Low-Level Network Attack Recognition: A Signature-Based Approach[*]

C. Taylor, W.S. Harrison, A.W. Krings, N. Hanebutte
University of Idaho
Moscow, Idaho
USA

M. McQueen
INEEL
Idaho Falls, Idaho
USA

ABSTRACT
This research presents a new method for detecting network attacks based on network traffic signatures. It is part of a survivability architecture, which focuses on attack recognition, fault-tolerance and recovery after malicious acts. The attack recognition portion emphasizes low-level analysis of network traffic, high efficiency, real-time operation, and accurate identification of attacks. Attack recognition is based on the analysis of TCP protocol flags with respect to specific attacks and is characterized by its simplicity.

KEY WORDS
Attack recognition, intrusion detection, attack signatures.

1. Introduction

In recent years the growth of the Internet has resulted in unprecedented numbers of security related incidents. Networking greatly complicates security because there are no well-defined physical boundaries or static sets of users. It is generally recognized by security professionals that today's heterogeneous computing environment makes it nearly impossible to completely safeguard a system [4].

Attacks against computers vary in both their intent and severity. Hackers perform relatively harmless scans and probes to gather information which might later lead to a more serious Denial of Service (DoS) or root escalation attack where the attacker gains root privilege and unlimited access to system resources [13]. Whereas fault-tolerant systems design has solved many problems related to reliability and availability, it has not resulted in extensive technology transfer into the network security domain.

Fault recognition, a first step in a sequence of events leading to fault-tolerance, is very similar to intrusion detection (ID), which may serve as the stepping stone to system and network survivability. Intrusion detection is a branch of computer security that investigates attacks or violations of security policies for a single system or an entire network [15]. Intrusion Detection takes one of two main approaches to detecting intrusions: signature detection or anomaly detection. Signature or rule-based Intrusion Detection takes one of two main approaches to detecting a set of known patterns of attack behavior, similar to a virus scanner. Anomaly detection tries to capture the normal machine state and compares new behavior against this baseline to identify deviations from normal behavior [1].

This paper addresses attack recognition, a very focused form of Intrusion Detection, and presents a low-level strategy for detecting attack signatures developed from low-level network traffic. In Section 2 signatures are described. The target environment, the model definition and the statistical approach to attack recognition are presented in Section 3. Section 4 discusses experimental results, and Section 5 concludes the paper with a summary.

2. Attack Signatures

2.1 Current Research

Signature-based detection is by far the most common intrusion detection technique employed by both research and commercial Intrusion Detection Systems (IDS's) [4]. Implicit in signature-based methods is a priori knowledge of attack patterns. In essence, a signature is a rule that defines a recognized pattern whose content depends on the detection focus of the IDS. In host-based systems, attack patterns are often created from system log data or user behavior profiles. Asax [8] and Securenet [20] are host-based systems that use these types of signatures. Another type

[*]This work has been supported in part by an LDRD grant of the Idaho National Environmental and Engineering Laboratories (INEEL).

of host-based signature is created from program system call sequences [9]. Network IDS's screen network traffic for known attacks detectable from either packet sequences or packet content. Netprowler [10], NFR [18], and Bro [16] are three network IDS's that utilize network traffic signatures for network attack recognition.

2.2 Network Traffic Signatures

In monitoring network traffic, attack signatures can be captured based on either packet distribution or packet contents. Packet distribution analysis examines the frequency and sequences of the network packets as opposed to looking at the packet contents or payload [14]. An example of a packet distribution signature is the number of Syn packets received per second, which signifies a possible Syn flood attack [3]. A packet-content signature is characterized by an embedded string. This may cause buffer overflows, e.g. an IMAP buffer overflow attack [19]. In distribution analysis, it is sufficient to analyze just the packet headers. This translates into a fast and efficient analysis. Content packet analysis is more thorough, but slower, so that typically only certain types of network traffic are examined [14].

2.3 Protocol Characteristics for Signature Creation

In packet distribution analysis, we examine network traffic features as indications of potential attacks. The Internet Protocol (IP) is the most widely used network layer protocol. IP provides an unreliable, connection less packet delivery service [2]. One characteristic of IP traffic important for detecting malicious traffic is packet fragmentation. Packet fragmentation occurs when packets enter a network segment that has a maximum transmission unit (MTU) that is too small to carry the packet [3]. This results in the packet being fragmented into a number of packets, each delivered separately, with the destination machine being responsible for packet reconstruction. Attackers often use fragmentation in an attempt to crash machines, and avoid detection by routers that filter traffic. This creates a large number of fragments that are easily detected [3].

The Transmission Control Protocol (TCP) is the most popular transport layer protocol and provides reliable communication to applications that need delivery guarantees [2]. To coordinate the connection between two machines, six control flags are used, i.e. Syn, Push, Ack, Fin, Reset, and Urgent. The Syn flag is sent to initiate, and Fin is sent to terminate the connection. The Push and Ack flags coordinate data sent between two machines. A Reset flag is sent when unexpected events occur, such as hardware or software failures, and the connection must be reset. Finally, the Urgent flag causes data to be processed immediately [2]. TCP uses what is called a three-way handshake to set up the connection. A host A initiates a connection with a host B by sending a packet with the Syn flag set. Next, host B sends back a packet with the Syn+Ack flags set, acknowledging the first Syn packet. Now, host A sends a packet with the Ack flag set. At this point a TCP connection is established. The TCP connection terminates in a similar fashion.

Attackers exploit wait states in the TCP protocol by tying up resources on a victim machine by opening many incomplete connections, i.e. never sending back Ack packets that complete the connection establishment or termination [13]. Large numbers of Syn or Fin packets are strong indications of these attacks.

3. Attack Recognition Approach

3.1 Network Attack Recognition

This research uses signature-based identification of network intrusions. What distinguishes the approach from most other network IDS's is the emphasis on efficiency, low-level measurement, and the focused attack scope. Efficiency is achieved by measuring the minimal amount of information for attack recognition, i.e. the packet header. The method is further characterized by its simple statistical attack recognition test.

3.2 Target Architecture

Our target system is a single networked computer operating in a *standard* user environment. We view such an environment as a typical desktop computer, operated mostly by single individuals. Characteristics of such systems are powerful computing capabilities, low resource utilization, and a limited number of applications. Our focus is on the Linux operating system. This environment allows for the development of a very low-level approach to attack recognition. The network detection component is consistent with the overall goal of our research, to develop simple, efficient detection methods that achieve high accuracy in identifying specific attacks. A high-level view of the network attack recognition environment is presented in Figure 1.

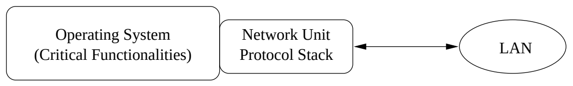

Figure 1. Network Environment

3.3 Model Definition

As previously mentioned, TCP traffic exhibits certain patterns that can be measured to distinguish normal from attack traffic. Signatures are based on frequencies of specific attributes. The *attributes* considered are a subset of the TCP control flags, i.e. Syn, Fin, and Reset, as well as the number of fragmented packets. The total number of attributes is n, e.g. in our case $n = 4$.

The attack recognition is based on comparisons of signatures to the network profile at run-time. A *profile* is defined as a vector $P = (f_1, f_2, ..., f_n)$, where f_i represents the frequency of the i^{th} attribute, $1 \leq i \leq n$. Frequencies f_i in P are in one-to-one correspondence with the attributes considered.

Rather than relating profiles to time, in this research it is more beneficial to relate the profiles to the total number of attributes counted. Therefore we define the size Z of a profile as the summation of the frequencies of the individual attributes, i.e.

$$Z = \sum_{i=1}^{n} f_i.$$

It should be noted that Z is not necessarily equal to the number of packets, since a single packet may affect several attributes.

We view an attack as a deliberate effort to exploit vulnerabilities of a specific protocol. Specific attacks will be denoted by A_i. We assume that each individual attack, such as a Syn scan, produces a distinct set of frequencies in a profile. This profile is distinct and differs noticeably from normal traffic profiles. An attack profile captured in a clean environment, e.g. an isolated network, is saved as a signature. Therefore, a signature of a specific A_i is defined as a vector $S_i = (f_1, f_2, ..., f_n)$. Since f_j represents frequencies of profiles P and S_i, we will use superscripts in order to differentiate the association of f_j, i.e. f_j^P and $f_j^{S_i}$ are the j^{th} frequencies of P and S_i respectively.

3.4 Attack Signature Generation

In creating the attack signatures, our goal was to capture each attack in its purest form without the presence of outside noise. Thus, we isolated two Linux Pentium III machines and set them up as attacker and victim systems. Two well-known attack suites, toast [7] and nmap [6] were used to create the attack signatures. Whereas the general environment in which signatures are collected is identical to that described in [12], it should be noted that the information selected, i.e. the signatures, are of a fundamentally different nature. Network traffic was captured in an off-line process by running tcpdump [11], a packet capture tool, on the victim machine. Attacks were selected from toast and nmap that targeted the TCP network protocol.

To create an attack signature, the attributes discussed in the previous subsection were measured. The size of the profile, Z, is important for later identification of attacks. If Z is chosen too large, the attack signature will be obscured by normal network traffic. On the other hand, if Z is chosen too small, a high false positive rate has been observed, due to the fact that the profile was not large enough to capture the characteristics of attacks.

Analysis of the attack network traffic for the attacks considered resulted in attribute counts of approximately 100 to 6000. Whereas Z is a tunable parameter, experience has shown that $Z = 100$ is a reasonable choice.

3.5 Statistical Comparison

If attacks were run under the ideal conditions that existed during the attack signature generation, then attack identification would be just a simple comparison between a signature S_i and the profile P. However, the host computer is connected to the world via a network that handles many types of normal traffic. As such, there is a need for a method to recognize attacks in such noisy environment. Specifically, we consider the proportions of attributes of the signature and the current profile, i.e. the relationship between f_i^S and f_i^P. We also require that the comparison method be efficient, so the comparison can be performed in real-time. We identified the Chi-square test [5] as a suitable statistical comparison technique. The Chi-square test measures the difference between proportions in two independent samples.

The signature and the profile can be laid out in a two-by-two contingency table to illustrate the concept. For our case, an example contingency table for the Syn flag attribute of the misfrag attack is presented in Figure 2. Recall that the Z is the size of the profile. A

	SYN packets	Other packets	
misfrag signature	A	B	A+B
current profile	C	D	C+D
	A+C	B+D	N

Figure 2. Two-by-Two Contingency Table

represents the frequency of Syn flags in the misfrag signature and $B = Z - A$. The second row of the figure is with reference to the current profile. C represents the frequency of Syn flags in the current profile and $D = Z - C$. Furthermore, $N = 2Z$ and is the combined count of the signature and the profile. The Chi-square test is computed by the following formula:

$$\chi^2(A,C) =$$

$$N(AD - BC)^2/(A+B)(C+D)(A+C)(B+D).$$

$\chi^2(A,C)$ is compared against a threshold value from a standard Chi-square distribution table with one degree of freedom. The degree of freedom is associated with the number of parameters that can vary in a statistical model. A significance level of 0.05 was selected. This means that 95% of the time we expect $\chi^2(A,C)$ to be less than or equal to 3.84 [5]. Chi-square values greater than 3.84 provide evidence of a real difference between the profile and the attack signature.

The Chi-square value is computed separately for each attribute of attack signatures. An attack can be recognized when for each attribute of its signature, the result of the Chi-square test is below 3.84. If any one attribute has a Chi-square value greater than 3.84, the attack does not match.

We can now formally state the above matching rules. Let ϵ be the Chi-square threshold value 3.84. Given P and S_k,

$$\forall i, 1 \leq i \leq n, \qquad \chi^2(f_i^{S_k}, f_i^P) \leq \epsilon$$

$$\Rightarrow A_k \ is\ possible$$

$$\exists i, 1 \leq i \leq n, \qquad \chi^2(f_i^{S_k}, f_i^P) > \epsilon$$

$$\Rightarrow A_k \ is\ not\ possible$$

Under our model, and with the realistic expectation of noise in detecting attack signatures in mixed traffic, false positives and false negatives are possible. A false positive for a single S_k is defined as the probability

$$Pr\left(\prod_{i=1}^{n} \chi^2(f_i^{S_k}, f_i^P) \leq \epsilon \mid no\ attack\right).$$

A false negative for A_k occurs when the data in the profile is distorted, and therefore S_k cannot be detected. For a specific S_k, this is defined as

$$Pr\left(\prod_{i=1}^{n} \chi^2(f_i^{S_k}, f_i^P) > \epsilon \mid A_k\right).$$

4. Experimental Results

4.1 Empirical Data

To test the effectiveness of the attack recognition method and identify potential false positives, network traffic with embedded attacks was observed. The experimental environment[1] consisted of concurrently running a web browser, telnet, and ftp session on the victim machine, while the machine was subjected to specific attacks. Embedded attacks included, nmapsS (a syn scan), gewse (a port flooder), syndrop (a fragment attack), and nmapsF (a Fin scan).

Table 1 shows the results of a typical attack recognition experiment. The first column indicates the profile P, which has the named attack embedded. Now the Chi-square values are shown for the Syn, Fin, Reset and Fragment attributes, as they were computed from profile P and attack signatures S_i, shown in the last column. Thus, the attribute entries show $\chi^2(f_j^{S_i}, f_j^P)$. As can be seen from Table 1, all of the embedded attacks were correctly identified. The Chi-square values were either zero or very small, far below the threshold value of 3.84. Of special interest is the last row of the table, where the profile did not contain an attack. It should be noted that among all attacks, gewse5 was the attack that came closest to a match. The high Chi-square value of 15 indicates the absence of the gewse5 attack signature. Actually, this value was the lowest Chi-square value observed within a large sample of profiles in the absence of attack.

[1]In the current implementation, the network data is analyzed off-line.

Profile P	Syn	Fin	Reset	Fragment	Signature S_i
nmapsS*	.02	.00	.02	.00	nmapsS
nmapsF*	.00	.37	.37	.00	nmapsS
syndrop*	.00	.00	.02	1.63	syndrop
gewse5*	.00	.00	.00	.00	gewse5
Normal	15	.00	.02	.00	gewse5

Table 1. Minimum Chi-square Values for Embedded Attacks and Normal Sessions

4.2 Discussion

Results from the initial testing show that the proposed method, based on generation of network attack signatures, shows promise. While more thorough testing is needed, preliminary results showed accurate attack identification when attacks were embedded in "noisy" traffic. So far, we have not managed to produce false positives in the laboratory environment, however, we have yet to conduct extensive field testing.

5. Conclusions

This research presents an efficient method for attack recognition based on attack signatures, which is characterized by its simplicity and its real-time potential. The method described has proven to be successful in recognizing specific attacks based on statistical analysis of TCP protocol attributes. The intent of the approach is to detect specific attacks that exploit network protocol vulnerabilities, and not to provide general intrusion detection. Preliminary testing was promising, in that attacks embedded in normal data were accurately identified with no false positives.

References

[1] J. Allen, A. Christie, W. Fithen, J. McHugh, J. Pickel and E. Stoner, State of the Practice of Intrusion Detection Technologies, Carnegie Mellon, SEI, *Technical Report CMU/SEI-99-TR-028*, ESC-99-028, January 2000.

[2] B. Baccala, Connected: An Internet Encyclopedia, www.freesoft.org/CIE/index.htm, April, 1997.

[3] M. Cooper, S. Northcutt, M. Fearnow, and K. Frederick, *Intrusion Signatures and Analysis* (New Riders, Indianapolis, IN, 2001).

[4] H. Debar, M. Dacier, and A. Wespi, Towards a taxonomy of Intrusion Detection Systems, *Computer Networks* Vol. 31, 1999, 805-822.

[5] G. Ferguson, *Statistical Analysis in Psychology and Education* (McGraw-Hill, New York, N.Y. 1981).

[6] Fyodor, *nmap*, www.insecure.org/nmap, 1999.

[7] Gridmark, *toast*, packetstorm.securify.com/DoS/indexsize.shmtl, March, 2000.

[8] N. Habra, B. Le Charlier, A. Mounji, and I. Mathieu, Asax: Software architecture and rule-based language for universal audit trail analysis, *2nd European Symp. on Res. in Comp. Sec.*, (ESORICS), Toulouse, Lecture Notes in CS, Vol. 648, Springer Verlag, Nov. 1992.

[9] S. Hofmeyr, and S. Forrest, Intrusion Detection Using Sequences of System Calls, *Journal of Computer Security*, Vol. 6, pp. 151-180, 1998.

[10] Hurwitz Group, *Hurwitz Report: Axent Technologies' NetProwler and Intruder Alert* (Hurwitz Group, Framingham, MA, Sept. 2000).

[11] V. Jacobson, C. Leres, and S. McCanne, *tcpdump*, LBNL, University of California, June 1997, ftp://ftp.ee.lbl.gov/tcpdump.tar.Z.

[12] A. Krings, W. Harrison, N. Hannebutte, C. Taylor and M. McQueen, Attack Recognition Based on Kernel Attack Signatures, *Proc.: 2001 International Symp. on Information Systems and Eng.*, (ISE'2001), Las Vegas, June 25-28, 2001.

[13] S. McClure, J. Scambray, and G. Kurtz, *Hacking Exposed*, (Osborne/McGraw-Hill, Berkeley, CA 1999).

[14] S. Northcutt, V. Irwin, and B. Ralph, *Shadow*, Naval Surface Warfare Center, Dahlgren Lab, 1998.

[15] S. Northcutt, *Network Intrusion Detection: An Analysts Handbook* (New Riders, Indianapolis, IN, 1999).

[16] V. Paxson, *Experiences Learned from Bro*, login; The Usenix Assoc. Magazine, Sept. 1999, 21-22

[17] P. Proctor. *The Practical Intrusion Detection Handbook* (Prentice-Hall, Upper Saddle River, N.J., 2001).

[18] M. Ranum, K. Landfield, M. Stolarchuk, M. Sienkiewicz, A. Lambeth, and E. Wall, Implementing a Generalized Tool for Network Monitoring, *Proc. of 11th Syst. Admin. Conf. (LISA 97)*, San Diego, CA, Oct. 1997

[19] M. Roesch, *Snort - Lightweight Intrusions Detection for Networks*, http://www.clark.net/ roesch/security.html

[20] T. Spyrou, and J. Darzentas, Intention Modeling: Approximating Computer User Intentions for Detection and Prediction of Intrusions, *Proc. Information System Security*, Samos, Greece, May 1996, pp. 319-335.