V.Sivakumar, T.Yoganandh, R.Mohan Das / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 2, Issue 2,Mar-Apr 2012, pp.370-373

Preventing Network From Intrusive Attack Using Artificial Neural Networks

V.Sivakumar[1],T.Yoganandh[2],R.Mohan Das[3]
[1]Student, Department of Computer Science and Engineering, Hindustan University Chennai, India
[2]Student, Department of Computer Science and Engineering, Hindustan University Chennai, India
[3]Asst. Professor, Department of Computer Science and Engineering, Hindustan University Chennai, India

Abstract –
With the growth of computer networking, electronic commerce, and web services, security of networking systems has become very important. Many companies now rely on web services as a major source of revenue. Computer hacking poses significant problems to these companies, as distributed attacks can render their cyber-storefront inoperable for long periods of time. This happens so often, that an entire area of research, called Intrusion Detection, is devoted to detecting this activity.

We show that evidence of many of these attacks can be found by a careful analysis of network data. We also illustrate that neural networks can efficiently detect this activity. We test our systems against denial of service attacks, distributed denial of service attacks, and port scans. In this work, we explore network based intrusion detection using classifying, self-organizing maps for data clustering and MLP neural networks for detection.

***Keywords*— NIDS,HIDS,Information Gain**.

I.INTRODUCTION

Intrusion Detection attempts to detect computer attacks by examining data records observed by processes on the same network. These attacks are typically split into two categories, host-based attacks and network-based attacks. Host-based attack detection routines normally use system call data from an audit-process that tracks all system calls made on behalf of each user on a particular machine. These audit processes usually run on each monitored machine. Network-based attack detection routines typically use network traffic data from a network packet sniffer (e.g., tcpdump).

Many computer networks, including the widely accepted Ethernet (IEEE 802.3) network, use a shared medium for communication. Therefore, the packet sniffer only needs to be on the same shared subnet as the monitored machines. We believe that denial of service and other network-based attacks leave a faint trace of their presence in the network traffic data. Ours is an anomaly detection system that detects network-based attacks by carefully analyzing this network traffic data and alerting administrators to abnormal traffic trends. It has been shown that network traffic can be efficiently modeled using artificial neural networks.

Intrusion detection is the first step for defending against attacks. Attack alarms from IDSs are usually reported to auto-response systems or security staff for automatic or manual appropriate response actions according to the specific attacks. Identifying attacks in real-time is therefore crucial for taking appropriate response actions as soon as possible before substantial damage is done. However, nearly all the current anomaly detection methods can only detect network behavior as normal or abnormal but cannot identify the type of attack. Relying on current anomaly detection systems, therefore, is not adequate for real-time effective intrusion prevention.

On the other hand, most current intrusion detection methods lack the capacity of real-time processing large amounts of typically high dimensional audit data produced during daily operation in a computer system. In experiments carried out by MIT Lincoln Lab for the 1998 DARPA evaluation , for example, network traffic over 7 weeks contains four gigabytes of compressed binary tcpdump data which were processed into about five million connection records. Processing a large amount of audit data in real-time is therefore essential for a practical IDS so that actions for response can be taken as soon as possible.

II.EXISTING SYSTEM

A firewall is a device or set of devices designed to permit or deny network transmissions based upon a set of rules and is frequently used to protect networks from unauthorized access while permitting legitimate communications to pass.

Many personal computer operating systems include software-based firewalls to protect against threats from the public Internet. Many routers that pass data between networks contain firewall components and, conversely, many firewalls can perform basic routing functions.

1. Firewalls evolve due to cracker's ability to circumvent those increases.
2. "Always on" connections created by Cable and DSL connections create major problems for firewalls. This can be compared to leaving your car running with the keys in it and the doors unlocked which a thief may interpret as an invitation to "Please steal me".
3. Firewalls cannot protect you from internal sabotage within a network or from allowing other user's access to your PC.
4. Firewalls cannot edit indecent material like pornography, violence, drugs and bad language. This would require you to adjust your browser security options or purchase special software to monitor your children's Internet activity.
5. Firewalls offer weak defense from viruses so antiviral software and an IDS (intrusion detection system) which protects against Trojans and port scans should also complement your firewall in the layering defense.
6. Some firewalls claim full firewall capability when it's not the case. Not all firewalls are created equally or offer the same protection so it's up to the user to do their homework.
7. Cost varies. There are some great free firewalls available to the PC User but there are also a few highly recommended products, which can only be purchased. The difference may be just the amount of support or features that a User can get from a free product as opposed to a paid one and how much support that user thinks he or she will require.
8. A firewall protection is limited once you have an allowable connection open. This is where another program should be in place to catch Trojan horse viruses trying to enter your computer as unassuming normal traffic.
9. There have been claims made by IDS (Intrusion Detection System) companies where Trojan's were detected such as the RuX FireCracker v 2.0 which disabled certain Firewalls programs thus leaving the PC vulnerable to malicious actions.

III.PROPOSED SYSTEM

An intrusion detection system (IDS) is a device or software application that monitors network and/or system activities for malicious activities or policy violations and produces reports to a Management Station. Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, logging information about them, and reporting attempts. In addition, organizations use IPSec for other purposes, such as identifying problems with security policies, documenting existing threats, and deterring individuals from violating security policies. IPSec have become a necessary addition to the security infrastructure of nearly every organization.
For the purpose of dealing with IT, there are two main types of IDS:

Network intrusion detection system (NIDS)

Is an independent platform that identifies intrusions by examining network traffic and monitors multiple hosts. Network intrusion detection systems gain access to network traffic by connecting to a network hub, network switch configured for port mirroring, or network tap. In a NIDS, sensors are located at choke points in the network to be monitored, often in the demilitarized zone (DMZ) or at network borders. Sensors capture all network traffic and analyze the content of individual packets for malicious traffic.

Host-based intrusion detection system (HIDS)

It consists of an agent on a host that identifies intrusions by analyzing system calls, application logs, file-system modifications (binaries, password files, capability databases, Access control lists, etc.) and other host activities and state. In a HIDS, sensors usually consist of a software agent. Some application-based IDS are also part of this category.

IV.EXPERIMENTS AND TESTING

Data set

We used network data in the experiments to validate the proposed model. The network data is distributed by MIT Lincoln Lab for 1998 DARPA evaluation and has been widely used for evaluating various intrusion detection methods. The data contains traffic in a simulated military network that consists of hundreds of hosts. Altogether the data includes 7 weeks of training set and 2 weeks of test set that were not from the same probability distribution as the training set. Since the probability distribution is not the same, in our experiments, we only use the training set and sample one part of the data for training and another part of the data for testing. The raw training set of the data contains about 4 gigabytes of compressed binary tcpdump data of network traffic and it was pre-processed into about 5 million connection records by Lee et al. A connection is a sequence of TCP packets starting and ending at some well defined times, between which data flows from a source IP

address to a target IP address under some well defined protocol.

In the 10% subset data, each network connection is labeled as either normal, or as an
Exactly one specific kind of attack. A connection of the network data contains 41 features. These features were extracted by Lee et al. from the raw data divided into three groups: basic features of individual TCP connections, traffic features and content features within a connection suggested by domain knowledge. Among these 41 features, 34 are numeric and 7 are symbolic. Only the 34 numeric features were used in the experiments. Each connection in the data set is thus transformed into a 34-column vector as data input for detection and identification. There are 494,021 connection records in the training set in which 97,277 are normal and 396,744 are attacks. There are 22 types of attacks in total in the subset and these attacks fall into one of 4 categories:

1. DOS: denial-of-service (e.g. teardrop).
2. R2L: unauthorized access from a remote machine (e.g. password guessing).
3. U2R: unauthorized access to local super user (root) privileges by a local
a. Unprivileged user (e.g. buffer overflow attacks).
4. PROBE: surveillance and other probing (e.g. port scanning).

We group the network data into individual type of attack connections and normal connections for data preparation.

V.the learning phase and neural network structure

Prior to collecting and monitoring the network traffic trends, which we believe holds the information revealing intrusions; we must first determine the neural network structure. We present the current intensity of network traffic to the neural network in the form of the number of times a host is accessed through its' different services across the network in a certain interval dt. To allow a machine to differentiate one application's traffic from another, hosts have up to 216 or 65,536 ports to which or from which traffic travels (Stevens, 94). Monitoring all of these ports through a neural network is unnecessary. It is highly unlikely that all ports on a particular host are used at the same time. Therefore, we must establish which ports are important. To determine the important ports to monitor and thus determine the architecture of our neural network, we introduced an architectural learning phase.

We first establish an architectural multiplier F, which is multiplied by the time interval dt to develop the length of our architectural learning phase. We then observe the network traffic intensity for F * dt period of time, cataloging the number of times sources access different ports on the target machine. The number of packets received at the target machine in this period of time form a set A.

The systems administrator defines for our architectural learning phase a list of known ports to watch (the set KP) and how many extra ports (ep) the algorithm can choose to add to the list. Our system simply uses the ports given by the administrator KP and the top ep ports from the remaining Figure 2: Combining given ports with most active ones in the observed active ones to find the final set of ports traffic.

The final set of ports used is FINALSET = KP υ max (ep, A). The process of determining the final set of ports to use is illustrated in Fig. 2 (the administrator requests the addition of two ports to FINALSET). Once we have determined the ports to monitor, the neural network structure can be established as having N * M input nodes in which N is the number of sources and M is the number of monitored ports (|FINALSET|).

The following section will show the reason why the sources are clustered for input to the neural network. The first M nodes of the neural network input layer represent the total number of packets sent from the first source to the corresponding monitored port. The next M nodes of the input layer receive the respective total numbers of packets for the second source in the same order as the first layer, and so forth.

If only three sources have communicated with the target machine, then the architecture would contain three sets of M nodes in the input layer. Once the neural network is created, the number of inputs it receives is fixed by its structure. As mentioned before, the sources that we should monitor may change frequently from interval to

interval. At any given interval, the observed source activity could rise far above or sink far below the activity level of N sources used by the neural network in the previous interval.

Vi.performance measurement

	True	False
Positive	**A legitimate attack which triggers an IDS to produce an alarm**	**When no attack has taken place and no alarm is raised**
Negative	**An event signaling an IDS to produce an alarm when no attack has taken place**	**A failure of an IDS to detect an actual attack**

Information Gain.

The information gain of a given attribute X with respect to the class attribute Y is the reduction in uncertainty about the value of Y when we know the value of X, I(Y ;X). The uncertainty about the value of Y is measured by its entropy, H(Y). The uncertainty about the value of Y when we know the value of X is given by the conditional entropy of Y

Given X, H(Y |X).I(Y; X) = H(Y) − H(Y |X). When Y and X are discrete variables that take values in {y1...yk} and {x1...xl} then the entropy of Y is given by:

$$H(Y) = -\sum_{I=1}^{I=K} P(Y = yi)\log_2(P(Y = yi))$$

The conditional entropy of Y given X is:

$$H(Y|X) = -\sum_{j=1}^{j=1} P(X = xj)H(Y|X = xj)$$

Alternatively the information gain is given by:
I(Y; X) = H(X) + H(Y) − H(X, Y)
Where H(X, Y) is the joint entropy of X and Y:

$$H(X.Y) = -\sum_{I,J} P(X = xj, Y = yi)\log_2 P(X = xj, Y = yi)$$

If the predictive variable X is not discrete but continuous then in order to Compute its information gain with the class attribute Y we will consider all possible binary attributes, X_, that arise from X when we choose a threshold_ on X. _ takes values from all the values of X. Then the information gain is simply:

$$I(Y; X) = \text{argmax}x_{X\text{ø}}\, I(Y, X_{\text{Ø}})$$

Vii.conclusion

Many methods have been employed for intrusion detection. However, modeling networking traffic for a simple representation to a neural network shows great promise, especially on an individual attack basis. Also, using SOMs as a clustering method for MLP neural networks is an efficient way of creating uniform, grouped input for detection when a dynamic number of inputs are present. Once trained, the neural network can make decisions quickly, facilitating real-time detection. Neural Networks using both supervised and unsupervised learning have many advantages in analyzing network traffic and will be a continuing area of our research.

Viii.refrences

1 .Aussem, A., Mahul, A., and Marie, R., 2000, "Queueing Network Modelling with Distributed Neural Networks for Service Quality Estimation in B-ISDN Networks,"

http://www.sans.org/resources/idfaq/, 2004.

2. http://www.windowsecurity.com/articles/IDS-Part2-Classification-methodstechniques.html, 2004.

3. http://www.ll.mit.edu/IST/ideval/index.html, 2001.

4. http://www.mathworks.com/products/matlab/, 2005.

5. Fausett, Laurene. *Fundamental of Neural Networks.* 1st Edition, Prentice Hall, 1994.

6. http://www.cs.stir.ac.uk/~lss/NNIntro/InvSlides.html,