Anomaly Intrusion Detection Using an Hybrid Of Decision Tree And K-Nearest Neighbor

Article · January 2015

CITATION
1

READS
152

1 author:

Abdullateef Balogun
University of Ilorin
13 PUBLICATIONS **4** CITATIONS

SEE PROFILE

A Multidisciplinary Journal Publication of the Faculty of Science, Adeleke University, Ede, Nigeria

2015 Edition Vol 2.

Anomaly Intrusion Detection Using an Hybrid Of Decision Tree And K-Nearest Neighbor

Balogun, A. O. & Jimoh, R. G.
Department of Computer Science
University of Ilorin
Ilorin, Nigeria
jimoh_rasheed@unilorin.edu.ng

ABSTRACT
With the dictate of the 21st century making the economy to be information-driven, securing this valuable asset becomes interesting engagement. Intrusion detection plays a vital role in this regard by allowing prospective attacks or threats to information resources by unauthorized person(s) be detected and prevented. Previous researchers have identified the need for a robust approach to intrusion detection. This paper presents an overview of intrusion detection and a hybrid classification algorithm based on decision tree and K Nearest neighbour. The data set is first passed through the decision tree and node information is generated. Node information is determined according to the rules generated by the decision tree. This node information (as an additional attribute) along with the original set of attributes is passed through the KNN to obtain the final output. The key idea here is to investigate whether the node information provided by the decision tree will improve the performance of the KNN. A performance evaluation is performed using a 10-fold cross validation technique on the individual base classifiers (decision tree and KNN) and the proposed hybrid classifier (DT-KNN) using the KDD Cup 1999 dataset on WEKA tool. Experimental results show that the hybrid classifier (DT-KNN) gives the best result in terms of accuracy and efficiency compared with the individual base classifiers (decision tree and KNN).

Keywords: Network security, intrusion detection system, classifiers, k nearest neighbour, decision tree.

1. INTRODUCTION

With the enormous growth of computer networks usage and the huge increase in the number of applications running on top of it, network security is becoming increasingly more important. As it is shown in (Landwehr, Bull, McDermott & Choi, 1994), all the computer systems suffer from security vulnerabilities which are bothtechnically difficult and economically costly to be solved by the manufacturers. Therefore, the role of Intrusion Detection Systems (IDSs), as special-purpose devices to detect anomalies and attacks in the network, is becoming more important. It should be noted that IDSs is not only used for detection but also in monitoring attempts to break security (Chen, Hsu &Shen, 2005). Traditional protection techniques such as user authentication, data encryption, avoiding programming errors and firewalls are used as the first line of defence for computer security. If a password is weak and is compromised, user authentication cannot prevent unauthorized use, firewalls are vulnerable to errors in configuration and suspect to ambiguous or undefined security policies (Summers, 1997). They are generally unable to protect against malicious mobile code, insider attacks and unsecured modems. Programming errors cannot be avoided as the complexity of the system and application software is evolving rapidly leaving behind some exploitable weaknesses. Consequently, computer systems are likely to remain unsecured for the foreseeable future. Therefore, intrusion detection is required as an additional wall for protecting systems despite the prevention techniques.

Intrusion detection is useful not only in detecting successful intrusions, but also in monitoring attempts to break security, which provides important information for timely counter measures(Heady, Luger, Maccabe, &Servilla, 1990; Sundaram, 1996). Intrusion detection has emerged to gather and analyse a number of key points in computer systems and networks, to find if there are abnormal behaviours against the policy of system or violent sign in the network. The combination of software and hardware for detection intrusion is IDS (Zhao, Chen & Lou, 2011). Researchers have developed two main approaches for intrusion detection: misuse and anomaly intrusion detection. Misuse consists of representing the specific patterns of intrusions that exploit known system vulnerabilities or violate system security policies.

On the other side, anomaly detection assumes that all intrusive activities are necessarily anomalous. This means that if we could establish a normal activity profile for a system, we could, in theory, flag all system states varying from the established profile as intrusion attempts. These two kinds of systems have their own strengths and weaknesses. The former can detect known attacks with a very high accuracy via pattern matching on known signatures, but cannot detect novel attacks because their signatures are not yet available for pattern matching. The latter can detect novel attacks but in general for most such existing systems, have a high false alarm rate because it is difficult to generate practical normal behaviour profiles for protected systems. We designed a hybrid algorithm which not only have low false alarm rate and but also increasing detection accuracy on detection known and unknown attacks.

In our experiments, we use the data which originates from NSL-KDD 1999; a benchmark datasets. It was developed for KDD CUP 1999. During the experiment, we examine the attack in four types, denial of service, user to root, root to local and probe, distinguish with normal. The rest of the paper is organized as follows. Section 2 presents the related works using corresponding machine learning Algorithms for proposed model. Section 3 described the KDD 99 intrusion detection cup dataset. Section 4 introduce the proposed model for AIDS. Using those machine learning algorithms in our proposed system, which presented in Section 2, Section 5 describes the experimental results obtained by using WEKA tool (Weka, 2015). Section 6 for conclusion for this paper.

2. RELATED WORK

In 1980, the concept of intrusion detection began with Anderson's seminal paper (Anderson, 1980); he introduced a threat classification model that develops a security monitoring surveillance system based on detecting anomalies in user behavior. In 1986, Dr. Denning proposed several models for commercial IDS development based on statistics, Markov chains, time-series, etc (Denning, 1987). In the early 1980's, Stanford Research Institute (SRI) developed an Intrusion Detection Expert System (IDES) that monitors user behavior and detects suspicious events (Denning & Neumann, 1985).

In 1988, a statistical anomaly-based IDS was proposed by Haystack (Smaha& Haystack, 1988), which used both user and group-based anomaly detection strategies. In 1996, Forrest et al. proposed an analogy between the human immune system and intrusion detection that involved analyzing a program's system call sequences to build a normal profile (Forest, Hofmeyr, Somayaji&Longstaff, 1996). In 2000, Valdes et al. (Valdes & Skinner, 2000) developed an anomaly based IDS that employed naïve Bayesian network to perform intrusion detecting on traffic bursts.

In 2003, Kruegel et al. (Kruegel, Mutz, Robertson, &Valeur, 2003) proposed a multisensory fusion approach using Bayesian classifier for classification and suppression of false alarms that the outputs of different IDS sensors were aggregated to produce single alarm. In the same year, Shyu et al. (Shyu, Chen, Sarinnapakorn& Chang, 2003) proposed an anomaly based intrusion detection scheme using principal components analysis (PCA), where PCA was applied to reduce the dimensionality of the audit data and arrive at a classifier that is a function of the principal components. In 2003, Yeung et al. (Yeung& Ding, 2003) proposed an anomaly based intrusion detection using hidden Markov models that computes the sample likelihood of an observed sequence using the forward or backward algorithm for identifying anomalous.

Lee et al. (Lee &Stolfo, 1998) proposed classification based anomaly detection using inductive rules to characterize sequences occurring in normal data. In 2000, Dickerson at al. (Dickerson & Dickerson, 2000) developed the Fuzzy Intrusion Recognition Engine (FIRE) using fuzzy logic that process the network data and generate fuzzy sets for every observed feature and then the fuzzy sets are used to detect network attacks. In 2003, Ramadas et al. (Ramadas&Tjaden, 2003) presented the anomalous network traffic detection with self-organizing maps using DNS and HTTP services that the neurons are trained with normal network traffic then real time network data is fed to the trained neurons, if the distance of the incoming network traffic is more than a preset threshold then it raises an alarm.

3. DATASETS DISCRIPTION

Since 1999, KDD'99 (Yimin, 2004) has been the most widely used data set for the evaluation of anomaly detection methods. This data set is built based on the data captured in DARPA'98 IDS evaluation program (KDD, 1999). DARPA'98 is about 4 gigabytes of compressed raw (binary) tcpdump data of 7 weeks of network traffic. The two weeks of test data have around 2 million connection records. KDD training dataset consists of approximately 4,900,000 single connection vectors each of which contains 41 features and is labeled as either normal or an attack, with exactly one specific attack type.

The simulated attacks fall in one of the following four categories:

1. Denial of Service Attack (DoS): is an attack in which the attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine.
2. User to Root Attack (U2R): is a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system.
3. Remote to Local Attack (R2L): occurs when an attacker who has the ability to send packets to a machine over a network but who does not have an account on that machine exploits some vulnerability to gain local access as a user of that machine.
4. Probing Attack: is an attempt to gather information about a network of computers for the apparent purpose of circumventing its security controls. Table 1 showed the four categories and their corresponding attacks on each category.

Classification of Attacks	Attack Name
DoS	smurf, land, pod, teardrop, neptune, back
R2L	ftp_write, guess_passwd, imap, multihop, phf, spy,
U2R	perl, buffer_overflow, rootkit, loadmodule
Probe	ipsweep, nmap, satan, portsweep

Figure 3.1: Classification of attacks on KDD data set

4. PROPOSED HYBRID ALGORITHM

Given a training data D = $\{t_1,\ldots,t_n\}$ where $t_i = \{t_{i1},\ldots,t_{ih}\}$ and the training data D contains the following attributes $\{A_1, A_2,\ldots,A_n\}$ and each attribute Ai contains the following attribute values $\{A_{i1}, A_{i2},\ldots,A_{ih}\}$. The attribute values can be discrete or continuous. Also the training data D contains a set of classes C = $\{C_1, C_2,\ldots,C_m\}$. Each example in the training data D has a particular class C_j. The algorithm first searches for the multiple copies of the same example in the training data D, if found then keeps only one unique example in the training data D (suppose all attribute values of two examples are equal then the examples are similar). Then the algorithm discretizes the continuous attributes in the training data D by finding each adjacent pair of continuous attribute values that are not classified into the same class value for that continuous attribute. Next the algorithm ranks the selected node's neighbor among the training data set, and uses the class label of the k most similar neighbor to predict the class of the new data.

The classes of the neighbors are weighted using the similarity of each neighbor of X, where similarity is measured by cosine similarity which is defined as:

$$Sim(X, D_j) = \frac{\sum_{t_i \varepsilon (X \cap D_j)} X_i * d_{ij}}{\|X\|_2 * \|D_j\|_2} \quad \ldots\ldots 1$$

Where:

X is the test node, represented as a vector;
D_j is the jth training dataset;
t_i is the attribute shared by X and D_j
X_i is the weight of the attribute t_i in X
$d_{ij}$is the weight of attribute t_i in D_j
$\|X\|_2$ is the normalization of X
$\| D_j\|_2$ is the normalization of D_j

Then the algorithm classifies all the examples in the training data D with these methods. If any of the training example is misclassified then the algorithm calculates the information gain for each attributes $\{A_1, A_2,\ldots,A_n\}$ in the training data D

$$Info(D) = -\sum_{j=1}^{m} \frac{freq(C_j, D)}{|D|} \log_2 \left(\frac{freq(C_j, D)}{|D|} \right) \quad \ldots\ldots 2$$

$$Info(T) = \sum_{i=1}^{n} \frac{|T_i|}{|T|} \inf o(T_i) \quad \ldots\ldots 3$$

$$Information\ Gain\ (A_i) = Info(D) - Info(T) \quad \ldots\ldots 4$$

and chooses one of the best attributes Ai among the attributes $\{A_1, A_2,\ldots,A_n\}$ from the training data D with highest information gain value, Then split the training data D into sub-datasets $\{D_1,D_2,\ldots,D_n\}$ depending on the chosen attribute values of A_i.

The algorithm will continue this process until all the examples of sub/sub-sub-datasets are correctly classified. When the algorithm correctly classifies all the examples then information for each datasets are preserved for future classification of unseen examples. The main procedure for the proposed algorithm is described as follows:

Algorithm DT-KNN (D, A, T)

If D contain s only training examples of the same class $c_j \varepsilon C$ **then**
Make *T* a leaf node labeled with class c_j,
elseif A = ϕ **then**
make*T* a leaf node labeled with c_j, which is the most frequent class in D
else // D contains examples belonging to a mixture of classes.
// we select a single attribute to partition D into subsets so that each subset is purer
P_0 = impurityEval -1(D);
For each attribute $A_i \varepsilon \{A_1, A_2, A_3,\ldots..,A_k\}$ **do**
P_i = impurityEval – 2(A_i, D)
End
Select $A_g \varepsilon \{A_1, A_2, A_3,\ldots..,A_k\}$ that gives the biggest impurity reduction, computed using $P_0 – P_i$,
If $P_0 – P_g <$ threshold **then** // A_g does not significantly reduce impurity P_0
Make*T* a leaf node labeled with c_j the most frequent class in D
else // A_g is able to reduce impurity P_0
Make*T* a decision node on A_g,
Let the possible values of A_g be $V_1, V_2, V_3,\ldots\ldots, V_m$
Partition D into m disjoint subsets $D_1, D_2,\ldots..,D_m$ based on the m values of A_g
For each D_j in $\{ D_1, D_2,\ldots\ldots, D_m \}$ **do**
If$D_j \neq$ ϕ **then**
Create a branch (edge) node T_j for V_j as a child node of T;
Decision Tree $\{D_j, A – \{A_g\}, T_j\}$ // A_g is removed
For each process D_j in training data **do**
calculateSim(X, D_j);
ifSim(X, D_j) == 1.0 **then**
X is normal and exit;
Order Sim(X, D_i) from Lowest to highest, (i = 1,...,N);
Find K biggest scores of Sim(X, D);
Select the K nearest instances to X: D^K_X;
Assign to x the most frequent class in D^K_X;
Calculate Sim_Avg for k-nearest neighbours;
If Sim_Avg> threshold **then**
X is normal;
else then
X is abnormal;
end
end
end
end

Fig 2: Pseudocode for DT-KNN Classification algorithm

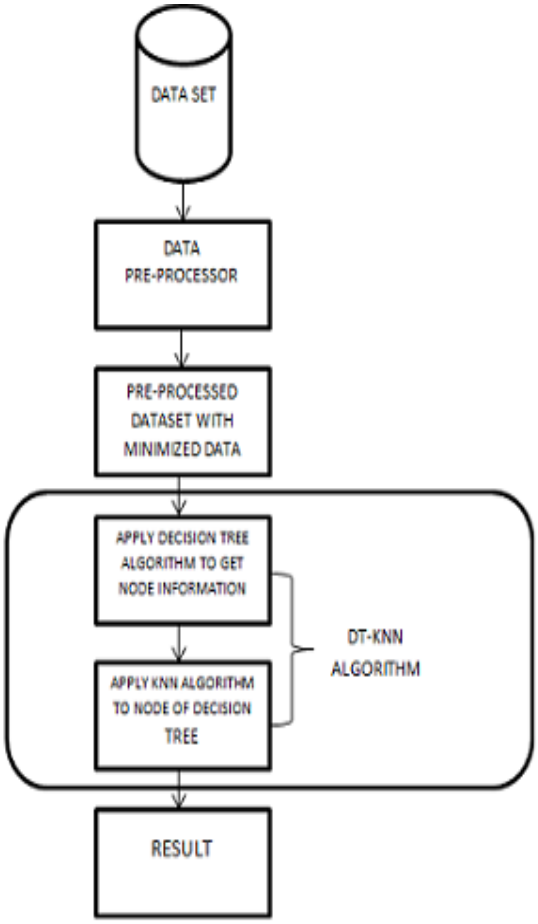

Fig 3: Proposed IDS model

5. EXPERIMENTAL ANALYSIS

Our experiments were done using Weka 3.6.7. Weka(Waikato Environment for Knowledge Analysis) is a popular suite of machine learning software written in Java, developed at the University of Waikato, New Zealand. Weka supports several standard data mining tasks, more specifically, data preprocessing, clustering, classification, regression, visualization, and feature selection. The Experiments were carried out on a 32-bit Windows 8 Professional operating system, with 2 GB of RAM and a Pentium (R) Dual-core CPU at 2.20Hz per core. Due to the iterative nature of the experiments and resultant processing power required, the java heap size for weka-3-6.7 was set to 1024 MB.

To assess the effectiveness of the algorithms, each one of them was trained on the KDD data set using a 10-fold cross validation test mode in a Weka (Waikato Environment for Knowledge Analysis) environment. To test and evaluate the algorithms we use 10-fold cross validation. In this process the data set is divided into 10 subsets. Each time, one of the 10 subsets is used as the test set and the other k-1 subsets form the training set. Performance statistics are calculated across all 10 trials. This provides a good indication of how well the classifier will perform on unseen data. Hybrid DT-KNN works better than the individual decision tree and KNN for normal class. For Probe and Normal classes it performed better than an individual decision tree and KNN approach.

Table 1: Performance evaluation of the three classifiers

ATTACK TYPE	ACCURACY OF CLASSIFIERS		
	DECISION TREE	KNN	DT-KNN
NORMAL	100%	99.97%	100%
U2R	75.00%	86.54%	86.54%
PROBE	99.49%	99.44%	99.50%
R2L	98.05%	98.76%	98.76%
DOS	96.83%	98.32%	99%

From the above results, we can conclude that although the node information generated by the decision tree did enhance the performance of KNN, on the whole the hybrid DT-KNN model did not give the expected 100 percent performance in the various classes of attacks but it supersedes the individual decision tree and K Nearest Neighbor classification algorithm.

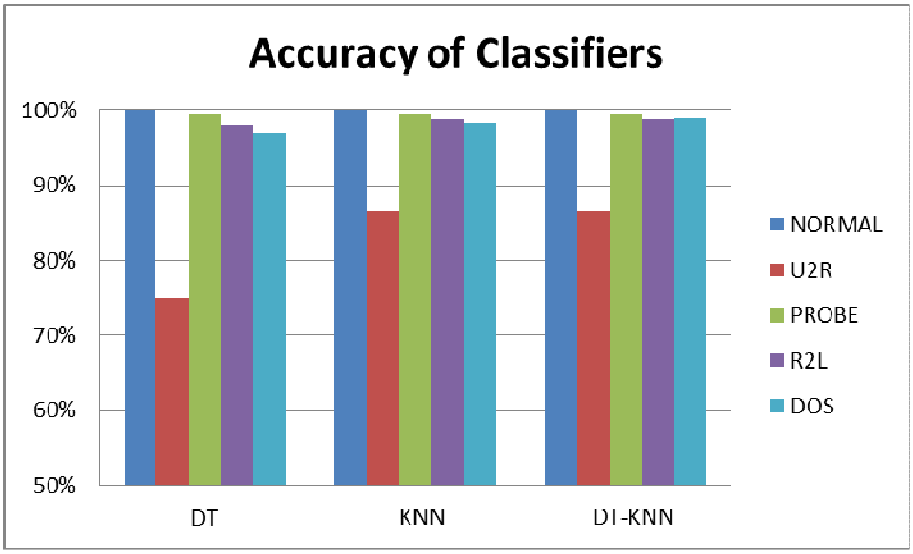

Fig 4: Graphical representation of the accuracies of the three classifiers (DT,KNN and DT-KNN)

Figure 4 above describes the accuracy of the correctly classified instances of each algorithm. After classification of KDD test dataset, it is clearly shown that the hybrid DT-KNN algorithm shows the higher detection accuracy.

6. CONCLUSION AND FUTURE WORKS

According to the experiment on the KDD Cup 1999, the proposed hybrid classifier could reach an accuracy of 100% with a false positive rate of 0%. Compared with other NIDSs that also applied KDD Cup 1999 as a dataset, this hybrid classifier showed superior performance in U2R, R2L, DoS and Probe attacks, though it was not the best for U2R and R2L attacks even as it gives equal detection rate as KNN but it took longer time which can be overlooked. However, in terms of accuracy, the proposed classifier could obtain the best performance at 100%. This experiment was performed on 10% KDD Cup 1999 dataset without sampling. Some new attack instances in the test dataset, which never appeared in training, could also be detected by this system.

REFERENCES

1. Chen, W.H., Hsu, S.H., and Shen, H.P. (2005).Application of SVM and ANN for intrusion detection.Comput.Oper. Res.,32:2617-2634.
2. Dickerson,J.E. , Dickerson, J.A. (2000). "Fuzzy network profiling for intrusion detection," InProc. of the 19th International Conference of the North American Fuzzy Information Processing Society (NAFIPS), Atlanta, GA, 2000, pp. 301-306.
3. Dorothy E. Denning, (1987)."An intrusion detection model," IEEE Transaction on Software Engineering, SE-13(2), 1987, pp. 222-232.
4. Dorothy E. Denning, and P.G. Neumann (1985)."Requirement and model for IDES- A real-time intrusion detection system," Computer Science Laboratory, SRI International, Menlo Park, CA 94025-3493, Technical Report # 83F83-01-00, 1985.
5. Forrest, S., Hofmeyr, S.A.,Somayaji, A., and Longstaff, T.A., (1996). "A sense of self for Unix processes," in Proc. of the IEEE Symposium on Research in Security and Privacy, Oakland, CA, USA, 1996, pp. 120-128.
6. Heady,.R. , Luger, .G., Maccabe, .A., and Servilla,.M.(1990)The architecture of a network levelintrusion detection system. Technical Report, Department of Computer Science, University NewMexico, August, 1990.
7. James P. Anderson (1980). "Computer security threat monitoring and surveillance," Technical Report 98-17, James P. Anderson Co., Fort Washington, Pennsylvania, USA, April 1980.
8. KDD'99 datasets, The UCI KDD Archive, http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html , Irvine, CA, USA, 1999.
9. Kruegel, C., Mutz, D., Robertson, W. and Valeur, F. (2003)."Bayesian event classification for intrusion detection," in Proc. of the 19th Annual Computer Security ApplicationsConference, Las Vegas, NV, 2003.
10. Landwehr, C.E., Bull, A.R., McDermott, J.P., and Choi, W.S.(1994). A taxonomy of computer program security flaws. ACM Comput.Surv., vol.26,no.3,pp.211–254,199.
11. Lee, W., Stolfo, S.J. (1998)."Data mining approaches for intrusion detection," In Proc. of the 7thUSENIX Security Symposium (SECURITY-98), Berkeley, CA, USA, 1998, pp. 79-94.
12. Ramadas, M.,Tjaden, S.O.B., (2003). "Detecting anomalous network traffic with self-organizing maps," In Proc. of the 6th International Symposium on Recent Advances in Intrusion Detection, Pittsburgh, PA, USA, 2003, pp. 36-54.
13. Shyu, M.L., Chen, S.C.,Sarinnapakorn, K., and Chang, L. (2003). "A novel anomaly detection scheme based on principal component classifier," in Proc. of the IEEE Foundations and New Directions of Data Mining Workshop, Melbourne, FL, USA, 2003, pp. 172-179.
14. Smaha, S.E., and Haystack, "An intrusion detection system," in Proc. of the IEEE Fourth Aerospace Computer Security Applications Conference, Orlando, FL, 1988, pp. 37-44.
15. Summers, R.C. Secure computing: threats and safeguards. NewYork: McGraw-Hill; 1997.
16. Sundaram, A. (1996). An introduction to intrusion detection. ACM Cross Roads 1996;2(4).Valdes, A., and Skinner, K., (2000)."Adaptive model-based monitoring for cyber-attack detection," in Recent Advances in Intrusion Detection Toulouse, France, 2000, pp. 80-92.
17. WEKA software, Machine Learning, http://www.cs.waikato.ac.nz/ml/weka/, The University of Waikato, Hamilton, New Zealand.
18. Yimin Wu, (2004). High-dimensional Pattern Analysis in Multimedia Information Retrieval and Bioinformatics, Doctoral Thesis, State University of New York, January 2004.
19. Zhao, J., Chen, M., and Lou, Q. (2011).Research of intrusion detection system based on neura networks. Proceedings of IEEE 3rd international Conference on Communication Software and Networks, May 27-29, 2011,Xi'an, China, pp:174-178