DATA MINING TECHNIQUES FOR SECURITY OF WEB SERVICES

Manu Malek and Fotios Harmantzis
Steven Institute of Technology, Castle Point on the Hudson, Hoboken, NJ 07030, USA

Keywords: Security services, Security attack, Denial of service, Intrusion detection, Security safeguards

Abstract: The Internet, while being increasingly used to provide services efficiently, poses a unique set of security issues due to its openness and ubiquity. We highlight the importance of security in web services and describe how data mining techniques can offer help. The anatomy of a specific security attack is described. We then survey some security intrusions detection techniques based on data mining and point out their shortcomings. Then we provide some novel data mining techniques to detect such attacks, and describe some safeguard against these attacks.

1 INTRODUCTION

Cyberspace is used extensively for commerce. For years banks and other financial organizations have conducted transactions over the Internet using various geographically dispersed computer systems. Businesses that accept transactions via the Internet can gain a competitive edge by reaching a worldwide customer base at relatively low cost. But the Internet poses a unique set of security issues due to its openness and ubiquity. Indeed, security is recognized as a critical issue in Information Technology today. Customers will submit information via the Web/Internet only if they are confident that their private information, such as credit card numbers, is secure. Therefore, today's Web/Internet-based services must include solutions that provide security as a primary component in their design and deployment.

Web services generally refer to web-based applications that make it possible for enterprises to do transactions on the web and for users to share documents and information with each other over the Web. The standard that makes it possible to describe the communications in some structured way is Web Services Definition Language (WSDL). WSDL is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information [1].

But openness and integration have their price. Without adequate security protections and effective security management, these features can be used to attack the availability and integrity of information systems and the networks connecting to them. Here we highlight a few typical ways an attacker may gain illegal access to an information system, or to make it unavailable to legitimate users. We identify the profiles or signatures for the sequence of actions an attacker may perform to perpetrate such attacks. We use data mining techniques to discover such attack profiles to detect the attacks.

Data mining refers to a technique to intelligently and automatically assist humans in analyzing the large volumes of data to identify valid, novel, and potentially useful patterns in data. It offers great promise in helping organizations uncover patterns hidden in their data that can be used to predict the behavior of customers, so that they can better plan products and processes. Data mining takes advantage of advances in the fields of artificial intelligence (AI) and statistics. Both disciplines help in pattern recognition and classification. Other disciplines used in data mining include rule-based and case-based reasoning, fuzzy logic, and neural networks. The techniques used in data mining include rule induction, clustering, projection, and visualization (e.g., see [2] for details).

This paper provides a glimpse at the cyberspace security situation, and offers some techniques to manage the security of web services. The paper

Malek M. and Harmantzis F. (2004).
DATA MINING TECHNIQUES FOR SECURITY OF WEB SERVICES.
In *Proceedings of the First International Conference on E-Business and Telecommunication Networks*, pages 61-67
DOI: 10.5220/0001382300610067

describes some security attacks, and provides some techniques to detect and defend against them. In Section 2, we present some statistics related to security attacks to highlight the urgency of the issue. Some typical security vulnerabilities and attacks are discussed in Section 3. In Section 4, we provide a survey of data mining applications in intrusion detection and point out their shortcomings. We then define attack signatures and outline how to use them in conjunction with data mining techniques for efficient intrusion detection. Section 5 summarizes the paper.

2 BACKGROUND

Based on data provided by CERT/CC, the number of incidents and vulnerabilities for cyber attacks have increased exponentially during the period 1998 to 2002 [3]. Figure 1 shows that intrusions were relatively few in the early1990s, but there has been a major increase since 2000. About 25,000 intrusions were reported in the Year 2000 [3]. Keep in mind that not all enterprises that suffer security breaches report them. The line moving upward in this figure shows various types of threats, starting with very simple ones in the early '90s, like password guessing. The sophistication of attacks increased with self-replicating codes, such as viruses, then password cracking (where the cryptographic password is broken), and on to the more sophisticated threats shown. Against this rising sophistication in threats, we have easy availability of hacking tools: hackers no longer have to be experts in computer science or security; they could use available tools. For example, a tool such as nmap [4] can be used to find all the open ports, a first stage in an attack. This combination of decreasing knowledge required of the attackers and the increasing sophistication of the attacks is giving rise to major security concerns.

According to the Federal Computer Incidence Response Center (FedCIRC), the incident handling entity for the federal government, 130,000 government sites totaling more than one million hosts were attacked in 1998 [5]. Also, a 1999 survey conducted by the Computer Security Institute (CSI) and the Federal Bureau of Investigation (FBI) revealed that 57% of organizations cited their Internet connections as a frequent point of attack, 30% detected actual network intrusion, and 26% reported theft of proprietary information [6]. A similar survey in 2002, showed that 90% of the 507 participating organizations detected computer security breaches within the past 12 months, 74% cited their Internet connections as a frequent point of attack, but only 34% reported security intrusions to law enforcement agencies. These numbers must be considered observing that they relate to only known attacks and vulnerabilities. However, they do indicate the magnitude of the problem.

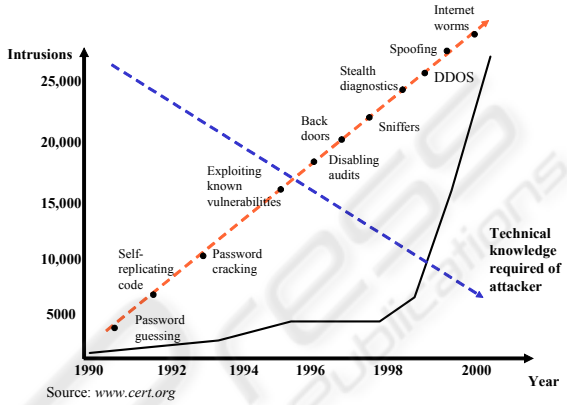

Figure 1: Security vulnerabilities and threats

A key to preventing security attacks is to understand and identify vulnerabilities, and to take corrective action. A threat to computing systems or communication network is a potential violation of security for unauthorized, illegitimate, malicious or fraudulent purposes. An attack is the implementation of a threat using the system vulnerabilities. Vulnerability is a weakness in the security system that might be exploited to launch an attack. Finally, a control is a protective measure – an action, device, procedure, or technique – that reduces vulnerabilities.

Table 1 sows the top 10 security vulnerabilities as reported periodically by The SANS Institute [7]. The reasons for the existence of these vulnerabilities include: buggy software design and development, system administrators being too busy to install security patches in a timely manner, and inadequate policies and procedures. Another factor is that due to the ubiquity of the Internet, vulnerabilities are quickly and widely published.

Table 1: Top 10 security vulnerabilities [7]

Vulnerabilities of Windows Systems	***Vulnerabilities of Unix Systems***
1. Internet Information Services (IIS)	*1. Remote Procedure Calls (RPC)*
2. Microsoft Data Access Components (MDAC) -- Remote Data Services	*2. Apache Web Server*
3. Microsoft SQL Server	*3. Secure Shell (SSH)*
4. NETBIOS -- Unprotected Windows Networking Shares	*4. Simple Network Management Protocol (SNMP)*
5. Anonymous Logon -- Null Sessions	*5. File Transfer Protocol (FTP)*
6. LAN Manager Authentication -- Weak LM Hashing	*6. R-Services -- Trust Relationships*
7. General Windows Authentication -- Accounts with No Passwords or Weak Passwords	*7. Line Printer Daemon (LPD)*
8. Internet Explorer	*8. Sendmail*
9. Remote Registry Access	*9. BIND/DNS*
10. Windows Scripting Host	*10. General Unix Authentication -- Accounts with No Passwords or Weak Passwords*

The motive of attackers could be anything from pure joy of hacking to financial benefit. The attackers are either highly technically capable, or they sometimes breaks into the network by trial and error. Disgruntled employees have more access rights to enterprise computer networks compared to outside attackers. According to the CSI/FBI 2002 Survey [6], 60% of attacks in the US were inside attacks (attacks that originated inside the institutions) and 40% were outside attacks.

3 SOME TYPICAL SECURITY ATTACKS

As mentioned, a security attack occurs when an attacker takes advantage of one or more security vulnerabilities. To improve security, one needs to minimize security vulnerabilities. In this section we present some typical security attacks, point out the vulnerabilities abused to perpetrate the attacks. Some safeguards against these attacks will be described in the next section. An attack that we deal with specifically in this section and in Section 4 is the HTTP GET attack.

3.1 Denial-of-Service Attack

In a Denial-of-Service (DoS) attack, the attacker attempts to use up all the victim system's resources like memory or bandwidth. When the attack is successful, legitimate users can no longer access the resources and the services offered by the server will be shut down. According to the 2002 CSI/FBI survey [6], 40% of all attacks are DoS attacks.

An attack can be directed at an operating system or at the network. The attacker may send specially crafted packets that crash remote software/services running on the victim server. It will be successful if the network is unable to distinguish between legitimate traffic and malicious or bogus traffic. Some common DoS attacks follow.

ICMP Flooding and Smurf Attack
These are both ICMP-based attacks. Flooding with ICMP packets slows down the victim server so that it can no longer respond quickly enough for the services to work properly. If packets are sent with forged IP addresses, the victim server not only has to allocate system resources to receive, but to reply to packets to addresses which do not exist. The Smurf attack uses a similar idea: the attacking machine sends Echo requests with broadcast IP addresses, thus not only the victim server but the attached network will be flooded by a large amount of ICMP traffic.

SYN Flooding
SYN flooding exploits the weakness of the TCP Three-way Handshake [8]. In a normal TCP connection request, the source sends a SYN (synchronization) packet to the destination to initiate the connection; then waits for a SYN ACK (synchronization acknowledged) packet from the destination. The connection is established when the destination receives a FIN ACK (finishing acknowledged) packet from the source. In the SYN flooding attack, the attacker sends a large number of SYN packets, often from bogus IP addresses, to the victim server, which adds the entry to the connection queue and replies with SYN ACKs. As the source addresses are incorrect or non-existent, FIN ACKs will never be received by the victim server, so the last part of the Three-way Handshake never completes and the connection queue of the victim server fills up.

Badly-formed Packets
In this type of attack, the attacker sends badly-formed IP packets, e.g., packets that consist of invalid fragments, protocol, packet size, or header values, to the victim server. Once the destination TCP stack receives such invalid packets, the operating system must allocate resources to handle them. If the operating system cannot handle the misbehavior, it will crash. An example of this is the Ping-of-Death attack [9] which causes buffer overflow in the operating system. In this attack, the attacker sends a larger than standard ICMP (Internet Control Message Protocol) packet, such as a ping, in fragments to the target server. Since the allowed maximum size of such a packet is 65,535 bytes, the server allows a corresponding buffer space to collect the fragments. A clever attacker may create a ping with many fragments destined to a target server. The server receives the fragments and starts to reassemble them. When reassembled, the buffer will overflow, leading to program termination, overwriting other data or executable code, kernel dump, etc. More than 50% of attacks on servers are due to buffer overflow [3].

Distributed Denial of Service Attack
With the speed and power of computing resources today, an attacker may not be able to simply use one computer to craft a DoS attack. In the Distributed Denial of Service (DDoS) attack, many computers may be hijacked by the attacker as agents (zombies) to simultaneously flood a victim system's resources. A typical way to recruit zombie computers is for the perpetrator to send viruses to multiple computers, or to break into computer systems and load them with DDoS programs. Each infected system then finds other vulnerable systems and loads them with the programs, etc. The perpetrator uses the first system that was overtaken to instruct all the other compromised systems to launch the attack simultaneously.

3.2 HTTP GET Attack

For many web applications, a client should be able to send information to the server. HTML 2.0 and later versions support the Form element within an HTML document to allow data to be sent to web servers [10]. One of the attributes of Form is Method which indicates how data is submitted to the web server. Valid choices for the Method attribute are GET and POST. In METHOD=GET, the values inputted by the user are concatenated with the URL, separated by a special character (usually ?); fields are separated by &; space is represented by +. For example, the following URL:

http://www.gadgets.com?customer=**John+Doe**&address=
101+Main+Street&cardno=**1234567890**&cc=**visacard**
indicates that the customer's name and address with the customer's credit card number are to be sent to the web server at www.gadgets.com. A savvy user (attacker) may be able to use this feature to get access to proprietary information if appropriate security mechanisms are not in place. The following scenario, adopted from Ref. [11], is an HTTP GET attack on a typical web server which has some vulnerabilities.
The server http://www.acme.com runs Apache 1.3.12 on a Linux operating system. Firewalls prevent all but HTTP traffic via ports 80 (HTTP default port) and 443 (SSL port). Perl CGI scripts are used for the online store. A visitor to this site first begins browsing through the www.acme.com site, viewing the site's main page and a few images on it. The visitor notices that for the last selection (viewing the picture of a sunset), the URL in the browser window shows:
http://www.acme.com/index.cgi?page=sunset.html.
Following this pattern, the visitor (now attacker) issues a request for index.cgi by typing the following URL:
http://www.acme.com/index.cgi?page=index.cgi
Now, if the program does not validate the parameters passed to the index.cgi script, the filename passed as a parameter from the URL is captured by the CGI script, appended to the absolute path, and causes to open the index.cgi script as requested. Consequently, the browser display shows the source code of the index.cgi script!

At this point the attacker realizes that this technique can be further exploited to retrieve arbitrary files from the server. So the attacker may send the following request through the browser:
http://www.acme.com/index.cgi?page=/../../etc/passwd
If permissions are not set properly on the /etc/passwd file, its contents will be displayed by the browser, providing the attacker with user's password information. The attacker could now execute arbitrary commands on the server, for example, by sending
http://www.acme.com/index.cgi?page=|ls+-la/%0aid%0awhich+xterm|
(% plus the hex character 0a indicates line feed) requesting
ls –al (to show a file list of the server's root directory)
id (the effective user id of the process running index.cgi)

which xterm (path to the xterm binary code, to gain interactive shell access to the server
and the attacker could gain full interactive shell-level access to the web server.

Note again that the vulnerabilities: the program does not validate the parameters passed to the index.cgi script, and permissions are not set properly on the /etc/passwd file.

4 DATA MINING TECHNIQUES FOR INTRUSION DETECTION

In this section we review server logs, introduce attack signatures, and present our main contribution: how security attack signatures are used in conjunction with data mining to detect security intrusions. More specifically, we first describe the relevance and importance of the different log files that are available; we then define specific patterns in the log files for an attack (the individual log records as well as their sequence/order) as the attack signature; and use data mining to search and find such patters for attack detection. The efficiency and speed of the overall process can even lead to attack prediction capabilities.

4.1 Logs

Every visit to a Web site by a user creates a record of what happens during that session in the server's log. A busy site may generate thousands of log entries per hour, compiled in various log files. A log file entry contains items like the IP address of the computer requesting the Web page, the date and time of the request, the name and the size of the file requested. Logs vary by the type of server and the file format. Following are some typical logs and what they record:

- Access Log records every transaction between server and browsers (date, time, domain name or IP address, size of transaction, …).
- Referrer Log records the visitor's path to the site (the initial URL from which the visitor came).
- Agent Log records the type and version of the browser.

For secure systems, the standard logs and directories may not be sufficient and one must employ additional logging tools, e.g., information about which computer is connecting to which services on the system. There are many programs under the heading of IP loggers available for this purpose, e.g., EnviroMon [12] and ippl [13].

4.2 Mining Logs

The data available in log files can be "mined" to gain useful information. Data mining offers promise in uncovering hidden patterns in the data that can be used to predict the behavior of (malicious) users. Using data mining in intrusion detection is a relatively new concept. In [14], the authors outline a data mining framework for constructing intrusion detection models. The central idea is to utilize auditing programs to extract an extensive set of features that describe each network connection or host session, and apply data mining to learn rules that capture the behavior of intrusions and normal activities. Detection models for new intrusions are incorporated into an Intrusion Detection Systems (IDS) through a meta-learning (or co-operative) learning process. The strength of this approach is in classification, meta-learning, and association rules.

In [15], the authors present an intrusion detection tool aimed at protecting servers. However, their method does not effectively handle all matches of the signature (e.g., of the 404 type: document not found). Attacks that have no matching signature and are sent by a previously unknown host may be missed.

Ref. [16] represents a first attempt to analyze sequences of system calls issued by a process for intrusion detection. The authors introduce a method based on sequences of Unix system calls at the process level for anomaly detection resulting in intrusion detection. They address sendmail, lpr, and ftpd processes and obtain some good results in terms of false-positives. Ref. [17] also uses sequences of system calls for intrusion detection. The authors choose to monitor behavior at the level of privileged processes. . Their proposed approach of detecting irregularities in the behavior of privileged programs is to regard the program as a black-box, which, when it runs, emits some observable behavior. Privileged processes are trusted to access only relevant system resources, but in cases where there is a programming error in the code that the privileged process is running, or if the privileged process is incorrectly configured, an ordinary user may be able to gain super-user privileges by exploiting the problem in the process. The system-call method, however, is specific to processes and cannot detect generic intrusion attempts, e.g., race condition attacks, session hijacking (when one user masquerades as another), and cases in which a user violates policy without using privileged processes.

Artificial intelligence techniques have also been applied to help in decision making for intrusion detection. In [18], the author presents a survey of such methods and provides an example of using feature selection to improve the classification of network connections. In [19], the authors present a comparison of some neural-network-based method and offer some "classifiers" for anomaly detection in Unix processes. All the techniques based on artificial intelligence, however, suffer from lack of scalability: they work only for small size networks and data sizes.

4.3 Attack Signatures

We use attack signatures in combination with data mining to not only detect, but predict attacks. An attack signature encapsulates the way an attacker would navigate through the resources and the actions the attacker would take. For example, in a denial-of-service attack, the attacker may send a large number of almost simultaneous TCP connect requests from one or more IP addresses without responding to server acknowledgements.

To illustrate a specific attack signature, let us look at the log lines stored by the web server in the HTTP GET attack example described in the previous section. The log line in the Access Log corresponding to the visitor's (attacker's) first attempt is

A. 10.0.1.21 – [31/Oct/2001:03:02:47] "GET / HTTP/1.0" 200 3008

where 10.0.1.21 is the visitor's IP address, followed by date and time of visit, the Method and the Protocol used. The number 200 indicates the "normal" code, and 3008 indicates the byte size of the file retrieved. The following log line corresponds to the visitor's selection of the sunset picture:

B. 10.0.1.21 – [31/Oct/2001:03:03:18] "GET / sunset.jpg HTTP/1.0" 200 36580

and the following log line corresponds to the visitor's first attempt at surveillance of the site (issuing a request for index.cgi):

C. 10.0.1.21 – [31/Oct/2001:03:05:31] "GET / index.cgi?page= index.cgi HTTP/1.0" 200 358

The following log lines correspond to the visitor (by now, attacker) attempting to open supposedly secure files:

D. 10.0.1.21 – [31/Oct/2001:03:06:21] "GET / index.cgi?page=/../../etc/passwd HTTP/1.0" 200 723

E. 10.0.1.21 – [31/Oct/2001:03:07:01] "GET / index.cgi?page=|ls+- la+/%0aid%0awhich+xterm| HTTP/1.0" 200 1228

This pattern of log lines from the same source IP address can be recognized as a signature of an HTTP GET attack. In the above example, the sequence of log lines A-B-C-D-E, A-C-D-E, B-C-D-E, or C-D-E constitutes the signature of this HTTP GET attack. Even some individual log lines from a source IP address could provide tell-tale signs of an impending HTTP GET attack. For example, the existence of a "pipe" (i.e., |) in the URL, as in log line E above, would indicate that the user is possibly trying to execute operating system commands.

In our research, we try to establish signatures for various types of attacks. Note the importance of good comprehensive attack signatures in detecting attacks. Incomplete signatures result in false-positive or false-negative detection. Another point is the use of data mining to detect attack signatures. One can imagine the tremendous amount of data collected by web services, resulting in multi-tera-byte databases. With such large amounts of data to analyze, data mining could become quite computationally expensive. Therefore, efficiency becomes a major issue. Currently, we are continuing our efforts to identify ways data should be efficiently analyzed in order to provide accurate and effective results.

In our research, we use the Rule Induction Kit (RIK) and Enterprise Data-Miner (EDM) tools [20] to detect and mine attack signatures. The RIK package discovers highly compact decision rules from data, while the EDM software kits implement the data-mining techniques presented in [21] and includes programs for (a) data preparation (b) data reduction or sampling, and (c) prediction. Our selection of this tool package was based on criteria related to efficiency (speed, especially when it comes to large amounts of data, as is the case with log files), and portability (multiple platforms), as well as extensibility (where the user can compose new methods with the existing building blocks). Based on this software platform, we are able to create a sophisticated data mining methodology for efficient intrusion detection.

4.4 Security Safeguards

Safeguards are applied to reduce security risk to an acceptable/desirable level. They may be Proactive to prevent security incidents, or Reactive, to protect information when an incidence is detected. In either case, they must be cost effective, difficult to bypass, and with minimal impact on operations. Examples of safeguards are: avoidance (keeping security incidents from occurring, e.g., by removing vulnerabilities), limiting access (e.g., by reducing the number of entry points where attacks may

originate), transference (shifting risk to someone else, e.g., via insurance or outsourcing), and mitigation (minimizing the impact of an incidence, e.g., by reducing its scope or improving detection).

One of the major safeguards is to detect and reduce/remove vulnerabilities. The main reasons for existing vulnerabilities are buggy software design and development, or system administration problems. Existence of bugs in software are due to

- programming for security not being generally taught,
- good software engineering processes not being universal, as well as
- existence of legacy code.

The system administration problems are due to inadequate policies and procedures, or the system administrators being too busy with many machines to administer, too many platforms and applications to support, and too many updates and patches to apply.

For the attack examples given in the previous section, we can offer some rather simple safeguards. For attacks that are based on making multiple requests and ignoring the server acknowledgments, such as ICMP Flood and Smurf Attack, and SYN Flooding, one could employ a timer: if the response does not arrive within a reasonable time, the request could be dropped and the resources freed. For attacks that are based on buffer overflow, one could use operating systems written in "safe" languages that perform range checking (like Java). The HTTP GET attack could be prevented by making sure that programs validate the parameters passed to them, and that file permissions are set properly.

5 CONCLUSIONS

We have first set the stage emphasizing the magnitude of the security problem, raising awareness and focusing on the impact of security. We have detailed two attacks: Denial of Service and the HTTP GET attack, and defined the signature of the latter. The application of data mining techniques for detecting attacks was described. The novelty of our approach is in determining the relevance/importance of different log records, defining intelligent signatures, and using efficient data mining techniques. Preliminary results have been encouraging.

There is significant work still to be done, e.g., improving the effectiveness of attack signatures, developing distributed algorithms for detection/prediction, and improving the efficiency of pattern searching. We are currently working on these issues.

REFERENCES

[1]. http://www.w3.org/TR/wsdl

[2]. Michael J. A. Berry and Gordon Linoff, Data Mining Techniques, Wiley Computer Publishing, 1997

[3]. Computer Emergency Response Team/Coordination Center (CERT/CC) at Carnegie Mellon University's Software Engineering Institute, http://www.cert.org/

[4].www.insecure.org/nmap/nmap-fingerprinting-article.html

[5]. NIST ITL Bulletin, "Computer attacks: what they are and how to defend against them," May 1999.

[6]. CSI, "2002 CSI/FBI Computer Crime and Security Survey," http://www.gocsi.com/.

[7]. The SANS Institute (http://www.sans.org/top20/), May 2003

[8]. Douglas Comer, Internetworking with TCP/IP Vol.1: Principles, Protocols, and Architecture (4th Edition), Prentice Hall, 2000

[9]. www.insecure.org/sploits/ping-o-death.html

[10]. www.w3c.org?

[11]. S. McClure, S. Shah, and S. Shah, Web Hacking: Attacks and Defenses, Addison Wesley, 2003

[12]. http://www.interwld.com/pico/subs/pico_Environ_IP_Logging.htm

[13]. http://packages.debian.org/unstable/net/ippl.html

[14]. W. Lee and S. J. Stolfo, "Data Mining Approaches for Intrusion Detection," Usenix Security Symposium, San Antonio, Texas, July 1998

[15]. Magnus Almgren, Herve Deba, and Marc Dacier, "A Lightweight Tool for Detecting Web Server Attacks," http://www.ce.chalmers.se/~almgren/Publications/almgren-ndss00.pdf

[16]. S. Forrest, S. A. Hofmeyr, A. Somayaji, and T. A. Longstaff, "A Sense of Self for Unix Processes," Proc. 1996 IEEE Symp. Security and Privacy, Los Alamitos, CA, pp. 120-128, 1996

[17]. S. A. Hofmeyr, A. Somayaji, and S. Forrest, "Intrusion Detection using Sequences of System Calls," Journal of Computer Security Vol. 6, pp. 151-180, 1998.

[18]. Jeremy Frank, "Artificial Intelligence and Intrusion Detection: Current and Future Directions," June 1994 (http://citeseer.nj.nec.com/frank94artificial.html)

[19]. Zhen Liu, German Florez, and Susan Bridges, "A Comparison of Input Representation in Neural Networks: A Case Study in Intrusion Detection," Proc. International Joint Conference on Neural Networks, May 12-17, 2002, Honolulu, Hawaii.

[20]. http://www.data-miner.com

[21]. S. Weiss and N. Indurkhya, Predictive Data Mining: A Practical Guide, Morgan Kaufmann, 1997.