key words: Data reduction, intrusion detection, anomaly detection, computer security

1 Introduction

The importance of computer network security is growing with the pervasive involvement of computers in people's daily lives and in business processes within most organizations. As an important technique in the defense-in-depth network security framework, intrusion detection has become a widely studied topic in computer networks in recent years.

In general, the techniques for intrusion detection can be categorized as signature-based detection and anomaly detection. Signature-based detection (e.g., Snort [31]) relies on a database of signatures from known malicious threats. Anomaly detection, on the other hand, defines a profile of a subject's normal activities and attempts to identify any unacceptable deviation as a potential attack. Typically, machine learning techniques are used to build normal profiles of a subject. Any observable behavior of a system, such as a network's traffic [13,19], a computer host's operating system [11,36] or a mobile application [2,39], can be used as the subject information.

Anomaly detection has a potential to detect unforeseen attacks. As new attacks appear very frequently and signature-based detection methods may be overwhelmed by an abundance of polymorphic attacks, using anomaly detection sensors to discover zero-day attacks has become a necessity rather than an option [8]. We are entering the era of "big data" [23]. The increasing volume of information generated by enterprises, the rise of social media and the Internet are fueling an exponential growth of data. Anomaly intrusion detection techniques

Email addresses: wangwei1@bjtu.edu.cn, http://infosec.bjtu.edu.cn/wangwei (Wei Wang), jqliu@bjtu.edu.cn, http://infosec.bjtu.edu.cn/jqliu (Jiqiang Liu), georgios.pitsilis@gmail.com (Georgios Pitsilis), xiangliang.zhang@kaust.edu.sa, http://mine.kaust.edu.sa (Xiangliang Zhang).

are therefore challenged by the demand to process more massive data in higher dimensions at high speeds. A practical and efficient Intrusion Detection System (IDS) capable of detecting potential attacks is required so that resolutions can be implemented as quickly as possible.

In general, as shown in Figure 1, there are four steps in anomaly intrusion detection: *data collection*, *attribute construction*, *model building* and *anomaly detection*. Prior to building the models, attributes are typically normalized [42]. Many methods [3,4,7,12,16,17,24,32,36,44,45] have been employed for anomaly intrusion detection. However, most of them mainly focus on attribute construction [12,16,17,32,44] or on detection algorithms [3,4,7,24,35,36,45]. In this paper, we aim at building a lightweight IDS with enhanced capability to process big data. Intuitively, the approach to high-speed processing of massive audit data in intrusion detection is to reduce the amount of data without losing the valuable information in the data before the detection model is built. We call this process *data abstraction*. In this work, we introduce a "*data abstraction*" step between the "attribute construction" and the"model building" steps, as shown in Figure 1. This step yields a dataset we call abstracted audit data.

We propose three strategies to perform the data abstraction. The first strategy is to perform *attribute abstraction* after the attributes have been constructed. In this paper, we use Principal Component Analysis (PCA) to transform audit data in a high-dimensional space onto a space of fewer dimensions. The transformed attributes in the low-dimensional space are *not* part of the original attributes. The second strategy is *attribute selection*, which locates a subset of original attributes to represent the whole data. In this work, we employ Information Gain (IG) to select a small subset of key attributes from the originals. Different from attribute abstraction or attribute selection where we analyze the attributes, we propose a third strategy called *exemplar extraction* which focuses on the data samples. The process of exemplar extraction is also based on the training data. The strategy is to extract a smaller set of representative exemplars from the large amount of training data, so that the training is based on a smaller set and the test is based on a compressed detection model. An *exemplar* refers to a factual data item (e.g., a http request, a network connection) that represents a number of similar data items. Compared to randomly sampling data items (e.g., Netflow based network intrusion detection [5]), *exemplars* summarize massive audit data and thus better represent the audit data for anomaly detection.

For *exemplar extraction*, we employed two clustering methods, a newly developed Affinity Propagation (AP) [10] as well as traditional k-means [21] to cluster the original training data before the detection model is built. After the clustering process is finished, each cluster can be represented by an exemplar for AP, and by a mean center for k-means. We then use the exemplars or the cluster centers as the data input for building the detection models. In this way, the data is largely reduced while the valuable information is preserved to build a lightweight detection model with fewer exemplars for processing. This paper extends our previous work [40,43] by conducting more extensive experiments and comparing the three strategies of data abstraction in terms of the accuracy and efficiency of the detection. In addition, we comprehensively discuss the advantages and the disadvantages of these three strategies of data abstraction during the detection.

Two http traffic data sets collected in a real computing environment as well as the KDD'99 benchmark data are used to validate the three strategies of data abstraction. The extensive test results show that the AP-based exemplar extraction significantly improves the detection efficiency and achieves a more robust detection performance than the information-gain-based attribute selection or the PCA-based attribute abstraction.

In this paper, we make three contributions:

- We propose three strategies of data abstraction to accelerate the process of intrusion detection in massive data using lightweight models: exemplar extraction, attribute selection and attribute abstraction. We compare the three strategies in terms of detection accuracy and efficiency. The comprehensive study provides a valuable reference for the processing of big data in intrusion detection.
- We employ Affinity Propagation (AP) to extract *exemplars* from the massive training data. We thus build a lightweight IDS based on a smaller data set extracted from original training data such that the detection is based on a compressed model. We prove this process of **extracting *exemplars*** to be effective for anomaly intrusion detection. In most cases, the AP-based exemplar extraction outperforms attribute abstraction and attribute selection in terms of detection efficiency. To the best of our knowledge, this is the first work to extract exemplars from audit data for intrusion detection with AP.
- We use two types of data, real http traffic data and synthetic KDD'99 benchmark data, to validate the three strategies of data abstraction for intrusion detection. We provide extensive test results that demonstrate the effectiveness and high-speed performance of our methods for lightweight anomaly intrusion detection.

The remainder of this paper is organized as follows. Section 2 briefly introduces the related work. Section 3 describes the three strategies of data abstraction for anomaly intrusion detection. Extensive experiments and comparative results are reported in Section 4 and Section 5. Concluding remarks follow in Section 6.

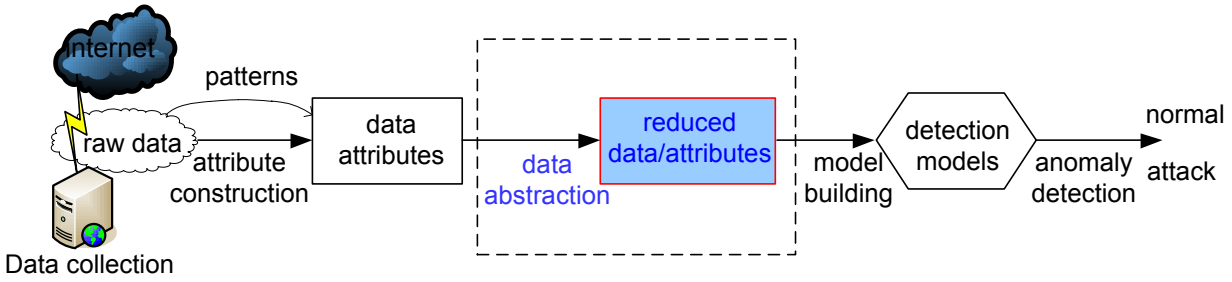


Fig. 1. Steps for anomaly intrusion detection

2 Related Work

Our work focuses on data reduction in anomaly intrusion detection based on http traffic and network data. The related work thus can be summarized into two subsections. In the first subsection, we survey anomaly detection techniques in general, Web-based anomaly intrusion detection and network-based anomaly intrusion detection methods. We review data reduction techniques for anomaly intrusion detection in the second subsection.

2.1 Anomaly intrusion detection

Anomaly intrusion detection has been an active research area. Network traffic as an important data source has been widely used for anomaly intrusion detection. Cretu et al. [8] cleaned a training data set of network traffic by combining it with the "micro-model" in a voting scheme to remove some of the attacks in the data. Krüegel and Vigna [17] were the first to use http traffic (web server log files) to detect attacks against web servers and web-based applications. They investigated the parameters (e.g., length and structure) contained in the client queries to detect potential attacks.

Ariu et al. [1] proposed an HMMPayl-based method for detecting web-based attacks. HMMPayl uses payload represented by a sequence of bytes as data input and used Hidden Markov Models (HMM) for the analysis. Lee and Kim [18] proposed a system called WarningBird to detect suspicious URL for Twitter. The system investigated correlated redirect chains of URLs found in many tweets. Razzaq et al. [26] presented an ontology-based approach that specifies web application attacks using the context of consequence, semantic rules and specifications of application protocols by analyzing the specified portion of user requests. Robertson et al. [27] addressed the problem of undertraining using global knowledge built by exploiting similarities between web application parameters of a similar type.

Based on the 1998 DARPA Intrusion Detection Evaluation Program, Lee et al. [19] extracted 41 attributes for each network connection and formed a well-known KDD Cup 1999 data [15]. The attributes of each network connection include basic attributes (e.g., protocol type), content attributes (e.g., number of "root" accesses) and traffic attributes (e.g., percentage of connections to the same service in a two-second time window). Many research groups [3,4,7,13,19,25,35,36,38,45] have used the KDD'99 data to validate their detection methods. Davis and Clark [9] reviewed data preprocessing techniques for anomaly based network intrusion detection, but they mainly focused on network traffic analyzed and on feature construction and selection methods used.

Most existing anomaly detection methods mainly focus on attribute construction or on detection algorithms. They have used either most of the attributes constructed from audit data or most of the original data to build detection models. For example, many network intrusion detection methods that have used KDD'99 data as their data source used most of the 41 attributes for the detection [3,4,7,19,20,35–37,45]. However, some of the attributes may be redundant or may even be the effect of noise and therefore may decrease the performance of an IDS. Moreover, many data may be very similar or even exactly the same, using all the data hence decreases the efficiency of building the detection models. Different from related work, to avoid these these types of redundancies, in this paper, we are motivated to abstract massive data so as to build lightweight models for efficient intrusion detection.

2.2 Data reduction in anomaly detection

There are related work regarding data reduction for intrusion detection. Sung and Mukkamala [33] used Artificial Neural Networks (ANN) and SVM to identify some important attributes based on the performance comparison with KDD'99 data. For example, an attribute is identified as important if the detection accuracy decreases and/or computation time increases after the attribute has been deleted from the training set. In our previous work [40], we used Information Gain (IG) and the combination of Bayesian Networks and decision tree classifiers to select some key subsets from the 41 attributes. We also used Principal Component Analysis (PCA) to reduce the dimensions of the audit data for intrusion detection [36,43,35]. Sengupta and Sen [29] proposed an intrusion detection method by integrating Q-learning algorithm and Rough Set Theory (RST) based on KDD'99

data. Like PCA, Partial Least Square (PLS) was employed by Gan et al. [30] to extract attributes from the KDD'99 data followed by Core Vector Machine (CVM) for network intrusion detection. In Netflow-based network anomaly detection, many methods sampled data with a certain probability to deal with massive traffic [5]. However, this approach involves sampling data (e.g., flow or packet) which is inherently inaccurate because potentially useful information can be discarded.

Different from attribute selection [29,33] or randomly sampling [5], in this paper, we employ a strategy that finds exemplars from original data and then use the exemplars for training and detection. We call this process "exemplar extraction". We aim to comprehensively compare the accuracy and efficiency of exemplar extraction with that of attribute selection and attribute abstraction method for fast anomaly detection. To the best of our knowledge, this is the first work to make this type of in-depth comparison between exemplar extraction, attribute selection and attribute abstraction for anomaly intrusion detection.

3 Abstracting big audit data for intrusion detection

To compare the accuracy and efficiency the three strategies of data abstraction, we use a number of classifiers for results comparison. First, we use k-Nearest Neighbor(NN) [37] and one class Support Vector Machine (SVM) [28] for anomaly detection, as both two methods have been shown effective for intrusion detection in our previous work [37]. Second, we use Principal Component Analysis (PCA) for attribute abstraction as this is a widely used attribute transformation algorithm. In fact, PCA itself can be considered to be an anomaly intrusion detection method. Third, we use Information Gain (IG) for attribute selection. Finally, for exemplar extraction, we employ Affinity Propagation (AP) [10] and k-means [21]. In this work, we only use normal data for building normal models (using attack-free data for training the model) based on which anomalies are detected.

3.1 No data abstraction: direct use of k-NN and one-class SVM

3.1.1 k-NN based anomaly intrusion detection method.

k-NN is a method for classifying objects by finding the closest training examples in the feature space. It uses the class labels of the k most nearest neighbors to predict the class of the test vector. A data vector normally consists of attributes. It represents an event or an object (e.g., an http request, a network connection) that needs to detect whether it is normal or not. In the remainder of this paper, *data vector* and *data item* are used interchangeably. Euclidean distance is usually used for measuring the similarity between two vectors T and X:

$$d(T, X) = \parallel T - X \parallel \tag{1}$$

Given a test vector T, the Euclidean distance between the test vector and each vector in the training data set is calculated. The k nearest neighbors of the test vector are chosen to determine whether the test vector is normal or not. In anomaly detection, we define the averaged k closest distance score as *anomaly index*. If the *anomaly index* of a test sequence vector is above a threshold, the test vector is then classified as abnormal. Otherwise it is considered as normal[41].

3.1.2 One-class SVM based anomaly intrusion detection method.

To facilitate comparison, we use one-class SVM proposed by Schölkopf et al. [28] for anomaly detection. One-class SVM algorithm is to map the data into a feature space H by a mapping function $\Phi(X)$, such that the dot product in H can be computed using a kernel $k(X_i, X_j) = \Phi(X_i) \cdot \Phi(X_j)$. The mapped data in H are separated from the origin with maximum margin using a hyperplane $\mathrm{w} \cdot \Phi(X) - \rho = 0$, while w is a weight vector and ρ is offset parameterizing the hyperplane in H. A small $\parallel \mathrm{w} \parallel$ corresponds to a large margin of separation from the origin.

Given training vectors $X_1, X_2, ..., X_n$ belonging to the normal class, the primal form of optimization function is

$$\min_{\mathrm{w},\xi,\rho} \frac{1}{2}\|\mathrm{w}\|^2 + \frac{1}{\nu n}\sum_{i=1}^{n}\xi_i - \rho,$$

$$\text{subject to} \quad \mathrm{w} \cdot \Phi(X_i) \geq \rho - \xi_i \quad \text{and} \quad \xi_i \geq 0,$$

where $\nu \in (0, 1]$ is an upper bound on the fraction of data that may be outliers.

Solving the optimization problem, the decision function is defined as

$$f(X) = sgn(\mathrm{w} \cdot \Phi(X) - \rho) \tag{2}$$

This function returns the value $+1$ in a "small" region capturing the training data, and -1 elsewhere.

In anomaly detection, we use the normal data to learn the function $f(X)$. If the decision function (2) gives a positive value for a test data item, it is classified as *normal*. Otherwise, it is considered as *anomalous*.

3.2 Attribute abstraction and anomaly intrusion detection with PCA

Principal Component Analysis (PCA) [14] is based on transforming a relatively large number of variables into

a *smaller* number of uncorrelated variables by finding a few orthogonal linear combinations of the original variables with the largest variance.

Given a set of original observations be $X_1, ..., X_n$, and suppose each observation is represented by a row vector of length m (the number of attributes), then the data set can be represented by a matrix $X_{n\times m}$. The average observation is defined as $\mu = \frac{1}{n}\sum_{i=1}^{n} X_i$. The observation deviation from the average is defined as $\Phi_i = X_i - \mu$. The sample covariance matrix of the data set is defined as $C = \frac{1}{n}\sum_{i=1}^{n}(X_i - \mu)(X_i - \mu)^T$.

Suppose $(\lambda_1, \mu_1), (\lambda_2, \mu_2), ..., (\lambda_m, \mu_m)$ are m eigenvalue-eigenvector pairs of the sample covariance matrix C, we choose k eigenvectors having the largest eigenvalues. Often there are just few large eigenvalues. This implies that k is the inherent dimensionality of the subspace governing the "signal", while the remaining $(m - k)$ dimensions generally contain noise. The dimensionality of the subspace k can be determined by $\frac{\sum_{i=1}^{k}\lambda_i}{\sum_{i=1}^{m}\lambda_i} \geq \alpha$, where α is the ratio of variation in the subspace to the total variation in the original space. We form a $(m \times k)$ (usually $k \ll m$ for data reduction) matrix U, which has columns comprising the k eigenvectors. The representation of the data by principal components consists of projecting the data onto the k-dimensional subspace according to the rule $Y_i = (X_i - \mu)U = \Phi U$.

The number of principal eigenvectors, $U_1, U_2, ..., U_k$, used to represent the distribution of the original data, is determined by α. They can also be regarded as the attributes abstracted from the original data attributes.

For anomaly detection, given an incoming vector T that represents a test sample, we project it onto the k-dimensional subspace representing the normal behavior. The distance between the test data vector Φ and its reconstruction onto the subspace Φ_r is the distance between the mean-adjusted input data vectors $\Phi = T - \mu$ and $\Phi_r = (T - \mu)UU^T = \Phi UU^T$ [36]. If the test data vector is normal, that is, if the test data vector is very similar to the training vectors corresponding to normal behavior, the test data vector and its reconstruction should be very similar and therefore the distance between them should be small. Our intrusion identification model is based on this property.

As PCA seeks a projection that best represents the data in a least-square sense, we use the squared Euclidean distance in the experiments to measure the distance between these two vectors: $\varepsilon = \| \Phi - \Phi_r \|^2$. ε is characterized as the *anomaly index*. If ε is below a predefined threshold, the vector is then identified as normal. Otherwise it is identified as anomalous.

3.3 Attribute selection with information gain

The Information Gain (IG) of a given attribute X with respect to the class attribute Y (e.g., normal or one of the attacks) is the reduction in uncertainty about the value of Y after observing values of X. It is denoted as $IG(Y|X)$. The uncertainty about the value of Y is measured by its entropy defined as

$$H(Y) = -\sum_i P(y_i)\log_2 P(y_i) \tag{3}$$

where $P(y_i)$ is the prior probabilities for all values of Y. The uncertainty about the value of Y after observing values of X is given by the conditional entropy of Y given X defined as

$$H(Y|X) = -\sum_j P(x_j)\sum_i P(y_i|x_j)\log_2 P(y_i|x_j) \tag{4}$$

where $P(y_i|x_i)$ is the posterior probabilities of Y given the values of X. The information gain is thus defined as $IG(Y|X) = H(Y) - H(Y|X)$. According to this measure, an attribute X is regarded as more correlated to class Y than attribute Z if $IG(Y|X) > IG(Y|Z)$. By calculating information gain, we can rank the correlations of each attribute to the class and select key attributes based on this ranking.

3.4 Exemplar extraction with Affinity Propagation and with k-means

3.4.1 Affinity Propagation based exemplar extraction.

Affinity Propagation (AP) is an exemplar-based clustering algorithm [10]. Let $E = \{e_1, \ldots, e_N\}$ be a set of data items and let $d(e_i, e_j)$ denote the distance (e.g., an Euclidean distance) between items e_i and e_j: $d(e_i, e_j) = \|e_i - e_j\|$.

The fitness function is defined by

$$\mathbf{E}(c) = \sum_{i=1}^{n} S(e_i, e_{c(i)}) \tag{5}$$

where $c(i)$ is the index of the exemplar representing the item e_i in a cluster; $S(e_i, e_j)$ is set to $-d(e_i, e_j)^2$ if $i \neq j$, and otherwise is set to a small constant $-s^*$ ($s^* \geq 0$). $-s^*$ represents a preference that e_i itself be chosen as an exemplar. AP finds the mapping $\mathbf{c}$ that maximizes the fitness function $\mathbf{E}(c)$ defined by (5) to cluster the data items. The resolution of this optimization problem is achieved by a message passing algorithm [10].

In practical use, there may be some items that are exactly the same in the audit data. In our study, we

used Weighted AP (WAP) [46–48] because it adds more weight to the multiple-appear items so as to let one of them have more probability of being an exemplar. Let data set $E' = \{(e_i, n_i)\}$ involve n_i copies of item e_i, for $i = 1, \ldots, L$. WAP considers the similarity metric defined as

$$S'(e_i, e_j) = \begin{cases} -n_i d(e_i, e_j)^2 & \text{if } i \neq j \\ -s^* & \text{otherwise.} \end{cases}$$

Unlike k-means or k-centers, AP has no need to predefine how many exemplars or clusters exist in the data. Instead, AP specifies the penalty s^* for allowing an item to become an exemplar. Note that for $s^* = 0$, the best solution is a trivial one, selecting every item as an exemplar.

3.4.2 K-means based exemplar extraction.

Given a set of observations $(X_1, X_2, \ldots, X_n)$, where each observation is an attribute vector, k-means clustering aims to partition the n observations into k sets $(k < n)$ $S = \{S_1, S_2, \ldots, S_k\}$. This is to minimize the within-cluster sum of squares

$$\sum_{i=1}^{k} \sum_{x_j \in s_i} \parallel X_j - \mu_i \parallel^2 \qquad (6)$$

where μ_i is the mean of points in S_i. It is clear that k-means can only generate k-mean cluster centroids rather than real exemplars.

4 Experiments

4.1 Overview

To comprehensively validate our methods, in this work, we performed a large number of experiments. In detail, we evaluate the three strategies of data abstraction on three data sets by combining the following detection methods and strategies together:

- **exemplar extraction**: AP + k-NN; k-means + k-NN; AP + PCA; k-means + PCA; AP + SVM; k-means + SVM.
- **attribute selection**: IG + k-NN; IG + PCA; IG + SVM.
- **attribute extraction**: PCA.
- **combinations**: IG + AP + PCA; IG + AP + k-NN; IG + k-means + k-NN; IG + k-means + PCA; IG + AP + SVM; IG + k-means + SVM.

4.2 Data

In the experiments, we used two different types of data for the evaluation. The first is a real http traffic dataset [1] collected, from the main http server of a intermediate research institute in Europe. The second is the KDD'99 benchmark data [15].

4.2.1 Http data streams.

We detect web attacks with real http traffic streams. In some ways our method is analogous to an anomaly detection version of the Snort sensor [31], focusing on port 80. Two large http traffic data sets were collected on the main Apache server. These two real traffic data sets represent two attack scenarios in the network. The first traffic (http large) contains rare attacks while the second (http small) contains bursty attacks. We labeled the data by three methods to guarantee the correctness of the labeling information. Firstly, we examined the alert information recorded by the signature-based IDS to find known attacks, and then we used several anomaly detectors to check for unknown anomalies. Finally we manually examine the data, especially the anomalies identified by the previous detectors.

The attacks in http large traffic mainly includes JS XSS attacks, input validation error, URL decoding error, SQL-injection attacks, PHP remote file inclusion attacks and DoS attacks. The attacks in http small traffic mainly includes JS XSS attacks, SQL-injection attacks and PHP remote file inclusion attacks. In detail, the http large traffic contains 36 attack requests distributed in more than 5.7 million requests collected in 10 days and 21 hours while the http small traffic contains 239 attacks occurring over a very short interval (between request 7923th and 9743th after filtering) in more than 1.4 million requests collected in 3 days and 3 hours. These two traffic data sets are described in Table 1.

Our method examines individual http requests and models the content of script inputs. To reduce noise contained in the data streams, we filtered out most static requests (e.g., .html, .wav, .txt, .swf) and widely known search engine robots (e.g., googlebot, msnbot) before the detection, because a static request cannot be an attack to the server. Note that we only filter out widely known attack-free static requests to guarantee that no true attack is removed from the data. The data are largely reduced by this process of filtering.

We used character distribution of each path source in http requests as the attributes. Character distribution was first introduced for anomaly detection by Krüegel and Vigna [17] who computed them in a coarse way.

[1] All the preprocessed data and the programs used in this paper are available upon request.

Table 1
The http traffic used in the experiments

Data set	Before Filtering		After Filtering		# normal	# attack
	File size	# requests	File size	# requests	requests	requests
http large	1,536MB	5,700,949	53.5MB	265,752	265,716	36
http small	561.2MB	1,449,379	9.5MB	40,095	39,856	239

They first sorted the 256 ASCII characters by frequency, aggregated them into 6 groups (0, 1-3, 4-6, 7-11, 12-15, and 16-255), and computed one single uniform distribution model for these 6 groups for all attributes. Also, they only modeled the query attributes. Their results [17] showed that even the attributes of this type of coarse character distribution detect most web attacks. Furthermore, Wang et al. [34] used full character distribution of payload of network traffic for anomaly network detection and the results also showed its effectiveness. Nevertheless, in this work, we seek to refine this computation by modeling the full-byte distribution of each path source (including queries) in http requests. We find there are 256 types of ASCII code in total but only 95 types (between 33 and 127) appear in the http request. The character distribution is computed as the frequency of each ASCII code in the path source of a http request. In contrast, we model. For example, the character distribution of printable ASCII codes of the path source "/acacia/project/edccaeteras/wakka.php?wiki=Acti onOrphanedPages/referrers" is computed as:

0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.014 0.069 0 0 0 0 0 0 0 0 0 0
0 0 0 0.014 0 0.014 0 0.014 0 0 0 0 0 0 0 0 0 0 0 0 0 0.014
0.014 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.125 0 0.083 0.028
0.125 0.014 0.014 0.028 0.056 0.014 0.042 0 0 0.028 0.028
0.056 0 0.0976 0.042 0.042 0 0 0.028 0 0 0 0 0 0 0 0 0 0

As a consequence, each http request (data item) is represented by a vector of 95 dimensions. Our goal is to identify whether an item is normal or anomalous. There are some other different attributes that can be extracted from the http traffic [17,22]. The character distribution can be regarded as a 1-gram method (w.r.t. n-gram [32]) to consider the content of the http request. New attributes can be added in our future work, but here, we mainly focus on abstracting massive data for fast anomaly detection.

4.2.2 KDD'99 data.

The raw KDD'99 data contains traffic in a simulated military network that consists of hundreds of hosts. It has been processed into about 5 million connection records by Lee et al. [19] as part of the UCI KDD archive [15]. A connection is a sequence of TCP packets starting and ending at some well-defined times, between which data flows from a source IP address to a target IP address under some well defined protocol [15]. In the data set, each network connection is labeled as either normal, or as a specific kind of attack. The network connection data contains 41 attributes among which 34 are numeric and 7 are alphanumeric. We only used the 34 numeric attributes in the experiments. Each connection in the data set is thus transformed into a 34-dimensional vector as data input for the detection.

These attributes are divided into three groups: basic attributes of individual TCP connections, traffic attributes and content attributes within a connection suggested by domain knowledge. There are 494,021 connection records in the training set in which 97,278 are normal and 396,744 are attacks. There are 22 types of attacks in the data set in total and these attacks fall into one of 4 categories: DoS: denial-of-service (e.g., teardrop); PROBE: surveillance and other probing (e.g., port scanning); R2L: unauthorized access from a remote machine (e.g., password guessing); U2R: unauthorized access to local superuser (root) privileges by a local unprivileged user (e.g., buffer overflow attacks).

In the attack data set, because DoS attacks take quite a large proportion in the attack data and they are relatively easy to detect [36], we randomly selected 20% from each DoS attack category to form the DoS attack data. All PROBE, R2L and U2R attacks are used in our work. For normal data, we randomly selected 7000 connections for training and other 10000 for the test. The original data and the data used in our work is described in Table 2. In the experiments, we normalize the KDD'99 data with statistical normalization method before conducting the detection.

5 Comparative results

5.1 Experiment settings

For exemplar extraction, to generate a different number of exemplars, the preference parameter ($-s^*$) of AP needs to be set to various values. In total there are N^2 values that can be set as $-s^*$ to generate the exemplars. In the experiments, we set $(l * N^2)$-th largest value of the N^2 similarities between all pairs of items as $-s^*$ and set $l = 1/2, 1/4, 1/8, 1/16, 1/32, 1/64$, respectively. For k-means, to facilitate comparison, we set k as the same number of exemplars generated by AP to generate clusters. For attribute abstraction with PCA, as mentioned earlier, α is the ratio of variation in the subspace to the

Table 2
The KDD'99 data (number of connections) used in the experiments

Data set	Normal (#)	DoS (#)	PROBE (#)	R2L (#)	U2R (#)
Origin	97,278	391,458	4,107	1,126	52
Training	7,000	0	0	0	0
Test	10,000	78,291	4,107	1,126	52

total variation in the original space. If α is chosen as 99.9%, then the variation in the subspace spanned by the former k eigenvectors has only 0.1% loss of variation in the original space. Basically a larger α results in better representation of the original space with subspace. Usually there is a tradeoff between α and the detection efficiency. In this work, we set α as 99.9% in the experiments to obtain satisfactory detection results. This is also consistent with our previous finding in [35,36].

Different from exemplar extraction or attribute abstraction in which only normal data is needed for training, attribute selection requires additional attack data. For http traffic data, we used all the normal training data *(http large and http small)* and randomly selected half the attack data to rank the importance of attributes. For KDD'99 data, we used all the normal training data and randomly selected a subset of attack data for the ranking. For detection algorithm k-NN, we set $k = 1$ and it is thus in fact Nearest Neighbor (NN). We made our own MATLAB programs for all the algorithms except SVM for which LibSVM tools [6] were used.

The three data sets and the number of exemplars generated with AP from their corresponding training data are described in Table 3. Table 4 shows the number of attributes after attribute selection using IG and after attribute extraction with PCA.

We performed extensive experiments on the three data sets with different combinations of detection algorithms (i.e., k-NN, SVM, PCA) and strategies of data abstraction (i.e., PCA, IG, AP and k-means). In this paper, we report the exemplars extracted and attributes selected, the detection speed and the detection accuracy.

5.2 Exemplars extracted and attributes selected

An exemplar is nicely representative of itself and some other data items. As shown in Figure 2, an exemplar is a factual data item (i.e., an http request) presented in rows while an attribute is a value describing the data item and is presented in columns. For instance, the second http request is extracted by AP from the http large data set as an exemplar and attribute 2, 5 and 92 are selected by IG for anomaly detection.

Two exemplars extracted by AP based on the http large traffic data are shown as

1. /axis/publications/login.php?ref=/axis/publications /add.php&forgot=y
2. /mascotte/Stages/?ID=138&annee=2006 _2007&lang=fr

The two exemplars are clearly shown to represent a set of similar http requests in the training data. For attribute selection with IG, the experiments show that only 52 out of 95 attributes (characters) contribute to the detection. The important attributes (characters) are listed (**ranked** by their contribution to the detection) as

. / r ? e \ - = w c o —) (; b g t p h j A f ¿ " a k O P m s ¡ u v _ d R [ ] I 5 X * % T l i 7 , ‘ x W

The attributes ". / r ?" have high ranking w.r.t. the contribution to the detection because many attack requests contain them. For example, an http request of a remote file inclusion attack is "./query.php?loginID=/content /base/build/explorer/none.php?..:..:..:..:..:..:..:etc:passwd", where the mentioned attributes appear frequently.

The attributes abstracted by PCA, however, are not the original ones, as they represent attributes in a new feature space.

5.3 Detection efficiency

For exemplar extraction, the computation time for generating exemplars, for training the models and for the detection is summarized in Table 5. For attribute selection, the computation time for attribute selection (IG), for training the models and for the detection is listed in Table 6. All the experiments were carried out on a machine with Intel Dual Core 2.80GHz and memory 4GB.

It is seen from Tables 5 that the detection efficiency for all the detection algorithms is significantly improved by the process of exemplar extraction. For example, while using all the data to build the detection models, $k-$NN needs 7670 seconds for the detection of web attacks in http large data set, but only 235 seconds are required for the detection with 298 exemplars extracted from 8000 items by AP. Generating exemplars is not time consuming. For instance, generating 298 exemplars costs only 94 seconds. As a result, in a best case scenario, the process

Table 3
Three data sets and the number of their exemplars generated from the training data with AP

Data	# training items	# test items	# exemplars $l = 1/2$	$l = 1/4$	$l = 1/8$	$l = 1/16$	$l = 1/32$	$l = 1/64$
http large	8000	257751	298	384	461	532	678	977
http small	7000	33095	420	529	613	692	781	975
KDD'99	7000	93576	191	327	614	1172	/	/

Table 4
The number of attributes after reduction in the three data sets

Data set	# attributes	# attribute after PCA	# attribute after IG
http large	95	66	52
http small	95	71	52
KDD'99	34	23	16

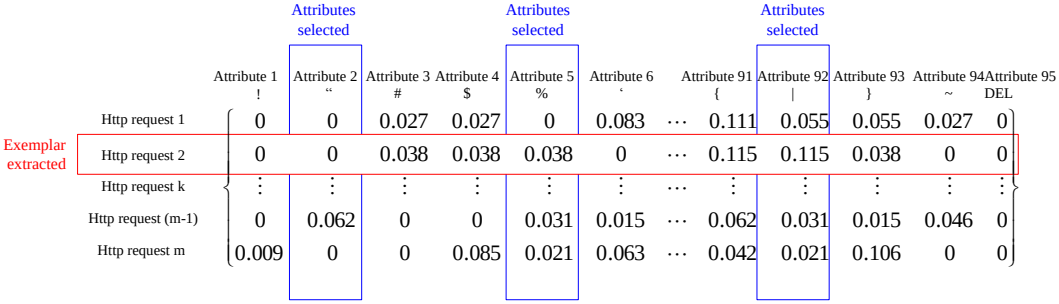


Fig. 2. Exemplars and attributes in the http traffic data

of exemplar extraction improves the computation time by more than 23 times in the best case. In most cases AP is more efficient than k-means for generating exemplars.

As expected, Table 6 shows that attribute selection with IG also improves the detection efficiency. For example, k-NN requires 7170 seconds for the detection of web attacks in http large data set with original data (95 attributes). In contrast, it only needs 4114 seconds with reduced attributes (52 attributes). By comparing the computation time during the detection in Table 5 and in Table 6, we see that exemplar extraction with AP is more effective than attribute selection for enhancing the detection efficiency.

5.4 Comparative detection results

The comparative results on http large data set are shown in Figures 3 and 4. These curves in the Figures represent ROC (Receiver Operating Characteristic) of the Detection Rates (DR), calculated as the percentage of intrusions/attacks detected against False Positive Rates (FPR), calculated as the percentage of normal events falsely classified as intrusions/attacks.

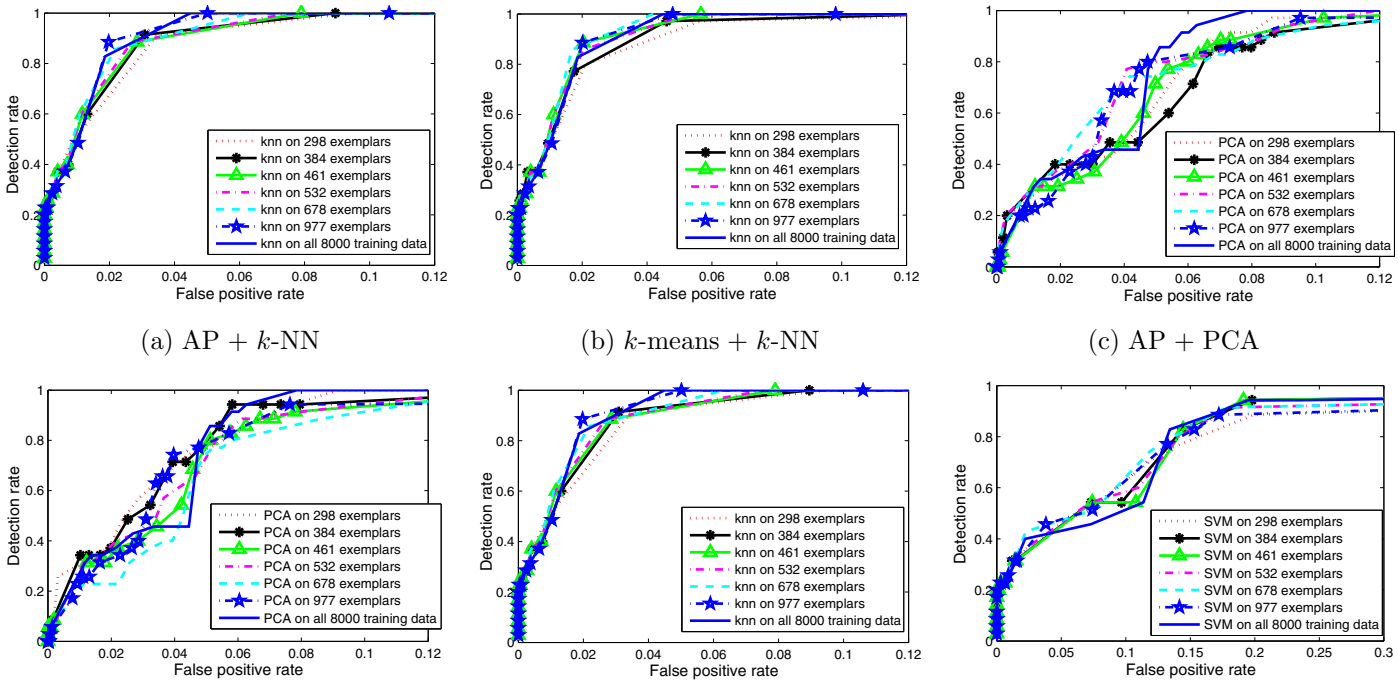
Figure 3 shows that using a smaller set of exemplars results in similar or improved detection accuracy in many cases. Normally, as the number of exemplars increases, the detection accuracy is improved too. However, as shown in the Figures, selecting approximately 10 percent of the total number of data items as exemplars gives satisfactory detection results. In many cases the results are better than using all the data.

AP performs similar to k-means in exemplar extraction for k-NN and PCA based detection methods. It outperforms k-means for SVM-based detection. Similarly, Figure 4 shows that IG-based attribute selection improves detection rates. Thus k-NN appears to be more effective than PCA or SVM for the detection of anomalies in http large data.

Comparison with KDD'99 data in Figure 5 shows that exemplar extraction is a good tool. Selecting a smaller set of exemplars for training has either no affect on detection accuracy and in most cases it even improves the detection accuracy. These results are consistent with those on http traffic, and show that k-NN is better than PCA or SVM for anomaly detection.

Table 5
The computing time (sec.) for anomaly detection **with or without exemplar extraction process** (AP and k-means).

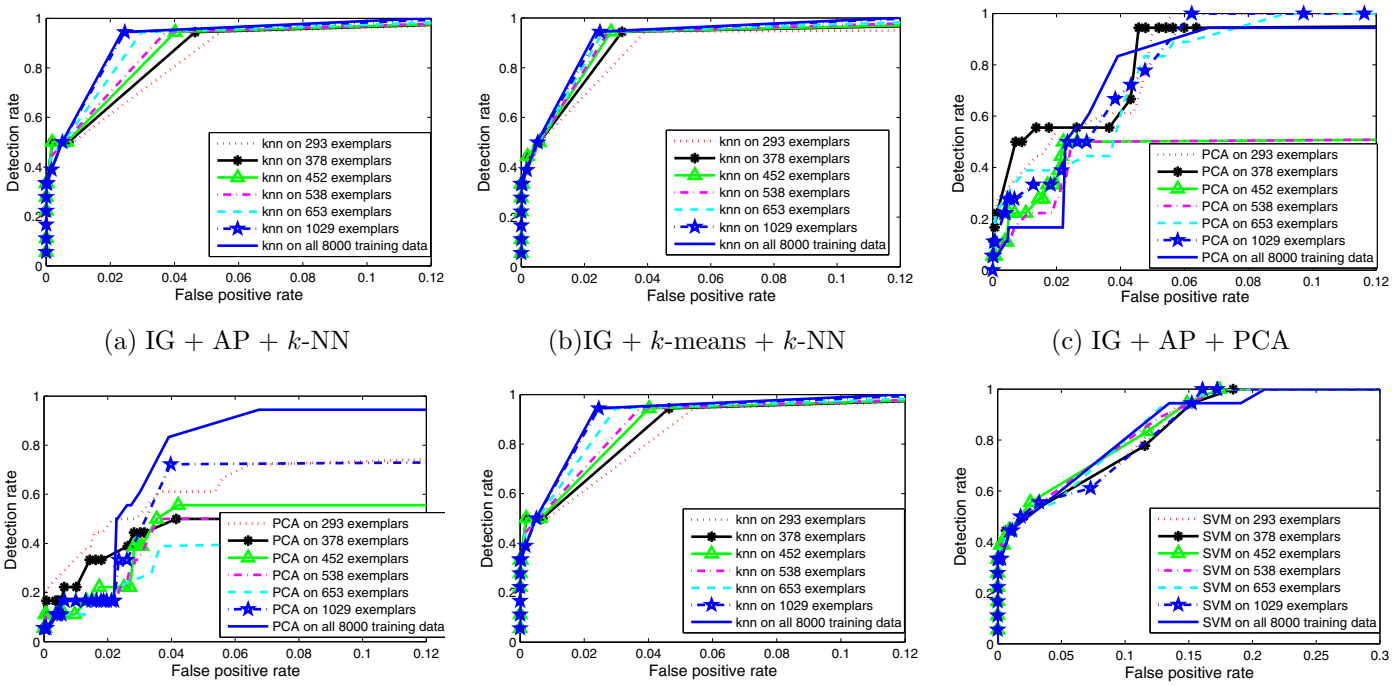
Data set	training data	time for generating exemplars		time for anomaly detection		
		AP	k-means	k-NN	PCA	SVM
http large	8000 items	/	/	7670	36	919
	298 exemplars	94	99	235	3	28
	384 exemplars	108	128	294	2	36
	461 exemplars	246	207	478	2	43
	532 exemplars	251	225	645	2	51
	678 exemplars	97	394	617	3	63
	977 exemplars	125	275	628	2	94
http small	7000 items	/	/	592	29	101
	420 exemplars	139	190	29	0.3	5
	529 exemplars	141	365	38	0.3	6
	613 exemplars	158	215	45	0.3	8
	692 exemplars	158	408	48	0.3	8
	781 exemplars	176	495	56	0.3	10
	975 exemplars	162	307	67	0.7	12
KDD'99	7000 items	/	/	989	11	74
	191 exemplars	548	302	27	0.3	2
	327 exemplars	557	414	46	0.3	3
	614 exemplars	611	603	83	0.3	6
	1172 exemplars	753	1023	153	0.6	12



(a) AP + k-NN (b) k-means + k-NN (c) AP + PCA

(d) k-means + PCA (e) AP + SVM (f) k-means + SVM

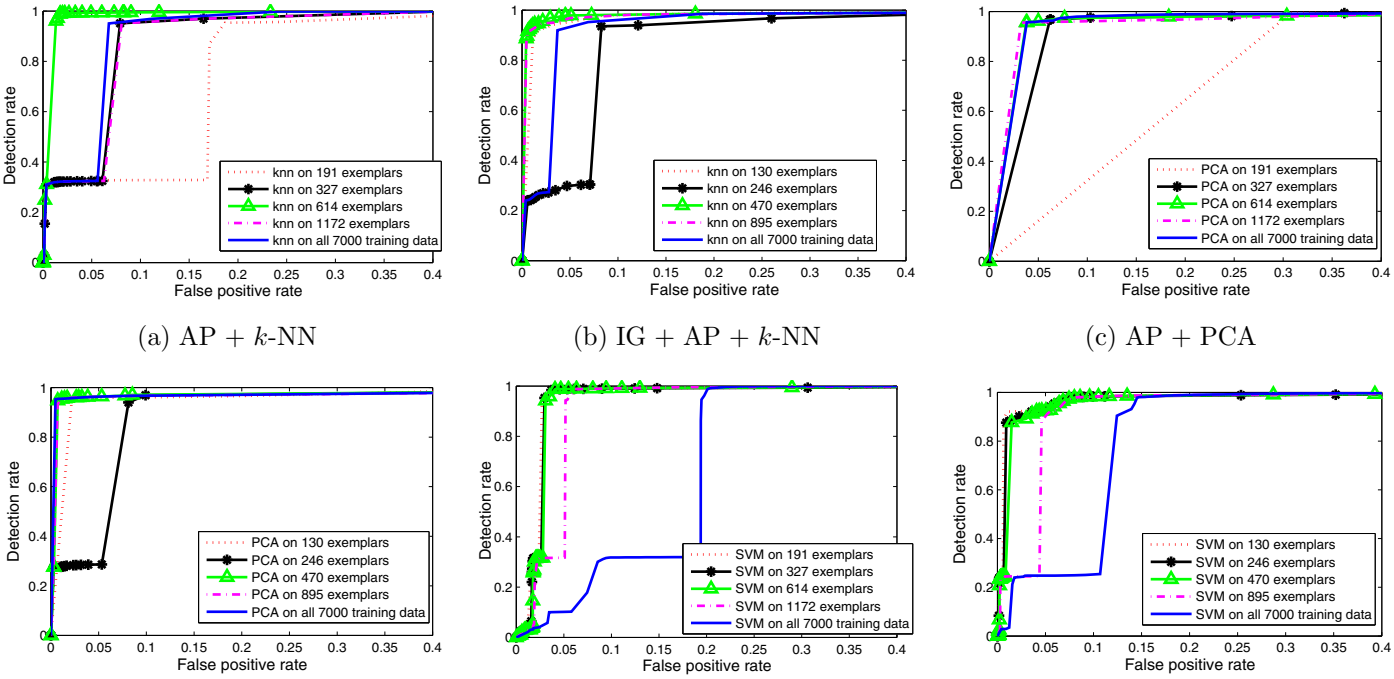
Fig. 3. Results on http large data set with exemplar extraction.



(a) IG + AP + k-NN (b)IG + k-means + k-NN (c) IG + AP + PCA

(d) IG + k-means + PCA (e) IG + AP + SVM (f) IG + k-means + SVM

Fig. 4. Results on http large data set with exemplar extraction after attribute selection with IG.



(a) AP + k-NN (b) IG + AP + k-NN (c) AP + PCA

(d) IG + AP + PCA (e) AP + SVM (f) IG +AP +PCA

Fig. 5. Results on KDD'99 data with exemplar extraction as well as its performance after attribute selection with IG.

Table 6
The computing time (sec.) for anomaly detection **after attributes were selected with IG** (52 out of 95 attributes for http data and 16 out of 34 attributes for KDD'99 data, see Table 4)

Data set	training data	exemplars time cost		classifying time cost		
		AP	k-means	k-NN	PCA	SVM
http large	8000 items	/	/	4114	21	558
	293 exemplars	218	59	116	1.2	19
	378 exemplars	180	70	154	1.2	25
	452 exemplars	187	79	175	1.2	31
	538 exemplars	190	90	215	1.2	37
	653 exemplars	291	100	287	1.3	44
	1029 exemplars	200	114	448	1.5	70
http small	7000 items	/	/	382	17	72
	395 exemplars	84	86	20	0.2	3
	497 exemplars	87	102	26	0.2	4
	592 exemplars	85	116	31	0.2	5
	664 exemplars	97	133	35	0.2	6
	736 exemplars	371	148	39	0.2	6
	819 exemplars	247	146	45	0.3	7
KDD'99	7000 items	/	/	806	7	51
	130 exemplars	475	158	16	0.2	1
	246 exemplars	894	354	29	0.2	2
	470 exemplars	720	559	53	0.2	3
	895 exemplars	1732	1008	101	0.3	6

6 Concluding remarks

The amount of data in anomaly intrusion detection is becoming increasingly massive in current computing environments. Building a lightweight model for anomaly intrusion detection to achieve real-time detection therefore becomes an important challenge. In this paper, we abstract big audit data by finding a small set of exemplars from a large set of original data. An exemplar is nicely representative of other data items. Exemplars are identified among data items and clusters of data items are formed around these exemplars. The exemplars are then fed as data input for training the detection models. This method improves detection efficiency for two reasons: first, only a smaller set of data needs to be processed for the training, and second, the detection process only needs to be based on a compressed model. For a comparative view of different strategies of data abstraction in intrusion detection, in this paper we also introduced Information Gain based attribute selection and PCA based attribute abstraction for anomaly detection.

PCA is a widely used attribute abstraction algorithm that performs a coordinate rotation that aligns the transformed axes with the directions of maximum variance. The new coordinates can be regarded as the new attributes abstracted from the originals. However, PCA is effective only when the observed data has a high signal-to-noise ratio, as the principal components with larger variance correspond to interesting dynamics and lower ones correspond to noise. Therefore, PCA is not effective when data items are all centralized. In cases where variance is low, exemplar extraction is more suitable because it works through grouping similar data items to form a cluster in which the most representative factual data item are selected as an exemplar.

Most existing methods have used all data items or all attributes for anomaly intrusion detection. However, because some data items or attributes could be redundant or even represent only noise, including all data items can have a negative impact on the detection performance. The correlated data items or redundant features thus should be reduced before the detection models are built. Exemplar extraction considerably improves intrusion detection by reducing reduplicate and similar data items that usually exist in the original data set. To illustrate the ability of exemplar abstraction to minimize duplicates in the data sets, we used the KDD'99 data set to find that only 1172 data items were unique out of 7000

training items. Similarly, in the http large data set, only 1943 data items were unique among 8000 data items, and in http small data set, only 2308 out of 7000 were unique. Using a clustering algorithm Affinity Propagation (AP) [10], we extract real exemplars from original data and used traditional k-means clustering method to generate average exemplars.

Attribute selection can also improve detection efficiency. In this paper, we used Information Gain (IG) to rank the attributes. To generate an effective subset of attributes from the original set, IG requires a large amount of attack data and normal data. In practice, however, a large amount of attack data is difficult to collect. This may, in some circumstances, prevent the use of IG based attribute selection method for intrusion detection.

Two http data sets collected from a real computing environment as well as the KDD'99 data have been used to validate the two strategies of data abstraction. The extensive experimental results confirm that both AP and k-means clustering methods improve the accuracy and efficiency of anomaly intrusion detection through exemplar extraction. In many cases AP outperforms k-means. For example, the test results show that the AP based exemplar extraction method improves the detection efficiency by 23 times compared to k-NN. In addition, AP has two major advantages over k-means. First, AP extracts factual exemplars from the data while k-means can only generate averaged ones. This is important because an exemplar has a physical significance in many cases, which in practice means, a set of real http requests can be extracted, replacing the originals. Second, AP does not need to define the number of clusters (exemplars) beforehand while k-means does. This is an important advantage because it is often difficult to have *a priori* knowledge of this data, especially for a very large amount of streaming data like http traffic. The process of exemplar extraction is more robust than attribute selection with IG and than attribute abstraction with PCA. To enhance detection performance, we recommend to perform exemplar extraction (in particular AP-based exemplar extraction) prior to building detection models.

In the future, we will explore more attributes of http traffic to improve detection performance of anomalies and address the problem of "concept drift" in audit data.

7 Acknowledgements

The first author thanks the European Research Consortium for Informatics and Mathematics (ERCIM) fellowship program for their support. The work reported in this paper is supported in part by the Scientific Research Foundation through the Returned Overseas Chinese Scholars, Ministry of Education of China, under Grant K14C300020, in part by Shanghai Key Laboratory of Integrated Administration Technologies for Information Security under Grant AGK2015002, and in part by the 111 Project under Grant B14005.

References

[1] D. Ariu, R. Tronci, G. Giacinto, Hmmpayl: An intrusion detection system based on hidden markov models, Computers & Security 30 (4) (2011) 221–241.

[2] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, Drebin: Efficient and explainable detection of android malware in your pocket, in: NDSS, 2014.

[3] R. A. R. Ashfaq, X.-Z. Wang, J. Z. Huang, H. Abbas, Y.-L. He, Fuzziness based semi-supervised learning approach for intrusion detection system (to appear), Information Sciences.

[4] M. H. Bhuyan, D. Bhattacharyya, J. Kalita, A multi-step outlier-based anomaly detection approach to network-wide traffic, Inf. Sci. 348 (2016) 243–271.

[5] D. Brauckhoff, K. Salamatian, M. May, A signal-processing view on packet sampling and anomaly detection, in: INFOCOM, 2010.

[6] C.-C. Chang, C.-J. Lin, LIBSVM: A library for support vector machines, ACM Transactions on Intelligent Systems and Technology 2 (2011) 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[7] K. A. P. Costa, L. A. M. Pereira, R. Y. M. Nakamura, C. R. Pereira, J. P. Papa, A. X. Falcão, A nature-inspired approach to speed up optimum-path forest clustering and its application to intrusion detection in computer networks, Inf. Sci. 294 (2015) 95–108.

[8] G. F. Cretu, A. Stavrou, M. E. Locasto, S. J. Stolfo, A. D. Keromytis, Casting out demons: Sanitizing training data for anomaly sensors, in: IEEE S&P, 2008, pp. 81–95.

[9] J. J. Davis, A. J. Clark, Data preprocessing for anomaly based network intrusion detection: A review, Computers & Security 30 (6-7) (2011) 353–375.

[10] B. J. Frey, D. Dueck, Clustering by passing messages between data points, Science 315 (5814) (2007) 972–976.

[11] X. Guan, W. Wang, X. Zhang, Fast intrusion detection based on a non-negative matrix factorization model, J. Network and Computer Applications 32 (1) (2009) 31–44.

[12] K. L. Ingham, H. Inoue, Comparing anomaly detection techniques for http, in: RAID, 2007, pp. 42–62.

[13] S.-Y. Jia, B.-K. Jeongb, S. Choia, D. H. Jeongc, A multi-level intrusion detection method for abnormal network behaviors, J. Network and Computer Applications 62 (2) (2016) 9–17.

[14] I. T. Jolliffe, Principal Component Analysis, 2nd ed., Springer-Verlag, Berlin, 2002.

[15] KDD-Data, Kdd cup 1999 data (retrieved january 2016), online available at `http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html` (1999).

[16] D. Kong, G. Yan, Discriminant malware distance learning on structural information for automated malware classification, in: KDD, 2013, pp. 1357–1365.

[17] C. Krügel, G. Vigna, Anomaly detection of web-based attacks, in: ACM CCS, 2003, pp. 251–261.

[18] S. Lee, J. Kim, Warningbird: Detecting suspicious urls in twitter stream, in: NDSS, 2012.

[19] W. Lee, S. J. Stolfo, K. W. Mok, A data mining framework for building intrusion detection models, in: IEEE S&P, 1999, pp. 120–132.

[20] W. Lin, S. Ke, C. Tsai, CANN: an intrusion detection system based on combining cluster centers and nearest neighbors, Knowl.-Based Syst. 78 (2015) 13–21.

[21] J. B. MacQueen, Some methods for classification and analysis of multivariate observations, in: Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability, 1967, pp. 281–297.

[22] F. Maggi, W. K. Robertson, C. Krügel, G. Vigna, Protecting a moving target: Addressing web application concept drift, in: RAID, 2009, pp. 21–40.

[23] E. O. of the President, Big data research and development initiative, `http://www.whitehouse.gov` (2012).

[24] V. Ramanathan, H. Wechsler, Phishing detection and impersonated entity discovery using conditional random field and latent dirichlet allocation, Computers & Security 34 (2013) 123–139.

[25] S. Rastegari, P. Hingston, C. Lam, Evolving statistical rulesets for network intrusion detection, Appl. Soft Comput. 33 (2015) 348–359.

[26] A. Razzaq, K. Latif, H. F. Ahmad, A. Hur, Z. Anwar, P. C. Bloodsworth, Semantic security against web application attacks, Information Sciences 254 (1) (2014) 19–38.

[27] W. K. Robertson, F. Maggi, C. Kruegel, G. Vigna, Effective anomaly detection with scarce training data, in: NDSS, 2010.

[28] B. Schölkopf, J. C. Platt, J. Shawe-Taylor, A. J. Smola, R. C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (7) (2001) 1443–1471.

[29] N. Sengupta, J. Sen, J. Sil, M. Saha, Designing of on line intrusion detection system using rough set theory and q-learning algorithm, Neurocomputing 111 (2013) 161–168.

[30] X. sheng Gan, J. shun Duanmu, J. fu Wang, W. Cong, Anomaly intrusion detection based on pls feature extraction and core vector machine, Knowl.-Based Syst. 40 (2013) 1–6.

[31] Snort, Snort (retrieved 2016), `http://www.snort.org/` (2016).

[32] Y. Song, A. D. Keromytis, S. J. Stolfo, Spectrogram: A mixture-of-markov-chains model for anomaly detection in web traffic, in: NDSS, 2009.

[33] A. H. Sung, S. Mukkamala, Feature selection for intrusion detection using neural networks and support vector machines, in: 82nd Annual Meeting of the Transportation Research Board, 2003.

[34] K. Wang, S. J. Stolfo, Anomalous payload-based network intrusion detection, in: RAID, 2004, pp. 203–222.

[35] W. Wang, R. Battiti, Identifying intrusions in computer networks with principal component analysis, in: Proceedings of the The First International Conference on Availability, Reliability and Security, ARES 2006, April 20-22, Vienna Austria, 2006, pp. 270–279.

[36] W. Wang, X. Guan, X. Zhang, Processing of massive audit data streams for real-time anomaly intrusion detection, Computer Communications 31 (1) (2008) 58–72.

[37] W. Wang, T. Guyet, R. Quiniou, M. Cordier, F. Masseglia, X. Zhang, Autonomic intrusion detection: Adaptively detecting anomalies over unlabeled audit data streams in computer networks, Knowl.-Based Syst. 70 (2014) 103–117.

[38] W. Wang, Y. He, J. Liu, S. Gombault, Constructing important features from massive network traffic for lightweight intrusion detection, IET Information Security 9 (6) (2015) 374–379.

[39] W. Wang, X. Wang, D. Feng, J. Liu, Z. Han, X. Zhang, Exploring permission-induced risk in android applications for malicious application detection, IEEE Transactions on Information Forensics and Security 9 (11) (2014) 1869–1882.

[40] W. Wang, X. Zhang, High-speed web attack detection through extracting exemplars from http traffic, in: SAC, 2011, pp. 1538–1543.

[41] W. Wang, X. Zhang, S. Gombault, Constructing attribute weights from computer audit data for effective intrusion detection, J. Sys. and Soft. 82 (12) (2009) 1974–1981.

[42] W. Wang, X. Zhang, S. Gombault, S. J. Knapskog, Attribute normalization in network intrusion detection, in: ISPAN, 2009, pp. 448–453.

[43] W. Wang, X. Zhang, G. Pitsilis, Abstracting audit data for lightweight intrusion detection, in: ICISS, 2010, pp. 201–215.

[44] G. Yan, N. Brown, D. Kong, Exploring discriminatory features for automated malware classification, in: DIMVA, 2013, pp. 41–61.

[45] J. Zhang, H. Li, Q. Gao, H. Wang, Y. Luo, Detecting anomalies from big network traffic data using an adaptive detection approach, Inf. Sci. 318 (2015) 91–110.

[46] X. Zhang, C. Furtlehner, C. Germain-Renaud, M. Sebag, Data stream clustering with affinity propagation, IEEE Trans. Knowl. Data Eng. 26 (7) (2014) 1644–1656.

[47] X. Zhang, C. Furtlehner, J. Perez, C. Germain-Renaud, M. Sebag, Toward autonomic grids: analyzing the job flow with affinity streaming, in: Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Paris, France, June 28 - July 1, 2009, pp. 987–996.

[48] X. Zhang, C. Furtlehner, M. Sebag, Data streaming with affinity propagation, in: Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, Proceedings, Part II, 2008, pp. 628–643.

Brief biography of the authors

Wei Wang is an associate professor at the School of Computer and Information Technology, Beijing Jiaotong University, China. He earned his PhD degree from Xi'an Jiaotong University in 2006. He was a postdoctoral researcher at the University of Trento, Italy, from 2005-2006 and at TELECOM Bretagne and at INRIA, France, from 2007-2008. He was an European ERCIM Fellow in the Norwegian University of Science and Technology (NTNU), Norway, and at the Interdisciplinary Centre for Security, Reliability and Trust (SnT), University of Luxembourg, from 2009-2011. He visited INRIA, ETH, NTNU, CNR, and New York University Polytechnic. He was guest editor of IEEE Network and currectly a young AE of Frontiers of Computer Science. He has authored or co-authored over 50 peer-reviewed papers in various journals and international conferences. His main research interests include network security and data mining.

Jiqiang Liu received BSc and PhD degrees from Beijing Normal University, Beijing, China, in 1994 and 1999, respectively. He is currently a Professor at the School of Computer and Information Technology, Beijing Jiaotong University, Beijing. He has authored over

90 scientific papers in various journals and international conferences. His main research interests are trusted computing, cryptographic protocols, privacy preserving, and network security.

Georgios Pitsilis received his BEng degree in engineering and BEng degree in informatics from the Technological Institution of Athens, Greece, in 1991 and 1999 respectively. He received his MSc degree from Oxford Brookes University, UK in 2000, and his PhD degree in computer science from Newcastle University, UK in 2007. He has worked as a postdoctoral researcher at the Universities of Luxembourg, NTNU and Newcastle University for 5 years. His research interest are in the areas of trust computing, recommender systems, data mining and distributed computing.

Xiangliang Zhang is currently an Assistant Professor and directs the Machine Intelligence and kNowledge Engineering (MINE) Laboratory (http://mine.kaust.edu.sa) in the Division of Computer, Electrical and Mathematical Sciences & Engineering at King Abdullah University of Science and Technology (KAUST). She was an European ERCIM research fellow in the Department of Computer and Information Science, NTNU, Norway, from April-August 2010. She earned her PhD degree in computer science with honors from INRIA-University Paris-Sud 11, France, in July 2010. She visited IBM T.J. Watson Research Center, Texas A&M University, University Paris-Sud 11, Concordia University, Microsoft Research Asia and Rochester University. She regularly serves on the TPC for top-tier conferences like AAAI 2017, WSDM 2017, KDD 2016 and ICDM 2016. She has authored or co-authored over 70 refereed papers in various journals and conferences. Her main research interests and experiences are in diverse areas of machine intelligence and knowledge engineering, such as complex system modeling, computer security and big data processing.