A Hybrid System for Reducing Memory and Time Overhead of Intrusion Detection System

Zhi-Guo Chen, Sung-Ryul Kim

A Hybrid System for Reducing Memory and Time Overhead of Intrusion Detection System

Zhi-Guo Chen and Sung-Ryul Kim

Division of Internet and Multimedia Engineering
Konkuk University ,Seoul, Rep. of Korea
chenzhiguo520@gmail.com,kimsr@konkuk.ac.kr
http://www.konkuk.ac.kr

Abstract. With the growing use of the internet worldwide, internet security becomes more and more important. There are many techniques available for intrusion detection. However, there remain various issues to be improved, such as detection rate, false positive rate, memory overhead, time overhead, and so on. In this paper, a new hybrid system for network intrusion detection system using principal component analysis and C4.5 is presented, which has a good detection rate and keeps false positive and false negative rate at an acceptable level for different types of network attacks. Especially, this system can effectively reduce the memory overhead and the time overhead of building the intrusion detection model. These claims are verified by experimental results on the KDD Cup 99 benchmark network intrusion detection dataset.

Keywords: Intrusion Detection System, Principal Component Analysis Algorithm, C4.5 Algorithm, Time Overhead and Memory Overhead.

1 Introduction

With the rapid development and application of computing and communication technologies, more and more people solve problems or handle things with the internet, using such things as email, internet banking, video conference, save personal information and so on. Therefore, internet security becomes one of the key problems in the world. Traditionally, firewall is a widely used security measure but the firewall alone does not provide enough security. The protection of computer systems, network systems and the securities of information infrastructure usually depend on a kind of intrusion detection technology also [1].

So far, many techniques for intrusion detection have been proposed and intrusion detection systems [2] by these techniques are broadly classified into two categories: misuse-based IDS and anomaly-based IDS [3]. Misuse-based IDS is based on signatures for known attacks. If unknown attacks or known attacks which do not match any signatures appear, then these attacks may not be detected with misuse-based IDS. So, in order to detect new attacks, IDS need to revise the set of signatures frequently. Anomaly-based IDS is different from misuse-based IDS, Anomaly-based IDS is able to detect known and unknown

attacks by building profiles of normal behaviors. If a new behavior is far from normal behaviors of the profiles, then the behavior will be treated as an attack. Anomaly based intrusion detection using data mining algorithms such as nave Bayesian classifier (NB), decision tree (DT), neural network (NN), k-nearest neighbors (KNN), support vector machine (SVM), and genetic algorithm have been widely used by researchers to improve the performance of IDS [4]. However, Anomaly detection suffers from low detection and high false positive rates and it incurs large memory and time overhead for the detection model.

In this paper, we present a new hybrid system for network intrusion detection using principal component analysis and C4.5 algorithm [5] which has good detection rate and keeps false positive and false negative rates at an acceptable level for different types of network attacks while having the benefit of lower the memory and the time overhead for building the detection model. PCA (principal component analysis) [6, 7] is used to reduce the dimension of original high dimensional data and remove noise effectively. Then we use new dataset which handled by PCA to make intrusion detection model by C4.5 algorithm. The experimental results show that the proposed method has acceptable detection rates (DR), false positive rate(FP), false negative rate(FN) and accuracy.

The rest of this paper is organized as follow: in section 2, we will introduce principal component analysis and C4.5 algorithm. Section 3, describes our proposed method. In section 4, we introduce experiment and result, performances estimation of IDS, experimental dataset and experimental analysis. Finally, in section 5, we conclude this paper.

2 Algorithm

2.1 PCA algorithm

The performance of the intrusion detection model depends on the quality of dataset. So noise in the dataset is one of the challenges in data mining. The reason for dealing with noisy data is that it will avoid over-fitting the dataset. The irrelevant and redundant attributes of dataset may lead to complex classification model and reduce the classification accuracy. So we need to reduce noises, irrelevant and redundant attributes firstly. Principal component analysis algorithm have been widely used in dimension reduction; it can be effective to deal with complex data that have high dimensions. It can identify the main elements and structure of data, remove noise and redundant attributes, express the data with simple format.

The following procedure describes the PCA algorithm we use to deal with dataset S $(D \times N)$ where N is the number of data example and D is the number of dimension of original dataset S. Each column represents one data that have

dimensionality$D$.

$$S = \begin{pmatrix} S_{11} & \cdots & S_{m1} & \cdots & S_{N1} \\ S_{12} & \cdots & S_{m2} & \cdots & S_{N2} \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ S_{1D} & \cdots & S_{mD} & \cdots & S_{ND} \end{pmatrix}$$

1. Map discrete attributes to continuous attributes;
2. Calculate the mean value of all data examples $\bar{S}$;
3. Subtract mean vector for each data example;
4. Find covariance matrix $\sum$ of dataset S, and then calculate all eigenvectors and eigenvalues of $\sum$;
5. According the eigenvalues to select the dimension number d of biggest eigenvectors to make the new dataset. The eigenvectors are represented as $u_1, u_2, ..., u_d$ and then find its matrix transpose U;
6. Get new dataset $D = (US')^T(N \times d)$, where N is the number of the data examples and d is the dimension of the new dataset D. Add the original datasets class identity in new Dataset D and we have completed the transformed dataset D and with every record included with d attributes and one class identity.

2.2 C4.5 Algorithm

C4.5 algorithm has been widely used in classification and decision making. C4.5 algorithm is a later version of the ID3 algorithm [8] proposed by R. Quinlan [9]. The majority of this algorithm uses a descendent strategy from the root to the leaves. To ensure this procedure, the following generic parameters are required. The gain ratio (attribute selection measure) taking into account the discriminative power of each attribute over classes in order to choose the best one as the root of the decision tree. C4.5 decision tree divides data items into subsets, based on the attributes. It finds the one attribute that maximizes gain ratio and divides the data using that attribute.

Let us assume that D is a dataset and that A_i is one of attribute of the dataset. Let us also assume that the dataset has a set of classes $C_1, C_2, ..., C_m$. The information gain is calculated as the follows:

$$Gain(D, A_i) = Entropy(D) - \sum_{i \in Values(A_i)} \frac{|D_i|}{|D|} Entropy(D_i) \tag{1}$$

where,

$$Entropy(D) = -\sum_{i=1}^{m} \frac{freq(C_i, D)}{|D|} log_2 \frac{freq(C_i, D)}{|D|} \tag{2}$$

where $freq(C_i, D)$ denotes the number of examples in the dataset$D$ belonging to class C_i and D_i is the subset of dataset D divided by the attribute $A_i's$ value

a_m. Then $SplitInfo(A_i)$ is defined as the information content of the attribute A_i itself [12]:

$$SplitInfo(D, A_i) = -\sum_{i \in Values(A_i)} \frac{|D_i|}{|D|} log_2 \frac{|D_i|}{|D|} \tag{3}$$

The gain ratio is the information gain calibrated by Split Info:

$$GainRatio(D, A_i) = \frac{Gain(D, A_i)}{SplitInfo(A_i)} \tag{4}$$

1. If A_i is a discrete attribute [10] then divide the dataset by discrete values of $A_i(a_1, a_2, ..., a_m)$.

2. If A_i is a continuous attribute then use information gain to efficiently find the best split point and disperse the dataset by this best split point [11]. If attribute A_i has values $(A_{i1}, A_{i2}, , A_{im}, , A_{ih})$ then firstly sort the continuous values (called candidate cut point) from small to large and then divide dataset with these candidate cut points with $> A_{im}$ and $\leqslant A_{im} (i \leqslant m < h)$. Then we calculate the information gain for each possible cut point. The cut point for which the information gain is maximized amongst all the candidate cut points is taken as the best split point. We divide the current dataset (subset dataset) according this point. Now we can build the tree by dividing the current dataset (subset dataset) by the selected attribute (discrete attributes according the attribute values, continuous attributes according to best split point).

3 Our proposed method

3.1 Using PCA Algorithm

To determine different types parameters of network attacks. Firstly, we need randomly select data in file "corrected" which is downloaded from KDD Cup 99 site to make training dataset and test dataset. The dataset is $S(N \times D)$, where N is the number of the data records and D is the dimension of the KDD data (41 dimension except for class identity). And then we use the PCA algorithm to reduce the dimension of the KDD data to get new Dataset $D(N \times d)$. Where N is the number of the new dataset and d is the dimension (the number of attributes) of new dataset D. We then add the original datasets class identity in the new Dataset D. After reducing the dimension by PCA we use the C4.5 algorithms to build the intrusion detection model.

3.2 Making Detection Model

After reducing the dimension by PCA, we get new dataset D where all of the attributes are continuous ones. So we should use the continuous attributes handling method of C4.5 algorithm that has been described earlier to make tree structure.

Dataset D contains attributes $A_1, A_2, ..., A_d$ and each attribute A_i contains the following continuous attribute values $(A_{i1}, A_{i2}, ..., A_{im}, ..., A_{ih})$. The training dataset also has a set of Class $C_1, C_2, ..., C_m$. Each data in the training data D have particular class C_j. We follow the next procedure.

1. Find the best split point of every attributes and calculate gain ratio in training dataset D with C4.5 continuous attribute handling method. Compare the gain ratio of every attribute. Select A_i among the attributes $A_1, A_2, ..., A_d$ which have maximize gain ratio to make the root attribute node, and then divide the training dataset D into subsets (D_1, D_2) depending on the best split point A_{im}.

2. Find the best split point of every attribute and calculate gain ratio in subset dataset D_i. Compare the gain ratio of every attribute. Select A_i among the attributes $A_1, A_2, ..., A_d$ which have maximize gain ratio to make the attribute node, Then divide the subset dataset D_i into subsets (D_{i1}, D_{i2}) depending on the best cut point A_{im}.

3. Continue this process until subset dataset's entropy is zero or all attributes have same maximizes gain ratio in the subset dataset D_s. If subset dataset's entropy is zero and we know that all of the Class C_i in subset dataset is same. In this case, a leaf node will be set up. And if all attribute's gain ratio is same, We use $freq(C_1, D_s), freq(C_2, D_s), ..., freq(C_m, D_s)$ to determine the class in dataset D_s.

When we test an example data X, we should use PCA algorithm to deal with this example. And then we use the tree structure to find the Class (leaf node).

Pseudo code: Since after reducing the dimension by PCA, we get new dataset D where all of the attributes are continuous ones. So we just need use the continuous attributes handling method of C4.5 algorithm to build tree:

```
C4.5 (Dataset D, Attributes Sets A, Attributes values)
 Create a root node for the tree by Maximum (Gain Ratio (D,A))
  If Entropy of dataset is 0, we are sure that the Class of this
  dataset(subset dataset) is same. In this case we stop dividing
  the dataset
  If all attribute have same maximum gain ratio in the subset
  dataset Ds. We do not know which attribute we can select to
  divide dataset. Use freq(C1,Ds),freq(C2,Ds),...,freq(Cm,Ds)
  to determine the classes in dataset Ds.
  Otherwise Begin
     Ai -- The attribute that best classifies examples(select by
     Maximum Gain Ratio (D,A)).
     Decision Tree attribute for node = Ai
     For best split point Aim of Ai (best split point find by C4.5
     continuous attributes handling method)
        Add a new tree branch below node according > Aim and <= Aim
        Let dataset (> Aim) be the subset of Dataset that have the
        values > Aim for Ai
```

```
        If all attribute have same maximum gain ratio in the subset
        dataset (> Aim)
            Then below this new branch add a leaf node with label
            = most common type in the subset dataset
        If Entropy of subset dataset (> Aim) == 0
            Then all with the same value of the categorical att-
            ribute, return a leaf
        Else below this new branch add the subtree C4.5 (subset
        dataset (> Aim), Attributes Set A, Attributes values)
End
Return Root
```

4 Experiment and Result

4.1 Performances Estimation

Accuracy, detection rate (DR), false positive rate (FP) and false negative rate (FN) are basic parameters that are used for performance estimation [4] of intrusion detection models. We can accord the following simple confusion matrix for calculating the parameters.

Table 1. Confusion Matrix

	Predicted Class Positive	**Predicted Class Negative**
Actual Class Positive	a	b
Actual Class Negative	c	d

In the Intrusion Detection System(IDS), a detection rate(DR) is an instance which is normal is classified as normal or attack is classified as attack. False positive(FP) means no attack but IDS detect as attack and false negative(FN) means attack occurs but IDS detect as normal. Accuracy is also an important parameter that determines the percentage of correctly classified instance. Form the confusion matrix; we can get these parameters by following calculations.

$$DetectionRate = a/(a+b) \quad and \quad d/(d+c)$$
$$FP = b/(a+b) \quad FN = c/(c+d)$$
$$Accuracy = (a+d)/(a+b+c+d)$$

4.2 Dataset

The KDD Cup 1999 dataset [12] is used in this experiment, which is network connection data collected from the U.S Air Force LAN in nine weeks. KDD-CUP99 dataset have a set of inherent constructed features. Each record has 41 characteristic attributes and one more attribute assigns the record type. And 34 attributes are continuous data types, 7 attributes are discrete data types in the

characteristic attributes. The list of the input attributes in KDD99 dataset for each network connections is shown below.

Table 2. Input attributes in KDD99 Dataset

No	Input Attribute	Type	No	Input Attribute	Type
1	Duration	*Con.*	22	is_guest_login	*Dis.*
2	protocol_type	*Dis.*	23	Count	*Con.*
3	Service	*Dis.*	24	srv_count	*Con.*
4	Flag	*Dis.*	25	serror_rate	*Con.*
5	src_bytes	*Con.*	26	srv_serror_rate	*Con.*
6	dst_bytes	*Con.*	27	rerror_rate	*Con.*
7	Land	*Dis.*	28	srv_rerror_rate	*Con.*
8	wrong_fragment	*Con.*	29	same_srv_rate	*Con.*
9	Urgent	*Con.*	30	diff_srv_rate	*Con.*
10	Hot	*Con.*	31	srv_diff_host_rate	*Con.*
11	num_failed_logins	*Con.*	32	dst_host_count	*Con.*
12	logged_in	*Dis.*	33	dst_host_srv_count	*Con.*
13	num_compromised	*Con.*	34	dst_host_same_srv_rate	*Con.*
14	root_shell	*Con.*	35	dst_host_diff_srv_rate	*Con.*
15	su_attempted	*Con.*	36	dst_host_same_src_port_rate	*Con.*
16	num_root	*Con.*	37	dst_host_srv_diff_host_rate	*Con.*
17	num_file_creations	*Con.*	38	dst_host_serror_rate	*Con.*
18	num_shells	*Con.*	39	dst_host_serror_rate	*Con.*
19	num_access_files	*Con.*	40	dst_host_rerror_rate	*Con.*
20	num_outbound_cmds	*Con.*	41	dst_host_srv_rerror_rate	*Con.*
21	is_host_login	*Dis.*			

The data contains attack types (marked at last attribute) that can be classified into four main categories as follows:

1. Denial of Service (DOS): Denial of Service (DOS) attack makes some computing or memory resources too busy or too full to handle legitimate requests, or denies legitimate users access to a machine.

2. Remote to User (R2L): Remote to User (R2L) is an attack that a remote user gains access of a local user account by sending packets to a machine over a network communication.

3. User to Root (U2R): User to Root (U2R) is an attack that an intruder begins with the access of a normal user account and then becomes a root-user by exploiting various vulnerabilities of the system.

4. Probing: Probing (Probe) is an attack that scans a network to gather infor-mation or find known vulnerabilities. An intruder with a map of machines and services that are available on a network can use the information to look for exploits.

So, all the classes in KDD99 dataset can be categorized into five main classes and four attacks are recorded into different attacks which are shown in Table 3.

Table 3. Different Data Types in KDD99 Dataset

Data Type	Mark
normal	normal
Denial of Service (DOS)	apache2, mailbomb, processtable, udpstorm, back, land, neptune, pod, smurf, teardrop
Remote to User (R2L)	ftp_write, guess_passwd, multihop, phf, spy, warezclient, warezmaster, imap, named, sendmail, snmpgetattack, snmpguess, worm, xlock, xsnoop
User to Root (U2R)	buffer_overflow, loadmodule, perl, rootkit, httptunnel, ps, sqlattack, xterm
Probing	Satan, ipsweep, nmap, portsweep, mscan, saint

4.3 Experimental Analysis

The number of training and test data is shown in the following:

Table 4. Number of training and test examples

Types	Training data	Test data
Normal	5869	5781
Probing	429	384
DOS	22086	22297
U2R	20	21
R2L	1596	1517

In order to evaluate the performance of the proposed algorithm for different types of network attacks, we performed 5-Class classification using KDD99 dataset. We compared our method with C4.5 algorithm.

Table 5. Confusion matrix of our method

a	b	c	d	e	classified as
5469	9	12	3	288	a = normal
13	357				b = Probing
7		22274			c = DOS
9			10		d = U2R
283				1226	e = R2L

Table 6. Confusion matrix of C4.5 method

a	b	c	d	e	classified as
5491	7	5	9	269	a = normal
1	378				b = Probing
1		22285			c = DOS
5			11		d = U2R
299				1217	e = R2L

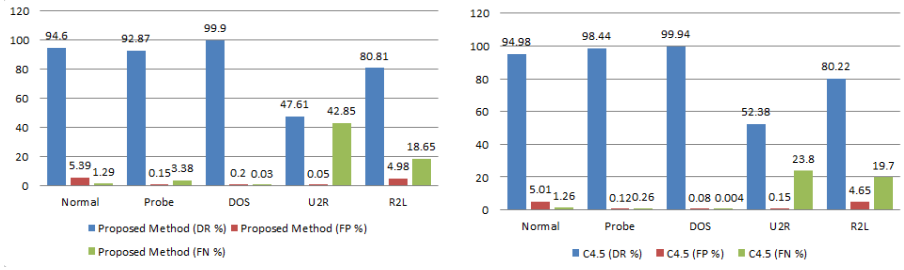

Fig. 1. Performance of our method and C4.5 algorithm

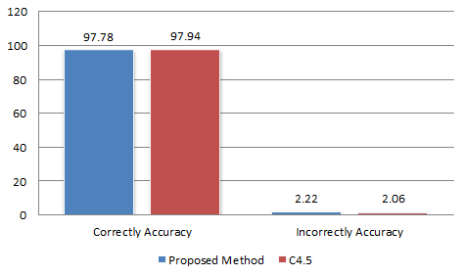

Fig. 2. Accuracy of two methods

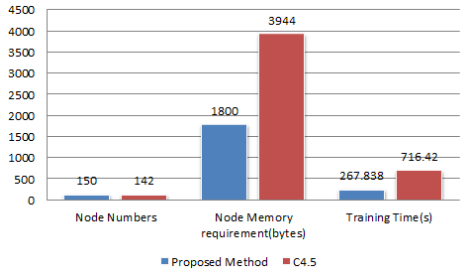

Fig. 3. Node numbers, node memory requirement and training time

Our experimental results show that detection rates (DR) and accuracy of our proposed method and C4.5 algorithm almost identical. But from Figure 3, we can see that our method is effective to reduce the memory overhead and the time overhead of building the intrusion detection model.

There are not many examples of U2R, R2L and Probing to build the intrusion detection model. Thus, the C4.5 method and our method have lower detection rates for these types of network intrusions. For similar reasons, even if one example is not correctly detected the resulting rate will show considerable differences. Therefore, the detection rate of Probe and false negative of U2R have considerable difference between our proposed method and C4.5 method.

From figure 3, we can see that the number of nodes in our proposed method is almost similar to C4.5 method, but one node in our method occupies just 12bytes and it can be stored as $[x, y]$ (x is the selected attribute written as *int* format, y is best split value that can be written as *double* format). If we use C4.5 algorithm to deal with the original dataset S. For continuous attribute, we also can store the node as $[x, y]$, but for discrete attribute, if it have lots of attribute values, we need to store a node as $[x, a_1, a_2, ..., a_i, ..., a_m]$ (x is the selected attribute written as *int* format, a_i is discrete attribute value that can be written as *int* format). In this case, the discrete attribute values requires more memory. So, we can conclude that our proposed method can effectively reduce the memory overhead of C4.5 method.

Because of we use PCA algorithm to reduce the dimension of the data, our proposed method just need 267.838 seconds to build the intrusion detection model. Therefore, after comparing the training time of two methods we can conclude that PCA algorithm can be effective to reduce the time overhead of building Intrusion Detection Model.

Overall consideration accuracy, detection rates, false positive rate, false negative rate, memory overhead and time overhead, we can conclude that our method has good performance for intrusion detection system and that our proposed method is effective to improve the performances of Anomaly-based intrusion detection system.

5 Conclusion

This paper introduced a new hybrid system for network intrusion detection using principal component analysis and C4.5 algorithm. The results show that it has acceptable detection rate, false positive and false negative to detect the 5 types of the KDD dataset. It also can be effective to reduce the memory overhead and the time overhead of building the intrusion detection model. But the false negative and false positive rates for R2L are high. The reason is there are not many examples of R2L to build detection model and the intrusion detection model has misclassified the data as normal. So the future work will focus on avoiding misclassified R2L, reducing the false positive and false negative rates of R2L and improving the detection rate of normal and R2L.

Acknowledgments. This research was supported by Next-Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT & Future Planning (2011-0029924).

References

1. Hui Lu, Jinhua Xu. Three-level Hybrid Intrusion Detection System.
2. http://www.sans.org/reading-room/whitepapers/detection/understanding-intrusion-detection-systems-337.
3. DOUGLAS J. BROWN, BILL SUCKOW, and TIANQIU WANG: A Survey of Intrusion Detection Systems.
4. Thuzar Hlaing. Feature Selection and Fuzzy Decision Tree for Network Intrusion Detection. International Journal of Informatics and Communication Technology (IJ-ICT) Vol.1, No.2, December 2012, pp. 109 118 ISSN: 2252-8776
5. N. Ben Amor, S. Benferhat, and Z. Elouedi. Naive bayes vs decision trees in intrusion detection systems. In ACM symp osium on Applied computing (SAC2004), pages 420424, Nic-osia, Cyprus, 2004.
6. Lindsay I. Smith: A tutorial on Principal Components Analysis New York (2002)
7. Lu Zhao, Ho-Seok Kang, Sung-Ryul Kim: Improved Clustering for Intrusion Detection by Principal Component Analysis with Effective Noise Reduction. Information and Communication Technology Lecture Notes in Computer Volume 7804, 2013, pp 490-495.
8. J. R. Quinlan: Induction of Decision Trees. Machine Learning, 1:81-106, 1986
9. J.R.Quinlan: Improved Use of Continuous Attributes in C4.5. Journal of Artifcial Intelligence Research 4 (1996) 77-90 Submitted 10/95; published 3/96
10. Yogendra Kumar Jain, Upendra: An Efficient Intrusion Detection Based on Decision Tree Classifier Using Feature Reduction. International Journal of Scientific and Research Publications, Volume 2, Issue 1, January 2012 ISSN 2250-3153
11. Salvatore Ruggieri: Efficient C4.5. IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL.14, N0.2, MARCH/APRIL 2002
12. The third international knowledge discovery and data mining tools competition dataset KDD99-Cup. http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html (1999)