Available at: www.researchpublications.org

Computer Attacks and Intrusion Detection System: A Need Review

Mr. Shailesh P. Thakare[#1], Ms. Prerana Chandurkar[*2] Ms. Maithili S. Deshmukh*[3]

[#]*Department of Information Technology, P.R.M.I.T. & R, Badnera, M.S, India*

[1]rohit7277@yahoo.com

[*]*Lecturer, Department of B.C.A., Mahatma Jyotiba Fule Mahavidyalaya, Amravati, M.S., India.*

[2]Prechandurkar24@gmail.com

**Assistant Professor Department of Information Technology, P.R.M.I.T & R, Badnera, M.S., India*

[3]msdeshmukh@mitra.ac.in

Abstract: **With recent advances in network based technology and increased dependability of everyday life on this technology, assuring reliable operation of network based systems is very important. During recent years, number of attacks on networks has dramatically increased and consequently interest in network intrusion detection has increased among the researchers. This paper provides a review on different attacks used in DARPA 1998 Intrusion Detection Evaluation, the first standard corpus for evaluating computer intrusion detection systems. Intrusion detection system (IDS) is increasingly becoming an important tool to secure the network. Many IDSs are unable to detect novel attacks because they are designed on limited environment to restricted applications. Many researchers proposed approaches which are able to achieve good detection accuracy for old attacks but poor detection performance for new attacks In this paper we have presented information about different attacks , Taxonomy of attacks, Attack Scenarios etc. Which will helpful for readers in their work?**

Keywords—Taxonomy of Computer Attacks, 1998 DARPA Evaluation, Intrusion, DOS Attack, Novel attacks, Attack Scenarios.

1. INTRODUCTION

In order to design security monitoring surveillance system, it is necessary to understand the types of threats and attacks that can be mounted against the computer system, and how these threats may manifest themselves in audit data. It is also important to understand the threats and their sources from viewpoint of identifying other data sources by which the threats may be recognized. Use of firewalls and frequent software updates cannot prevent attacks while fixing security holes is a vital part of maintaining a system, an administrator needs to keep a close eye on the system, including monitoring log for abnormal behaviour. A useful tool that can help automate this task is an Intrusion Detection System (IDS). The 1998 DARPA intrusion detection evaluation created the first standard corpus for evaluating computer intrusion detection systems. An important goal of the 1999 DARPA Intrusion Detection Evaluation was to promote the development of intrusion detection systems that can detect new attacks. The focus of this paper is on the attacks that were developed for use in the 1998 DARPA intrusion detection evaluation. In all, over 300 attacks were included in the 9 weeks of data collected for the evaluation. These 300 attacks were drawn from 32 different attack types and 7 different attack scenarios. The attack types covered the different classes of computer attacks and included older, well-known attacks, newer attacks that have recently been released to publicly available forums, and some novel attacks developed specifically for this evaluation. In some attacks, the attacker breaks into a computer system just for fun, while in others the attacker is interested in collecting confidential information or causing damage. In addition to providing detailed descriptions of each attack type, this paper also describes the methods of stealthiness and the attack scenarios.

2. OVERVIEW OF COMPUTER ATTACKS

A computer attack is any malicious activity directed at a computer system or the services it provides. Some Examples of computer attacks are viruses, use of a system by an unauthorized person, denial-of-service, probing, and physical attack against computer hardware. Possible types of computer attacks were included in the 1998 DARPA intrusion detection system evaluation, as follows:

(1) Attacks that allow an intruder to operate on a system with more privileges than are allowed by the system security policy.
(2) Attacks that deny someone else access to some service that a system provides.
(3) Attempts to probe a system to find potential weaknesses.

Some ways by which attacker can either gain access to a system or deny legitimate access by others, as follows.

Social Engineering: An attacker can gain access to a system by fooling an authorized user into providing information that can be used to break into a system.
For example, an attacker can call an individual on the telephone impersonating a network administrator in an attempt to convince the individual to reveal confidential information (passwords, file names, details about security policies). Or an attacker can deliver a piece of software to a user of a system which is actually a trojan horse containing malicious code that gives the attacker system access.
Implementation Bug: Bugs in trusted programs can be exploited by an attacker to gain unauthorized access to a computer system. Specific examples of implementation bugs are buffer overflows, race conditions, and mishandled of temporary files.
Abuse of Feature: There are legitimate actions that one can perform, when taken to the extreme can lead to system failure.
For example, include opening hundreds of telnet connections to a machine to fill its process table, or filling up a mail spool with junk e-mail.
System Misconfiguration: An attacker can gain access because of an error in the configuration of a system.

For example, the default configuration of some systems includes a "guest" account that is not protected with a password.

Masquerading: In some cases it is possible to fool a system into giving access by misrepresenting oneself.

An example is sending a TCP packet that has a forged source address that makes the packet appear to come from a trusted host. [1]

2.1 Intrusion Detection Systems

Intrusion detection systems gather information from a computer or network of computers and attempt to detect intruders or system abuse. Generally, an intrusion detection system will notify a human analyst of a possible intrusion and take no further action, but some newer systems take active steps to stop an intruder at the time of detection. Although there are many possible sources of data an intrusion detection system can use, three types of data were provided to participants in the 1998 Lincoln Laboratory intrusion detection evaluation. Most intrusion detection systems in existence today use one or more of these three types of data.

The first of these data sources is traffic sent over the network. All data that is transmitted over an Ethernet network is visible to any machine that is present on the local network segment. Because this data is visible to every machine on the network, one machine connected to this Ethernet can be used to monitor traffic for all the hosts on the network. During the DARPA evaluation, network traffic was sniffed using a single machine running the tcpdump program [39] to save the network traffic.

A second source of data for an intrusion detection system is system-level audit data. Most operating systems offer some level of auditing of operating system events. The amount of data that is collected could be as limited as logging failed attempts to log in, or as verbose as logging every system call. Basic Security Module (BSM) data from a Solaris victim machine was collected and distributed as part of the DARPA evaluation data.

A third source of data distributed to the evaluation participants was information about file system state. Daily file system dumps were collected from each of the machines used in the simulation. An intrusion detection system that examines this file system data can alert an administrator whenever a system binary file (such as the ps, login, or ls program) is modified. Normal users have no legitimate reason to alter these files, so a change to a system binary file indicates that the system has been compromised.

Although there are many other potential sources of data that can be used by an intrusion detection system to find attacks (such as real-time process lists, log files, processor loads, etc.), these three sources (sniffed network traffic, host-level audit files, and file-system state) were provided to participants in the 1998 Lincoln Laboratory DARPA intrusion detection evaluation because they were determined to be the sources most commonly used by the evaluation participants. After the three types of data were collected and aggregated, the data was distributed to participants via CD-ROM. Once participants obtained this data, each group used its particular intrusion detection system to the find intrusions and abuses that were inserted into the collected traffic. Although the 1998 DARPA evaluation tested only the ability to find attacks offline, some intrusion detection systems can evaluate data in real-time, allowing administrators (or the system itself) to take defensive action against the intruder. [2][3][12]

2.2 Strategies for Intrusion Detection

The different approaches that have been pursued to develop intrusion detection systems are described in many papers, including [7][8][10][11]. Figure 1 shows four major approaches to intrusion detection and the different characteristics of these approaches.

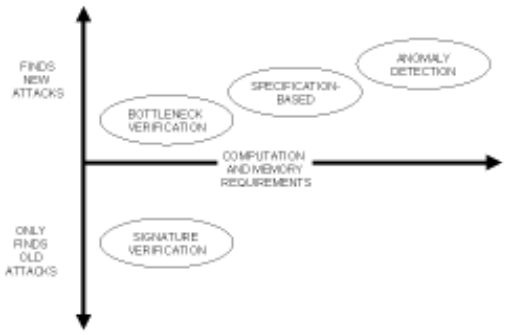

Figure1: Approaches to Intrusion Detection

The lower part of this figure shows approaches that detect only known attacks, while the upper part shows approaches that detect novel attacks. Simpler approaches are shown on the left and approaches that are both computationally more complex and have greater memory requirements are shown towards the right.[4]

2.3 Simulation Network(1998 DARPA Intrusion Detection System Evaluation)

The goal of the 1998 DARPA Intrusion Detection System Evaluation was to collect and distribute the first standard corpus for evaluation of intrusion detection systems. This corpus was designed to evaluate both false alarm rates and detection rates of intrusion detection systems using many types of both known and novel attacks embedded in a large amount of normal background traffic. One roadblock that has discouraged the creation of such a corpus is the reluctance of companies and government agencies to release data collected from operational computer networks. Data collected from an operational computer network is optimal for the evaluation of intrusion detection systems, but this data may contain personal or sensitive information that could not be released to the many parties who conduct intrusion detection research. For this reason, all data in the 1998 DARPA Intrusion Detection System evaluation was synthesized and recorded on a network which *simulated* an operational network connected to the Internet [5][12][13].

2.4 Exploits

A large sample of actual computer attacks is needed to test an intrusion detection system. These attacks should cover the different classes of attack types and contain exploits for both newly discovered as well as older well-known vulnerabilities. An attack instance might consist of several phases. For example, an attacker might copy a program onto a system, run this program that exploits a system vulnerability to gain root privileges, and then use this root privilege to install a backdoor into the system for later access. Each new exploit has a period of time during which it is most dangerous. Some of these older attacks were included in the set of attacks used for the 1998 DARPA evaluation. Intrusion detection systems were generally able to find these older, well-known attacks.

2.5 Sources

Many of the exploits developed for the 1998 DARPA evaluation were drawn from ideas or implementations available from public sources on the Internet. Rootshell is a web-site dedicated to

collecting computer exploits and has a sizeable archive of attacks for many popular operating systems. The "Bugtraq" mailing list also frequently hosts "exploit code"—ostensibly released for the purpose of testing one's own system for vulnerability—when a new vulnerability is discussed. A searchable archive of the Bugtraq mailing list can be found on the Internet at http://www.geek-girl.com. Other exploits were created from information released by computer security groups such as CERT and ISS X-force that frequently release information about new vulnerabilities. Additional sources of information about system vulnerabilities and possible exploits were vendor-initiated bulletins posted by operating system vendors like Sun Microsystems and Redhat Software. These bulletins are released to customers to encourage them to download patches that eliminate a new vulnerability. New and novel exploits were also created specifically for the purpose of the evaluation. These new exploits are useful for determining how well an intrusion detection system works against novel attacks that were not publicly known at the time the intrusion detection system was developed.[2][4]

3. TAXONOMY FOR COMPUTER ATTACKS

Taxonomy for classifying computer attacks was used to choose exploits for the evaluation. A good taxonomy makes it possible to classify attacks into groups that share common properties. Once these groups have been identified, the job of adequately testing an intrusion detection system becomes easier because instead of developing every possible attack we can choose a representative subset from each group. The taxonomy presented here was originally presented in [64]. The features of this taxonomy are:

- ❖ Each attack can be reliably placed in one category.
- ❖ All possible intrusions have a place in the taxonomy
- ❖ The taxonomy can be extended in the future.

This taxonomy was created for the express purpose of testing and evaluating intrusion detection systems. Within the taxonomy, each attack can be categorized as one of the following:

- A user performs some action at one level of privilege
- A user makes an unauthorized transition from a lower privilege level to a higher privilege level
- A user stays at the same privilege level, but performs some action at a higher level of privilege.

The taxonomy requires A**.** Way of describing each privilege level B**.** Way to describe transitions C**.** Way of categorizing actions.

3.1 Privilege Levels

The taxonomy defines an approach to Categorizing levels of privilege. The privilege categories that are applied in this paper are:

- R- *Remote network access:* "Remote network access" level refers to having privilege at the, via an interconnected network of systems, minimal network access to a target system.
- L -*Local network access:* "Local network access" represents the ability to read from and write to the local network that the target machine uses.
- U -*User access:* "User access" refers to the ability to run normal user commands on a system.
- S -*Root/Super-user access:* "Root/Superuser access" gives a user total software control of a system.
- P -*Physical Access to Host:* "Physical Access to Host" allows the operator to physically manipulate characteristics of the system (i.e. remove disk drives, insert floppy disks, and turn the system off).

This list represents a few possible access levels among available all access levels, but these were the most useful categories for describing the attacks in the 1998 DARPA intrusion detection evaluation.

3.2 Methods of Transition or Exploitation

An attacker needs to exploit some failure of a security framework in order to perform an attack. The five methods of transition that were explored for the 1998 DARPA evaluation and the single letters (**m**, **a**, **b**, **c**, **s**) used to represent the methods were:

- **M-Masquerading**: In some cases it is possible to fool a system into giving access by misrepresenting oneself.
- Examples of masquerading include using a stolen username/password or sending a TCP packet with a forged source address.

- **A-Abuse of Feature**: There are legitimate actions that one can perform, or is even expected to perform, that when taken to the extreme can lead to system failure.
- Example include filling up a disk partition with user files or starting hundreds of telnet connections to a host to fill its process table.

- **B-Implementation Bug:** A bug in a trusted program might allow an attack to proceed. Specific examples include buffer overflows and race conditions.

- **C-System Misconfiguration:** An attacker can exploit errors in security policy configuration that allows the attacker to operate at a higher level of privilege than intended.

- **S-Social Engineering**: An attacker may be able to coerce a human operator of a computer system into giving the attacker access. An individual attack may use more than one of these methods.
  For example, a bug in the implementation of the TCP stack on some systems makes it possible to crash the system by sending it a carefully constructed malformed TCP packet. This packet may also have the source address forged so as to avoid identification of the attacker. Such an attack would be exploiting both **m**asquerading and an implementation **b**ug, and it would be possible to detect the intrusion by noting either of these features.

3.2.1 Transitions between Privilege Levels

To show a transition between two privilege levels the strings for the two levels are written adjacent to one another with the method of transition between them. Two examples are shown in the following table:

Attack	String	Description
Format	U-b-S	User exploits a bug in the format program to become root/superuser
Ftp-write	R-c-U	A user with remote network access exploits a badly configured anonymous ftp server to gain local user

		access

Table1

3.3 Actions

There are many actions that can occur as part of a computer attack. Within the taxonomy, actions are represented with a string that represents a category, and a specification string that describes the specific action taken.

For example, the string **Probe (Users)** represents some action taken by an attacker to gather information about the users of a system.

Probes are actions taken by an attacker to gather information about one or more machine. Probes are represented within the taxonomy by the category label of "**Probe"**. The specific types of probes used in the DARPA evaluation were: (1) Probing a network to see how many and what types of machines are on that network **(Probe (Machines)**), (2) Probing a system to see what services the system supports (**Probe (Services)**), and (3) Probing a system to find out information about user accounts on that system (**Probe (Users)**).

Denial of service attacks are attempts to interrupt or degrade a service that a system provides. These attacks are represented within the taxonomy by the category label "**Deny"**. The classes of denial of service attacks used in the DARPA evaluation were: (1) Temporary denial of service with automatic recovery (**Deny(Temporary)**), (2)Denial of service requiring administrative action for recovery (**Deny(Administrative)**), and (3) Permanent denial of service with total system reconstruction required for recovery (**Deny(Permanent)).**

Another category of attacker actions is the interception of data. Interception of data is represented within the taxonomy by the category label "**Intercept**". The types of data interception actions used in the 1998 DARPA evaluation were: (1) Interception/Reading of files on a file system (**Intercept(Files)**), and (2) Interception of packets on a network (**Intercept(Network)**).

The following paragraphs describe the five categories of actions that were used to describe the actions taken during the 1998 DARPA intrusion detection evaluation.

An additional category of action is the alteration or creation of data on a system or network. Actions that involve data alteration or creation are represented with the category label "**Alter"**. The types of data alteration used in the 1998 DARPA evaluation were: (1) Alteration of data stored on a system, such as a password file or any other file (**Alter(Data)**), and (2) Removal of hints of an intrusion, such as entries in log files (**Alter(Intrusion-Traces)**).

The final category of attacker action described in the taxonomy is "**use**" of a system. Any use of the system that does not fall into the categories described above can be placed in the category represented by the category label "**Use**". The specific ways in which an attacker might use a system that were included in the 1998 DARPA evaluation were: (1) Use of the system by the intruder for enjoyment or recreational purposes such as playing games or bragging on IRC (**Use(Recreational)**), and (2) Use of a system as a staging ground or entry point for attacks on other systems (**(Use(Intrusion-Related)**).

The action categories and specifications described above paragraphs are summarized in Table 4-2. Each row of this table represents a specific type of action within a category. These specific actions have been grouped according to the categories presented above. The first column of the table is the action category, the second column is the string that represents the action in the taxonomy, and the third column in each row is a description of that particular type of action.[1][2][3]

Category	Specific Type	Description
Probe	Probe(Machines)	Determine types and numbers of machines on a network
	Probe(Services)	Determine the services a particular system supports
	Probe(Users)	Determine the names or other information about users with accounts on a given system
Deny	Deny(Temporary)	Temporary Denial of Service with automatic recovery
	Deny(Administrative)	Denial of Service requiring administrative intervention
	Deny(Permanent)	Permanent alteration of a system such that a particular service is no longer available
Intercept	Intercept(Files)	Intercept files on a system
	Intercept(Network)	Intercept traffic on a network
	Intercept(Keystrokes)	Intercept keystrokes pressed by a user
Alter	Alter(Data)	Alteration of stored data
	Alter(Intrusion-Traces)	Removal of hint of an intrusion, such as entries in log files
Use	Use(Recreational)	Use of the system for enjoyment, such as playing games or bragging on IRC
	Use(Intrusion-Related)	Use of the system as a staging area/entry point for future attacks

Table2: Summary of Possible Types of Actions

4. USING THE TAXONOMY TO DESCRIBE ATTACKS

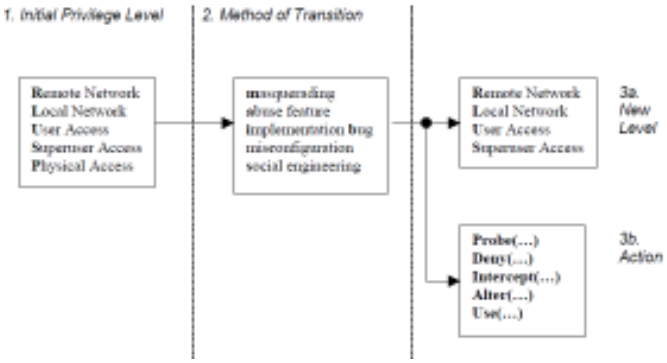

Figure2: A Summary of Possible Attack Descriptions

Figure 2 gives an overview of the approach the taxonomy uses for the classification of attacks. Each attack classified by this taxonomy is represented as a short alpha-numeric string. The initial privilege level is indicated by **R**, **L**, **U**, **S,** or **P,** the actions are indicated by any of the strings presented in section 3.3, and the method of exploitation is indicated by **m**, **a**, **b, c,** or **s** as defined in section 3.2. In order to describe an attack, select the privilege level that the attacker had before the attack occurred, from the possible choices of **R**emote Network, **L**ocal Network, Local **U**ser, **S**uperuser/Root, and **P**hysical Access. Next, select the method of exploitation, if the method is known. If the method is unknown, then a question mark ("?") is used to indicate the method of exploitation. The possible choices are **m**asquerading, **a**buse of feature, implementation **b**ug, mis**c**onfiguration, or **s**ocial engineering. Finally, either indicate the level of privilege the attacker gained as a result of the exploit (again with **R**, **L**, **U**, **S,** or **P**) or the actions the user performed at the current level of privilege.

Examples

The following table3 presents three examples that show the correct formatting of the alphanumeric string that specifies an action being performed at a specific privilege level. In a SYN flood (or neptune) attack the attacker sends a stream of SYN packets to a port on a target machine. For a short period of time after these packets have been sent, other users are unable to access the network services provided by that port. In the second example a user runs the crack program to decrypt the password file of a machine that has been compromised. In the third example, the attacker uses the Ffbconfig attack to make a transition from Local **U**ser privilege to Root/**S**uperuser privilege, and then uses this new privilege level to alter the password file on the victim system.[2]

Attack	String	Description
SYN flood	R-a-Deny(temporary)	A user with remote network access temporarily denies service
Cracking passwords	U-Use(Intrusion)	A user with a local account runs a program which attempts to decrypt entries in the password file.
ffbconfig	U-b-S-Alter(Files)	An attacker with a local account uses a bug in the Ffbconfig program to gain root access and alter files.

Table3

5. EXPLOITS FOR THE 1998 DARPA EVALUATION

Table 4 shows the 32 different exploits that were used in the 1998 DARPA intrusion detection evaluation. This table presents the attacks broken up into categories of type and vulnerable operating system. The four type categories represent groupings of the possible attack types listed in the taxonomy.

These four groups are: Denial of Service (R-?-Deny), Remote to Local User (R-?-U), Local User to Super-user (U-?-S), and Probes (R-?-Probe). The three columns of the table divide the exploits by target platform. Some attacks are listed in more than one column. The Smurf attack, for example, is listed three times—in the Solaris column, the SunOS column, and the Linux column—because all three operating systems are vulnerable to the Smurf attack.

The next sections of this paper present detailed descriptions of each class of attack, and the individual attacks from that class that were included in the 1998 DARPA intrusion detection evaluation.[2]

	Solaris	SunOS	Linux
Denial Of Service (R-Deny)	Apache2 Back Mailbomb Neptune Ping Of Death Process Table Smurf Syslogd UDP Storm	Apache2 Back Land Mailbomb Neptune Ping of death Process Table Smurf UDP Storm	Apache2 back Mailbomb Neptune Ping of death Process Table Smurf Teardrop UDP Storm
Remote to User (R-?-U)	dictionary ftp-write guest phf xlock xsnoop	dictionary ftp-write guest phf xlock xsnoop	dictionary ftp-write guest imap named phf sendmail xlock xsnoop
User to Superuser (U-?-S)	eject ffbconfig fdformat ps	loadmodule ps	perl xterm
Surveillance/ Probing (R-Probe)	ip sweep mscan nmap saint satan	ip sweep mscan nmap saint satan	ip sweep mscan nmap saint satan

Table4: The Attacks Used in the 1998 DARPA Intrusion Detection Evaluation

6. DENIAL OF SERVICE ATTACKS

A denial of service attack is an attack in which the attacker makes some computing or memory resource too busy or too full to handle legitimate requests, or denies legitimate users access to a machine. There are many varieties of denial of service (or DoS) attacks. Some DoS attacks (like a mailbomb, neptune, or smurf attack) abuse a perfectly legitimate feature. Others (teardrop, Ping of Death) create malformed packets that confuse the TCP/IP stack of the machine that is trying to reconstruct the packet. Still others (apache2, back, syslogd) take advantage of bugs in a particular network daemon.

Table5 provides an overview of the denial of service attacks used in the 1998 DARPA intrusion detection evaluation. Each row represents a single type of attack. The six columns show the attack name, a list of the services that the attack exploits, the platforms that are vulnerable to the attack, the type of mechanism that is exploited by the attack (implementation bug, abuse of feature, masquerading, or misconfiguration), a generalization of the amount of time the attack took to implement, and a summary of the effect of the attack

Name	Service	Vulnerable Platforms	Mechanism	Time to Implement	Effect

Apache 2	http	Any Apache	Abuse	Short	Crash httpd
Back	http	Any Apache	Abuse/Bug	Short	Slow server response
Land	N/A	SunOS	Bug	Short	Freeze machine
Mailbomb	smtp	All	Abuse	Short	Annoyance
SYN Flood	Any TCP	All	Abuse	Short	Deny service on one or more ports for minutes
Ping of Death	icmp	None	Bug	Short	None
Process Table	Any TCP	All	Abuse	Moderate	Deny new processes
Smurf	icmp	All	Abuse	Moderate/ Long	Network Slowdown
Syslogd	syslog	Solaris	Bug	Short	Kill Syslogd
Teardrop	N/A	Linux	Bug	Short	Reboot machine
Udpstorm	echo/ charge n	All	Abuse	Short	Network Slowdown

Table5: Summary of Denial of Service Attacks

6.1 Apache2- R-a-Deny(Temporary/Administrative)

The Apache2 attack is a denial of service attack against an apache web server where a client sends a request with many http headers. If the server receives many of these requests it will slow down, and may eventually crash [4].

Back R-a-Deny(Temporary)

In this denial of service attack against the Apache web server, an attacker submits requests with URL's containing many frontslashes. As the server tries to process these requests it will slow down and becomes unable to process other requests [55].

6.2 Land R-b-Deny(Administrative)

The Land attack is a denial of service attack that is effective against some older TCP/IP implementations. The only vulnerable platform used in the 1998 DARPA evaluation was SunOS 4.1. The Land attack occurs when an attacker sends a spoofed SYN packet in which the source address is the same as the destination address.

6.3 Mailbomb R-a-Deny(Administrative)

A Mailbomb is an attack in which the attacker sends many messages to a server, overflowing that server's mail queue and possible causing system failure.

6.4 SYN Flood (Neptune) R-a-Deny(Temporary)

A SYN Flood is a denial of service attack to which every TCP/IP implementation is vulnerable (to some degree). Each half-open TCP connection made to a machine causes the "tcpd" server to add a record to the data structure that stores information describing all pending connections. This data structure is of finite size, and it can be made to overflow by intentionally creating too many partially-open connections. The half-open connections data structure on the victim server system will eventually fill and the system will be unable to accept any new incoming connections until the table is emptied out. Normally there is a timeout associated with a pending connection, so the half-open connections will eventually expire and the victim server system will recover. However, the attacking system can simply continue sending IP-spoofed packets requesting new connections faster than the victim system can expire the pending connections. In some cases, the system may exhaust memory, crash, or be rendered otherwise inoperative.

6.5 Ping Of Death R-b-Deny(Temporary)

The Ping of Death is a denial of service attack that affects many older operating systems. Although the adverse effects of a Ping of Death could not be duplicated on any victim systems used in the 1998 DARPA evaluation, it has been widely reported that some systems will react in an unpredictable fashion when receiving oversized IP packets. Possible reactions include crashing, freezing, and rebooting.

6.6 Process Table R-a-Deny(Temporary)

The Process Table attack is a novel denial-of-service attack that was specifically created for this evaluation. The Process Table attack can be waged against numerous network services on a variety of different UNIX systems. The attack is launched against network services which fork() or otherwise allocate a new process for each incoming TCP/IP connection. Although the standard UNIX operating system places limits on the number of processes that any one user may launch, there are no limits on the number of processes that the superuser can create, other than the hard limits imposed by the operating system. Since incoming TCP/IP connections are usually handled by servers that run as root, it is possible to completely fill a target machine's process table with multiple instantiations of network

servers. Properly executed, this attack prevents any other command from being executed on the target machine.

An example of a service that is vulnerable to this attack is the finger service. On most computers, finger is launched by inetd. The authors of inetd placed several checks into the program's source code that must be bypassed in order to initiate a successful process attack. In a typical implementation (specifics will vary depending on the actual UNIX version used), if inetd receives more than 40 connections to a particular service within 1 minute, that service is disabled for 10 minutes. The purpose of these checks was not to protect the server against a process table attack, but to protect the server against buggy code that might create many connections in rapid-fire sequence.

To launch a successful process table attack against a computer running inetd and finger, the following sequence may be followed:
1. Open a connection to the target's finger port. 2. Wait for 4 seconds. 3. Repeat steps 1-2.

This attack has been attempted against a variety of network services on a variety of operating systems. It is believed that the imap and sendmail servers are vulnerable. Most imap server software contains no checks for rapid-fire connections. Thus, it is possible to shut down a computer by opening multiple connections to the imap server in rapid succession. With sendmail the situation is reversed. Normally, sendmail will not accept connections after the system load has jumped above a predefined level. Thus, to initiate a successful sendmail attack it is necessary to open the connections very slowly, so that the process table keeps growing in size while the system load remains more or less constant.

6.7 Smurf R-a-Deny(Temporary)

In the "smurf" attack, attackers use ICMP echo request packets directed to IP broadcast addresses from remote locations to create a denial-of-service attack. There are three parties in these attacks: the attacker, the intermediary, and the victim (note that the intermediary can also be a victim). The attacker sends ICMP "echo request" packets to the broadcast address (xxx.xxx.xxx.255) of many subnets with the source address spoofed to be that of the intended victim.

Any machines that are listening on these subnets will respond by sending ICMP "echo reply" packets to the victim. The smurf attack is effective because the attacker is able to use broadcast addresses to amplify what would otherwise be a rather innocuous ping flood. In the best case (from an attacker's point of view), the attacker can flood a victim with a volume of packets 255 times as great in magnitude as the attacker would be able to achieve without such amplification.

The attacking machine sends a single spoofed packet to the broadcast address of some network, and every machine that is located on that network responds by sending a packet to the victim machine. Because there can be as many as 255 machines on an Ethernet segment, the attacker can use this amplification to generate a flood of ping packets 255 times as great in size (in the best case) as would otherwise be possible.

6.8 Syslogd R-b-Deny(Administrative)

The Syslogd exploit is a denial of service attack that allows an attacker to remotely kill the syslogd service on a Solaris server. When Solaris syslogd receives an external message it attempts to do a DNS lookup on the source IP address. If this IP address doesn't match a valid DNS record, then syslogd will crash with a Segmentation Fault.

6.9 Teardrop R-a-Deny(Temporary)

The teardrop exploit is a denial of service attack that exploits a flaw in the implementation of older TCP/IP stacks. Some implementations of the IP fragmentation re-assembly code on these platforms does not properly handle overlapping IP fragments.

6.10 Udpstorm R-a-Deny(Administrative)

A Udpstorm attack is a denial of service attack that causes network congestion and slowdown. When a connection is established between two UDP services, each of whichproduces output, these two services can produce a very high number of packets that can lead to a denial of service on the machine(s) where the services are offered. Anyone with network connectivity can launch an attack; no account access is needed. For example, by connecting a host's chargen service to the echo service on the same or another machine, all affected machines may be effectively taken out of service because of the excessively high number of packets produced. [2][4][5].

7. USER TO ROOT ATTACKS

User to Root exploits are a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system. There are several different types of User to Root attacks. The most common is the buffer overflow attack. Buffer overflows occur when a program copies too much data into a static buffer without checking to make sure that the data will fit. For example, if a program expects the user to input the user's first name, the programmer must decide how many characters that first name buffer will require.

Assume the program allocates 20 characters for the first name buffer. Now, suppose the user's first name has 35 characters. The last 15 characters will overflow the name buffer. When this overflow occurs, the last 15 characters are placed on the stack, overwriting the next set of instructions that was to be executed. By carefully manipulating the data that overflows onto the stack, an attacker can cause arbitrary commands to be executed by the operating system. Despite the fact that programmers can eliminate this problem through careful programming techniques, some common utilities are susceptible to buffer overflow attacks. Another class of User to Root attack exploits programs that make assumptions about the environment in which they are running.

A good example of such an attack is the loadmodule attack, which is discussed below. Other User to Root attacks take advantage of programs that are not careful about the way they manage temporary files. Finally, some User to Root vulnerabilities exists because of an exploitable race condition in the actions of a single program, or two or more programs running simultaneously.

Although careful programming could eliminate all of these vulnerabilities, bugs like these are present in every major version of UNIX and Microsoft Windows available today. Table 6 summarizes the User to Root attacks used in the 1998 DARPA evaluation.

Name	Service	Vulnerable Platforms	Mechanism	Time to Implement	Effect
Eject	Any user session	Solaris	Buffer Overflow	Medium	Root Shell

Ffbconfig	Any user session	Solaris	Buffer Overflow	Medium	Root Shell
Fdformat	Any user session	Solaris	Buffer Overflow	Medium	Root Shell
Loadmodule	Any user session	SunOS	Poor Environment Sanitation	Short	Root Shell
Perl	Any user session	Linux	Poor Environment Sanitation	Short	Root Shell
Ps	Any user session	Solaris	Poor Temp File Management	Short	Root Shell
Xterm	Any user session	Linux	Short	Buffer Overflow	Root Shell

Table6: Summary of User to Root Attacks

7.1 Eject U-b-S

The Eject attack exploits a buffer overflow is the "eject" binary distributed with Solaris 2.5. In Solaris 2.5, removable media devices that do not have an eject button or removable media devices that are managed by Volume Management use the eject program. Due to insufficient bounds checking on arguments in the volume management library, libvolmgt.so.1, it is possible to overwrite the internal stack space of the eject program. If exploited, this vulnerability can be used to gain root access on attacked systems.

7.2 Ffbconfig U-b-S

The Ffbconfig attack exploits a buffer overflow is the "ffbconfig" program distributed with Solaris 2.5. The ffbconfig program configures the Creator Fast Frame Buffer (FFB) Graphics Accelerator, which is part of the FFB Configuration Software Package, SUNWffbcf. This software is used when the FFB Graphics accelerator card is installed. Due to insufficient bounds checking on arguments, it is possible to overwrite the internal stack space of the ffbconfig program.

7.3 Fdformat U-b-S

The Fdformat attack exploits a buffer overflow is the "fdformat" program distributed with Solaris 2.5. The fdformat program formats diskettes and PCMCIA memory cards. The program also uses the same volume management library, libvolmgt.so.1, and is exposed to the same vulnerability as the eject program.

7.4 Loadmodule U-b-S

The Loadmodule attack is a User to Root attack against SunOS 4.1 systems that use the xnews window system. The loadmodule program within SunOS 4.1.x is used by the xnews window system server to load two dynamically loadable kernel drivers into the currently running system and to create special devices in the /dev directory to use those modules. Because of a bug in the way the loadmodule program sanitizes its environment, unauthorized users can gain root access on the local machine.

7.5 Perl U-b-S

The Perl attack is a User to Root attack that exploits a bug in some Perl implementations. Suidperl is a version of Perl that supports saved set-user-ID and set-group-ID scripts. In early versions of suidperl the interpreter does not properly relinquish its root privileges when changing its effective user and group IDs. On a system that has the suidperl, or sperl, program installed and supports saved set-user-ID and saved set-group-ID, anyone with access to an account on the system can gain root access.

7.6 Ps U-b-S

The Ps attack takes advantage of a race condition in the version of "ps" distributed with Solaris 2.5 and allows an attacker to execute arbitrary code with root privilege. This race condition can only be exploited to gain root access if the user has access to the temporary files. Access to temporary files may be obtained if the permissions on the /tmp and /var/tmp directories are set incorrectly. Any users logged in to the system can gain unauthorized root privileges by exploiting this race condition.

7.6 Xterm U-b-S

The Xterm attack exploits a buffer overflow in the Xaw library distributed with Redhat Linux 5.0 (as well as other operating systems not used in the simulation) and allows an attacker to execute arbitrary instructions with root privilege. Problems exist in both the xterm program and the Xaw library that allow user supplied data to cause buffer overflows in both the xterm program and any program that uses the Xaw library. These buffer overflows are associated with the processing of data related to the input Method and preedit Type resources (for both xterm and Xaw) and the *Keymap resources (for xterm). Exploiting these buffer overflows with xterm when it is installed setuid-root or with any setuid-root program that uses the Xaw library can allow an unprivileged user to gain root access to the system [2][4][5].

8. REMOTE TO USER ATTACKS

A Remote to User attack occurs when an attacker who has the ability to send packets to a machine over a network—but who does not have an account on that machine—exploits some vulnerability to gain local access as a user of that machine.

There are many possible ways an attacker can gain unauthorized access to a local account on a machine. Some of the attacks exploit

buffer overflows in network server software (imap, named, sendmail). The Dictionary, Ftp-Write, Guest and Xsnoop attacks all attempt to exploit weak or misconfigured system security policies. The Xlock attack involves social engineering—in order for the attack to be successful the attacker must successfully spoof a human operator into supplying their password to a screensaver that is actually a trojan horse. Table7 summarizes the characteristics of the Remote to User attacks that were included in the 1998 DARPA intrusion detection evaluation. The following sections provide details of each of these attacks.

Name	Service	Vulnerable Platforms	Mechanism	Time to Implement	Effect
Dictionary	telnet, rlogin, pop, imap, ftp	All	Abuse of Feature	Medium	User-level access
Ftp-write	ftp	All	Misconfiguration	Short	User-level access
Guest	telnet, rlogin	All	Misconfiguration	Short	User-level access
Imap	imap	Linux	Bug	Short	Root Shell
Named	dns	Linux	Bug	Short	Root Shell
Phf	http	All	Bug	Short	Execute commands as user http
Sendmail	smtp	Linux	Bug	Long	Execute commands as root
Xlock	X	All	Misconfiguration	Medium	Spoof user to obtain password
Xsnoop	X	All	Misconfiguration	Short	Monitor Keystrokes remotely

Table7: Summary of Remote to User (Local) Attacks

8.1 Dictionary R-a-U

The Dictionary attack is a Remote to Local User attack in which an attacker tries to gain access to some machine by making repeated guesses at possible usernames and passwords. Users typically do not choose good passwords, so an attacker who knows the username of a particular user (or the names of all users) will attempt to gain access to this user's account by making guesses at possible passwords. Dictionary guessing can be done with many services; telnet, ftp, pop, rlogin, and imap are the most prominent services that support authentication using usernames and passwords.

8.2 Ftp-write R-c-U

The Ftp-write attack is a Remote to Local User attack that takes advantage of a common anonymous ftp misconfiguration. The anonymous ftp root directory and its subdirectories should not be owned by the ftp account or be in the same group as the ftp account. If any of these directories are owned by ftp or are in the same group as the ftp account and are not write protected, an intruder will be able to add files (such as an rhosts file) and eventually gain local access to the system.

8.3 Guest R-c-U

The Guest attack is a variant of the Dictionary attack. On badly configured systems, guest accounts are often left with no password or with an easy to guess password. Because most operating systems ship with the guest account activated by default, this is one of the first and simplest vulnerabilities an attacker will attempt to exploit.

8.4 Imap R-b-S

The Imap attack exploits a buffer overflow in the Imap server of Redhat Linux 4.2 that allows remote attackers to execute arbitrary instructions with root privileges. The Imap server must be run with root privileges so it can access mail folders and undertake some file manipulation on behalf of the user logging in. After login, these privileges are discarded. However, a buffer overflow bug exists in the authentication code of the login transaction, and this bug can be exploited to gain root access on the server. By sending carefully crafted text to a system running a vulnerable version of the Imap server, remote users can cause a buffer overflow and execute arbitrary instructions with root privileges.

8.5 Named R-b-S

The Named attack exploits a buffer overflow in BIND version 4.9 releases prior to BIND 4.9.7 and BIND 8 releases prior to 8.1.2. An improperly or maliciously formatted inverse query on a TCP stream destined for the named service can crash the named server or allow an attacker to gain root privileges.

8.6 Phf R-b-U

The Phf attack abuses a badly written CGI script to execute commands with the privilege level of the http server. Any CGI program which relies on the CGI function escape_shell_cmd() to prevent exploitation of shell-based library calls may be vulnerable to attack. In particular, this vulnerability is manifested by the "phf" program that is distributed with the example code for the Apache web server.

8.7 Sendmail R-b-S

The Sendmail attack exploits a buffer overflow in version 8.8.3 of sendmail and allows a remote attacker to execute commands with superuser privileges. By sending a carefully crafted email message to a system running a vulnerable version of sendmail, intruders can force sendmail to execute arbitrary commands with root privilege.

\

8.8 Xlock R-cs-Intecept(Keystrokes)

In the Xlock attack, a remote attacker gains local access by fooling a legitimate user who has left their X console unprotected, into revealing their password. An attacker can display a modified version of the xlock program on the display of a user who has left their X

display open (as would happen after typing "xhost +"), hoping to convince the user sitting at that console to type in their password. If the user sitting at the machine being attacked actually types their password into the trojan version of xlock the password will be sent back to the attacker.

8.9 Xsnoop R-c-Intercept(Keystrokes)

In the Xsnoop attack, an attacker watches the keystrokes processed by an unprotected X server to try to gain information that can be used gain local access the victim system. An attacker can monitor keystrokes on the X server of a user who has left their X display open. A log of keystrokes is useful to an attacker because it might contain confidential information, or information that can be used to gain access to the system such as the username and password of the user being monitored. [2][4][5]

9. PROBES

In recent years, a growing number of programs have been distributed that can automatically scan a network of computers to gather information or find known vulnerabilities. These network probes are quite useful to an attacker who is staging a future attack. An attacker with a map of which machines and services are available on a network can use this information to look for weak points. Some of these scanning tools (satan, saint, mscan) enable even a very unskilled attacker to very quickly check hundreds or thousands of machines on a network for known vulnerabilities. Table 8 provides a summary of the probes. The following sections describe in detail each of the probes that was used in the 1998 DARPA intrusion detection evaluation.

Name	Service	Vulnerable Platforms	Mechanism	Time to Implement	Effect
Ipsweep	ICMP	All	Abuse of Feature	Short	Finds active machines
Mscan	many	All	Abuse of Feature	Short	Looks for known vulnerabilities
Nmap	many	All	Abuse of Feature	Short	Finds active ports on a machine
Saint	many	All	Abuse of Feature	Short	Looks for known vulnerabilities
Satan	many	All	Abuse of Feature	Short	Looks for known vulnerabilities

Table8: Summary of Probes

9.1 Ipsweep R-a-Probe(Machines)

An Ipsweep attack is a surveillance sweep to determine which hosts are listening on a network. This information is useful to an attacker in staging attacks and searching for vulnerable machines.

9.2 Mscan R-a-Probe(Known Vulnerabilities)

Mscan is a probing tool that uses both DNS zone transfers and/or brute force scanning of IP addresses to locate machines, and test them for vulnerabilities.

9.3 Nmap R-a-Probe(Services)

Nmap is a general-purpose tool for performing network scans. Nmap supports many different types of portscans—options include SYN, FIN and ACK scanning with both TCP and UDP, as well as ICMP (Ping) scanning. The Nmap program also allows a user to specify which ports to scan, how much time to wait between each port, and whether the ports should be scanned sequentially or in a random order.

9.4 Saint R-a-Probe(Known Vulnerabilities)

SAINT is the Security Administrator's Integrated Network Tool. In its simplest mode, it gathers as much information about remote hosts and networks as possible by examining such network services as finger, NFS, NIS, ftp and tftp, rexd, statd, and other services. The information gathered includes the presence of various network information services as well as potential security flaws. These flaws include incorrectly setup or configured network services, well-known bugs in system or network utilities, and poor policy decisions. Although SAINT is not intended for use as an attack tool, it does provide security information that is quite useful to an attacker.

9.5 Satan R-a-Probe(Known Vulnerabilities)

SATAN is an early predecessor of the SAINT scanning program described in the last section. While SAINT and SATAN are quite similar in purpose and design, the particular vulnerabilities that each tools checks for are slightly different. [2][4][5]

10. ATTACK SCENARIOS

Most of the attacks that were included in the evaluation consist of a single session, or a few sessions that all occur within some short period of time. In the real world, an attacker often has a goal in mind, and sometimes this goal cannot be achieved in a single session. Several complex scenarios were added to the collected data in an attempt to create a better simulation of real attacker behaviour. Each scenario consists of a planned sequence of sessions (sometimes over the course of a week or more) that represent the actions of a single individual in pursuit of a goal. The following subsections describe each scenario in detail.

10.1 Cracker

The default scenario that occurred in 95% of sessions is that a curious cracker was trying to gain access to a machine just to prove that it could be done. Usually these crackers are simply trying to break into as many machines as they can, and may install a backdoor or download the password file in order to guarantee that they can access the machine again. Crackers are represented as individuals of different skill levels, some perform all of their actions in the clear, while others are aware that an intrusion detection system is present and take actions to avoid detection.

10.2 Spy

The spy is an information collector who comes back to a compromised machine several times to collect information. A spy might be looking for confidential data files or reading user's personal mail. A spy will take steps to minimize the possibility of detection.

10.3 Rootkit

The rootkit scenario could be viewed as an extension of the default Cracker scenario. A rootkit is a collection of programs that are intended to help a hacker maintain access to a machine once it has been compromised. A typical rootkit consists of a sniffer, versions of login, su, and other programs with backdoors which allow for access, and new versions of ps, netstat, and ls that hide the fact that a sniffer is running and hide files in certain directories. Once the rootkit has been installed, the attacker comes back several times to download the sniffer logs.

10.4 Http Tunnel

The Http Tunnel scenario was originally developed as a method of defeating a firewall and for continuing to access a system while minimizing the chance of detection. The http tunnelling tools used in this scenario were developed specifically for use in this evaluation. Assuming that a hacker is able to penetrate a firewall once (perhaps by sending an email with the executable content of the client in it or by dialling into a modem) a server program can be installed that masquerades as a normal user browsing web pages. Once this server has been installed, the hacker can issue requests to execute commands or transfer files with interaction happening in web traffic between the server and the hacker. For the simulation, the data was exchanged through cookies that rode along with a web request. This general method is very flexible however, and the data could have been tunnelled using any cryptographic or steganographic technique. Tunnelling through http was chosen with hopes that the large amount of http traffic most networks see in a typical day would obscure the actions of the attacker.

10.5 SNMP Monitoring

An attacker who has guessed the SNMP community password of a router will then be able to monitor the traffic levels on that router, and may be able to issue commands to the router to change default routes or allow connections from a previously forbidden host or network.

10.6 Multihop

Some intrusion detection systems monitor traffic immediately outside of a router and only see traffic going into or coming out of that network. The multihop scenario was designed to test whether these systems could find attacks where an attacker first breaks into one inside machine, and then uses this inside machine for further attacks on the rest of the machines on the network. This would be a very effective means of launching a Denial of Service attack undetected, as an intrusion detection system which sits just outside the network being monitored would not see a Denial of Service attack that originated from the internal network.

10.7 Disgruntled/Malicious User

These sessions simulate an attacker who is not interested in collecting information from a system or gaining access to a system, but is simply interested in doing damage. This is a common threat in operational computer networks. Within the simulation, one malicious user re-formatted the primary disk partition of a victim machine [2][4][5].

11. STEALTHINESS AND ACTIONS

In addition to varying the methods and intentions of the simulated attackers, attention was given to the extent to which attackers tried to hide their actions from either an individual who is monitoring the system, or an intrusion detection system. There are several ways that attackers can reduce their chances of being detected by the administrator of a network. Skilled attackers might try to cover their tracks by editing system logs or resetting the modification date on files that they replaced or modified. These actions are generally intended to reduce the chance of detection by a human administrator. Attackers may also be aware that an intrusion detection system is monitoring a network, and may try to hide from the intrusion detection system as well. Methods for being stealthy vary depending on the type of attack.

11.1 Avoiding Detection of Denial of Service (R-Deny)

Denial of Service attacks are difficult to make stealthy. One method an attacker can use to hide a denial of service attack is to gain the cooperation of a large group and break up the attack so pieces of it are coming from several different sources. Another method an attacker can use is to send thousands of packets with different spoofed source-addresses. Sending these spoofed packets will not make identifying the attack any harder, but they will make it more difficult to track down and stop the attacker because the victim has no 101 way of knowing which of the thousands of addresses the actual attack is coming from. Both of these methods do not make the attack any harder to detect, but simply reduce the chances that the attacker will be caught. No stealthy denial of service attacks were included in the 1998 DARPA intrusion detection evaluation.

11.2 *Avoiding Detection of Probes (R-Probe)*

Several methods can be used either to hide the fact that a probe is occurring, or obscure the identity of the party who is performing the probe. The following paragraphs describe methods of increasing stealth that were used in the probes included in the 1998 DARPA intrusion detection evaluation.

- **Scan Slowly and Randomly:** If an attacker wants to hide the fact a probe is occurring, the probe can be configured to occur slowly and probe ports or machines in a nonlinear order. An intrusion detection system will have a very hard time identifying one stray connection per hour to a random port as a port sweep initiated by an attacker.
- **Probe With Half-Open or Other Unlogged Connections**: Another method of scanning stealthily is to probe with half-open connections. A connection for which the threeway TCP handshake is never completed will not be logged by the operating system. There are several tools available that will perform this type of half-open (FIN) scanning of a network.
- **Use an Intermediate Machine to Obscure the Real Source of the Scan:** One way attackers can hide their identity is to use an ftp bounce probe. Some ftp servers will allow anyone to tell them to send data to a particular port on a particular machine. An attacker can look at the response the ftp server gives from such a request and ascertain whether that port is listening on the victim machine. The portscan will appear to be coming from an anonymous ftp server, and this simple step may be enough to assure that the party who is really doing the scanning is never identified.

11.3 Avoiding Detection of User to Root (L-?-S) Attacks

There are many ways that User to Root attacks can be made stealthy. The following paragraphs discuss each of the methods that were used to make a number of the User to Root attacks in the 1998 DARPA intrusion detection evaluation stealthy.

- **Keyword Hiding:** Some intrusion detection systems that attempt to detect illegal User to Root transitions rely on keyword spotting to detect intruders. For example, if the system observes the text of the C source code for the publicly available Eject exploit in a telnet or rlogin session, it will flag this session as being suspicious. By uuencoding or Mime encoding the text of the code before sending it over the network connection, an attacker would avoid detection by such a system.
- **Output Hiding**: It is possible to identify attacks by looking that the output that is displayed on the terminal when the exploit is run. An attacker can avoid detection via this mechanism by sending all the output of commands that are run to a file and encoding (again with some method like ROT13, uuencode, or gzip) the file before displaying or transferring it.
- **Command Hiding:** A system might also look for an attacker to run some command that only the superuser should be able to run, such as displaying the contents of the shadow password file. The invocation of a command that the attacker wishes to hide from someone who is looking for certain suspicious commands or actions can be obfuscated by using glob constructs and character replacement. Instead of typing the command "cat /etc/passwd", the attacker can issue the command "/[r,s,t,b]?[l,w,n,m]/[c,d]?t/?t[c,d,e]/*a?s*". When the shell tries to interpret this input string it will do replacement of the glob characters and find that the only valid match for this string is "/bin/cat /etc/passwd".
- **Delayed Attack**: An attacker can also avoid detection (or at least reduce the chances of identification), by separating the time of exploit from the initial time of access. This can be accomplished by submitting a job to the "at" or "cron" daemon which will run the attack at some later time. An attacker can log in as a normal user (which would not be noticed by an intrusion detection system) and submit a shell script to the "at" daemon which would execute any actions that the attacker desired three weeks (or six months, or five years!) after the attacker initially logged into the system [2][4][5].

CONCLUSIONS

In This Paper We Have Presented Information About Different Attacks, Intrusion Detection System, We Also Present Information About The 1998/99 DARPA Intrusion Detection Evaluation With Attack Details Such As Denial Of Service Attacks(DoS), User To Root Attacks(U2R), Remote To User Attacks(R2L) And Probes. Further we present some information about attack scenarios and Stealthiness and Actions against attack. Finally we conclude that the 1998/99 DARPA Intrusion Detection Evaluation data set is very useful for researchers who are working in networks security especially on Intrusion Detection Systems (IDS). We are hopeful that our paper will helpful for researchers.

REFERENCES

[1] John Mchugh Carnegie Mellon University, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory"," ACM Transactions on Information and System Security, Vol. 3, No. 4, November 2000".

[2] Daniel Weber, "A Taxonomy of Computer Intrusions","Master's thesis, Massachusetts Institute of Technology, June 1998".

[3] J.P. Anderson, "Computer security threat monitoring and surveillance", Technical report, James P. Anderson Co.,Fort Washingon, PA, April 1980.

[4] Kristopher Kendall, "A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems", "Department of Electrical Engineering and Computer Science, Massachusetts Institute Of Technology May 21, 1999".

[5] J.W. Haines, R.P. Lippmann, D.J. Fried, E. Tran, S.B. Boswell "1999 DARPA Intrusion Detection Evaluation: Design and Procedures", "Technical Report 1062,Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Massachusetts, 26 February 2001"

[6] Kumar J. Das, "Attack Development for Intrusion Detection Evaluation", "Department of Electrical Engineering and Computer Science, Massachusetts Institute Of Technology,June 2000.

[7] Srinivas Mukkamala, Andrew H. Sung, Ajith Abraham, "Intrusion detection using an ensemble of intelligent paradigms", "S. Mukkamala et al. / Journal of Network and Computer Applications 28 (2005) 167–182"

[8] Hamdan.O.Alanazi, Rafidah Md Noor, B.B Zaidan, A.A Zaidan, "Intrusion Detection System: Overview ", "Journal Of Computing, Volume 2, Issue 2, February 2010, Issn 2151-9617 Https://Sites.Google.Com/Site/Journalofcomputing/"

[9] Network Attack and Defense," Security Engineering: A Guide to Building Dependable Distributed Systems, Chapter 18, pp-367-390"

[10] Peyman Kabiri and Ali A. Ghorbani," Research on Intrusion Detection and Response:A Survey ", "International Journal of Network Security, Vol.1, No.2, PP.84–102, Sep. 2005 (http://isrc.nchu.edu.tw/ijns/)"

[11] Understanding Intrusion Detection Systems," SANS Institute 2001, As part of the Information Security Reading Room."

[12] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, and Ali A. Ghorbani, "A Detailed Analysis of the KDD CUP 99 Data Set", "Proceedings Of The 2009 Ieee Symposium On Computational Intelligence In Security And Defense Applications (CISDA 2009)"