Intrusion Detection Technique using Hypothesis Testing

Aladesote O. I, Boniface K. Alese, Folasade Dahunsi

Abstract— **Intrusion detection systems (IDS) refer to a category of defense tools that is used to provide warnings indicating that a system is under attack or intrusion. The IDS monitors activities within a network and alerts security administrators of suspicious activities. This paper adopted an existing decision tree algorithm. The testing data was run on the algorithm to generate rules. The subsets of the 13 significant attributes of the KDD '99 dataset were coded and the mean of each rule was determined. C# Programming language was used for the implementation. This was used to set hypothesis using clustering method. The performance of hypothesis Testing System was tested with test data and also measured with confusion matrix and certain detection metrics.**

Index terms: ***Hypothesis Testing, Confusion Matrix, Clustering Analysis, KDD '99 dataset, Intrusion Detection System.***

I. Introduction

With the proliferation of cyber security threats, such as malicious viruses and worms, enormous growth of computer networks usage with the huge increase in the number of applications running on it and coupled with its benefits, the internet created numerous ways to compromise the stability and security of the system(s) connected to it. Sodiya et al (2004) explained that the dependency on computer systems by organizations and even individuals is ever on the increase and consequently, the vulnerability of these systems is also commensurably on the increase.

Securing a network against attacks has become of great importance because of the increase in the services on the network coupled with the sensitivity of information on it. Intrusion detection system is the solution to securing a network since intrusion prevention techniques have suffered a great setback (Mrutyunjaya and Manas et al, 2007).

An Intrusion is defined as any set of actions that attempt to compromise the integrity, confidentiality or availability of a resource.

This includes a deliberate unauthorized attempt to access information, manipulate information, or render a system unreliable or unusable (Dharamraj et al, 2010). Integrity can be compromised when intruders are able to modify the data, thus making the information unreliable (Tsai et al, 2006). Confidentiality can be breached when a non-privileged user is able to access the information. Availability, such as Denial of Service (DoS) attacks on the Internet, thus this can disrupt the web service and force information to be unavailable (Flora, 2009).

Manuscript received March 21, 2014; revised April 20,2014.

B. K. Alese is with the Federal University of Technology, Akure, Nigeria, e-mail: kaalfad@yahoo.com, +2348034540465

O. I. Aladesote is with the Federal University of Technology, Akure, Nigeria, e-mail: lomaladesote@yahoo.com

F. Dahunsi is with the Federal University of Technology, Akure, Nigeria, e-mail: fmdahunsi@futa.edu.ng, +2348101564160

Technology has also improved greatly on the speed of communication among hosts on the internet. This trend has contributed sophisticated, effectiveness and subtlety to the kinds and frequency of debilitating attacks to which computing network infrastructure are exposed to and fro which grave losses are incurred (Fatogun, 2012).

II. SURVEY MOTIVATION

The major means by which many organizations share information are networks. The increase in the use of networks has paved the way for intruders to attack the communication path and to steal the valuable asset (data) of any organization. The explosive increase in the number of computer networks and the use of internet in every organization has led to an increase of attacks, not only from external intruders but also internet intruders (Farid et al, 2009).

In the work of Jainganesh and Sumathi (2012), Support Vector Machine and Kernel Support Machine with Levenberg-Marquardt were applied on KDD dataset. The data was segmented in training and testing dataset. Support Vector Machine was applied on it and Kernel Support Machine was tuned with Levenberg-Marquardt on training dataset with the purpose of constructing and training the models. The trained models were evaluated on testing dataset. The accuracy of the proposed system was tested based on detection rate and false alarm rate. The detection rate for KSVM with LM was better than SVM in all other attacks. Applying feature extraction techniques(s) will give better classification result and combining more than one approach to intrusion detection would provide more accurate classification.

III. OBJECTIVE

The objective of the research work is to develop an hypothesis testing to classify various Denial of Service (DoS) attacks and normal traffic on KDD '99 dataset.

IV. METHODOLOGY

The existing works of the authorities in the field of Intrusion Detection System were reviewed. The selected thirteen attributes after Gain Ratio and Principal Component Analysis (PCA) techniques were employed to extract significant features of KDD '99 dataset were used.

Knowledge Discovery and Data Mining 1999 (KDD '99) dataset, which is an effective benchmark dataset to help researchers on intrusion detection problems. According to Chou et al. [2007], the DARPA (Defense Advanced Research Projects Agency) KDD '99 dataset is made up of a large number of network traffic connections and each connection is represented with 41 features. Further, each connection had a label of either normal or the attack type. The research work was limited to networks attacks under Denial of Service (DoS) only. The DoS dataset with normal traffic extracted from the KDD ’99 dataset contains Ninety-two thousand, five hundred and seven (92, 507) records. 89,409 were used as training data while 3098 were used as the test data.

The work of Fatogun (2012) on Denial of Service attack detection using Machine learning techniques was studied and her Decision Tree Algorithm was adopted for the hypothesis testing detailed in the paper. The training dataset was run on the adopted algorithm which resulted in thirty (30) rules. The rules were pruned using the thirteen significant attributes, which resulted in twenty-six (26) rules as shown in Table 1.

The rules generated were used for the hypothesis testing. The subsets of some of the significant attributes (protocol, service and flag) were determined. Protocol has three subsets (tcp, udp and icmp), service has seventy subsets (http, finger, smtp, private, telnet, etc) and flag has eight attributes (rstr, s0, s1, s2, s3, sf, rej and rsto). The remaining ten (10) attributes without subsets were coded with numbers. The mean of each rule was determined and this was used to set hypothesis using cluster analysis for each class of DoS attacks and normal traffic.

V. RESULT

A. *HYPOTHESIS FORMULATION*

1) Hypothesis formulation for Apache2

H_0: $\mu = (1.2, 1.333, 1.375)$

H_1: $\mu \neq (1.2, 1.333, 1.375)$

Decision rule

Accept H_0 if the mean (μ) is 1.2, 1.333 or 1.375. Reject if otherwise.

2) Hypothesis formulation for Back

H_0: $\mu = (1.75, 2.333)$

H_1: $\mu \neq (1.75, 2.333)$

Decision rule

Accept H_0 if the mean (μ) is either 1.75 or 2.333. Reject if otherwise.

1) Hypothesis formulation for Land

H_0: $\mu = (1.25)$

H_1: $\mu \neq (1.25)$

Decision rule

Accept H_0 if the mean (μ) is1.25. Reject if otherwise.

2) Hypothesis formulation for Mailbomb

H_0: $\mu = (1.25, 1.56)$

H_1: $\mu \neq (1.25, 1.56)$

Decision rule

Accept H_0 if the mean (μ) is either 1.25 or 1.56. Reject if otherwise.

3) Hypothesis formulation for Neptune

H_0: $\mu = (3.0, 3.333)$

H_1: $\mu \neq (3.0, 3.333)$

Decision rule

Accept H_0 if the mean (μ) is either 3.0 or 3.333. Reject if otherwise.

4) Hypothesis formulation for Normal

H_0: $\mu = (1.429, 1.60, 2.333, 28.5)$

H_1: $\mu \neq (1.429, 1.60, 2.333, 28.5)$

Decision rule

Accept H_0 if the mean (μ) is 1.429, 1.60, 2.333, 28.5. Reject if otherwise.

5) Hypothesis formulation for pod

H_0: $\mu = (2.50)$

H_1: $\mu \neq (2.50)$

Decision rule

Accept H_0 if the mean (μ) is 2.50. Reject if otherwise.

6) Hypothesis formulation for Processtable

H_0: $\mu = (1.0, 1.25, 1.286, 1.333, 1.50, 1.778, 2.0, 2.333)$

H_1: $\mu \neq (1.0, 1.25, 1.286, 1.333, 1.50, 1.778, 2.0, 2.333)$

Decision rule

Accept H_0 if the mean (μ) is 1.0, 1.25, 1.286, 1.333, 1.50, 1.778, 2.0, 2.333. Reject if otherwise.

Table I: Rules generated after pruning when the training data are run on Decision Tree Algorithm

No	Rule
1	Protocol = tcp, duration <= 8155, service = smtp, flag = SF, Num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, srv_rerror_rate > 0, mailbomb
2	Protocol = tcp, duration <= 8155, service = telnet, flag = SF, Num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, Dst_host_srv_diff_host_rate <= 0, srv_rerror_rate > 0, processtable
3	Protocol = tcp, duration <= 8155,service = http, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, srv_rerror_rate > 0, apache2
4	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, srv_rerror_rate > 0, apache2
5	Protocol = tcp, duration <= 8155, flag = RSTR, num_compromised <= 0, logged_in = 1, apache2
6	Protocol = tcp, duration <= 8155, service = http, flag = S0, num_compromised <= 0, apache2
7	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, processtable
8	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, processtable
9	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, srv_rerror_rate <= 0, processtable
10	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate <= 0, srv_rerror_rate <= 0, mailbomb
11	Protocol = icmp, src_bytes <= 1032, smurf
12	Protocol = icmp, src_bytes> 1032, pod
13	Protocol = udp, service = domain_u, normal
14	Protocol = udp, service = private, wrong-fragment <= 0,normal
15	Protocol = udp, service = private, wrong-fragment > 0,teardrop
16	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes > 293, normal
17	Protocol = tcp, duration <= 8155, flag = SF, num_compromised <= 0, dst_bytes <= 293, serror_rate <= 0, dst_host_srv_diff_host_rate > 0, normal
18	Protocol = tcp, duration <= 8155, flag = RSTO, Neptune
19	Protocol = tcp, duration <= 8155, flag = REJ, Neptune
20	Protocol = tcp, duration <= 8155, flag = S2, back
21	Protocol = tcp, duration <= 8155, flag = SF, num_compromised > 0, back
22	Protocol = tcp, duration <= 8155, flag = S0, land = 1, land
23	Protocol = tcp, duration <= 8155, flag = RSTR, logged_in = 0, processtable
24	Protocol = tcp, duration <= 8155, flag = S2, processtable
25	Protocol = tcp, duration <= 8155, flag = S3, processtable
26	Protocol = tcp, duration <= 8155, processtable

7) Hypothesis formulation for Smurf

H_0: μ = (2.00)
H_1: μ ≠ (2.00)
Decision rule
Accept H_0 if the mean (μ) is 2.00. Reject if otherwise

8) Hypothesis formulation for teardrop

H_0: μ = (2.667)
H_1: μ ≠ (2.667)
Decision rule
Accept H_0 if the mean (μ) is 2.667. Reject if otherwise.

VI. DISCUSSION

The performance of Hypothesis Testing System was measured based on the result generated from the Test data. The Test datasets were tested on the Hypothesis testing system. The performance is shown in table 3 where 97.93% was correctly classified, 1.97% was wrongly classified and 0.001% was not classified. Total percentage to normal connections that were classified as attacks was 10.78% and 0.2% of the normal connection was not classified. Figure 2 shows the graphical representation of the performance of hypothesis testing on test data.

A. Confusion Matrix obtained from the Classification

Confusion matrix is a specific table layout that allows visualization of the performance of an algorithm, typically a supervised learning one (in unsupervised learning it is usually called a matching matrix). Each column of the matrix represents the instance in a predicted class, while each row represents the instance in an actual class. Table III shows the confusion matrix obtained from the Hypothesis Testing classification.

Table II: Performance of Hypothesis Testing on Test data

Attacks	TCC	TWC	TUC	TOTAL
Apache2	330	1	0	331
Back	231	2	2	235
Land	10	0	0	10
Mailbomb	500	0	0	500
Neptune	601	0	0	601
Normal	446	54	1	501
Pod	57	3	0	60
Processtable	337	1	0	338
Smurf	508	0	0	508
Teardrop	14	0	0	14

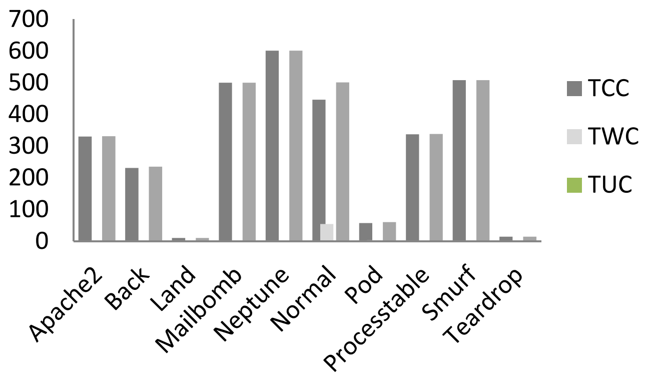

*Fig 1: Graphical representation of the performance of hypothesis testing on test data.

*Where TCC = Text Correctly Classified, TWC = Test Wrongly Classified, TUC = Test Unclassified

Out of 331 Apache2 that was classified, 330 was correctly classified while 1 was wrongly classified as Back, the Hypothesis Testing Classification showed that Apache2 was 100% accurate and 93% reliable in classification. 231 out of 245 back were correctly classified while 2 were wrongly classified and the remaining 2 was not classified, the system showed that Back was 98% and 99% accurate and reliable respectively. Out of 501 normal, 446 was correctly classified, 21 was wrongly classified as Apache2, 1 as Neptune, 28 as Process table, 4 as Smurf while 1 was not classified, the system showed that 89% accurate and 100% reliable. 57 was correctly classified as Pod out of 60 while 3 was wrongly classified as Smurf, the result showed that Pod was 95% and 100% accurate and reliable. The Classification on Teardrop showed that 337 was correctly classified while I was not classified, Teardrop was 100% accurate and 92% reliable. Smurf, Land, Mailbomb, Neptune and Teardrop were 100% correctly classified with 100% accuracy and reliability except Smurf with 99% reliability.

Table III: Confusion Matrix obtained from Hypothesis Testing on test data*

A/P	AP	BA	LA	MA	NE	NO	PO	PR	SM	TE	TUC	ACC
AP	330	1	0	0	0	0	0	0	0	0	0	1
BA	2	231	0	0	0	0	0	0	0	0	2	.98
LA	0	0	10	0	0	0	0	0	0	0	0	1
MA	0	0	0	500	0	0	0	0	0	0	0	1
NE	0	0	0	0	601	0	0	0	0	0	0	1
NO	21	0	0	0	1	446	0	28	4	0	1	.89
PO	0	0	0	0	0	0	57	0	3	0	0	.95
PR	0	1	0	0	0	0	0	337	0	0	0	1
SM	0	0	0	0	0	0	0	0	0	0	0	1
TE	0	0	0	0	0	0	0	0	14	14	0	1
REL	.93	.99	1	1	1	1	1	.92	1	1		

*Where A/P = Actual/Predicted, AP = Apache2, BA =back, LA = Land, MA= Mailbomb, NE = Neptune, NO = Normal, PO = Pod, PR = processtable, SM = Smurf, TE = teardrop, ACC = Accuracy, REL = Reliability, NM = Normal traffic misclassified, TN = Total number of normal traffic.

Average accuracy= 98.2%

Average reliability = 98.34%

Overall accuracy = 98.03%

$$\text{Classification rate} = \frac{\text{TCC+TWC}}{\text{TCC+TWC+TUC}} = \frac{3034+61}{3034+61+3}\ \frac{3095}{3098} = 99.90\%$$

$$\text{Detection rate} = \frac{TCC}{TCC+TWC+TUC} = \frac{3034}{3098} = 97.93\%$$

$$\text{False Alarm Rate} = \frac{NM}{TN} = \frac{54}{501} = 10.78\%$$

B. Hypothesis Testing Classification

The hypotheses set were categorized into three (see table IV):

- Accepted: those hypotheses with considerable evidence to support their claims.
- Rejected: those that have few evidence to support their claim and
- Eliminated: those hypotheses that were never used.

Table IV: summary of the Hypothesis Testing System Model*

R.No	NCA	NWU	Total	Category	Class
1	500	0	500	Accepted	MA
2	289	289	0	Accepted	PR
3	1	15	16	Rejected	AP
4	0	6	6	Rejected	AP
5	179	2	181	Accepted	AP
6	150	0	150	Accepted	AP
7	0	0	0	Eliminated	PR
8	0	27	27	Rejected	PR
9	0	0	0	Eliminated	PR
10	0	0	0	Eliminated	MA
11	508	7	515	Accepted	SM
12	57	0	57	Accepted	Pod
13	46	0	46	Accepted	NO
14	64	0	64	Accepted	NO
15	14	0	14	Accepted	TE
16	336	0	336	Accepted	NO
17	0	0	0	Eliminated	NO
18	2	0	2	Accepted	NE
19	599	1	600	Accepted	NE
20	3	1	4	Accepted	BA
21	228	1	229	Accepted	BA
22	10	0	10	Accepted	LA
23	0	1	1	Rejected	PR
24	0	0	0	Eliminated	PR
25	48	0	48	Accepted	PR
26	0	0	0	Eliminated	PR

*Where NCA = number of time Correctly Applied, NWA = number of time Wrongly Applied, R.No. = Rule number

Table IV shows that hypotheses set with Rule numbers. 7, 9, 10, 17, 24 and 26 were redundant, that is, they were unable to classify neither any class of DoS attack nor normal traffic and thus, they were eliminated. Four of these hypotheses were set for processtable (Rule Numbers 7, 9, 24, 26); one for Normal (Rule Number 17) and one for Mailbomb (Rule Number 10). Hypotheses with Rule Numbers 3, 4, 8 and 23 had fewer evidence to support their claim, that is, they never classified correctly but wrongly and therefore rejected. Two of these hypotheses were set for Apache2 (Rule Numbers 3 & 4) while the remaining two hypotheses (Rule Numbers 8 & 23) were set for Processtable. The remaining 16 hypotheses set with Rule Numbers 1, 2, 5, 6, 11, 12, 13, 14, 15, 16, 18, 19, 20, 21, 22 and 25 had considerable evidence for their formulation, that is, they were able to classify correctly and therefore accepted.

C. Acceptable Hypothesis for Intrusion Detection System

Below are the acceptable hypotheses for classifying both Denial of Service (DoS) attack and normal traffic:

1) Hypothesis formulation for Apache2

H_0: μ = (1.2, 1.333)
H_1: μ ≠ (1.2, 1.333)

2) Hypothesis formulation for Back

H_0: μ = (1.75, 2.333)
H_1: μ ≠ (1.75, 2.333)

3) Hypothesis formulation for Land

H_0: μ = (1.25)
H_1: μ ≠ (1.25)

4) Hypothesis formulation for Mailbomb

H_0: μ = (1.56)
H_1: μ ≠ (1.56)

5) Hypothesis formulation for Neptune

H_0: μ = (3.0, 3.333)
H_1: μ ≠ (3.0, 3.333)

6) Hypothesis formulation for Normal

H_0: μ = (1.60, 2.333, 28.5)
H_1: μ ≠ (1.60, 2.333, 28.5)

7) Hypothesis formulation for pod

H_0: μ = 2.50
H_1: μ ≠ 2.50

8) Hypothesis formulation for Processtable

H_0: μ = (1.778, 2.0)
H_1: μ ≠ (1.778, 2.0)

9) Hypothesis formulation for Smurf

H_0: μ = 2.00
H_1: μ ≠ 2.00

10) Hypothesis formulation for teardrop

H_0: μ = 2.667
H_1: μ ≠ 2.667

VII. CONCLUSION AND RECOMMENDATION

The combined results of test data using Decision Tree with Hypothesis Testing on KDD '99 dataset consistently perform better when measured with detection metrics, which are standard metrics used in research work.

It is hereby recommended that the decision tree approach be combined with hypothesis testing method for effective and efficient intrusion detection system.

REFERENCES

[1] Ankita G., and Richariya V. (2007): "A Layered Approach for Intrusion Detection using Meta- modelling with Classification Techniques," International Journal of Computer Technology & Electronics Engineering (IJCTEE) Vol. 1, Issues 2.

[2] Asha Gowda Karegowda, A. S. Manjunati & M. A. Jayaram (2010): "Comparative Study of Attributes Selection using Gain Ratio and Correlation Based Feature Selection", International Journal of Information Technology and Knowledge Management. Volume 2, No. 2, pp. 271 – 277, July – December 2010.

[3] Ashish S., Dale E. Seborg (2006): "Clustering Multivariate Time-series data", Journal of Chemometrics, vol. 2, No. 05, pp 47 – 438, January2006.

[4] Dharamraj R. Patil and V. P. Kshirsagar (2010): "An overview of adaboost-based NISD and Performance evaluation on NSL –KDD dataset", International Journal of Computer Engineering and Computer Application, vol. 1, 2010.

[5] Fatogun, B. A. (2012): "Denial of Service Attack Detection Using Machine Learning Techniques", A Thesis in the Department of Computer Science, Federal University of Technology, Akure, Nigeria.

[6] Flora S. T. (2009), "Network Intrusion Detection using Associative Rules," International Journal of Recent Trends in Engineering, Vol. 2, No. 2, November 2009.

[7] Jaiganesh, V. and Sumathi, P. (2012): "Intrusion Detection using Kernelized Support Vector Machine with Levenberg – Marquardt Learning", International Journal of Engineering Science and Technology (IJEST), vol. 4 No. 03, pp. 1153 – 1160, March 2012.

[8] Kayacik, H. G., Zincir-Heywood, A. N, and Heywood M. I. (1999): "Selecting Features for Intrusion Detection: A Feature Relevance Analysis on KDD '99 Intrusion Detection Datasets," http://www.cs.dal.ca/projectx/

[9] KDD Cup 1999 Data: Available on http://kdd.ics.uci.edu/database/kddcup99/Database/kddcup99/kddcup99.html, October 2007

[10] Mrutyunjaya Panda and Manas Ranjan Patra (2007): "Network Intrusion Detection using Naive Bayes", International Journal of Computer Science and Network Security (IJCSNS), Vol. 7, No. 12, December 2007.

[11] Rupati D, and Bhupendra V. (2010): "Feature Reduction for Intrusion Detection using Linear Discriminant Analysis", International Journal on Computer Science and Engineering (IJCSE) Vol. 02, No. 04, 2010, 1072 – 1078.

[12] Shilpa, L., Joseph S., and Bhupendra V. (2010): "Feature Reduction using Principal Component Analysis for Effective Anomaly-Based

Intrusion Detection on NSL-KDD," International Journal of Engineering Science and Technology, vol. 2(6), 2010, 1970 – 1977.

[13] Tsai F. S. And Chan C. K. (eds), Cyber Security, Pearson Education, Singapore, 2006.

[14] Yadolah Dodge (2008): "The Concise Encyclopaedia of Statistics", Published by Springer Science + Business Media, LLC (Textbook). ISBN:978-0-38732833-1.