Random-Forests-Based Network Intrusion Detection Systems

Jiong Zhang, Mohammad Zulkernine, and Anwar Haque

***Abstract*—Prevention of security breaches completely using the existing security technologies is unrealistic. As a result, intrusion detection is an important component in network security. However, many current intrusion detection systems (IDSs) are rule-based systems, which have limitations to detect novel intrusions. Moreover, encoding rules is time-consuming and highly depends on the knowledge of known intrusions. Therefore, we propose new systematic frameworks that apply a data mining algorithm called random forests in misuse, anomaly, and hybrid-network-based IDSs. In misuse detection, patterns of intrusions are built automatically by the random forests algorithm over training data. After that, intrusions are detected by matching network activities against the patterns. In anomaly detection, novel intrusions are detected by the outlier detection mechanism of the random forests algorithm. After building the patterns of network services by the random forests algorithm, outliers related to the patterns are determined by the outlier detection algorithm. The hybrid detection system improves the detection performance by combining the advantages of the misuse and anomaly detection. We evaluate our approaches over the Knowledge Discovery and Data Mining 1999 (KDD'99) dataset. The experimental results demonstrate that the performance provided by the proposed misuse approach is better than the best KDD'99 result; compared to other reported unsupervised anomaly detection approaches, our anomaly detection approach achieves higher detection rate when the false positive rate is low; and the presented hybrid system can improve the overall performance of the aforementioned IDSs.**

***Index Terms*—Computer network security, data mining, intrusion detection, random forests.**

I. Introduction

WITH the tremendous growth of network-based services and sensitive information on networks, network security is getting more important than ever. Although a wide range of security technologies such as information encryption, access control, and intrusion prevention are used to protect network-based systems, there are still many undetected intrusions [1]. Given that, intrusion detection systems (IDSs) for automatic monitoring of network activities for detecting network attacks play a vital role with respect to network security. There are two major intrusion detection techniques: misuse detection and anomaly detection. Misuse detection discovers attacks based on patterns extracted from known intrusions [9]. Anomaly detection identifies attacks based on significant deviations from normal activities [16]. Misuse detection has low false positive rate, but cannot detect novel attacks. Anomaly detection can detect unknown attacks, but usually has a high false positive rate. To combine the advantages of both misuse and anomaly detection, many hybrid approaches have been proposed [7], [8], [29].

Currently, many IDSs [4] are rule-based systems where the performances highly rely on the rules identified by security experts. Since the amount of network traffic is huge, the process of encoding rules is expensive and slow. Moreover, security people have to modify the rules or deploy new rules manually using a specific rule-driven language. To overcome the limitations of rule-based systems, a number of IDSs employ data mining techniques. Data mining is the analysis of large data sets to discover understandable patterns or models [18]. Data mining can efficiently extract patterns of intrusions for misuse detection, identify profiles of normal network activities for anomaly detection, and build classifiers to detect attacks. Data-mining-based systems are more flexible and deployable. The security experts only need to label audit data to indicate intrusions instead of hand-coding rules for intrusions. This paper proposes new systematic frameworks that apply a data mining algorithm called random forests [12] in misuse [32], anomaly [34], and hybrid [33] detection. The random forests algorithm is an ensemble classification and regression approach, which is one of the most effective data mining techniques. The random forests algorithm has been used extensively in different applications. For instance, it has been applied to prediction [17], [25] and probability estimation [31]. However, the algorithm has not been applied in automatic intrusion detection. In our proposed system, the misuse component uses the random forests algorithm for the classification in intrusion detection, while the anomaly component is based on the outlier detection mechanism of the algorithm.

One of the challenges in IDSs is feature selection. Many algorithms are sensitive to the number of features. Hence, feature selection is essential for improving detection rate. The raw data format of network traffic is not suitable for detection. IDSs must construct features from raw network traffic data, and it involves a lot of computation. Thus, feature selection can help reduce the computational cost for feature construction by reducing the number of features. However, in many current data-mining-based IDSs, feature selection is based on domain knowledge or intuition. We use the feature selection algorithm that can give estimates of what features are important in the classification. Another challenge of intrusion detection is imbalanced intrusion. Some intrusions such as denial of service (DoS) [23] have much more connections than others (e.g., user to root). Most

Manuscript received November 20, 2006; revised April 13, 2007, August 22, 2007, and November 8, 2007. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada, in part by Bell Canada through Bell University Laboratories, and in part by the Mathematics of Information Technology and Complex Systems of Canada. This paper was recommended by Associate Editor S. Rubin.

J. Zhang was with the School of Computing, Queen's University, Kingston, ON K7L 3N6, Canada. He is now with TELUS, Toronto, ON M5B 1N9, Canada (e-mail: John.Zhang2@telus.com).

M. Zulkernine is with the School of Computing, Queen's University, Kingston, ON K7L 3N6, Canada (e-mail: mzulker@cs.queensu.ca).

A. Haque is with the Network Planning Division, Bell Canada, Hamilton, ON L8P 4S6, Canada (e-mail: anwar.haque@bell.ca).

Digital Object Identifier 10.1109/TSMCC.2008.923876

of the data mining algorithms try to minimize the overall error rate, but this leads to increasing the error rate of minority intrusions. However, in real-world network environments, minority attacks are more dangerous than majority attacks. In this paper, we improve the detection performance for minority intrusions.

One of the problems of supervised anomaly intrusion detection approaches [8], [24], [30] is the high dependency on training data for normal activities. Since training data only contain historical activities, the profile of normal activities can only include the historical patterns of normal behavior. Therefore, new activities due to the change in the network environment or services are considered as deviations from the previously built profile and are detected as attacks. On the other hand, attack-free training data are difficult to obtain, since there is no guarantee that we can prevent all attacks in real-world networks. The IDSs trained by the data with hidden intrusions usually lose the ability to detect these kinds of intrusions [28]. To overcome the limitations of supervised anomaly-based systems, a number of IDSs employ unsupervised approaches [16], [21], [27]. Unsupervised anomaly detection does not need attack-free training data. It detects attacks by determining unusual activities from data under two assumptions [21]: the majority of activities are normal and the unusual activities are outliers that are inconsistent with the remainder of data set [10]. Thus, outlier detection techniques can be applied in the unsupervised anomaly detection. Actually, outlier detection has been used in a number of practical applications such as credit card fraud detection, voting irregularity analysis, and severe weather prediction [22]. We propose an approach to use outlier detection technique in anomaly intrusion detection. The outlier detection technique is effective to reduce false positives with a desirable detection rate. For hybrid detection, we propose a framework to combine misuse and anomaly detection. Therefore, the hybrid system not only achieves high performance provided by the misuse detection, but can also detect novel intrusions.

We carry out the experiments on the Knowledge Discovery and Data Mining 1999 (KDD'99) dataset and compare our results with the best results of the KDD'99 contest [14]. We are aware of the limitations of the KDD'99 datasets that are inherited from the limitations of the Defence Advanced Research Projects Agency (DARPA) datasets [38]. However, these are the most comprehensive and widely used datasets that can be employed to compare and contrast with other related IDSs. The datasets also do not require any further time-consuming preprocessing.

The major contributions of this paper are summarized as follows.

1) Propose new systematic frameworks that employ the random forests algorithm for network intrusion detection. The random forests algorithm has not been applied for automatic intrusion detection yet.
2) Apply sampling techniques and feature selection algorithm in misuse detection to improve the performance of the IDS. The sampling techniques increase the detection rate of minority intrusions. The feature selection technique improves the overall detection performance.
3) Employ a new service-based unsupervised outlier detection approach in anomaly detection. By building patterns of network services, the algorithm determines outliers related to the built patterns. The proposed approach does not need attack-free training data that are difficult to obtain in real-world network environments.
4) Combine misuse detection and anomaly detection. The combination improves the overall performance of the earlier IDSs.

The rest of the paper is organized as follows. Section II presents the related work, especially data-mining-based detection systems. Some other related work has been compared and contrasted with the proposed systems when the experimental evaluation of the systems is discussed. Section III presents the proposed misuse detection system framework and the supportive experimental evaluation. Similarly, Sections IV and V describe the anomaly and the hybrid IDSs, respectively, along with the corresponding experimental results. Finally, Section VI summarizes the paper and outlines the future research plans based on the limitations of the presented approaches.

II. Related Work

Before we compare and contrast the most related data-mining-based IDSs with the proposed intrusion detection frameworks, we discuss some representative misuse, anomaly, and hybrid IDSs.

In the expert system-based IDS [9], a set of rules are used to describe intrusions. Audit events are translated into facts that carry their semantic significance in the expert system. Then, an inference engine can draw conclusions using these rules and facts. State transition analysis expresses attacks with a set of goals and transitions based on state transition diagrams. Any event that triggers an attack state is detected as an intrusion. Signature analysis describes attacks using signatures that can be found in audit trail. Any activity that matches the signatures is identified as an attack.

Eskin *et al.* [16] investigate three algorithms for unsupervised anomaly detection: cluster-based estimation, k-nearest neighbor, and one-class support vector machine (SVM). Other researchers [21], [27] apply clustering approaches in unsupervised IDSs. Supervised anomaly detection has been studied extensively such as fuzzy data mining and genetic algorithms [24], neural networks [11], [26], and SVM [30]. Supervised anomaly detection uses attack-free training data to build profiles of normal activities. After that, it uses the deviation from the profiles to detect intrusions. Statistical methods and expert systems are also applied in supervised anomaly detection [9]. Statistical methods build the profiles of user and system normal behavior by a number of samples. Expert systems describe normal behavior of users and systems by a set of rules and then apply the rules to detect anomalous behavior.

The next-generation intrusion detection expert system (NIDES) [7] is a hybrid IDS. NIDES performs real-time monitoring of user activity on multiple-target systems connected on a network. It consists of a misuse detection component as well as an anomaly detection component. The rule-based misuse detection component employs expert rules to define known intrusive activities. The anomaly detection component is based on a statistical approach, and it flags activities as attacks if the activities deviate significantly from the expected behavior. By combining a statistical component and an expert system component, NIDES increases the chances to detect intrusions that may be missed by a single detection component.

Many data mining approaches have been proposed for intrusion detection based on association rules [8], [19], [20], [39]. Audit Data Analysis and Mining (ADAM) [8] is one of the most widely known projects in the area of data-mining-based intrusion detection. It is an online-network-based IDS. It can detect known attacks as well as unknown attacks. ADAM uses association rules algorithm for intrusion detection. It searches for all possible frequent associations among the set of given features, and usually generates many useless rules that cannot effectively describe the user and system activities. The goal of the association rules is to gather necessary knowledge about the nature of audit data. The framework of ADAM has two phases: a training phase and an online phase. In the training phase, the attack-free training data are fed to a module that performs offline association rule discovery. The output of this module is a rule-based profile of normal activities. After that, the labeled training data are fed into a classifier builder to train the classifier. In the online phase, the test data are fed into the system. With the built profile, the system finds the items classified as false alarms, attacks, and unknown attacks by the trained classifier. The unknown attacks are the suspicious items that cannot be classified as false alarms or attacks.

Mining Audit Data for Automated Models for Intrusion Detection (MADAM ID) [19] is another widely discussed data mining project for intrusion detection. It uses data mining algorithms to compute activity patterns from system audit data and extracts predictive features from the patterns. It is an offline IDS to produce anomaly and misuse intrusion models. Association rules are used to find intra-audit record patterns, and the frequent episodes algorithm is used to find inter-audit record patterns. However, MADAM ID heavily relies on intrusion detection expert knowledge. Expert knowledge is not only used to prune the number of rules produced by association and frequent episode mining, but also used to construct features.

Java Agents for Metalearning (JAM) [20] is a distributed, scalable, and portable agent-based data mining system. The main target of JAM is fraud and intrusion detection in financial information systems. Metalearning is one of the key techniques to combine and integrate separately learned classifiers or models. Hence, the distributed agents can exchange models. Zhou *et al.* [35] propose an algorithm for imbalanced intrusion data. The algorithm involves the adjustments in computing the information gain of each attribute, which achieves better precision in classifying some rare classes than the classical decision tree algorithm. The agent-based distributed IDS (DIDS) [36] includes two components: *agent nodes* and a *central node*. DIDS uses a clustering algorithm consisting of two parts. The first part is based on a clustering technique [16] implemented in the *agent nodes* to choose candidate anomalies. The other part installed in the *central node* is in charge of choosing attacks from the candidates. The algorithm employs an unsupervised clustering technique and thus breaks the dependency on attack-free data. Hu and Hu [37] call their detection system a "hybrid" one since their training data contain both normal and attack samples. The labeled samples are fed to the system to train multiple "weak" classifiers and a "strong" classifier.

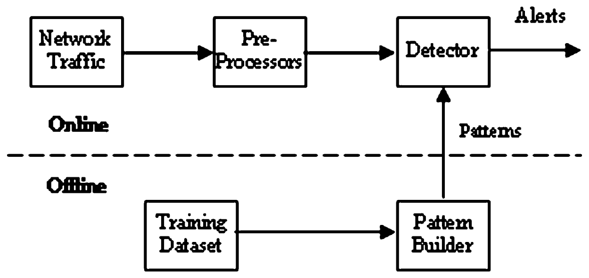

Fig. 1. Misuse detection framework.

Compared to the association-rules-based detection approaches [8], [19], [20], [39], the random forests algorithm can process large data sets with many features more efficiently. The volume of network traffic is huge, and network activities are complex with many features. Therefore, this algorithm is better suited for network intrusion detection. Our hybrid system has two phases (online phase and offline phase) similar to ADAM [8]. However, our system does not need attack-free data to detect novel intrusions using the outlier detection. Attack-free data are critical for ADAM. Due to the high complexity of the outlier detection, our system detects anomalies in the offline phase. ADAM can detect anomalies in the online phase. Compared to MADAM ID [19], our system can detect known intrusions in real time, but MADAM ID can detect intrusions only in offline mode. Although MADAM ID uses data mining techniques, it still has high reliance on expert knowledge.

Unlike JAM [20] and DIDS [36], our system employs a centralized architecture, in which all network packets are processed in the hybrid system (a single central location). Thus, there is no need to maintain separate agents scattered on the computers at each location. In our paper, we use unlabeled data in the anomaly detection. However, our misuse detection model still needs labeled data for building patterns. Using the same data sets of [35], we also discuss the scenarios in dealing imbalanced data. However, in our experiments, we make balanced data for training by replicating the rare classes. Our experiments demonstrate that the random forests algorithm not only achieves preferable detection rate but also greatly reduces training time using balanced data. Moreover, their work employs only a misuse detection technique that cannot detect unknown attacks. However, we are also able to detect some 'never-seen-before' anomalies. Unlike [37], our hybrid system uses a supervised random forests algorithm for misuse detection and an unsupervised one for anomaly detection. Furthermore, our system can automatically build known attack patterns from training data and learn to build unknown patterns from unlabeled testing data.

III. Misuse Detection

The proposed misuse detection framework (see Fig. 1) applies data mining techniques to build patterns for network intrusion detection. There are two phases in the framework: an offline phase and an online phase. The system builds patterns of intrusions in the offline phase and detects intrusions in the online phase. In the offline phase, we feed a training dataset into the pattern builder module that can build the patterns of intrusions. The module employs the feature selection algorithm, handles imbalanced intrusions, and builds the patterns by the random forests algorithm with optimal parameters. After mining the patterns for intrusions, the module outputs the patterns as the input of the detector module. In the online phase, the system captures the packets from network traffic. The features for each connection are constructed by the preprocessors from the captured network traffic. Then, the detector module classifies the connections as different intrusions or normal traffic using the patterns built in the offline phase. Finally, the system raises an alert when it detects any intrusion.

A. Optimization of Error Rate

The error rate of a forest depends on the correlation between any two trees and the strength of each tree in the forest. Increasing the correlation increases the error rate of the forest. The strength of a tree is determined by the error rate of the tree. Increasing the strength decreases the error rate of the forest. When the forest is growing, random features are selected at random out of all the features in the training data. The best split on these random features is used to split the node of the tree. The number of random features (*Mtry*) is held constant. Reducing (increasing) *Mtry* reduces (increases) both the correlation and the strength. The number of features employed in splitting each node for each tree is the primary tuning parameter (*Mtry*). To improve the performance of the random forests algorithm, this parameter should be optimized using training data. The minimum error rate corresponds to the optimal value. Therefore, we use the different values of *Mtry* to build the forests and evaluate the error rates of the forests. Then, we select the value corresponding to the minimum error rate to build the pattern.

There are two ways to evaluate the error rate. One is to split the dataset into training part and test part. We can employ the training part to build the forest and then use the test part to calculate the error rate. Another way is to use the oob (out of bag) error estimate. Because the random forests algorithm calculates the oob error during the training phase, we do not need to split the training data. We choose the oob error estimate since it is more effective by learning from the whole training dataset.

B. Minority Intrusions Detection

Intrusions are imbalanced. In other words, some intrusions produce much more connections than others. The random forests algorithm tries to minimize the overall error rate by lowering the error rate on majority classes (e.g., majority intrusions) while increasing the error rate of minority classes (e.g., minority intrusions) [12]. However, the cost of damage of minority intrusions is much higher than the damage cost of majority intrusions. Thus, for imbalanced intrusions, we need to improve the detection rate of minority intrusions while maintaining a reasonable overall detection rate. There are two solutions to deal with the imbalanced intrusions problem. One is to set different weights for different intrusions. Minority intrusions are assigned higher weights. Although the overall error rate goes up, the error rate of minority intrusions is reduced. The random forests algorithm supports this method by changing the weight parameters. The other method is to use sampling techniques: oversampling the minority intrusions and downsampling the majority intrusions. Since the network traffic is huge, downsampling the majority intrusions (e.g., normal traffic and DoS) can speed up building the patterns significantly by reducing the size of the datasets. Oversampling the minority intrusions (e.g., user to root and remote to local) can raise their weights to decrease their error rate. Therefore, we combine oversampling and downsampling in our IDS to solve the imbalanced intrusions problem instead of the first solution.

C. Feature Selection

The raw audit data of network traffic are not suitable for intrusion detection. Hence, feature construction is needed to extract a set of features that can detect intrusions effectively. Usually, the construction is based on each connection. Feature selection is one of the critical steps in building IDSs. The number of intrinsic features is fixed since the number depends on the information of packet header. However, traffic features and content features can be constructed using different methods. Hundreds of traffic and content features can be designed, while only some of them are essential for separating intrusions from normal traffic. Unessential features increase not only the computational cost but also the error rate, especially for some algorithms that are sensitive to the number of features. "Deciding upon the right set of features is difficult and time consuming" [20]. Currently, features are designed by security experts. Thus, we need an approach that can automate the feature selection. We employ variable importance calculated by the random forests algorithm in feature selection. The features with higher value of variable importance have more effect on classification. Therefore, we choose the features with the higher value of variable importance in the IDS.

D. Experiments and Results

We use the KDD'99 dataset, which was preprocessed by extracting 41 features from the tcpdump data in the 1998 DARPA datasets [3], [14]. It includes the full training set, the 10% training set, and the test set. The full training set has 4 898 431 connections. In TCP, a connection is established before two hosts on networks can communicate with each other. For UDP, each connectionless packet is also treated as a connection. The 10% training set has 494 020 connections containing all the minority classes (U2R and R2L) of the full training set and part of the majority classes (normal, DoS, and probing). The test set contains 311 029 connections. It is just like downsampling the majority classes such as normal, DoS, and probing. Hence, we use the 10% training dataset in our experiments. The task of the KDD'99 contest was to build a classifier capable of distinguishing between four kinds of intrusions and normal traffic numbered as one of the five classes: normal, probe, DoS, U2R, and R2L.

1) Performance Comparison on Balanced and Imbalanced Dataset: The original dataset (the 10% training set) is imbalanced (e.g., DoS has 391 458 connections but U2R has only 52 connections). To make a balanced training set, we downsample the Normal and DoS classes by randomly selecting 10% of connections belonging to normal and DoS from the original

TABLE I
PERFORMANCE ON THE BALANCED DATASET COMPARED TO THE ORIGINAL DATASET

Performance	Original dataset	Balanced dataset
Overall error rate	1.92%	0.05%
Time to build pattern	1975 seconds	65 seconds
True positive rate(Class 0)	0.948	0.999
True positive rate(Class 1)	0.989	0.994
True positive rate(Class 2)	1	1
True positive rate(Class 3)	0.862	1
True positive rate(Class 4)	0.83	1
False positive rate(Class 0)	0.011	0
False positive rate(Class 1)	0	0
False positive rate(Class 2)	0	0
False positive rate(Class 3)	0	0
False positive rate(Class 4)	0.01	0

dataset. We also oversample U2R and R2L by replicating their connections. The balanced training set with 60 620 connections is much smaller than the original one.

The first experiment is to compare the performance of detection between the patterns built on the original training set and the balanced training set with sampling. The experiment is carried out by using the default values of the parameters for the random forests algorithm in the Waikato environment for knowledge analysis (WEKA) [5]: 66% samples as training data, 34% samples as test data, ten trees in the forest, and six random features to split the nodes. The main objective of the experiment is to compare the performance between the balanced and the original datasets, but not to compare the effect of the parameters. As a result, for the sake of convenience, we just use the default values of the parameters for both datasets.

Table I lists overall error rate for classification, time to build pattern, true positive rate for all the classes, and false positive rate for the classes. It shows that the sampling techniques can improve the performance, especially for the detection rate (true positive rate) of the minority classes (U2R and R2L) and can reduce the time to build the patterns dramatically.

2) Selection of Important Features: The second experiment is to select the most important features. There are 41 features in the KDD'99 dataset numbered from 1 to 41. We employ the feature selection algorithm supported by the random forests algorithm to calculate the value of variable importance. To estimate the importance of the variable m, the number of votes for the correct class is counted using the oob cases in every tree. Then, the number of the correct votes is counted again after randomly permuting the values of variable m in the oob cases. The average of the margin between these two numbers over all the trees in the forest is the raw importance score for variable m. The raw score is divided by its standard error to get a z-score, and the value of variable importance is the negative z-score for variable m.

Fig. 2 plots the values of variable importance for all five categories, sorted in decreasing order. The figure shows the variable importance values of the last three features (features 7, 20, and 21) are much less than the other values. Therefore, we select the rest of the 38 most important features to build the patterns for intrusion detection. Feature 3 (service type such as http, telnet, and ftp) is the most important feature to detect intrusions. It means that the intrusions are sensitive to service type. Feature 7 (land) is used to indicate if a connection is from/to the same host. According to the domain knowledge, it is the most discriminating feature for land attacks. However, land attacks cause DoS, and they have much fewer connections than other types of DoS. After downsampling the DoS attacks, the land attacks are almost excluded from the balanced dataset. Therefore, feature 7 is not important to improve the detection rate of DoS attacks. Features 20 (number of outbound commands in an ftp session) and 21 (hot login to indicate if it is a hot login) do not show any variation for intrusion detection in the training set. The earlier analysis suggests that the feature selection can help in choosing features to detect intrusions without special domain knowledge. However, the method heavily depends on training sets.

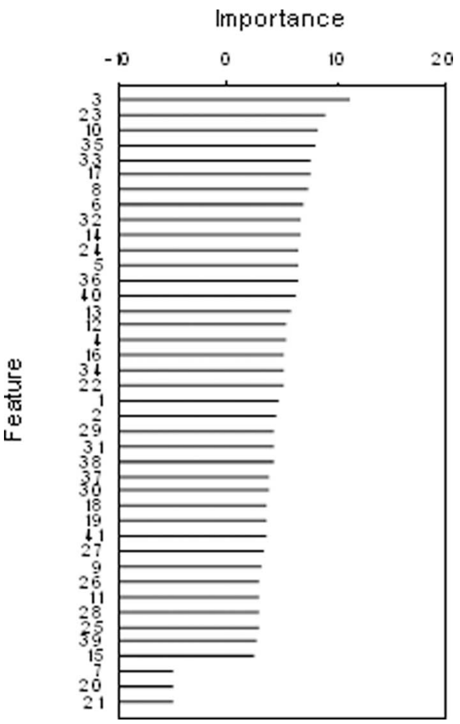

Fig. 2. Variable importance of the features.

3) Parameter Optimization: To improve the detection rate, we optimize the number of the random features (*Mtry*). We build the forest with different *Mtry* (5, 10, 15, 20, 25, 30, 35, and 38) over the balanced training set, and then plot the oob error rate and the time to build the pattern corresponding to different values for *Mtry*. As Fig. 3 shows, the oob error rate reaches the minimum when *Mtry* is 15, 25, or 30. Besides, increasing *Mtry* increases the time to build the pattern. Thus, we choose 15 as the optimal value, which reaches the minimum of the oob error rate and costs the least time among these three values.

4) Evaluation and Discussion: Different misclassifications have different levels of consequences. For example, misclassifying R2L as normal is more dangerous than misclassifying DoS as normal. We use the cost matrix as Table II published in the KDD'99 [14] to measure the damage of misclassification. M_{ij}

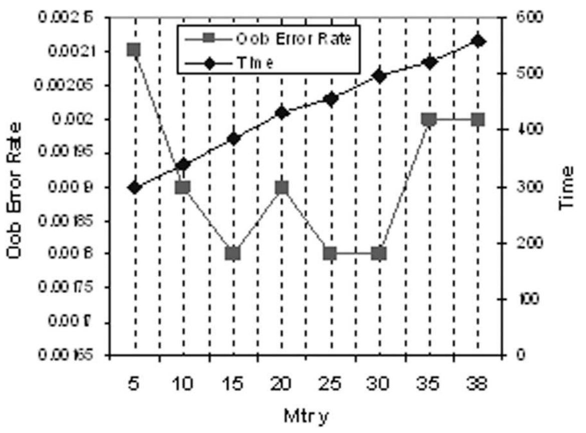

Fig. 3. Performance with different values for parameter *Mtry* of random forests.

TABLE II
COST MATRIX [14]

	Normal	Probe	DoS	U2R	R2L
Normal	0	1	2	2	2
Probe	1	0	2	2	2
DoS	2	1	0	2	2
U2R	3	2	2	0	2
R2L	4	2	2	2	0

denotes the number of samples in class i misclassified as class j, and C_{ij} indicates the corresponding cost in the cost matrix. Let N be the total number of the samples. The cost that indicates the average damage of misclassification for each connection is computed as

$$\text{cost} = \frac{\sum M_{ij} \times C_{ij}}{N}.$$

Similar to the KDD'99 contest, we evaluate our approach with the same test dataset that contains 311 029 examples. We carry out our experiment with 50 trees and 15 random features (optimized in the previous experiments). First, we build patterns on the balanced training set using all the 41 features. Then, we build patterns using the 38 most important features. The evaluation results of the patterns are reported in Table III along with the best result of the KDD'99 contest. There were 24 participants in total in the KDD'99 contest [14]. The experimental results show that our approach provides lower overall error rate and cost compared to the best KDD'99 result even without feature selection. The overall error rate is the ratio of the misclassified connections to the total connections in the test set. The results also show that the overall error rate, cost, and time to build patterns is reduced by selecting the most important features. Thus, feature selection can improve the performance of intrusion detection.

IV. ANOMALY DETECTION

The proposed anomaly detection framework applies the random forests algorithm to detect novel intrusions (see Fig. 4). The IDS captures the network traffic and constructs dataset by preprocessing. After that, service-based patterns are built over the dataset using the random forests algorithm. With the built patterns, we can find the outliers related to each pattern. The system raises alerts when any of the outliers are detected.

TABLE III
PERFORMANCE COMPARISON ON THE KDD'99 DATASET

Experiments	Overall error rate	Cost	Time (Seconds)
Best KDD Result	7.29%	0.2331	Not available
Experiment without feature selection	7.19%	0.2306	491
Experiment with feature selection	7.07%	0.2282	423

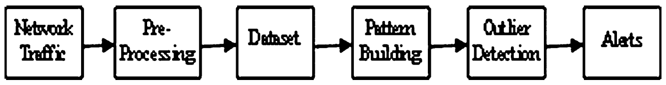

Fig. 4. Framework of the unsupervised anomaly IDS.

A. Building Patterns of Network Services

Network traffic can be categorized by services (e.g., http, telnet, and ftp). Each network service has its own pattern. Therefore, we can build patterns of network services using the random forests algorithm. However, the algorithm is supervised, and we need datasets labeled by network services. Since the information of network services is in network packets, network traffic can be labeled by the services automatically instead of time-consuming manual processing. Actually, many datasets used to evaluate IDSs can be labeled by network services with a little effort. For example, one of the features in the KDD'99 dataset is service type, which can be used as label.

Before building the patterns, we need to optimize the parameters of the random forests algorithm. The number of features employed in splitting each node for each tree is the primary tuning parameter (*Mtry*). Another parameter is the number of trees in a forest. We use the dataset to find the optimal value of the parameter *Mtry* and the number of the trees. The minimum error rate corresponds to the optimal values. Therefore, we use different values of *Mtry* and the number of trees to build the forests, and evaluate the error rate of each forest. Then, we select the values corresponding to the minimum error rate to build the service patterns.

B. Unsupervised Outlier Detection

There are two types of outliers detected here. The first type is an activity that deviates significantly from the others in the same network service. The second type is an activity whose pattern belongs to the services other than their own service. For instance, if an http activity is classified as an ftp service, the activity will be determined as an outlier. The random forests algorithm uses proximities to find the outliers whose proximities to all other cases in the entire data are generally small [12]. *Outlierness* indicates a degree of being an outlier. It can be calculated over proximities. class$(k) = j$ denotes that k belongs to class j. prox(n, k) denotes the proximity between cases n and k. The average proximity from case n in class j to case k (the rest of data in class j) is computed as

$$\overline{P}(n) = \sum_{class(k)=j} \text{prox}^2(n,k).$$

N denotes the number of cases in the dataset. The raw outlier-ness of case n is defined as

$$\frac{N}{\overline{P}(n)}.$$

The proximity is one of the most useful tools in the random forests algorithm. After the forest is constructed, all cases in the dataset are put down each tree in the forest. If cases k and n are in the same leaf of a tree, their proximity is increased by one. Finally, the proximities are normalized by dividing by the number of the trees. For a dataset with N cases, the proximities originally formed an $N \times N$ matrix. The complexity of calculation is $N \times N$. Datasets of network traffic are huge, so the calculation needs a lot of memory and CPU time. To improve the performance, we modify the algorithm to calculate the proximities. As we mentioned before, if a service activity is classified as another service, it will be determined as an outlier. Therefore, we do not care about the proximity between two cases that belong to different services. S_i denotes the number of cases in service i, the complexity will be reduced to $\sum S_i \times S_i$ after the modification.

In each class, the median and the absolute deviation of all raw outlier-ness are calculated. The median is subtracted from each raw outlier-ness. The result of the subtraction is divided by the absolute deviation to get the final outlier-ness. If the outlier-ness of a case is large, the proximity is small, and the case is determined as an outlier. To detect outliers in a dataset of network traffic, we build patterns of services over the dataset. Then, we calculate the proximity and outlier-ness for each activity. An activity that exceeds a specified threshold of outlier-ness will be determined as an outlier.

C. Experiments and Results

For the experiments, we choose five most popular network services: ftp, http, pop, smtp, and telnet. By selecting ftp, pop, telnet, 5% http, and 10% smtp normal connections, we generate a dataset called the normal dataset, which contains 47 426 normal connections. Finally, by injecting anomalies from the pool of attacks into the normal dataset, we generate four new datasets: 1%, 2%, 5%, and 10% datasets. 1% (2%, 5%, and 10%) dataset means that 1% (2%, 5%, and 10%) of connections in the dataset are attacks. We carry out the first experiment over the 1% attack dataset in a similar way used in [16] and [21].

Fig. 5 plots the outlier-ness of the 1% attack dataset. The IDS raises an alert if an outlier-ness of a connection exceeds a specified threshold. We evaluate the performance of our system by the detection rate and the false positive rate.

Fig. 6 plots receiver operating characteristic (ROC) curve to show the relationship between the detection rates and the false positive rates over the 1% attack dataset. The result indicates that our system can achieve a high detection rate with a low false positive rate. Compared to other unsupervised anomaly-based

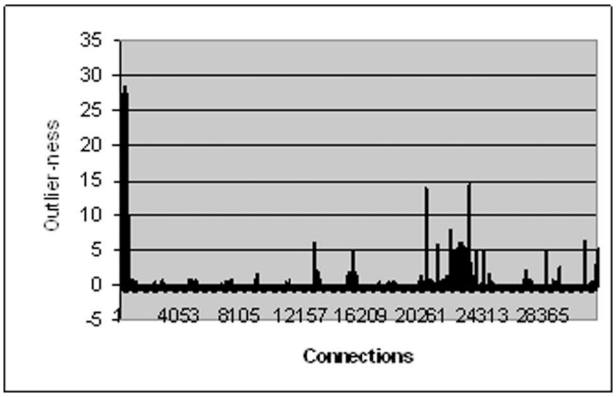

Fig. 5. Outlier-ness of the 1% attack dataset.

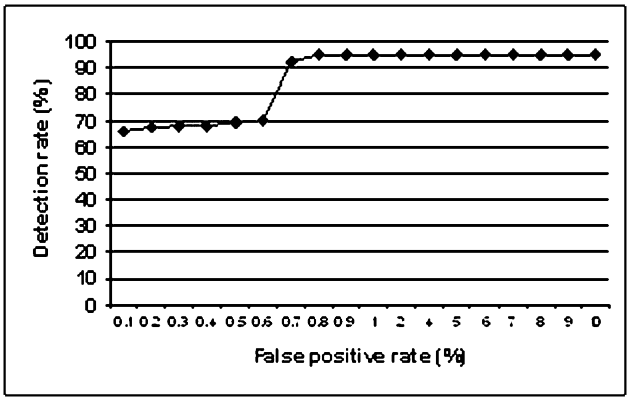

Fig. 6. ROC curve for the 1% attack dataset.

TABLE IV
OPTIMAL PARAMETERS OF RANDOM FORESTS

Dataset	Trees	Mtry
1% dataset	25	40
2% dataset	20	35
5% dataset	10	5
10% dataset	10	5

systems [16], [21], our system has better performance over the KDD'99 dataset while the false positive rate is low.

1) Detection Performance Over Different Datasets: To evaluate our system under different number of attacks, we carry out the experiments over the 1%, 2%, 5%, and 10% attack datasets. The optimal parameters for each dataset are listed in Table IV. With the optimized parameters, we build the patterns of the network services. Over the built patterns, the IDS calculates the outlier-ness of each connection. Fig. 7 plots the ROCs for each dataset. The result shows that the performance tends to be reduced with the increasing number of attacks, and the performance depends on the proportion of attacks in the datasets.

2) Detection Performance Over Minority Intrusions: Minority intrusions are more difficult to detect than majority intrusions. We evaluate the performance to detect minority intrusions using outlier detection. The optimal value of *Mtry* is 5. Fig. 8 plots ROC curve to show the relationship between the detection rates and the false positive rates over the minority attack dataset. The result indicates that the detection rate of minority intrusions

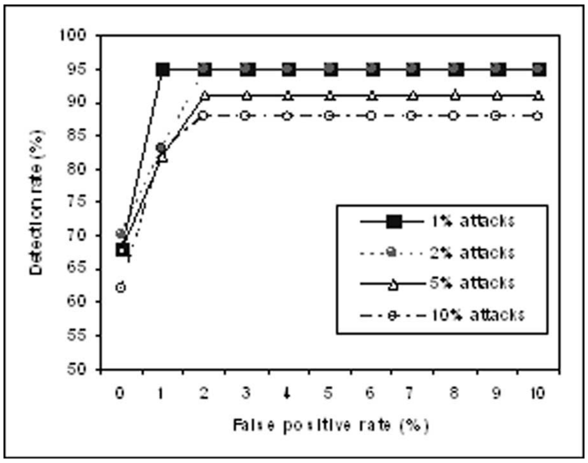

Fig. 7. ROC curves for different datasets.

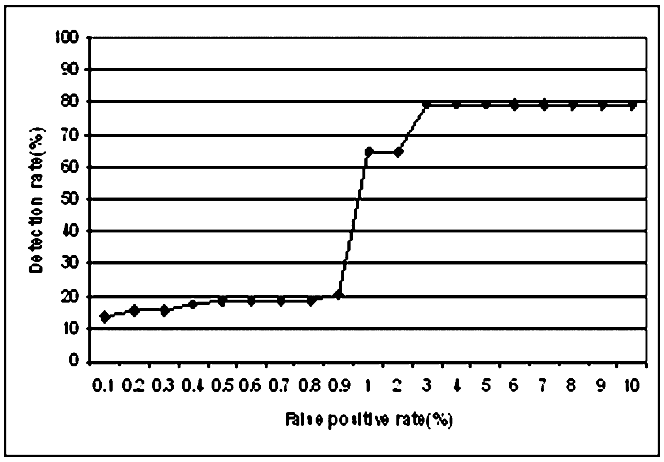

Fig. 8. ROC curve for the minority attack dataset.

is lower than the experiments on the detection performance over different datasets in the previous section. However, the result is still impressive. The detection rate can reach 65% when the false positive is 1%. In the KDD'99 contest, the detection rate of minority intrusion is much lower than that of majority intrusions even using misuse detection [14].

V. Hybrid Intrusion Detection

To address the problems of misuse and anomaly detection, we can combine our proposed misuse and anomaly IDSs in three ways: 1) anomaly detection followed by misuse detection; 2) misuse and anomaly detection in parallel; and 3) misuse detection followed by anomaly detection.

Since our anomaly approach has better performance when false positive rate is low, the third approach is more suitable than the others (see Fig. 9). For the first approach, the anomaly detection component should have very high detection rate. The low false positive rate is not critical. The misuse detection component needs the ability to identify false alarms to reduce the overall false positive rate. The high complexity of our anomaly detection does not match the high-speed detection performance of our misuse detection. Thus, combining both the approaches

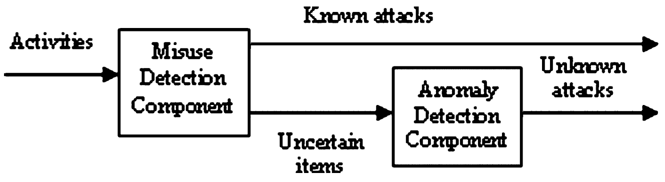

Fig. 9. Overview of the proposed hybrid detection approach.

in parallel makes real-time detection impossible. Moreover, the experimental results in the anomaly detection show that the performance tends to be reduced with increasing number of attack connections. That is a problem of unsupervised systems. Some attacks such as DoS attacks produce a large number of connections, which may undermine an unsupervised anomaly detection system. To overcome the problem, we use the third approach. By removing the known attacks through the misuse detection first, the number of attacks can be reduced significantly in the datasets fed for the unsupervised anomaly detection. Another reason to use the third approach is that the misuse detection by the random forests algorithm has high-speed performance. Thus, the hybrid system can be used to detect known intrusions in real time and to detect unknown intrusions offline. The low-speed performance of the anomaly detection makes real-time detection impossible using the first and second approaches.

A. *Hybrid Detection Framework*

The proposed hybrid system combines the misuse detection component and the anomaly detection component, as shown in Fig. 10. The figure also shows the details of each of the components. There are two phases in the framework: an offline phase and an online phase. The system can build patterns of intrusions for the misuse detection component and can detect unknown intrusions using the anomaly detection component in the offline phase. It detects known intrusions using the misuse detection component in the online phase. In the offline phase, after preprocessing the training data, the preprocessed data are stored in the training database. The intrusion pattern builder module is trained from the data in the training database, and it outputs patterns of intrusions to the misuse detector module. In the online phase, network traffic is captured by the sensors and fed to the misuse detector after being preprocessed. The misuse detector raises an alarm to the misuse alarmer if any connection matches an intrusion pattern. Then, the misuse alarmer delivers alarms to security analysts. If the connection does not match any intrusion pattern, it is sent to the anomaly database module that stores data for the anomaly detection component. In the offline phase, the system can detect novel intrusions using the anomaly detection component. First, the service pattern builder module retrieves data from the anomaly database to build patterns of network services and outputs the built patterns to the outlier detector module. The outlier detector retrieves the data from the anomaly database and uses the outlier detection technique to detect attacks. If it detects any attack, it raises alarms to the anomaly alarmer module for security analysts. It can also store the newly detected intrusions in the training database so that the new intrusion patterns can be built for misuse detection.

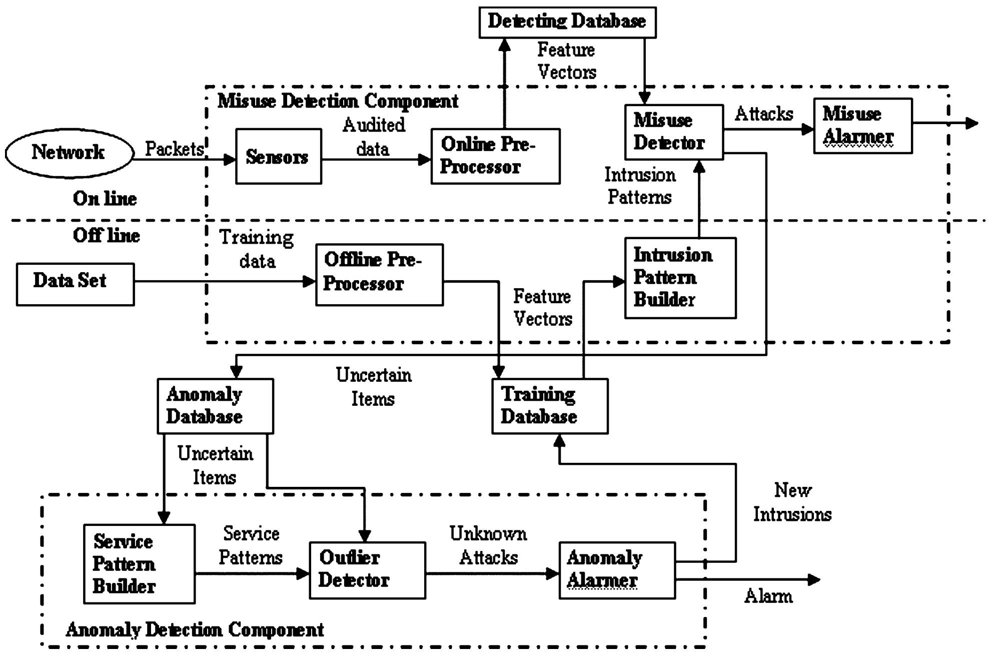

Fig. 10. Framework of the hybrid detection.

B. Experiments and Results

We again use the KDD'99 dataset to evaluate the hybrid approach. To calculate the detection rate and false positive rate easily, we change the label to 1 or 0 (1 if connection is an intrusion; 0 otherwise) instead of the types. Then, we choose the five most popular network services: ftp, http, pop, smtp, and telnet to generate a training set that contains 16 919 connections with downsampling normal connections. Similarly, we generate our test set that contains 49 838 connections without downsampling normal connections. The training set is used to build patterns of intrusions or misuse detection, while the test set is used to evaluate the hybrid approach.

To improve the performance of misuse detection, we employ the feature selection algorithm to calculate the value of variable importance over the training set. The result shows that the variable importance of the last seven features (features 2, 7, 8, 9, 15, 20, and 21) are less significant than others. Therefore, we select the 34 most important features to build patterns of intrusions. We also optimize the parameters (*Mtry* and the number of trees) of the random forests algorithm for the misuse detection. By building patterns of intrusions with different *Mtry* (5, 10, 15, 20, 25, 30, and 34) and different number of trees (10, 15, 20, 25, 30, 35, 40, 45, and 50), we get the oob error rates for each pattern. The minimum error rate corresponds to the optimized parameters (the number of trees is 15; the value of *Mtry* is 34). In misuse detection, with the optimized parameters, we build the patterns of intrusions. With the built patterns, we use the misuse approach to detecting intrusions over the test set. The detection rate is 94.2%, and the false positive rate is 1.1%.

By excluding the detected intrusions from the test set, we generate the anomaly test set. The anomaly test set is labeled by the services. We optimize the parameters for the anomaly detection over the anomaly test set. In anomaly detection, with the optimized parameters (the number of trees is 35, the value of *Mtry* is 34), we build the patterns of the services. With the built patterns, we use the outlier detection approach to calculate the outlier-ness of the connections in the anomaly test set. The connections are sorted by outlier-ness in descending order. The first 1% of the connections is determined as intrusions. We choose 1% as the threshold so that the false positive rate of the anomaly detection remains below 1%. Thirty connections are identified as new intrusions.

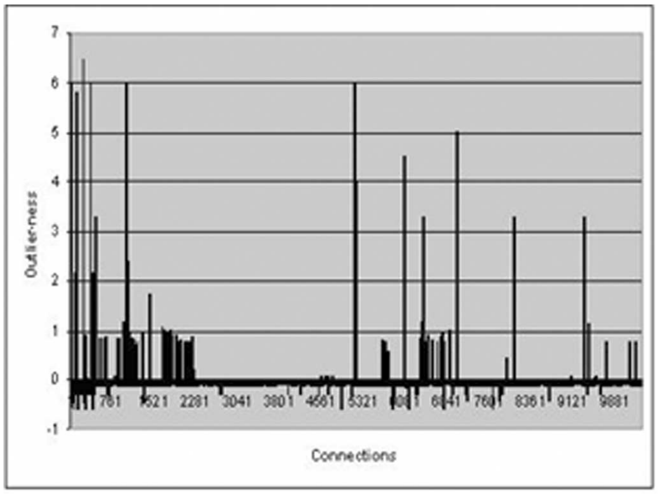

Fig. 11. Outlier-ness of the anomaly test set.

Fig. 11 plots the outlier-ness. There are 411 attacks in the anomaly set. The attacks are injected at the beginning of the dataset. As shown in the figure, some intrusive connections have much higher outlier-ness than the normal connections, but most of them have much lower outlier-ness than the normal ones. The explanation is that these intrusions are very similar to each other. They very likely fall into the same leaf of a tree built by the random forests algorithm, so they have very low outlier-ness.

The experimental results show that the proposed hybrid approach can achieve high detection rate with low false positive rate and can detect novel intrusions. The overall detection rate of the hybrid system is 94.7%. The overall false positive rate is 2%. The result shows that the anomaly approach detects some intrusions that are missed by the misuse approach.

VI. Conclusion and Future Work

In this paper, we outline three data-mining-based frameworks for network intrusion detection. We apply the random forests algorithm in misuse, anomaly, and hybrid detection. To address the problems of rule-based systems, we employ the random forests algorithm to build patterns of intrusions. By learning over training data, the random forests algorithm can build the patterns automatically instead of coding rules manually. The proposed approaches are implemented in Java program using the WEKA environment [5] and the Fortran 77 program [2]. We evaluate the implementations over different datasets obtained from the KDD'99 datasets, and the experimental results show that the performances of our approaches are better than the best KDD'99 results.

In our misuse detection framework, patterns of intrusions are built in the offline phase, and the system can automatically detect intrusions in real time using the built patterns. To improve the accuracy of the system, we use the feature selection algorithm and optimize the parameters of the random forests algorithm. We also use sampling techniques to increase the detection rate of minority intrusions in the framework. Since the misuse detection cannot detect novel intrusions, we propose a new approach in unsupervised anomaly detection. We apply the outlier detection of the random forests algorithm in anomaly detection. The outliers detected by the random forests algorithm are determined as intrusions. Since the random forests algorithm is a supervised data mining algorithm, it uses labeled training data to build patterns. Therefore, our approach builds patterns of network services. With the built patterns of services, the approach determines the outliers related to the patterns as intrusions. The approach breaks the dependency on attack-free training data, which is the major problem of supervised anomaly detection. To improve the performance of detecting outliers, we modify the original outlier detection algorithm to reduce the complexity of calculation. The results confirm that our approach achieves higher detection rate when the false positive rate is low, compared to the other reported unsupervised anomaly detection approaches. The results also show that the detection performance tends to decrease with the increasing number of attack connections in a dataset. We also propose a new framework to combine the misuse and the anomaly detection in which we apply the random forests algorithm. In the hybrid framework, the misuse detection is applied first to filter out the known intrusions from the datasets entering the anomaly detection component. Hence, the detection performance of the anomaly approach is improved. The experimental evaluation on our hybrid approach indicates that the proposed hybrid framework can achieve higher detection rate with low false positive rate in comparison with other hybrid systems.

The hybrid technique has two limitations due to the shortcomings of the proposed misuse and anomaly detection techniques. First, the intrusions in a dataset need to be much less than normal data. The outlier detection only works when the majority of data are normal. We use the misuse detection to filter out known intrusions. However, this cannot guarantee that the majority of activities are normal after removing known intrusions. For example, a new type of intrusion may produce large number of connections, which cannot be filtered out by the misuse detection. This could decrease the performance of the anomaly detection. Moreover, it may undermine the hybrid system. Second, some intrusions with high degree of similarity cannot be detected as outliers by the anomaly detection.

In future, other data mining algorithms such as clustering algorithm could be investigated to solve the earlier problems. Besides, we will analyze datasets and intrusions to predict certain intrusions using the random forests algorithm. The random forests algorithm has been successfully applied in various fields to find patterns that are suitable for prediction in large volumes of data. Basically, in intrusion prediction, we can predict a specific intrusion based on symptoms. There are some symptoms (predictor) for some intrusions. For example, IP scan activity is a predictor for worm propagation.

References

[1] CSI/FBI Computer Crime and Security Survey. (2004). Computer Security Inst., San Francisco, CA. [Online]. Available: http://www.issa-sac.org/docs/FBI2004.pdf

[2] L. Breiman and A. Cutler, Random Forests. (2006). [Online]. Available: http://stat-www.berkeley.edu/users/breiman/RandomForests/cchome.htm

[3] DARPA Intrusion Detection Evaluation. (2006). [Online]. Available: http://www.ll.mit.edu/IST/ideval/

[4] Snort Network Intrusion Detection System. (2006). [Online]. Available: http://www.snort.org

[5] WEKA software. (2006). [Online]. Available: http://www.cs.waikato.ac.nz/ml/weka/

[6] T. Abraham, "IDDM: Intrusion detection using data mining techniques," DSTO Electron. Surveill. Res. Lab., Salisbury, Australia, Tech. Rep. DSTO-GD-0286, May 2001.

[7] D. Anderson, T. Frivold, and A. Valdes, "Next-generation intrusion detection expert system (NIDES)—A summary," SRI Int., Menlo Park, CA, Tech. Rep. SRI-CSL-95-07, May 1995.

[8] D. Barbara, J. Couto, S. Jajodia, L. Popyack, and N. Wu, "ADAM: Detecting intrusions by data mining," in *Proc. 2nd Annu. IEEE Workshop Inf. Assur. Secur.*, New York, Jun. 2001, pp. 11–16.

[9] D. Barbara and S. Jajodia, *Applications of Data Mining in Computer Security*. Norwell, MA: Kluwer, 2002.

[10] V. Barnett and T. Lewis, *Outliers in Statistical Data*. New York: Wiley, 1994.

[11] A. Bivens, M. Embrechts, C. Palagiri, R. Smith, and B. Szymanski, "Network-based intrusion detection using neural networks," in *Proc. Artif. Neural Netw. Eng.*, St. Louis, MO, Nov. 2002, vol. 12, pp. 527–535.

[12] L. Breiman, "Random forests," *Mach. Learn.*, vol. 45, pp. 5–32, 2001.

[13] R. Duda, P. Hart, and D. Stork, *Pattern Classification*. New York: Wiley, 2000.

[14] C. Elkan, "Results of the KDD'99 classifier learning," *SIGKDD Explorations*, vol. 1, no. 2, pp. 63–64, 2000.

[15] C. Endorf, E. Schultz, and J. Mellander, *Intrusion Detection & Prevention*. New York: McGraw-Hill, 2004.

[16] E. Eskin, A. Arnold, M. Prerau, L. Portnoy, and S. Stolfo, "A geometric framework for unsupervised anomaly detection: Detecting intrusions in unlabeled data," in *Applications of Data Mining in Computer Security*. Norwell, MA: Kluwer, 2002.

[17] L. Guo, Y. Ma, B. Cukic, and H. Singh, “Robust prediction of fault-proneness by random forests,” in *Proc. 15th Int. Symp. Softw. Rel. Eng. (ISSRE)*, Brittany, France, Nov. 2004, pp. 417–428.

[18] D. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*. Cambridge, MA: MIT Press, Aug. 2001.

[19] W. Lee and S. Stolfo, “A framework for constructing features and models for intrusion detection systems,” *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 4, pp. 227–261, Nov. 2000.

[20] W. Lee and S. Stolfo, “Data mining approaches for intrusion detection,” in *Proc. 7th USENIX Secur. Symp.*, San Antonio, TX, Jan. 1998, pp. 79–83.

[21] K. Leung and C. Leckie, “Unsupervised anomaly detection in network intrusion detection using clusters,” in *Proc. 28th Australasian CS Conf.*, Newcastle, Australia, Jan. 2005, vol. 38, pp. 333–342.

[22] C. Lu, D. Chen, and Y. Kou, “Algorithms for spatial outlier detection,” in *Proc. 3rd IEEE Int. Conf. Data Mining*, Melbourne, FL, Nov. 2003, pp. 597–600.

[23] M. Mahoney and P. Chan, “An analysis of the 1999 DARPA/Lincoln Laboratory evaluation data for network anomaly detection,” in *Proc. Recent Adv. Intrusion Detect. (RAID)*, Pittsburgh, PA, Sep. 2003, Lecture Notes in Computer Science, vol. 2820, pp. 220–237.

[24] S. Bridges and R. Vaughn, “Fuzzy data mining and genetic algorithms applied to intrusion detection,” in *Proc. Nat. Inf. Syst. Secur. Conf. (NISSC)*, Baltimore, MD, Oct. 2000, pp. 13–31.

[25] B. Popescu and J. Friedman, “Ensemble learning for prediction,” Doctoral dissertation, Stanford Univ., Stanford, CA, Jan. 2004.

[26] M. Ramadas, S. Ostermann, and B. Tjaden, “Detecting anomalous network traffic with self-organizing maps,” in *Proc. Recent Adv. Intrusion Detect. (RAID)*, Pittsburgh, PA, Sep. 2003, Lecture Notes in Computer Science, vol. 2820, pp. 36–54.

[27] R. Smith, A. Bivens, M. Embrechts, C. Palagiri, and B. Szymanski, “Clustering approaches for anomaly based intrusion detection,” presented at the 1st Annu. Walter Lincoln Hawkins Graduate Res. Conf., New York, Oct. 2002.

[28] K. Tan, K. Killourhy, and R. Maxion, “Undermining an anomaly based intrusion detection system using common exploits,” in *Proc. Recent Adv. Intrusion Detect. (RAID)*, Zurich, Switzerland, Oct. 2002, pp. 54–73.

[29] E. Tombini, H. Debar, L. Me, and M. Ducasse, “A serial combination of anomaly and misuse IDSes applied to HTTP traffic,” in *Proc. 20th Annu. Comput. Secur. Appl. Conf.*, Tucson, AZ, Dec. 2004, pp. 428–437.

[30] Q. Tran, H. Duan, and X. Li, “One-class support vector machine for anomaly network traffic detection,” presented at the 2nd Netw. Res. Workshop 18th APAN, Cairns, Australia, Jul. 2004.

[31] T. Wu, C. Lin, and R. Weng, “Probability estimates for multi-class classification by pairwise coupling,” *J. Mach. Learn. Res.*, vol. 5, pp. 975–1005, Dec. 2004.

[32] J. Zhang and M. Zulkernine, “Network intrusion detection using random forests,” in *Proc. 3rd Annu. Conf. Privacy, Secur. Trust (PST)*, St. Andrews, NB, Canada, Oct. 2005, pp. 53–61.

[33] J. Zhang and M. Zulkernine, “A hybrid network intrusion detection technique using random forests,” in *Proc. Int. Conf. Availability, Reliability Secur. (AReS)*. Vienna, Austria: IEEE CS Press, Apr. 2006, pp. 262–269.

[34] J. Zhang and M. Zulkernine, “Anomaly based network intrusion detection with unsupervised outlier detection,” in *Proc. IEEE Int. Conf. Commun. (ICC)—Symp. Netw. Secur. Inf. Assur.*, Istanbul, Turkey, Jun. 2006, vol. 5, pp. 2388–2393.

[35] Q. Zhou, L. Gu, C. Wang, J. Wang, and S. Chen, “Using an improved C4.5 for imbalanced dataset of intrusion,” in *Proc. 4th Annu. Privacy Secur. Trust Conf.*, Markham, Canada, Oct. 2006, pp. 481–484.

[36] Y. Zhang, Y. Zhong, and X. Wang, “Distributed intrusion detection based on clustering,” in *Proc. 4th Int. Conf. Mach. Learn. Cybern.*, Guangzhou, China, Aug. 2005, vol. 4, pp. 2379–2383.

[37] W. Hu and W. Hu, “Network-based intrusion detection using adaboost algorithm,” in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell.*, Sep. 2005, pp. 712–717.

[38] J. McHugh, “Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA offline intrusion detection system evaluation as performed by Lincoln Laboratory,” *ACM Trans. Inf. Syst. Secur.*, vol. 3, no. 4, pp. 262–294, Nov. 2000.

[39] W. Lee, S. Stolfo, and K. Mok, “Adaptive intrusion detection: A data mining approach,” *Artif. Intell. Rev.*, vol. 14, pp. 533–567, Dec. 2000.

Jiong Zhang received the M.Sc. degree from the School of Computing, Queen’s University, Kingston, ON, Canada, in 2005.

He is currently a vulnerability research specialist at TELUS, Toronto, ON. His current research interests include intrusion detection and software vulnerability.

Mohammad Zulkernine received the B.Sc. degree in computer science and engineering from Bangladesh University of Engineering and Technology, Dhaka, in 1993, the M.Eng. degree in computer science and systems engineering from Muroran Institute of Technology, Muroran, Japan, in 1998, and the Ph.D. degree from the University of Waterloo, Waterloo, ON, Canada, in 2003.

He is currently a faculty member of the School of Computing, Queen’s University, Kingston, ON, where he is leading the Queen’s Reliable Software Technology Research Group, and is also with the Department of Electrical and Computer Engineering. He has been with the Bell Canada Software Reliability Laboratory, University of Waterloo. His current research interests include software engineering (software reliability and security), specification-based automatic intrusion detection, and software behavior monitoring. His research work is funded by a number of provincial and federal research organizations of Canada, while he is having an industry research partnership with Bell Canada.

Dr. Zulkernine is a member of the Association for Computing Machinery and the IEEE Computer Society. He is a Licensed Professional Engineer of the Province of Ontario, Canada.

Anwar Haque received the B.S. degree from the North South University, Dhaka, Bangladesh, in 1997, the M.Sc. degree from the University of Windsor, Windsor, ON, Canada, in 2001, and the M.Math. degree from the University of Waterloo, Waterloo, ON, in 2007, all in computer science.

He is currently a Manager — Network Planning at Bell Canada, Hamilton, ON. He has authored or coauthored more than 25 referred technical papers. He has served as a reviewer and Technical Program Committee (TPC) member for many international journals and conferences. His current research interests include IP network QoS, network design and planning with focus on survivability, and network security.

Mr. Haque is the recipient of the University of Waterloo Graduate Scholarship, the Natural Sciences and Engineering Research Council of Canada Industrial Postgraduate Scholarship (NSERC-IPS), the Ontario Graduate Scholarship (OGS), and the David R. Cheriton Graduate Scholarship.