

A SEMI-SUPERVISED MODEL FOR NETWORK TRAFFIC ANOMALY DETECTION

Nguyen Ha Duong*, Hoang Dang Hai**

*Faculty of Information and Technology, National University of Civil Engineering, Vietnam

** Ministry of Information and Communication, Vietnam

nghaduong@gmail.com, hddhai.hn@gmail.com

Abstract— Network traffic anomaly detection can help to early detect network attacks because hacker's activities may result in unusual changes of network traffic, that are significant fluctuations compared to normal traffic of the network. Among various anomaly detection approaches, principal component analysis (PCA) has been seen as an effective solution. Until now, PCA is basically applied to dimension reduction method. Several issues remain including: how effective can PCA be applied to semi-supervised models with a small training dataset, which components are significant for anomaly detection. This paper proposes a semi-supervised model using a modified Mahalanobis distance based on PCA for network traffic anomaly detection. We propose a K-means clustering method to build normal profile of traffic to improve the training dataset and propose to give weights to choose principal components of PCA.

Keywords— Network traffic anomaly, anomaly detection, semi-supervised model, intrusion detection, network security

I. INTRODUCTION

Today, network security is world-wide major concern of many countries. Organizations, companies and agencies are often facing with network attacks. The Intrusion Detection System (IDS) is implemented as an effective device to detect attacks outside or inside of a network. However, typical IDS often rely on signature database or pattern of known attack [1][2]. Therefore, intruders can change some parameters or characters that different from known patterns to make IDS unable to detect the new variances. Anomaly detection is the approach of recent IDS [3-6], since it does not require any prior knowledge about the attack signatures. Thus, it is capable to detect new attacks. Anomaly detection system (ADS) is used to detect the abnormal behaviour of a system. ADS can operate independently or as a component of IDS.

There are various network anomaly detection methods in recent years including machine learning techniques, statistical-based methods, principal component analysis (PCA) methods, etc. A review of different approaches for anomaly detection was given in [1-7]. Various methods and techniques proposed for anomaly detection indicate the difficulties of network traffic anomaly detection. There are several reasons: 1) the techniques of attackers become more sophisticated. There are many types of anomalies with different traffic data features. 2) No existing method is considered better than the others due to the complexity of the anomalies. Several issues remain unsolved regarding detection speed, accuracy, confidence, complexity,

etc. Among various anomaly detection approaches, PCA has been proposed as an effective solution [8-10]. PCA is useful to reduce the complexity of the dataset while maintaining significant dataset features. From high level viewpoint, anomaly detection can be categorized into 3 models [7]:

- Supervised: models both normality and abnormality. The entire area outside the normal class represents the outlier class. Supervised detection techniques fail to recognize behaviour that is not previously modelled, thus it lacks the ability to classify unknown anomalies.
- Unsupervised: no prior knowledge of the data is needed. It processes the data as a static distribution, pinpoints the most remote points, flags them as potential outliers (anomaly).
- Semi-Supervised: models only normality. It needs pre-classified data but only learns data marked normal. It is suitable for static or dynamic data as it only learns one class which provides the model of normality (baseline). If a point's distance exceeds the established threshold from the normal baseline, it is considered abnormal point.

Supervised models are developed as prediction models based on training datasets. If the anomalies are far fewer than normal pattern, the accuracy of the models is low. Unsupervised can be effective if the anomaly points are much lesser than the normal points to avoid high false positive rate. This techniques implicit assume that normal data pattern are far more present than anomalies, thus high false detection rate may occur. In semi-supervised models, normal data pattern are used for training dataset. This approach allows a high detection rate and promises a good accuracy if it is used in combination with clustering methods. Since this approach only models anomaly in the dataset, the complexity of the dataset can be remarkable reduced.

In this paper, we propose a semi-supervised model using a modified Mahalanobis distance based on PCA (M-PCA) for network traffic anomaly detection. In order to reduce the noise of anomalies, we propose to extend the K-means clustering algorithm to group similar data points and to build normal profile of traffic. This algorithm helps to improve the quality of the training dataset. On the other hand, we propose a modified Mahalanobis distance based on PCA and the different weights assigned to choose the principal components of the original PCA. The rest of this paper is organised as follows. Section 2 presents related works. In Section 3, we propose modified Mahalanobis distance based on PCA and weights. Section 4

describes our semi-supervised model. Section 5 presents our experiment and results. Section 6 concludes the paper.

II. RELATED WORKS

The authors in [3-7] presented a review of anomaly based intrusion detection systems. A version of apriori algorithm was used with systolic arrays to build efficient pattern matching similar to a signature based method. In [5], the existence of irrelevant and redundant features has been studied that affect the performance of machine learning part of the detection system. This work showed that a good selection of the features will result in better classification performance. The authors in [6] demonstrated that the elimination of the unimportant features and irrelevant features did not reduce the performance of the detection systems.

Anomaly detection models based on PCA was first proposed in [8]. PCA was used together with outlier detection in assumption that the anomalies appear as outliers to the normal data. PCA can reduce the dimensionality of the dataset. The authors in [7-10] further improved PCA algorithm in combination with several algorithm such as sketch-based and signal-analysis based in a framework. The authors in [11] showed that an important advantage of combining redundant and complementary classifiers is to increase accuracy and better overall generalization. Several authors [9-12] also identified important input features in building IDS that are computationally efficient and effective. This work showed the performance of various feature selection algorithms including bayesian networks, classification and regression trees.

Several works provided experiments using KDD-CUP'99 dataset [13-17], which is a subset of the Intrusion Detection Evaluation dataset of DAPRA. Almost works proposed to use 41 features from the raw data of this dataset.

III. M-PCA METHOD

A. Principal component analysis (PCA)

Principal Component Analysis (PCA) is a method for identifying patterns in data, and expressing the data in such a way as to highlight their similarities and differences [8]. PCA produces a set of principal components, based on eigenvalue/eigenvector pairs. Eigenvalues/eigenvectors can be built from covariance or correlation matrix. Principal components (PCs) are particular linear combinations of p random variables $X_1, X_2, \dots, X_p$. These PCs need to have three important properties:

- They are uncorrelated.
- They are sorted in descending order of variance.
- The total variance is equal to the sum of the original variables

These PCs often have different measures with different units. In covariance matrix, the original variables have larger magnitude often have larger weight in the leading PCs. Correlation matrix require all variables be standardized with mean, standard deviation and the resulting PCs have more balance weights with original variables. Due to this reason, correlation matrix is often used for computing eigenvalue and eigenvector.

The correlation matrix R consists of $p \times p$ variables $X_1, X_2, \dots, X_p$. We denote $x = (x_1, x_2, \dots, x_p)'$ the observed data on the

variables $X_1, X_2, \dots, X_p$. Let $(\lambda_1, e_1), (\lambda_2, e_2), \dots, (\lambda_p, e_p)$ the p eigenvalue/eigenvector pairs of the R , where $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$. The i^{th} principal component has variance λ_i .

We calculate the i^{th} PC of an observation data x as follows:

$$y_i = e_i'z = e_{i1}z_1 + e_{i2}z_2 + \dots + e_{ip}z_p, i = 1, 2, \dots, p \quad (1)$$

where: $e_i = (e_{i1}, e_{i2}, \dots, e_{ip})'$ is the i^{th} eigenvector, $z = (z_1, z_2, \dots, z_p)'$ is the standardized observed vector along the variables $X_1, X_2, \dots, X_p$. We have:

$$z_k = \frac{x_k - \bar{x}_k}{\sigma_k}, k = 1, 2, \dots, p \quad (2)$$

Where $\bar{x}_k$ and σ_k is the mean and standard deviation of the variable x_k in trained data set respectively. The covariance or correlation of any pair of PCs is equal to zero. The PCA produces a set of independent variables so the total variance of a sample is the sum of all the variances accounted for by the PCs.

B. PCA Detection with Mahalanobis Distance and Weights

Outlier detection techniques are used to calculate the distance of captured live network data to the 'normal data' projected by PCA procedure. We proposed using Mahalanobis distance for outlier detection, thus outliers measured are presumably anomalous network connections. Any network connection with a distance greater than an established threshold value is considered an outlier. The equation of Mahalanobis distance d between observation x and the sample mean μ is [8]:

$$d^2(x, \mu) = (x - \mu)'S^{-1}(x - \mu) \quad (3)$$

where S^{-1} is the sample covariance matrix. This equation takes into account the covariance matrix, thus, it can measure correlation between variables. In this paper, we use correlation matrix instead of covariance matrix since many variables in the training dataset were measured on different scales and ranges. The correlation matrix will represent more effectively the relationships between the data variables. The drawback of this method is the computationally demanding when calculating the inverse of the correlation matrix for feature vectors with a large number of dimensions. We need a method to calculate this distance more efficiently for each new connection. The sum square of standardized PCs score is equivalent to the Mahalanobis distance of the observation x from the mean vector μ of the trained data set as follows:

$$d(x, \mu) = \sum_{i=1}^p \frac{y_i^2}{\lambda_i} \quad (4)$$

where y_i is i^{th} the PC score, λ_i is the i^{th} eigenvalue.

The eigenvalue of each PC corresponds to the relative amount of variation it encompasses. The larger the eigenvalue, the more significant its corresponding projected eigenvector. Therefore, it is reasonable to sort all PCs from most to least significant and give weights to PCs based on their eigenvalues. Intuitively, we can divide PCs into the set of *major PCs* (most significant) with the weight W_1 and *minor PCs* (least significant) with the weight W_2 . Thus, we have:

$$d(\text{total sum}) = d(W_1.\text{major}) + d(W_2.\text{minor}) \quad (5)$$

where W_i is the weight for choosing the number of the principal components in each sub sum. W_i is chosen based on experiment and the variance of eigenvalues.

If a new data item is projected along the upper set of the significant PCs (with higher weight), it is likely that the data

item can be classified without projecting along all PCs. Eigenvectors of PCs represent axes which best suit a data sample. Points, which lie at a far distance from these axes would exhibit abnormal behaviour. The presented major-minor method has better detection rate than the use the total sum of distance with all PCs. The sum of all of the square of standardized PCs score is dominant by a few first component. PCs with larger magnitude of variance than others will have more weight in the sum of distance. We split the equation into small equations as (5) and calculate each sub PCs score distance.

$$d(\text{total sum}) = \sum_{i=1}^p \frac{y_i^2}{\lambda_i}$$

$$d(\text{major}) = \sum_{i=1}^q \frac{y_i^2}{\lambda_i}$$

$$d(\text{minor}) = \sum_{i=r}^p \frac{y_i^2}{\lambda_i} \quad (6)$$

where: $0 < q < r < p$

In (6), *major* and *minor* is the distance threshold. Any observation has distance greater than *major* or *minor* distance threshold is considered abnormal or attack.

IV. SEMI-SUPERVISED MODEL FOR ANOMALY DETECTION

Figure 1 shows our proposed semi-supervised model for anomaly detection using M-PCA and K-means clustering method to reduce the noise data of the dataset.

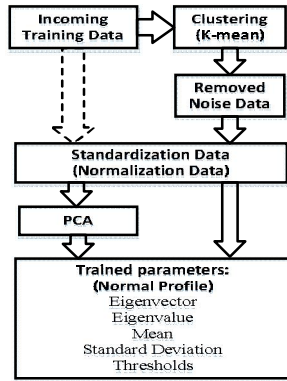


Figure 1. A semi-supervised model for anomaly detection

The core principle of anomaly detection is calculating the distance and building normal traffic profile as presented in the previous section. Distance describes how far a point compared to a centre of the known distribution. Our semi supervised anomaly detection scheme require 2 phases: Training phase and detection phase. The K-means clustering in training phase also included the algorithm to compute distance, which we used correlation distance. Clustering required to have a set certain amount of data before classifying. And the precision of the classification is not high enough to use it as a reliable and stable method of anomaly detection. So we need a method to detect each connection independently as it arrives. But the system need to be trained to know what behaviour is normal. That is why we need the training phase with normal data.

Phase 1 help to clean the data when it have more noise than it should be. The noise can be recognized as attack and normal connection which are outlier with other normal connection.

A. Training Phase

The purpose of training phase is building normal traffic profile from normal data pattern. It has following steps:

- Choose the variables, which affect normal profile most. Build the normal profile on selected features will reduce the number of variables needed to process. PCA is used to analyse the contribution of each variable to PCs of normal data.
- The network traffic needs to be free of attacks at training time in order to get a snapshot of captured network traffic for training dataset. In reality, this traffic can contain some small attacks considered as noise. Thus, we need to clean it beforehand. We propose using K-means clustering algorithm to remove outliers (noise) of the input data. We assume that the noise is much lesser than the normal data and recommend accepted noise level approximately at 10% of all incoming training data.
- Calculate the correlation matrix and eigenvector/eigen-value pairs.
- The cleaned training data needs to standardized (normalized) before computing PCs. We use 2 methods for this step.

Method 1: normalize training data with mean and standard deviation of the original variables (z-score).

$$z_k = \frac{x_k - \bar{x}_k}{\sigma_k} \quad (7)$$

Where $\bar{x}_k$ and σ_k is the mean and standard deviation of the variable x_k in trained data set respectively.

Method 2: normalize training data only with the mean of the original variables (zero-mean). This method reduces one division compared to the above method.

$$z_k = x_k - \bar{x}_k \quad (8)$$

In (7) and (8), z is the standardized (normalized) vector of training data set, and $z = (z_1, z_2, \dots, z_p)'$, where p is the number of the variable used in training.

- Compute the PC score of each sample in training data with z and eigenvector:

$$y_i = z e_i \quad (9)$$

- Sort the PCs by eigenvalues in descending order:
 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_p \geq 0$
- Compute distance for each observation of training dataset based on major and minor distances:

$$d(\text{major}) = \sum_{i=1}^q \frac{y_i^2}{\lambda_i}$$

$$d(\text{minor}) = \sum_{i=q+1}^p \frac{y_i^2}{\lambda_i} \quad (10)$$

- Build the empirical cumulative distribution function (ECDF) of distances. Choose the thresholds corresponding to estimated false positive ratio. ECDF is more fined grained with the above presented method 1.

• Apply K-means Algorithm

K-means is a clustering analysis algorithm that groups objects based on their feature values into K disjoint clusters. Objects classified into the same cluster have similar feature values. K is a positive integer number specifying the number of clusters, it has to be given in advance. Here are the four steps of the K-means clustering algorithm.

- Define the number of clusters K . and initialize K cluster centroids. This can be done by arbitrarily dividing all

objects into K clusters, computing their centroids, and verifying that all centroids are different from each other. The centroids can be initialized to various objects chosen arbitrarily.

- Iterate over all objects and compute the distances to the centroids of all clusters. Assign each object to the cluster with the nearest centroid.
- Recalculate the centroids of both modified clusters.
- Repeat step 2 until the centroids do not change any more.

The distance we used in K-mean algorithm is Pearson correlation distance [15]. Pearson Correlation measures the similarity in shape between two profiles. The formula for the Pearson Correlation distance is one minus the sample correlation between points (treated as sequences of values) as follows:

$$d = 1 - r \quad (11)$$

where: $r = z(u) \cdot z(v) / n$ is the dot product of the z -scores of the vectors u and v . The z -score of u is constructed by subtracting from u its mean and dividing by its standard deviation. Each centroid is the component-wise mean of the points in that cluster, after centering and normalizing those points to zero mean and unit standard deviation.

B. Detection Phase

In detection phase we use the sub score of Mahalanobis distance to detect each new point from the distribution of the trained data point.

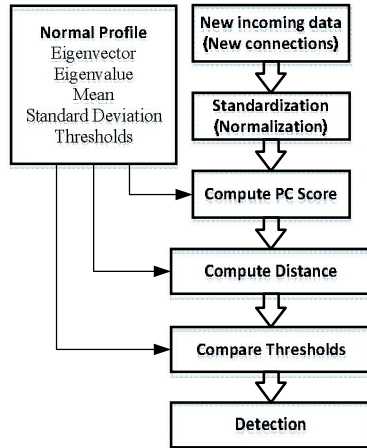


Figure 2. Detection phase of the model

This phase match each new observation arrives with established normal profile to detect anomaly. This include following steps.

- Standardize (normalize) data: In traditional way, the means and standard deviations have to be re-computed for each new data set, but the system need many of new observations before running detection engine. Our approach uses the standard deviation and the mean from trained data set. This way the distances can be computed for each connection individually when it arrives and the computational overhead is reduced.
- Compute PC score of each observation with trained eigenvectors which map observed data to eigenspace.

- Compute distances of each observation (equation 10). A new connection will have 2 distance values.
- If new connection's distance is greater than any of the established threshold, it marks as anomaly connection. Otherwise, it is normal connection.

C. Training Data Feedback

Figure 3 shows the enhancement of building normal profile. The idea is connections that pass negatively of attack should feedback to aid the training phase. At first, input training data can go through the signature-based module to filter normal data. The system has to detect known attacks in the data. Connection pass all the check as negative will be considered training data for clustering or data for PCA analysis.

Data after pass negatively of the anomaly detection module will go through validation checking. This step validates the correct of detection phase. Manual check can validate some random of the negatively connection for normal feedback. Other option is signature database can use again to automatic check for attacks in negative records. After checking, normal records use as feedback to the normal profile. This will help regulate the training data more efficiently.

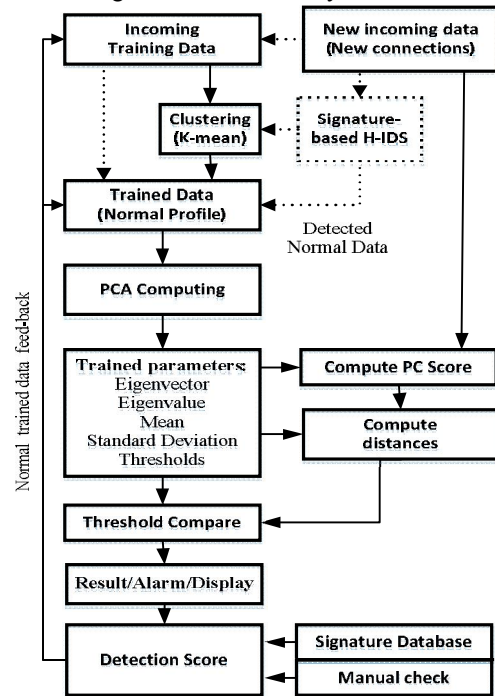


Figure 3. Enhancement with training data feedback

V. EXPERIMENT AND RESULTS

A. NSL-KDD Dataset

In [8], Shyu's test results use KDD CUP 99, the old data set contains 75% of redundant records. KDD CUP'99 is the mostly widely used data set for anomaly detection. The inherent problem of KDD dataset leads to new version of NSL KDD dataset that are mentioned in [14-17]. In [17], the authors conducted a statistical analysis on this data set and found two

important issues which highly affect the performance of evaluated system, and results in very poor evaluation of anomaly detection approaches. To solve these issues, they proposed a new data set, NSL-KDD, which consists of selected records of the complete KDD data set [14] and does not suffer from any of the mentioned shortcomings.

We don't use the KDDCUP99 but the NSL-KDD instead, since the advantages of NSL KDD dataset are as follows:

- No redundant records in the train set, so the classifier will not produce any biased result.
- No duplicated records in the test set which have better reduction rates.
- The number of selected records from each difficult level group is inversely proportional to the percentage of records in the original KDD data set.

The NSL-KDD data includes 41 features and 5 classes that are normal and 4 categories of attacks: Denial of Service Attack (DoS), Probe, Remote to Local Attack (R2L), and User to Root Attack (U2R). Attack categories and types in NSL-KDD data set were given in [14].

B. Performance Measures

We use the following performance measures:

- True Positive (TP): the event when an attack connection correctly detected.
- True Negative (TN): the event when a normal connection correctly detected.
- False Positive (FP): the event when a normal connection detected falsely as attack connection.
- False Negative (FN): the event when an attack connection detected falsely as normal connection.
- Precision: The ratio of true positive and the number of detected connection as attack.
- Recall: The ratio of true positive and the number of real attack in the sample data set.
- False Positive Ratio (FPR): The ratio of false positive and the real number of normal connection in sample data set.
- Total Accuracy: The overall successful prediction of both attack and normal connection.

C. Experiment Results

We implemented experiments in Matlab R2013a. We use the same subset of the KDDTrain+ dataset [14] for both training and detection phase.

TABLE 1. FEATURE USED IN EXPERIMENT

Features	Meaning
Protocol_type	Protocol types (tcp, udp, ...)
src_bytes	Number of bytes from source
count	Number of connections to the same host as the current connection in the past two seconds
diff_srv_rate	% of connections to different services
dst_host_same_srv_rate	% of connections to the same service for destination host
dst_host_serror_rate	% of connections that have SYN errors for destination host

In [8], Shyu used 34 features of KDDCUP99'. We focus on other approach that use features affected normal connections. Through experiments we found 6 features in NSL-KDD are significantly affects the normal profile (Table 1). This will make the processing data faster.

The K-means clustering algorithm was used with input training data to derive a more cleaned data. The goal of clustering is to remove attack points that not follow the baseline of major normal data. Attacks should be much lesser than normal traffic, otherwise malicious connection will dominate the normal traffic. We recommend malicious volume below 10% of the total incoming training data. In daily condition, many networks have that upper bound limit. Our first experiments use 7000 connections for input data, which are chosen randomly from KDDTrain+ data and include above 900 attack connections. The distance used of K-means algorithm is correlation distance. We use the number of clustering $K=2$ for K-means. The step of clustering input training data are:

- Choose target cluster for training set.
- Run K-means several times until target cluster have adequate data point. With 7000 records, the training data should have above 5000 records.

Table 2 depicted the number of attack connection before and after clustering input data. The result showed that clustering the input data significantly reduced the noise which includes attacks in the background traffic.

TABLE 2. REDUCED NOISE OF INPUT DATA

Test	Number of attacks before clustering	Number of attacks after clustering	Attack Reduced Ratio (%)
C1	928	152	83.6
C2	923	148	84
C3	937	157	83.2
C4	935	164	82.5
C5	913	142	84.4

After clustering, the targeted cluster becomes the data for training. The system will derive the training parameters needed. Through experiments we found that it is better to keep major PCs $W_1 = 3$ and minor PCs $W_2=3$. Major PCs affect the attack detection rate while minor PCs tend to affect normal detection rate. Increase or decrease PCs more or less than 3 will make the total accuracy decrease.

For measuring the test accuracy, we use random 50,000 connections in KDDTrain+ data. The results are shown in table 3 and 4. Table 3 showed 5 test result using training of method 1. In Table 4 is the result with method 2 using the same data.

TABLE 3. RESULT OF METHOD 1

Test	Precision (%)	Recall (%)	FPR (%)	Total Accuracy (%)
Test1	94.7	82	5.2	88
Test2	90.6	88.9	8.2	90.5
Test3	92.2	83.2	7.5	87.7
Test4	91.4	84.8	8	88.4
Test5	94.7	81.9	5.3	87.8

The result is good for both method 1 and method 2. Method 1 uses z-score data, so the thresholds of tests are not changed too much. Method 2 uses zero-mean data, it has to choose threshold for each new test, which often fluctuates with new input data. These can make the system have to change thresholds frequently.

TABLE 4. RESULT OF METHOD 2

Test	Precision (%)	Recall (%)	FPR (%)	Total Accuracy (%)
Test1	87.2	92.2	10.5	90.7
Test2	87.4	90.8	10.5	90
Test3	87	92.8	10.7	90.8
Test4	87	92.2	10.8	90.5
Test5	87	93	10.6	91

D. Small Training Dataset

Training often required a large training dataset. By using 6 chosen features, we demonstrate that even a small set of training data can have good result. We use both method 1 (M1) and 2 (M2) for testing. Table 5 showed these test results, where Test 1 and 2 use 1000 normal records, Test 3 uses only 500 normal records. As showed in the table 5, it is more difficult for method 2 to choose thresholds because the data is not fine grained due to the small input data. In case of small training dataset, we recommend use method 1.

TABLE 5. RESULT OF SMALL TRAINING DATASET

Test	Precision (%)	Recall (%)	FPR (%)	Total Accuracy (%)
Test1 / M1	87.3	91	10.5	90.2
Test1 / M2	89.8	87	9	89
Test2 / M1	82.3	94	13.7	89.4
Test2 / M2	85.3	87.3	10.3	88.6
Test3 / M1	86.6	92.7	10.9	90.6
Test3 / M2	82.7	88.72	14.5	87

VI. CONCLUSION

Principal component analysis (PCA) is an effective approach for anomaly detection. Previous works showed several issues with PCA implementation including the applied model, problem of noise data, complexity of dataset, selection of principal components. In this paper, we propose a modified Mahalanobis distance based on PCA and the different weights for network traffic anomaly detection. Using PCA with Mahalanobis distance and weights, we concentrated on building normal traffic profile. Through experiments we also showed that 6 features of NSL-KDD dataset are efficient with the normal profile. We propose a K-means clustering algorithm

to reduce noise with input training data. The experiments showed that even with small training dataset (less than 1000 points), our approach has good performance including detection accuracy. The training data feedback will be an enhancement to the normal profile of the proposed model for a combination of the anomaly detection module with a signature-based IDS.

REFERENCES

- [1] S. Axelsson, *Intrusion Detection Systems: A Survey and Taxonomy*, Chalmers University of Technology, Goteborg Sweden, 2000. <http://www.cs.chalmers.se/~sax/pub/>
- [2] S.S. Rajan, V.K. Cherukur, *An Overview of Intrusion Detection Systems*, 2010. <http://www.idt.mdh.se/>
- [3] V. Jyothana, V. V. R. Prasad, K. M. Prasad, *A Review of Anomaly based Intrusion Detection Systems*, Interl. Journal of Computer Applications, Vol. 28–No.7, pp. 28-34, August 2011.
- [4] A. Jain, B.Verma, J. L. Rana, *Anomaly Intrusion Detection Techniques: A Brief Review*, Interl. Journal of Scientific & Engineering Research, Vol. 5, Iss. 7, pp.1372-1383, July 2014.
- [5] S. Myers, J. Musacchio, N. Bao, *Intrusion Detection Systems: A Feature and Capability Analysis*. Tech.Report UCSC-SOE-10-12. Jack Baskin School of Engineering, 2010.
- [6] C. Kacha, K. A. Shevade, *Comparison of Different Intrusion Detection and Prevention Systems*, Intl. Journal of Emerging Technology and Advanced Engineering, Vol.2, Iss.12, pp.243-245, Dec.2012.
- [7] M.H.Bhuyan, D.K.Bhattacharyya, J.K.Kalita, *Survey on Incremental Approaches for Network Anomaly Detection*, Journal of Communication Networks and Information Security (IJCNIS), Vol.3, No.3, pp. 226-239, Dec.2011.
- [8] M.L. Shyu, S.C.Chen, K.Sarinnapakorn, L.W.Chang, *A Novel Anomaly Detection Scheme Based on Principle Component Classifier*, Proc. of the IEEE foundation and New Directions of Data Mining Workshop in 3rd IEEE Intl. Conference on Data Mining (ICDM03), pp. 172-179, 2003.
- [9] D. Brauckhoff, K. Salamatian, and M. May: *Applying PCA for Traffic Anomaly Detection: Problems and Solutions*, IEEE INFOCOM, pp.2866-2870, April, 2009.
- [10] H.Huang, H.AI-Azzawi, H.Brani, *Network Traffic Anomaly Detection*, ArXiv:1402.0856v1.
- [11] C.A.P.Boyce, A.N.Zincir-Heywood, *A Comparison of Four Intrusion Detection Systems for Secure E-Business*, 2003.
- [12] V.J.Hodge, J.Austin, *A survey of outlier detection methodologies*, 2004
- [13] The KDD Archive, KDD99 cup dataset, 1999, <http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html>
- [14] The NSL-KDD Data set: <http://nsl.cs.unb.ca/NSL-KDD>, 2009.
- [15] M.K. Siddiqui, S.Naahid, *Analysis of KDD CUP 99 Dataset using Clustering based Data Mining*, 2013.
- [16] NSL-KDD data set for network-based intrusion detection systems, <http://nsl.cs.unb.ca/KDD/NSLKDD.html>, March 2009.
- [17] M. Tavallaei, E. Bagheri, W. Lu, A.A. Ghorbani, *A Detailed Analysis of the KDD CUP 99 Data Set*, In the Proc. of the IEEE Symposium on Computational Intelligence in Security and Defense Applications (CISDA 2009), pp. 1-6, 2009.

Nguyen Ha Duong, BSc (2001) and MSc. (2003) at TU Hanoi, Vietnam. Lecturer at Faculty of Information Technology, National University of Civil Engineering, Vietnam. Research interests: communication network, network security, telecommunication.

Hoang Dang Hai, Prof. Dr.Ing-habil., PhD. (1999), DSc. (2003) at TU Ilmenau, Germany. Institution: Ministry of Information and Communication of Vietnam, invited Professor at Post and Telecommunication Institute.