Journal of Theoretical and Applied Information Technology
31st December 2011. Vol. 34 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

USING INFORMATION THEORY IN PATTERN RECOGNITION FOR INTRUSION DETECTION

[1]MEYSAM. MADANI, [2]ALIREZA. NOWROOZI
[1]PhD Student, Department of mathematics, Sharif University, Tehran, Iran
[2]PhD Candidate, Department of Mathematics and Computer Science, Amirkabir University of Technology
E-mail: madani@mehr.sharif.ir , ar_nowroozi@aut.ac.ir

ABSTRACT

There are so many methods of pattern recognition for a dataset, but some datasets are special! KDDCUP 99 dataset have some properties that can help us to do a better pattern recognition. For example there are many unused future that can be omitted in some manner, moreover when we want to do pattern recognition for a particular goal such as finding attacks we can find the relevant futures for any attack by information theory and just use these futures to detect attacks. we do pattern recognition for any attack by a simple way and after all we combine these results to deduce the final result.

Keywords: *Intrusion Detection Systems (Ids), Pattern Recognition, Partitioning, Kddcup99*

1. INTRUSION DETECTION SYSTEMS

An intrusion detection system (IDS) is a software, hardware or combination of both for detecting anomaly acts and intruder activities.
In other words ids do Monitoring and analysis of user and system activity, Auditing of system configurations and vulnerabilities, Assessing the integrity of critical system and data files, Statistical analysis of activity patterns based on the matching to known attacks, Abnormal activity analysis and Operating system audit[19].
An IDS consists of several components:

1- Sensors which generate security events
2- Console to monitor events and alerts
3- Engine that record events in a database and uses some rules to generate alerts

We have two types of ids that can be setting up on a network.

1- Network Based IDS (NIDS) which used more. This IDS detect attacks by capturing and analyzing network packets
2- Host Based IDS (HIDS) which monitors and analysis the internals of a computing system rather than on its external interfaces.

2. KDD CUP DATASET

The KDD Cup '99 dataset was created by processing the tcpdump portions of the 1998 DARPA Intrusion Detection System (IDS) Evaluation dataset, created by Lincoln Lab under contract to DARPA. This data sets is about 5 million record as

```
0,tcp,private,S0,0,0,0,0,0,0,0,0,0,0,
00,tcp,http,SF,334,1684,0,0,0,0,0,1,0
,0,0,0,0,0,0,0,0,0,1,9,0.00,0.00,0.00
,0.00,0.33,0,0,0.00,0.00,0.00,0.00,0.
00,0.00,0.00,0.00, normal.
```

, Where the last array is the type of attack. If we set all of this data in a set then we have a matrix by dimension 5000000×42 .Our propose is that by a part of this data decide a new data is normal or attack(and at the next which attack). In other words we want to recognizing patterns in this dataset. By this view we want to do a pattern recognition method on this set.
This data set has some really good properties which help us to gain a better result. At first we want to skim these properties.
Each column of dataset related to a future. We can see the name of each column respectively in the (table 1).

The last column of our dataset is the type of attack. If it is not an attack then we write in 42-th column normal, otherwise we write the name of that attack in it. Attacks fall into four main categories:

- **DOS:** denial-of-service, e.g. syn flood;
- **R2L:** unauthorized access from a remote machine, e.g. guessing password;

Journal of Theoretical and Applied Information Technology
31st December 2011. Vol. 34 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

- **U2R:** unauthorized access to local super user (root) privileges, e.g., various ``buffer overflow'' attacks;
- **Probing:** surveillance and other probing, e.g., port scanning.

Table 1: future names and their positions

	feature name		feature name
1	duration (c)	22	is_guest_login (s)
2	protocol_type: (s)	23	Count (c)
3	service (s)	24	srv_count (c)
4	flag (s)	25	serror_rate (c)
5	src_bytes (c)	26	srv_serror_rate (c)
6	dst_bytes (c)	27	rerror_rate (c)
7	land (s)	28	srv_rerror_rate (c)
8	wrong_fragment (c)	29	same_srv_rate (c)
9	urgent (c)	30	diff_srv_rate (c)
10	hot (c)	31	srv_diff_host_rate (c)
11	num_failed_logins (c)	32	dst_host_count (c)
12	logged_in (s)	33	dst_host_srv_count (c)
13	num_compromised (c)	34	dst_host_same_srv_rate (c)
14	root_shell (c)	35	dst_host_diff_srv_rate (c)
15	su_attempted (c)	36	dst_host_same_src_port_rate
16	num_root (c)	37	(c)
17	num_file_creations (c)	38	dst_host_srv_diff_host_rate
18	num_shells (c)	39	(c)
19	num_access_files (c)	40	dst_host_serror_rate (c)
20	num_outbound_cmds (c)	41	dst_host_srv_serror_rate (c)
21	is_host_login (s)		

One can find some further information about attacks in [6, 7 and 8]. Every attacks effect just on some futures. One can ask that we it will be understood if we do a pattern recognition method, but we can answer this by telling that we use this we use it before pattern recognition not after and this help the method to do a better work.

We want to detect these attacks in a well manner. There are so many articles [10-18] that trying to do a better approach, but up to day we can`t find a perfect way to do detecting attacks without any error, hence every one want to do design a better way by less error and time for detecting attacks.

3. INFORMATION GAIN

In [4] we see what the information of a future is. Let S be a set of training set samples in *m* classes and the training set contains s_i samples of class *I* and s is the total number of samples. Expected information needed to classify a given sample is calculated by

$$I(s_1, s_2, \ldots, s_m) = -\sum_{1}^{m} \frac{s_i}{s} \log\left(\frac{s_i}{s}\right)$$

A feature *F* with values $\{ f_1, f_2, \ldots, f_v \}$ can divide the training set into v subsets $\{ S_1, S_2, \ldots, S_v \}$ where S_j is the subset which has the value f_j for feature *F*. Furthermore let S_j contain s_{ij} samples of class *i*. Entropy of the feature *F* is

$$E(F) = \sum \frac{s_{1j} + \ldots + s_{mj}}{s} I(s_{1j}, \ldots, s_{mj})$$

Information gain for F can be calculated as:

$$Gain(F) = I(s_1, \ldots, s_m) - E(F).$$

In the following table we see the most related futures to any attacks.

Table 2: attacks and related futures

	Attcks	Related futures
1	buffer_overflow.	1,3,6
2	Loadmodule	1,3
3	perl.	14
4	neptune.	38,39
5	smurf.	2,3
6	guess_passwd.	11
7	pod.	8
8	teardrop.	2,3,23
9	portsweep.	3,4,5,23
10	ipsweep.	2,3,32
11	land.	4,23,32
12	ftp_write.	9,23
13	back.	1,5,6
14	imap.	39,3
15	Satan	1,2,3,23
16	phf.	14
17	nmap.	2,3,4,5
18	Multihop	6,13,23

We can see the schematic of this machine as follow

Now by this tool we can gain some relation between any future and attacks. Some attacks just effect on a few future. We can divide our machine to parallel machines and any machine decides just for one attack.

4. NORMALIZING DATASET

It is better to normalize our dataset to gain a better estimation. We work offline; hence without loss of generality we can normalize all data. We use

Journal of Theoretical and Applied Information Technology
31st December 2011. Vol. 34 No.2

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

the following algorithm to normalizing real value data:

$$N(f)=\begin{cases}1 & f > Max(F)\\ \dfrac{f}{\max(F)} & otherwise\end{cases}$$

Where $Max(F)$ is estimating maximum amount of future F. We neglect the outlier values, hence $Max(F)$ is not exactly the maximum of the future F. For other columns that haven't real value, we assign a number between 0 and 1 in a uniform way. For example if a future have $n+1$ string values, then we assign $0,\frac{1}{n},\frac{2}{n},\ldots,1$ to each value respectively.

5. OUR METHOD

After calculating related futures for any attacks, we should take a looking at data. We partition data to 50 parts and studding the real relations between futures and attacks. Every time we sort data according to a future, then if it partition attacks in a good manner then it can be added to the related futures. After all if one future does not separate a special attack, then we remove this future from related futures.

Table 3 attacks and related norms

	attack	Related future	Related norm
1	buffer_overflow.	1,3,6	inf
2	loadmodule	1,3	inf
3	perl.	14	2
4	neptune.	38,39	2
5	smurf.	2,3	inf
6	guess_passwd.	11	2
7	pod.	8	2
8	teardrop.	2,3,23	inf
9	portsweep.	3,4,5,23	inf
10	ipsweep.	2,3,32	inf
11	land.	4,23,32	inf
12	ftp_write.	9,23	inf
13	back.	1,5,6	inf
14	imap.	39,3	2
15	satan	1,2,3,23	inf
16	phf.	14	2
17	nmap.	2,3,4,5	inf
18	multihop	6,13,23	inf

Now we have some futures for every attack. For *i*-th attack we construct a machine M_i which decides whether it is *i*-th attack or not?

In our procedure we use a simple method to gain a pattern for any attack. In our sample we calculate the mean of related futures for normal state and *i*-th attack state. We show these means m_i and $m_{N,i}$ respectively. Now we can gain a parameter s where determine the type of our partitioning. We can gain this parameter by practice or by a machine learning methods. These parameters have showed in the 4-th column of (table 3). Now if a data is closer to m_i than $m_{N,i}$ we say that it is an attack of type i. If for all $i=1,2,\ldots,19$ it is not an attack of type i then we say that it is in normal state.

6. ALGORITHM

We can see sketch of our method in the following algorithm

1. j=1
2. Normalize j-th row and replace on x
3. Divide x to s part x_i (this part is not essentially distinct.
4. For any i, i-th machine M_i precede x_i. if it is the i-th attack then alarm
5. if for all I it don't alarm then it is a normal data
6. j=j+1
7. Go to 1 if any data remained.

For example the 2-th machine M_2 doing a calculation as follows

$$\begin{aligned}&\text{if } norm_2\left|(x_1,x_3)-(t_{3,1},t_{3,3})\right| < norm_2\left|(x_1,x_3)-(n_{3,1},n_{3,3})\right|/h\\ &\quad \text{then } d(2)=1\end{aligned}$$

Where $t_{i,j}$ is the mean of *j*-th column in which their state is the *i*-th attack. In these calculations h is a constant that can be optimized to give a better algorithm. By practice we assign 5 to h. One can obtain better value for this parameter by machine learning methods.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

7. RESULTS

In general we have two type of error.

- False positive error
- False negative error

A false positive, occurs when a statistical test rejects a true null hypothesis and false negative, occurs when the test fails to reject a false null hypothesis.

The false negative error is a dangerous error. Just imagine that a hacker hurt the system and we can't understand this. If there is no option we prefer to reduce false negative than false positive.

The machine learning methods often have big rate of false positive error, because dataset values are distributed and sporadic.

We compare mentioned methods that have run in **MATLAB**, with similar methods in the following table.

Methods	False positive error	False negative error
Em	52.3	2.2
k-mean	35	0
x-mean	26.8	12.6
Fcm	19.8	0
Sib	34	0
Our method	20	0. 1

Table 4 comparing methods

In complexity, our method is very good Because of its simple calculations. (it has eighteen 2-clustering)

8. CONCLUSION

We can itemize some conclusions as follow:

- Our method recognizes attacks well and the most of error accord when normal data assumed as *neprune.* Or *smurf.* We can develop the method so that have less error on recognizing mentioned attacks (*neprune.* & *smurf.*) We can achieve this by machin learning methods or neural networks.
- We use a simple 2 clustering in our method. One can use some better or faster methods in any part of data to enhance results.

REFRENCES:

[1] Bishop M. C, "Pattern Recognition and Machine Learning" *springer 2006*

[2] http://kdd.ics.uci.edu, 1999**.**

[3] Anazida Zainal, Mohd Aizaini Maarof and Siti Mariyam Shamsuddin" Ensemble Classifiers for Network Intrusion Detection System" *Journal of Information Assurance and Security 4 (2009) 217-225*

[4] H. Güneş Kayacık, A. Nur Zincir-Heywood, Malcolm I. Heywood "Selecting Features for Intrusion Detection"

[5] Mrutyunjaya Panda and Manas Ranjan Patra "A Novel Classification via Clustering Method for Anomaly Based Network intrusion Detection System" *International Journal of Recent Trends in Engineering, uVol 2, No. 1, November 2009*

[6] Kristopher Kendal, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems" *International Journal of Recent Trends in Engineering, Vol. 2, No. 1, November 2009*

[7] Kdd Cup 1999 data, *http://kss.ics.uci.edu/databases/kddcup/kddcup.html*

[8] D. Dittrich, "Distributed Denial of service (DDos) attacks/Tool page" *http://staff.washington.edu/dittrich/ddos*

[9] C. Endorf, E. Schultz & J. Mellander "Intrusion Detection & Prevention" *McGraw Hill, 2004*

[10] C. Elkan, "Results of the KDD99 classifier learning", *ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, Boston MA 2000*

[11] Rebecca Bace and Peter Mell "Special Publication on Intrusion Detection Systems", *16 August 2001*.

[12] Eleazar Eskin. "Anomaly Detection over Noisy Data Using Learned Probability Distributions", *Proceedings of the Seventeenth International Conference on Machine Learning (ICML-2000), Palo Alto, California, July 2000*.

[13] Anup K. Ghosh, James Wanken, and Frank Charron. "Detecting Anomalous and Unknown Intrusions Against Programs", *Annual Computer Security Applications Conference (ACSAC'98), Scottsdale, Arizona, 7-11 December 1998.*

[14] Mark Handley, Vern Paxson, and Christian Kreibich. "Network Intrusion Detection: Evasion, Trafic Normalization, and End-to-End Protocol Semantics", *10th USENIX*

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Security Symposium, Washington, D.C., 13-17 August 2001.

[15] Steven A. Hofmeyr and S. Forrest. "Architecture for an Artificial Immune System", *Evolutionary Computation Journal, 2000.*

[16] Koral Ilgun, Richard A. Kemmerer, and Phillip A. Porras. "A *State Transition Analysis Tool for Intrusion Detection", IEEE Transactions on Software Engineering*, *1995.*

[17] Sandeep Kumar and Eugene H. Spafiord. "A Pattern Matching Model for Misuse Intrusion Detection", *Proceedings of the 17th National Computer Security Conference, pp. 11-21, October 1994.*

[18] Terran Lane and Carla E. Brodley. "An Application of Machine Learning to Anomaly Detection", *20th Annual National Information Systems Security Conference, 1, pp. 366-380, 1997.*

[19] Vern Paxson. "Bro: A System for Detecting Network Intruders in Real-Time", *Computer Networks, 31, pp. 2435-2463, Dec.1999.*