Buletin Teknik Elektro dan Informatika
(Bulletin of Electrical Engineering and Informatics)
Vol. 2, No. 3, September 2013, pp. 225~232
ISSN: 2089-3191

Naïve Bayes Decision Tree Hybrid Approach for Intrusion Detection System

Bekti Maryuni Susanto
Information Management Study Program, AMIK BSI Yogyakarta, Indonesia
Jalan Ring Road Barat Ambarketawang Gamping Sleman 55294, Telp. (0274)4342536
e-mail: bekti.bms@bsi.ac.id

Abstract

Internet is also increasing exponentially increasing intrusion or attacks by crackers exploit vulnerabilities in Internet protocols, operating systems and software applications. Intrusion or attacks against computer networks, especially the Internet has increased from year to year. Intrusion detection system is main stream in the information security. The main purpose of intrusion detection system is a computer system to help deal with the attack. This study presents a hybrid approach to decision tree algorithm and naïve Bayes to detect computer network intrusions. Performance is measured based on the level of accuracy, sensitivity, precision and spesificity. Dataset used in this study is a dataset KDD 99 intrusion detection system. Dataset is composed of two training data and testing data. The selection of attributes is done using the chi-square, selected the top ten attributes based on the calculation of chi-square. From the experimental results obtained by the accuracy of naïve Bayes decision tree algorithm was 99.82%.

Keywords: *NBTree, machine learning, intrusion detection*

1. Introduction

Internet is also increasing exponentially increasing intrusion or attacks by crackers exploit vulnerabilities in Internet protocols, operating systems and software applications. Intrusion or attacks against computer networks, especially the Internet has increased from year to year. Based on the report from Kaspersky Lab amount of attacks through the internet browser number 23,680,646 in 2007, increased to 73,619,767 in 2009 and increased again to 580 371 937 in 2010. Internet browsers become a major tool in spreading malicious programs among the majority of computer users in 2010. Algorithm Kaspersky Security Network (KSN) is only able to detect web attacks by 60% [1].

Intrusion detection is the process of monitoring the events on a computer system or network and analyzing them for incidents that might give an indication, which is an offense or a breach of computer security policies, policies that use approved or standard security practices. Intrusion prevention is the process to show the intrusion detection and attempting to stop detected possible incidents. Intrusion detection and prevention systems is a major concern in identifying the incident, log the information, trying to stop and report to the security administrator. In addition to using the organization's intrusion detection and prevention systems (IDPS) for other purposes, such as identifying problems of security policies, documenting existing treatments, and inhibits individual in violation of security policy. IDPS become a necessary addition to the security infrastructure for each organization [2].

Intrusion detection system is main stream in the information security. The main purpose of intrusion detection system is a computer system to help deal with the attack. There are two types of intrusion detection system based on the type of surgery that is used to detect disorders, anomaly detection and misuse detection system. Anomaly detection systems create a database of normal behavior and it deviation of normal behavior that occurs; a warning is triggered by a disturbance. Misuse detection systems store attack patterns that have been previously defined in a database if a similar situation occurred and the data is classified as an attack. Based IDS data sources are classified into host-based IDS and network-based. Netwok based IDS analyze individual packets through the network. Host-based IDS analyzes the activity on a single computer or a host [3].

Received March 20, 2013; Revised June 15, 2013; Accepted June 30, 2013

Most of the current IDS use rule-based systems or expert. Strength is highly dependent on the ability of security personnel to develop IDS. Formerly, IDS can only detect known types of attacks and is now likely to generate false positive alarms. This led to the use of Intelligence technique known as data mining or machine learning as an alternative to costly human ability and weight. This technique automatically learn the data or extract useful patterns from the data as a reference profile of normal behavior or an attack on the existing data for further classification of network traffic [4]. Machine learning is a field of study that provide computers with the capability of learning from previous experience. Machine learning based on a statistical analysis of the data is very large and some algorithms can use the patterns found in the data prior to making decisions on new data [5].

A wide range of machine learning algorithms is used in intrusion detection system (IDS). Machine learning algorithm accuracy rate in detecting intrusion are shown in Table 1.

Table 1. Accuracy of Machine Learning Algorithm in Intrusion Detection [5]

Algorithm	Accuracy
J48	93,82%
Naïve Bayes	81,66%
NBTree	93,51%
Random Forest	92,79%
Random Tree	92,53%
Multilayer Perceptron	92,26%
SVM	65,01%

Table 2. Confusion Matrix

		Predicted class	
		True	False
Actual class	True	True positive (TP)	False positive (FP)
	False	False negative (FN)	True negative (TN)

This study presents a hybrid approach to decision tree and naïve Bayes algorithm to detect computer network intrusions. Performance is measured based on the level of accuracy, sensitivity, precision and spesificity. Dataset used in this study is a dataset KDD 99 intrusion detection system. Dataset is composed of two training data and testing data.

2. Algorithm

C4.5 algorithm called decision tree because the algorithm produces a decision tree. C4.5 algorithm is a rule-based algorithm is obtained indirectly, because the rule is obtained from the decision tree generated by C4.5 algorithm. C4.5 algorithm has two types of nodes, internal nodes and leaf nodes. Internal node associated with tests for samples on the attributes of individuals or groups and assigns the class label leaf nodes based on the sample distribution of the class record. C4.5 algorithms classify samples with a top-down manner, starting from the root node and keep the movement in accordance with the results of tests on the internal node, until the leaf node reached and assigned class labels [6].

C4.5 decision tree construction is based on the separation of internal nodes recursively. The selection of attributes that separated the internal nodes is very important during the construction process and determines the final structure of the wide range of decision trees. Many attempts have been made on this aspect and the range of separation criteria, such as the Gini index, information gain and chi square test. Entropy theory is adopted to select the appropriate attribute by separation algorithm C4.5. Suppose N is the size of the dataset D and Nj is the number of samples in class j. Assuming there are K class labels, the entropy theory states that the average amount of information needed to classify a sample is as follows:

$$Info(D) = -\sum_{j=1}^{k} \frac{N_j}{N} log_2 \left(\frac{N_j}{N}\right)$$

When the dataset D split into several subsets D1, D2, D3, ..., Dn according to the results of X attributes, information gain is defined as:

$$Gain(X,D) = Info(D) - \sum_{i=1}^{n} \frac{N^i}{N} Info(D_i)$$

```
GenerateTree(X)
    If NodeEntropy(X)<O1
        Create leaf labelled by majority class in X
        Return
    I←SplitAttribute(X)
    For each branch of xi
        Find Xi falling in branch
        GenerateTree(X)

SplitAttribute(X)
    MinEnt←MAX
    For all attributes i=1,...,d
        If xi is discrete with n values
            Split X into X1,...,Xn by xi
            E←SplitEntropy(X1,...,Xn)
            If e<MinEnt MinEnt←e;bestf←i
        Else
            For all possible splits
                Split X into X1,X2 on xi
                e←SplitEntropy(X1,X2)
                if e<MinEnt MinEnt←e;bestf←i
    Return bestf
```

Figure 1. Pseudcode for C4.5 Algorithm [7]

Where N^i is the number of samples in the subset D_i. C4.5 apply Gain_ratio, of the gain, as criteria:

$$Gain\_ratio(X,D) = \frac{Gain(X,D)}{\left(-\sum_{i=1}^{n} \frac{N^i}{N} log_2\left(\frac{N^i}{N}\right)\right)}$$

C4.5 are greedy partitioning nodes until Gain_ratio nontrivial value achieved. A prune procedure is then executed to avoid trees that overfit the data complex [6].

Formation of leaves on each node uses the naïve Bayes algorithm. Naïve Bayes algorithm is a statistical classifier based on Bayes theorem. Bayesian classification is very simple and shows a higher accuracy and speed when applied to large databases. Naïve Bayes works on the assumption that the effect of an attribute value on a given class is independent of the value of the other attributes. This assumption is called class conditional independent [8].

Bayesian classification can predict class membership probabilities, such as the probability of a given tuple owned by a particular class. Naïve Bayes classification predicts that tuple X into class Ci using the formula:

$$P\left({}^{C_i}/_{X}\right) = \frac{P\left({}^{X}/_{C_i}\right)P(C_i)}{P(X)}$$

Where $P(Ci/X)$was maximum posteriori hypothesis for class Ci.

If the probability of a previously unknown class, it is generally assumed that the class has the same probability, namely:

$$P(Ci) = P(C2) = \cdots = P(Cm)$$
$$P(Ci/X) = P\left(\frac{Xj}{Ci}\right)$$

Otherwise:

P(Ci/X)=P(X/Ci)P(Ci)

Earlier class probabilities estimated by P (Ci) = | Ci, D | / | D |, where | Ci, D | is the number of tuples in the training class Ci in D.

Given dataset with many attributes, it requires the calculation of the extreme computational expensive to compute P (X / Ci). To reduce computation in evaluating P (X / Ci), the conditional independence assumption of naïve class is made. Alleged value of this attribute is conditionally independent of each other, given the class label of a tuple for example, that there is no dependency relationship between attributes, defined:

$P\left(\frac{X}{Ci}\right) = \sum_{k=1}^{n} p\left(\frac{Xk}{Ci}\right)$
= P(Xi/Ci)xP(X2/Ci)x…..P(Xn/Ci)

Probability P(Xi/Ci), P(X2/Ci), … easily predicted form training tuple.

3. Research Method

This study combines the naïve Bayes algorithm and decision tree to detect attacks on computer networks, sometimes referred naïve Bayes tree. Algorithm was first proposed by Kohavi [9]. Formation of nodes using C4.5 algorithm, while the formation of class labels on leave using the naïve Bayes algorithm. Tool used Waikato Environment Knowledge Analysis (WEKA). This study uses IDS KDD 99 dataset. Dataset is divided into two, the data training and data testing. Training data consists of 125 973 instances, and data testing a number of 22 544. Attributes used in this dataset 41 attributes and one class attribute label. Algorithm Performance is measured based on confusion matrix [10].

Algorithm performance is evaluated based on [10]:

$Accuracy=\frac{TP+TN}{TP+FP+FN+TN}$

$Sensitivity=\frac{TP}{TP+FN}$

$Precision=\frac{TP}{TP+FP}$

$Spesivicity=\frac{TN}{TN+FP}$

4. Results and Analysis

Experiments were performed using the software Waikato Environment Knowledge Analysis (WEKA). Experiments were performed using a dataset that is divided into two, namely the data training and data testing. Training data used by the algorithm to learning, and then tested on the data testing. Based on the experimental results obtained confusion matrix, which will be used to evaluate the algorithm. Confusion matrix consists of True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN). True positive is a positive class (yes) that can be predicted algorithm correctly. False positive is a positive class (yes) which is predicted as negative class (not). False negatives are predicted negative class as the positive class by the algorithm. Meanwhile, the negative is true negatis class correctly predicted by the algorithm. Confusion matrix of each algorithm is shown in the following table.

Table 3. Atribute of Intrusion Detection System Dataset [11]

Attribute	Description	Type	Attribute	Description	Type
1.Duration	Durration of the connection (in seconds)	Cont.	22.Is guest login	1 if the login is the guest login, 0 otherwise	Disc.
2.Protocol type	Type of the connection protocol	Disc.	23.Count	Number of the connection to the same host as the current connection in the past 2 second	Cont.
3.Service	Destination service	Disc.	24.Srv. count	Number connection to same service as the current connection in the past 2 second	Cont.
4.Flag	Status flag of the connection	Disc.	25.Serror rate	% connection that have 'SYN' errors	Cont.
5.Source bytes	Number of bytes send from source to destination	Cont.	26.Srv error rate	% connection that have 'SYN' errors	Cont.
6.Destination bytes	Number of bytes send from destination to source	Cont.	27.Rerror rate	% connection that have 'REJ' errors	Cont.
7.Land	1 if connection is from/to same host/port	Disc.	28.Srv error rate	% connection that have 'REJ' errors	Cont.
8.Wrong	Number of wrong fragments	Cont.	29.Same srv rate	% connection to the same service	Cont.
9.Urgent	Number of urgent packets	Cont.	30.Diff srv rate	% connection to different service	Cont.
10.Hot	Number of hot indicators	Cont.	31.Srv diff host rate	% connection to different host	Cont.
11.Failed login	Number of failed logins	Cont.	32.Dst host count	Count of connection having the same destination host	Cont.
12.Logged in	1 if succesfully logged in,0 if otherwise	Disc.	33.Dst host srv count	Count of connection having the same destination host and using the same service	Cont.
13.Compromised	Number of compromised indicators	Cont.	34.Dst host same srv rate	% connection having the same destination host and using the same service	Cont.
14.Root shell	1 if root shell obatained, 0 if otherwise	Cont. .	35.Dst host diff srv rate	% different service on the current host	Cont.
15.Su attempted	1 if 'su root' command attempted,0 otherwise	Cont.	36.Dst host same src port rate	% connection to the current host having the same src port	Cont.
16.Root	Number of root acess	Cont.	37.Dst host srv diff host rate	% connection to the same service coming from different host	Cont.
17.File creations	Number of file creation operations	Cont.	38.Dst host serror rate	% connection to the current host that have an S0 error	Cont.
18.Shells	Number of shell prompts	Cont.	39.Dst host srv serror rate	% connection to the current host and spesified service that have an S0 error	Cont.
19.Access file	Number of operations on acess control files	Cont.	40.Dst host rerror rate	% connection to the current host that have an RST error	Cont.
20.Outbound cmds	Number of outbound commands in ftp session	Cont.	41.Dst host srv rerror rate	% connection to the current host and spesified service that have an RST error	Cont.
21.Is hot login	1 if the login is the hot list, 0 otherwise	Disc.			

Table 4. Confusion Matrix for Decision Tree

Decision tree		prediction	
		yes	no
actual	yes	9448	263
	no	3900	8933

Based on Table 4 it can be seen the number of true positives 9448, 263 false positive, false negative number 3900 and 8933 the number of true negatives. Number of true positives is greater than the number of true negatives, meaning that the number of correctly predicted positive class is greater than the negative class is predicted correctly by the decision tree algorithm.

According to the Table 5 it can be seen the number of true positives 9041, 670 the number of false positives, false negatives number 4714 and 8119 the number of true negatives.

Number of true positives out weighs the true negatives, the number of false negatives is greater than the false positives. When compared with the decision tree confusion matrix in Table 4, it can be seen that the number of classes predicted true positive (true positive) naïve Bayes algorithm is lower than the true positive decision tree algorithm. Naïve Bayes algorithm has a number of false positives and false negatives larger than the decision tree.

Table 5. Confusion Matrix for Naïve Bayes

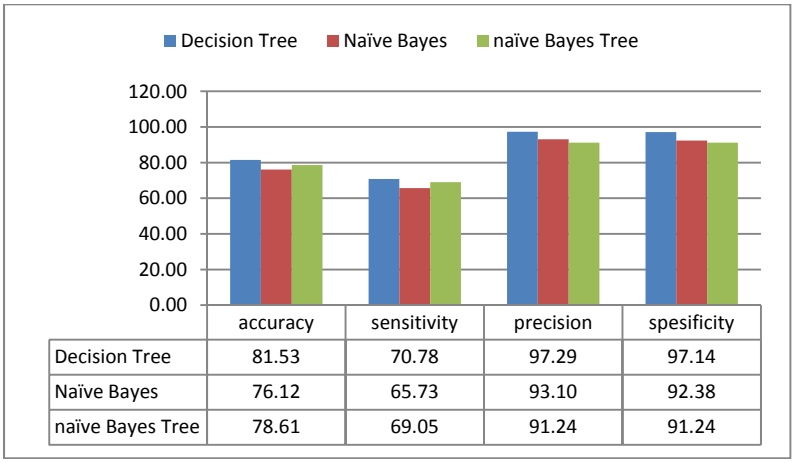
Naïve bayes		Prediction	
		yes	no
Actual	yes	9041	670
	no	4714	8119

Based on Table 6, it can be seen the number of true positives 8860, 851 the number of false positives, false negatives number 3971 and 8862 the number of true negatives. Number of true positives and true negatives are not much different. Compared with both the number of true positives previous algorithms naïve Bayes tree algorithm has the lowest number. While true negative is greater than the naïve Bayes but still lower than the decision tree. Naïve Bayes tree capable meningkkatkan number of true negatives naïve Bayes. It means that the application of the naïve Bayes decision tree can improve the performance of naïve Bayes algorithm.

Table 6. Confusion Matrix for Naïve Bayes Tree

naïve bayes tree		Prediction	
		yes	no
Actual	yes	8860	851
	no	3971	8862

Based on the confusion matrix table above we can calculate the value of accuracy, sensitivity, precision and spesificity, shown in Figure 2.



	accuracy	sensitivity	precision	spesificity
Decision Tree	81.53	70.78	97.29	97.14
Naïve Bayes	76.12	65.73	93.10	92.38
naïve Bayes Tree	78.61	69.05	91.24	91.24

Figure 2. Algorithm Performance

According to Figure 2, decision tree algorithm has highest accuracy, sensitivity, precision and spesificity among the others. Number of true positives and true negatives decision tree algorithm is bigger than other algorithms. This shows that the number of positive class and negative class are predicted correctly by the decision tree algorithm is greater than the other algorithms. Naïve Bayes algorithm acccuracy tree has a greater level than the naïve Bayes, although still smaller than the decision tree. That is, the hybrid approach used in naive Bayes

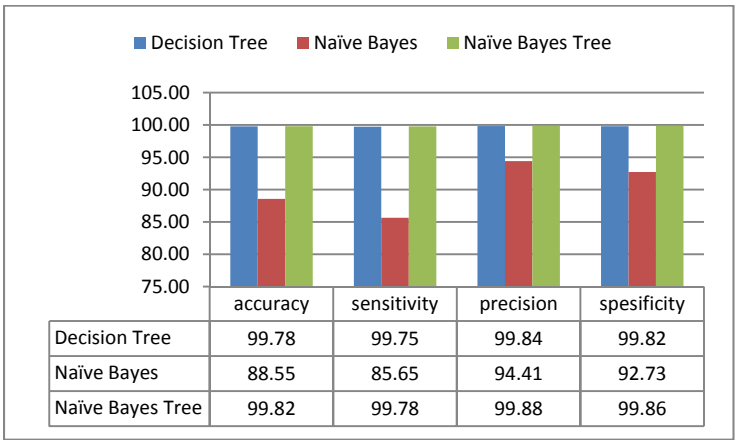
tree is able to improve the accuracy naïve Bayes. Of the three known algorithms spesificity value greater than sensitivity, this suggests that the greater the number of false negatives than false positive for all algorithms. Spesificity value that mendekkati 100% (90s%) indicate that the number of false positives is very small compared with the true negative. Precision value is also close to 100% (90s%) indicate that the number of true positives is far greater than the number of false positives. Sensitivity values are greater than 50% indicates that the greater the number of true positives than false negatives.

Further experiments applying attribute selection, using the chi-square formula. The next attribute is ranked and selected 10 (ten) top attribute for further applied to machine learning.

Table 7. Ten Attribute of Chi-Square Attribute Selection

No	Attribute	Type
1	Src_bytes	Cont.
2	Service	Disc.
3	Dst_bytes	Cont.
4	Flag	Disc.
5	Diff_srv_rate	Cont.
6	Same_srv_rate	Cont.
7	Dst_host_srv_count	Cont.
8	Dst_host_same_srv_rate	Cont.
9	Dst_host_diff_srv_rate	Cont.
10	Logged_in	Cont.

Further datasets that have been top 10 attributes of the calculation of chi-square applied to machine learning. The dataset used is the training dataset 125 973. Testing using X-10 Validation, meaning that the dataset is divided into ten equal parts, then nine parts are used as training data and testing data is a part of being, and so on until each of these sections into data testing. Then, the accuracy of the calculated value of the average is tenth part of the testing of data.



	accuracy	sensitivity	precision	spesificity
Decision Tree	99.78	99.75	99.84	99.82
Naïve Bayes	88.55	85.65	94.41	92.73
Naïve Bayes Tree	99.82	99.78	99.88	99.86

Figure 3. Algorithm Performance after Attribute Selection

Based on Figure 3 it can be seen the application of chi-square for attribute selection algorithm is able to improve significantly the accuracy of all attributes used although the number is much less than the first experiment. Numbers of attributes are used in the first experiment a number of 41 attributes, while in the second experiment a number of 10 attributes. Sensitivity, precision and spesificity also increased after the application of selection attributes using chi-square. This means the application of selection attributes using chi-square improve the performance of the three algorithms are decision tree, naïve Bayes and naive Bayes tree. The most significant improvement is the naïve Bayes tree algorithm, which has the highest level of accuracy than the other two lagoritma. After application of selection attributes using the chi-

square, naïve Bayes tree has the highest performance. Underneath, which has a very small difference is decision tree, naïve Bayes followed in third.

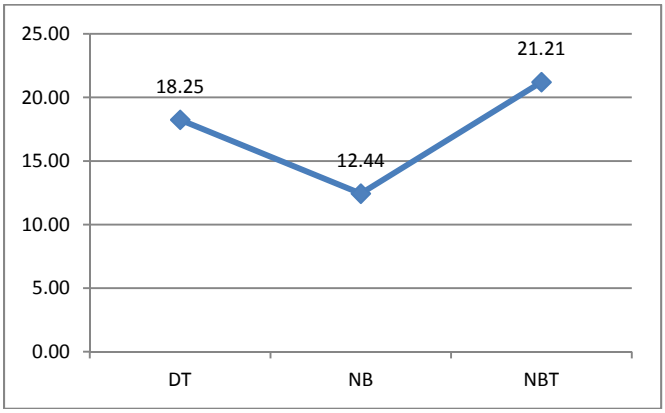


Figure 4. Accuration Increasing After Attribute Selection

5. Conclusion

The application of a hybrid approach to naïve Bayes decision tree can improve the performance of naïve Bayes, although still below the decision tree. Application of the chi-square attribute selection can improve the performance of the three algorithms significantly. After application of selection attributes using chi-square, naïve Bayes tree has the best performance than two other algorithms. Future studies can compare these results with the application of other attributes selection such information gain.

References

[1] Alexander Gostev, Yury Namestnikov. Securelist. http://www.securelist.com/en/analysis/204792162/Kaspersky_Security_Bulletin_2010_Statistics_2010
[2] K Scarfone, P Mell. Special Publication 800-94: Guide To Intrusion Detection and Prevention Systems. Gaithersburg, Maryland. 2007.
[3] B Neethu. Classification of Intrusion Detection Dataset Using Machine Learning Approaches. *International Journal of Electronics and Computer Science Engineering*. 2012; 1044-1051.
[4] AA Olusola, AS Oladele, DO Abosede. Analysis of KDD '99 Intrusion Detection Dataset for Selection of Relevance Feature. *World Congress on Engineering and Computer Science 2010.* San Fransisco. 2010.
[5] M Tavallaee, E Bagheri, W Lu, Ali A Ghorbani. A Detailed Analysis Of The KDD Cup 99 Data Set. *Proceedings Of The 2009 IEEE Symposium On Computational Inteligence in Security and Defense Application (CISDA)*, Otawa. 2009; 53-58.
[6] Lan Yu, Guoqing Chen, Andy Koronios, Shiwu Zu, Xunhua Guo. Application and Comparison of Classification Techniques in Controlling Credit Risk. *Recent Advances in Data Mining of Enterprise Data: Algorithms and Applications*. 2007; 6: 111-146.
[7] Ethem Alpaydin. *Introduction To Machine Learning*, Second Edition ed. Massachusetts: Massachusetts Institutes of Technology. 2010.
[8] Bendi Venkata Ramana, M Surendra Prasad Babu, NB Venkateswarlu. A Critical Study Of Selected Classification Algorithms For Liver Disease Diagnosis. *International Journal Of Database Management Systems*. 2011; 3(2): 101-114.
[9] Ron Kohavi. Scaling Up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybrid. *Second International Conference on Knoledge Discovery and Data Mining*, Portland. 1996: 202-207.
[10] Ian H Witten, Eibe Frank, Mark A Hall. *Data Mining Practical Machine Learning Tools and Technique Third Edition*. New York: Morgan Kaufmann. 2011.
[11] Shaik Akbar, K Nageswara Rao, JA Chandulal. Intrusion Detection System Methodologies Based On data Analysis. *International Journal of Computer Application*. 2010; 5(2): 10-20.