Knowledge Required for Detecting and Defending against Denial of Service Attacks

[1]S. Ramamoorthy, [2]V. Shanthi, [3]Srinivas Mukkamala, [4]Andrew H. Sung
[1, 2,] St. Joseph's College of Engineering, Chennai- 600119.
[3, 4,] New Mexico Institute of Mining and Technology, Socorro, New Mexico 87801, U.S.A.
E-mail : mailrmoorthy@yahoo.com, srinivas|sung@cs.nmt.edu

Abstract

The complexity, openness, and increasing accessibility of the Internet have all greatly increased the risk of information system security availability. A serious type of network attacks is Denial of Service (DoS), which is performed against an information system to prevent legitimate users from accessing the compromised system for service. This paper concerns detecting DoS attacks using Support Vector Machines (SVMs). The key idea is to train SVMs using already discovered patterns (signatures) that represent DoS attacks. Using a benchmark data from a KDD competition designed by DARPA (U.S. Defense Advanced Research Projects Agency), we demonstrate that highly efficient and accurate classifiers can be constructed by using SVMs to detect DoS attacks. Further, we also perform feature ranking of the DARPA intrusion data to identify the key features that are important to DoS detection.

Key words : Denial of service protocols, support vector machines

I. INTRODUCTION

Information assurance is an issue of serious global concern. The complexity and openness of client/server technology combined with Internet have brought about great benefits to the modern society; meanwhile, the rapidly increasing connectivity and accessibility to the Internet has posed a tremendous security threat. Malicious usage, attacks, and sabotage have been on the rise as more and more computers are put into use. Connecting information systems to networks such as the Internet and public telephone systems further magnifies the potential for exposure through a variety of attack channels. These attacks take advantage of the flaws or omissions that exist within the various information systems and software that run on many hosts in the network.

Ware and Steven Levy pointed out the need for computer security [1,2]. Since most of the intrusions can be located by examining patterns of user activities and audit records [3], many Intrusion Detection Systems (IDSs) have been built by utilizing the recognized attack and misuse patterns. IDSs are classified, based on their functionality, as misuse detectors and anomaly detectors. Misuse detection systems use the attack of well-know patterns as the basis for detection [3,4,5,6,7]. Anomaly detection systems use user profiles as the basis for detection; any deviation from the normal user behavior is termed as intrusions [3,4,5,8,9,10]. Several intrusion detection systems are built with human intervention and without human intervention. Rule based intrusion detection systems; these systems are characterized by their expert system properties that fire the rules when audit records or system status begin to turn illegal [3,4,11,12]. Statistical based intrusion detection systems; these systems seek to identify the deviations from the normal looking at the state and audit records [8,13,14,15,16,17,18]. Various artificial intelligence techniques are developed to automate the process by reducing human intervention; several such techniques include neural networks [16,17,18,19,20,21], autonomous agents [22,23,24], and machine learning [20,25,26]. Several data mining techniques have been introduced to identify key features or parameters that define intrusions [25,26,27,28].

This paper concerns detecting DoS attacks and the related issue of identifying important input features for DoS attacks. We use support vector machines to build IDSs. Since the ability to identify the important inputs and redundant inputs of a classifier leads directly to reduced size, faster training and possibly more accurate results, it is critical to be able to identify the important features of network traffic data for detecting misuse of the computational resources in order for the IDS to achieve maximal performance. Therefore, we also study feature ranking and selection, which is itself a problem of great interest in constructing models based on experimental data.

In the rest of the paper, a brief introduction to the data we used is given in section II. In section III. a brief introduction to DoS attacks is given. Experiments for detecting DoS attacks are given in section IV. In section V. a brief introduction to SVMs and the experimental results of using SVMs for DoS attacks is given. In section VI. we present our feature ranking methodologies for identifying key features for DoS detection. In section VII. we summarize our results.

II. DATA

In the 1998 DARPA intrusion detection evaluation program, an environment was set up by the Lincoln labs to acquire nine weeks of raw TCP dump data for a network by simulating a typical U.S. Air Force LAN. The LAN was operated simulating a real environment, but being blasted with multiple attacks. A connection is sequence of TCP packets with well defined connection time between a source and a destination IP address under known protocols [30,31]. All the connections are either labeled as normal or an attack instance. Lee and Stolfo came up with a data mining approach and identified 41 various quantitative and qualitative features for each TCP/IP connection [26]. Of this database a subset of 494021 data were used, of which 20% represent normal patterns.

Attack types fall into four main categories

1. DoS: denial of service
2. R2L: unauthorized access from a remote machine
3. U2Su: unauthorized access to local super user (root) privileges
4. Probing: surveillance and other probing

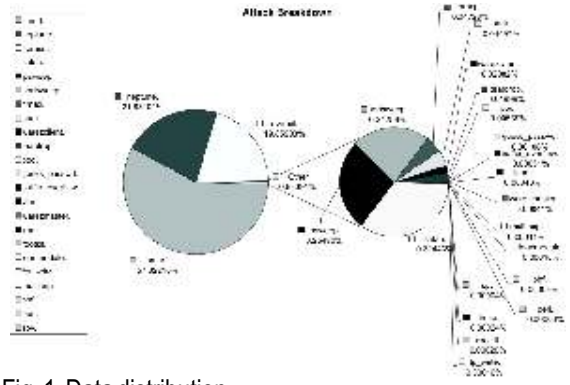

Fig. 1. Data distribution

III. DENIAL OF SERVICE ATTACKS

Attacks designed to make a host or network incapable of providing normal services are known as denial of service attacks. There are different types of DoS attacks: a few of them abuse the computers legitimate features; a few target the implementations bugs; and a few exploit the misconfigurations. DoS attacks are classified based on the services that an adversary makes unavailable to legitimate users. A few examples include preventing legitimate network traffic, preventing access to services for a group or individuals [32].

IV. EXPERIMENTS

We partition the data into the two classes of "Normal" and "DoS" patterns, where the DoS attack is a collection of six different attacks (back, neptune, ping of death, land, smurf, and teardrop). The objective is to separate normal and DoS patterns. We apply SVMs to the DARPA data set as described in Section 2. In our experiments we use the SVMs to classify patterns in several different ways. In the first set of experiments

SVMs are used to classify normal patterns vs. DoS patterns. In the second set of experiments we classify DoS patterns vs. the rest of the patterns, which include other types of attacks. Further we extended our experiments for DoS instance-specific classifications.

Table 1. Overview of denial of service attacks

Attack Type	Service	Mechanism	Effect of the attack
Apache2	http	Abuse	Crashes httpd
Back	http	Abuse/Bug	Slows down server response
Land	http	Bug	Freezes the machine
Mail bomb	N/A	Abuse	Annoyance
SYN Flood	TCP	Abuse	Denies service on one or more ports
Ping of Death	Icmp	Bug	None
Process table	TCP	Abuse	Denies new processes
Smurf	Icmp	Abuse	Slows down the network
Syslogd	Syslog	Bug	Kills the Syslogd
Teardrop	N/A	Bug	Reboots the machine
Udpstrom	Echo/Chargen	Abuse	Slows down the network

We also apply two different feature ranking methods for DoS attack patterns for the purpose of identifying the key features that can help in recognizing DoS attacks with better accuracy and/or faster detection.

V. SUPPORT VECTOR MACHINES

Support Vector Machines (SVMs) are learning machines that plot the training vectors in high-dimensional feature space, labeling each vector by its class. SVMs

classify data by determining a set of support vectors, which are members of the set of training inputs that outline a hyper plane in the feature space [33,34,35]. SVMs provide a generic mechanism to fit a hyper plane to perform a linear classification of the patterns through the use of a kernel function. The user may provide a function (e.g., linear, polynomial, or sigmoid) to the SVMs during the training process, which selects support vectors. The number of free parameters used in the SVMs depends on the margin that separates the data points but not on the number of input features, thus SVMs do not require a reduction in the number of features in order to avoid over fitting--an apparent advantage in applications such as intrusion detection. Another primary advantage of SVMs is the low expected probability of generalization errors.

A. Detecting DOS attacks using SVMs

To build SVMs for DoS detection, the input vectors are extracted from raw TCP/IP dump in the DARPA data and preprocessed; the output is a single value that indicates whether the pattern is a DoS attack.

DoS detection using SVMs consists of three phases:

- Preprocessing: an automated parser is used to process the raw TCP/IP dump data into an appropriate form.
- Training: SVM is trained on different types of attacks and normal data. We have 41 features and two classes, one is normal (-1) and the other is DoS attack data (+1).
- Testing: the trained SVM is performance tested to ensure that it has acquired adequate classification capability.

Table 2. Performance of SVMS for DOS attacks

Experiment	Accuracy (%)	Training time (sec)	Testing time (sec)	Train/Test Data sets
DoS/Normal	99.69	24.09	15.30	11593/5187 5
DoS/Rest	99.25	22.87	1.92	5092/6890
Smurf/Rest	100.00	4.79	2.45	5092/6890
Neptune/Rest	99.96	21.18	0.87	5092/6890
Back/Rest	99.70	8.34	2.87	5092/6890
Land/Rest	99.93	0.82	0.15	5092/6890
PoD/Rest	99.99	3.18	1.75	5092/6890
Teardrop/Rest	99.61	16.16	0.07	5092/6890

VI. RANKING THE SIGNIFICANCE OF INPUTS

Feature selection is an important issue in intrusion detection. Of the large number of features that can be monitored for intrusion detection purpose, which are truly useful, which are less significant, and which may be useless? The question is relevant because the elimination of useless features (or audit trail reduction) enhances the accuracy of detection while speeding up the computation, thus improving the overall performance of an IDS. In cases where there are no useless features, by concentrating on the most important ones we may well improve the time performance of an IDS without affecting the accuracy of detection in statistically significant ways.

The feature ranking and selection problem for intrusion detection is similar in nature to various engineering problems that are characterized by:

- Having a large number of input variables x = (x1, x2,_,_,_ xn) of varying degrees of importance; i.e., some elements of x are essential, some are less important, some of them may not be mutually independent, and some may be useless or noise
- Lacking an analytical model or mathematical formula that precisely describes the input-output relationship, Y = F (x).
- Having available a finite set of experimental data, based on which a model (e.g. neural networks) can be built for simulation, modeling, and prediction purposes

Due to the lack of an analytical model, we can only seek to determine the relative importance of the input variables through empirical methods. A complete analysis would require examination of all possibilities, e.g., taking two variables at a time to analyze their dependence or correlation, then taking three at a time, etc. This, however, is both infeasible (requiring 2n experiments!) and not infallible (since the available data may be of poor quality in sampling the whole input space). In the following, therefore, we apply the technique of deleting one feature at a time [36] to rank the input features and identify the most important ones for detecting DoS attacks.

A. Performance-based method for ranking importance

We first describe a general (i.e., independent of the modeling tools being used), performance-based input ranking methodology: One input feature is deleted from the data at a time, the resultant data set is then used for the training and testing of the classifier. Then the classifier's performance is compared to that of the original classifier (based on all features) in terms of relevant performance criteria. Finally, the importance of the feature is ranked

according to a set of rules based on the performance comparison.

The procedure is summarized as follows:

1. compose the training set and the testing set; for each feature do the following
2. delete the feature from the (training and testing) data;
3. use the resultant data set to train the classifier;
4. analyze the performance of the classifier using the test set, in terms of the selected performance criteria;
5. rank the importance of the feature according to the rules [Refer to our papers];

According to the rules in [29], the 41 features are ranked into the 3 types {Important}, <Secondary>, or (Unimportant), for the DoS patterns, as follows:

Important:{1,3,5,6,8,19,23-28,32,33,35,36,38-41}Secondary:<2,7,9,11,14,17,20,22,29,30,34,37>

Unimportant:(4,12,13,15,16,18,19,21,3)

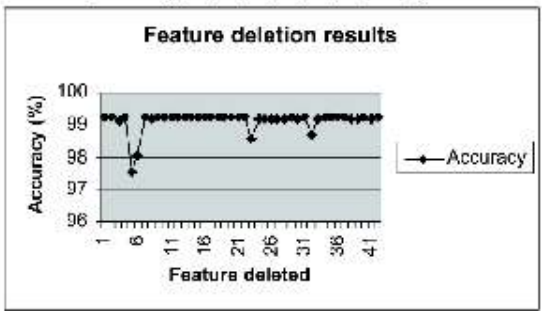

Fig. 2. DoS detection accuracy of SVMs with 40 features, obtained by deleting one feature at a time.

Table 3. SVM detection efficacy for dos using different features obtained from performance based ranking

Features	No of features	Accuracy (%)	Training Time (sec)	Testing Time (sec)
All	41	99.25	22.87	1.92
Important	19	99.22	22.79	1.84
Important + Secondary	32	99.25	19.72	2.11

B. SVM-Specific feature ranking method

Information about the features and their contribution towards classification is hidden in the support vector decision function. Using this information one can rank their significance, i.e., in the equation

$$F(X) = WiXi + b$$

The point X belongs to the positive class if F(X) is a positive value. The point X belongs to the negative class if F(X) is negative. The value of F(X) depends on the contribution of each value of X and Wi. The absolute value of Wi measures the strength of the classification. If Wi is a large positive value then the ith feature is a key factor for positive class. If Wi is a large negative value then the ith feature is a key factor for negative class. If Wi is a value close to zero on either the positive or the negative side, then the ith feature does not contribute significantly to the classification. Based on this idea, a ranking can be done by considering the support vector decision function.

The input ranking is done as follows: First the original data set is used for the training of the classifier. Then the classifier's decision function is used to rank the importance of the features. The procedure is:

1. Calculate the weights from the support vector decision function;
2. Rank the importance of the features by the absolute values of the weights;

According to the ranking method, the 41 features are placed into the 3 categories of {Important}, <Secondary> or (Unimportant), as follows:

Important: {1,5,6,23,24,25,26,32,36,38,39}

Secondary: <2,3,4,10,12,29,33,34>

Unimportant: (7,8,9,11,13,22,27,28,30,31,35,36,37, 40,41)

Table 4. SVM detection efficacy for dos using different features obtained from svdf ranking

Features	No of features	Accuracy (%)	Training Time (sec)	Testing Time (sec)
All	41	99.25	22.87	1.92
Important	11	99.16	18.93	1.00
Important Secondary	32	99.56	73.55	1.50

As indicated in the two tables, the two feature ranking methods give most consistent results in the selection of "important" features. (Results are less consistent for the other two classes of features). A few important features, as identified by both ranking methods, are given below:

- Duration: Length of the connection made by the destination system to the host system
- Source bytes: Number of bytes sent from the host system to the destination system
- Destination bytes: Number of bytes sent from the destination system to the host system
- Count: Number of connections made to the same host system in a given interval of time
- Same service rate: Percentage of connections from the destination system to the host system with the same service in a given interval of time
- Connections with SYN errors: Percentage of connections with SYN errors in a given interval of time
- Connections-Same service-SYN errors: Percentage of connections to the same service with SYN errors in a given interval of time
- Destination-Host Count: Number of connections made by the same destination system to the same host system in a given interval of time
- Destination-Host-Same source- port rate: Percentage of connections made by the destination system to the same port on the host system in a given interval of time
- Destination-Host-SYN error rate: Percentage of connections from the destination system to the host system with SYN errors in a given interval of time
- Destination-Host-Same-Service error rate: Number of connections made by the destination system using the same service to the same host system in a given interval of time

VII. CONCLUSION

We have implemented SVMs for detecting DoS patterns and validate their performance using the DARPA intrusion evaluation data. We also apply heuristic methods for ranking the features that are relevant to DoS detection.

In the IDS application (and, specifically, DoS detection) SVMs perform well and outperform other machine learning techniques like neural networks in the important respects of scalability, training time, running time and detection accuracy [21,22,28]. In particular, the training time of SVMs is frequently an order of magnitude shorter than that of neural networks, while the detection accuracy is markedly higher.

SVMs easily achieve very high detection accuracy (greater than 99%) for each of the attack instances of data. SVM feature ranking method revealed that using only the important or using important plus secondary features achieved only slightly lower accuracy, which suggests that a sensitivity selector may be included in an IDS--i.e., depending on the security requirements, different sets of features may be used in the DoS detection engine.

ACKNOWLEDGEMENTS

Support for this research received from ICASA (Institute for Complex Additive Systems Analysis, a division of New Mexico Tech) and a U.S. Department of Defense IASP capacity building grant is gratefully acknowledged. We would also like to acknowledge many insightful conversations with David Duggan that helped clarify some of our ideas. The support rendered by Dr.B.Babu Manoharn, Director, St. Joseph's College of Engineering and Dr. N. Manoharan, Dean Research, Sathyabama University is gratefully acknowledged.

REFERENCES

[1] W. H. Ware, 1979, “Security Controls for Computer Systems,” Report of Defense Science Board, Task Force on Computer Security. Santa Monica, CA: The Rand Corporation.

[2] S. Levy, 1984, “Hackers - Heroes of the Computer Revolution,” Dell.

[3] Lunt and F. Teresa, June 1993,“A Survey of Intrusion Detection Techniques,” Computers and Security 12, pp. 405-418.

[4] D. E. Denning, et al, February 1987, “Views for Multilevel Database Security,” IEEE Transactions on Software Engineering SE-13, 2 pp.129-140.

[5] Luo, Jianxiong, and S. M. Bridges, 2000, “Mining Fuzzy Association Rules and Fuzzy Frequency Episodes for Intrusion Detection,” International Journal of Intelligent Systems, Vol. 15, pp. 687-704

[6] S. Kumar, August 1995, “Classification and Detection of Computer Intrusions,” Ph.D. Dissertation.

[7] M. Toure, 1994, “An Interdisciplinary Approach for Adding Knowledge to Computer Security Systems,” IEEE Carnahan Conference on Security Technology.

[8] Ilgun and Koral, May 1993, “USTAT: A Real-time Intrusion Detection System for UNIX,” Proceedings of the 1993 Computer Society Symposium on Research in Security and Privacy. Oakland, California, Los Alamitos, CA: IEEE Computer Society Press.

[9] B. Mukherjee, L. T. Heberlein and K. N. Levitt, May/June 1994, “Network Intrusion Detection,” IEEE Networks, pp. 26-41.

[10] L. T. Heberlein, G. V. Dias, K.N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, May 1990, "A Network Security Monitor," Proceedings of the 1990 IEEE Symposium on Research in Security and Privacy, pp. 296-303.

[11] H. S. Teng, K. Chen and C. Lu. Stephen, May 1990, "Security Audit Trail Analysis Using Inductively Generated Predictive Rules," Sixth Conference on Artificial Intelligence Applications. Santa Barbara, CA,Los Alamitos, CA: IEEE Computer Society Press.

[12] Kemmerer, A. Richard, 1994, "Computer Security," Encyclopedia of Software Engineering. New York, NY: John Wiley and Sons, pp. 1153-1164.

[13] M. Bishop, "Reconciling Application and Kernel Logs," Deliverable report for DoE Lawrence Livermore National Laboratory Project W-7505-Eng-48.

[14] A. Sundaram, 1996, "An Introduction to Intrusion Detection," http://www.acm.org/crossroads/xrds2-4/xrds2-4.html.

[15] S. Forrest and T. Longstaff, May 1996, "A Sense of Self for Unix Processes," Proceedings of the 1996 IEEE Symposium on Security and Privacy, Oakland, CA 6-8 pp.120-128.

[16] H. S. Teng, K. Chen and C. Lu. Stephen, March 1990, "Security Audit Trail Analysis Using Inductively Generated Predictive Rules," In Proceedings of the 11th National Conference on Artificial Intelligence Applications, IEEE, IEEE Service Center, Piscataway, NJ, pp. 24-29.

[17] H. Debar, B. Dorizzi, 1992, "An Application of a Recurrent Network to an Intrusion Detection System," Proceedings of the International Joint Conference on Neural Networks, pp. 78-483.

[18] J. Ryan, M. J. Lin, R. Miikkulainen, 1998, "Intrusion Detection with Neural Networks," Advances in Neural Information Processing Systems 10, Cambridge, MA: MIT Press.

[19] J. Cannady, 1998, "Artificial Neural Networks for Misuse Detection," National Information Systems Security Conference.

[20] A. K. Ghosh, 1999, "Learning Program Behavior Profiles for Intrusion Detection," USENIX.

[21] S. Mukkamal, G. Janoski, A. H. Sung, 2001, "Monitoring Information System Security," Proceedings of the 11th Annual Workshop on Information Technologies & Systems, pp.139-144.

[22] M. Crosbie and E. Spafford, "Defending a Computer System Using Autonomous Agents," Technical Report CSD-TR-95-022.

[23] L.Prodromidis, S.J.Stolfo, "Agent-Based Distributed Learning Applied to Fraud Detection," Technical Report, CUCS-014-99.

[24] D. Dasgupta, 1999, "Immunity Based Intrusion Detection System: A general Framework," 22nd National Information Security Conference.

[25] S. Mukkamala, G. Janoski, A. H. Sung, 2002, "Intrusion Detection Using Neural Networks and Support Vector Machines," Proceedings of IEEE International Joint Conference on Neural Networks, pp.1702-1707.

[26] W. Lee, S. J. Stolfo, and K. Mok, August 1998, "Mining Audit Data to Build Intrusion Detection Models," Proc. KDD-98, honorable mention best application paper.

[27] W. Lee and S. Stolfo, 1998, "Data Mining Approaches for Intrusion Detection," Proc. 1998 7th USENIX Security Symposium.

[28] S. Mukkamala, A. H. Sung, August 2002, "Audit Data Reduction Using Neural Networks and Support Vector Machines," Digital Forensics Research Workshop (DFRWS-2002), Syracuse University, Center for Systems Assurance.

[29] S. Mukkamala, A H. Sung, August 2002, "Identifying Key Features for Intrusion Detection Using Neural Networks," Proceedings of ICCC International Conference on Computer Communications 2002.

[30] K. Kendall, 1998, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems", Master's Thesis, Massachusetts Institute of Technology.

[31] S. E. Webster, June 1998, "The Development and Analysis of Intrusion Detection Algorithms," S.M. Thesis, Massachusetts Institute of Technology,.

[32] T. Draelos, et. Al, 2003, "Distributed Denial of Service Characterization," Technical Report, Sandia National Laboratories.

[33] V. N. Vapnik, 1995, "The Nature of Statistical Learning Theory," Springer.

[34] T. Joachims, 2000, "Estimating the Generalization Performance of a SVM Efficiently," Proceedings of the International Conference on Machine Learning, Morgan Kaufman.

[35] T. Joachims, 2000, "SVMlight is an Implementation of Support Vector Machines (SVMs) in C," http://ais.gmd.de/~thorsten/svm_light, University.