Threat Assessment of Malicious Code and Human Threats

Lawrence E. Bassham & W. Timothy Polk

National Institute of Standards and Technology
Computer Security Division

October 1, 1992

Contents

1 Introduction **1**

2 Malicious Code **3**

2.1 Viruses 3

2.1.1 History of Viruses 4

2.1.2 Current Protection Against Viruses 6

2.2 Worms 7

2.2.1 History of Worms 7

2.2.2 Current Protection Against Worms 9

2.3 Trends for the Future 10

3 Human Threats **11**

3.1 Insider Attacks 11

3.2 Hackers 11

3.3 Phone Phreaks 13

3.4 Trends for the Future 13

3.4.1 Configuration Errors and Passwords 13

3.4.2 Internal Threats 14

3.4.3 Connectivity 14

3.4.4 Information Dissemination 15

4 Summary **17**

References **19**

1 Introduction

As a participant in the U.S. Army Computer Vulnerability/Survivability Study Team, the National Institute of Standards and Technology has been tasked with providing an assessment of the threats associated with commercial hardware and software. This document is the second and final deliverable under the Military Interdepartmental Purchase Request number: W43P6Q-92-EW138. This report provides an assessment of the threats associated with malicious code and external attacks on systems using commercially available hardware and software. The history of the threat is provided and current protection methods described. A projection of the future threats for both malicious code and human threats is also given.

Today, computer systems are under attack from a multitude of sources. These range from malicious code, such as viruses and worms, to human threats, such as hackers and phone "phreaks." These attacks target different characteristics of a system. This leads to the possibility that a particular system is more susceptible to certain kinds of attacks.

Malicious code, such as viruses and worms, attack a system in one of two ways, either internally or externally. Traditionally, the virus has been an internal threat, while the worm, to a large extent, has been a threat from an external source.

Human threats are perpetrated by individuals or groups of individuals that attempt to penetrate systems through computer networks, public switched telephone networks or other sources. These attacks generally target known security vulnerabilities of systems. Many of these vulnerabilities are simply due to configuration errors.

Certain commercial products are identified in this paper in order to adequately specify procedures being described. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology, nor does it imply that the material identified is necessarily the best for the purpose.

2 Malicious Code

Viruses and worms are related classes of malicious code; as a result they are often confused. Both share the primary objective of replication. However, they are distinctly different with respect to the techniques they use and their host system requirements. This distinction is due to the disjoint sets of host systems they attack. Viruses have been almost exclusively restricted to personal computers, while worms have attacked only multi-user systems.

A careful examination of the histories of viruses and worms can highlight the differences and similarities between these classes of malicious code. The characteristics shown by these histories can be used to explain the differences between the environments in which they are found. Viruses and worms have very different functional requirements; currently no class of systems simultaneously meets the needs of both.

A review of the development of personal computers and multi-tasking workstations will show that the gap in functionality between these classes of systems is narrowing rapidly. In the future, a single system may meet all of the requirements necessary to support both worms and viruses. This implies that worms and viruses may begin to appear in new classes of systems. A knowledge of the histories of viruses and worms may make it possible to predict how malicious code will cause problems in the future.

Basic Definitions

To provide a basis for further discussion, the following definitions will be used throughout the report.

- Trojan Horse - a program which performs a useful function, but also performs an unexpected action as well.
- Virus - a code segment which replicates by attaching copies to existing executables.
- Worm - a program which replicates itself and causes execution of the new copy.
- Network Worm - a worm which copies itself to another system by using common network facilities, and causes execution of the copy on that system.

2.1 Viruses

The following are necessary characteristics of a virus:

- replication
- requires a host program as a carrier

- activated by external action
- replication limited to (virtual) system

In essence, a computer program which has been infected by a virus has been converted into a trojan horse. The program is expected to perform a useful function, but has the unintended side effect of viral code execution. In addition to performing the unintended task, the virus also performs the function of replication. Upon execution, the virus attempts to replicate and "attach" itself to another program. It is the unexpected and generally uncontrollable repliction that makes viruses so dangerous.

Viruses are currently designed to attack single platforms. A platform is defined as the combination of hardware and the most prevalent operating system for that hardware. As an example, a virus can be referred to as an IBM-PC virus, referring to the hardware, or a DOS virus, referring to the operating system. "Clones" of systems are also included with the original platform.

2.1.1 History of Viruses

The term "computer virus" was formally defined by Fred Cohen in 1983, while he performed academic experiments on a Digital Equipment Corporation VAX system. Viruses are classified as being one of two types: research or "in the wild." A research virus is one that has been written for research or study purposes and has received almost no distribution to the public. On the other hand, viruses which have been seen with any regularity are termed "in the wild." The first computer viruses were developed in the early 1980s. The first viruses found in the wild were Apple II viruses, such as Elk Cloner, which was reported in 1981 [Den90]. Viruses have now been found on the following platforms:

- Apple II
- IBM PC
- Macintosh
- Atari
- Amiga

Note that all viruses found in the wild target personal computers. As of today, the overwhelming number of virus strains are IBM PC viruses. However, as of August 1989, the number of PC, Atari ST, Amiga, and Macintosh viruses were almost identical (21, 22, 18, and 12 respectively [Den90]). Academic studies have shown that viruses are possible for multi-tasking systems, but they have not yet appeared. This point will be discussed later.

Viruses have "evolved" over the years due to efforts by their authors to make the code more difficult to detect, disassemble, and eradicate. This evolution has been especially apparent

in the IBM PC viruses; since there are more distinct viruses known for the DOS operating system than any other.

The first IBM-PC virus appeared in 1986 [Den90]; this was the *Brain* virus. *Brain* was a boot sector virus and remained resident. In 1987, *Brain* was followed by *Alameda* (*Yale*), *Cascade*, *Jerusalem*, *Lehigh*, and *Miami* (*South African Friday the 13th*). These viruses expanded the target executables to include COM and EXE files. *Cascade* was encrypted to deter disassembly and detection. Variable encryption appeared in 1989 with the *1260* virus. Stealth viruses, which employ various techniques to avoid detection, also first appeared in 1989, such as *Zero Bug*, *Dark Avenger* and *Frodo* (*4096* or *4K*). In 1990, self-modifying viruses, such as *Whale* were introduced. The year 1991 brought the *GP1* virus, which is "network-sensitive" and attempts to steal Novell NetWare passwords. Since their inception, viruses have become increasingly complex.

Examples from the IBM-PC family of viruses indicate that the most commonly detected viruses vary according to continent, but *Stoned*, *Brain*, *Cascade*, and members of the *Jerusalem* family, have spread widely and continue to appear. This implies that highly survivable viruses tend to be benign, replicate many times before activation, or are somewhat innovative, utilizing some technique never used before in a virus.

Personal computer viruses exploit the lack of effective access controls in these systems. The viruses modify files and even the operating system itself. These are "legal" actions within the context of the operating system. While more stringent controls are in place on multi-tasking, multi-user operating systems, configuration errors, and security holes (security bugs) make viruses on these systems more than theoretically possible.

This leads to the following initial conclusions:

- Viruses exploit weaknesses in operating system controls and human patterns of system use/misuse.
- Destructive viruses are more likely to be eradicated.
- An innovative virus may have a larger initial window to propagate before it is discovered and the "average" anti-viral product is modified to detect or eradicate it.

It has been suggested that viruses for multi-user systems are too difficult to write. However, Fred Cohen required only "8 hours of expert work" [Hof90] to build a virus that could penetrate a UNIX[1] system. The most complex PC viruses required a great deal more effort.

Yet, if we reject the hypothesis that viruses do not exist on multi-user systems because they are too difficult to write, what reasons could exist? Perhaps the explosion of PC viruses (as opposed to other personal computer systems) can provide a clue. The population of PCs and

[1]UNIX is a registered trademark of AT&T

PC compatibles is by far the largest. Additionally, personal computer users exchange disks frequently. Exchanging disks is not required if the systems are all connected to a network. In this case large numbers of systems may be infected through the use of shared network resources.

One of the primary reasons that viruses have not been observed on multi-user systems is that administrators of these systems are more likely to exchange source code rather than executables. They tend to be more protective of copyrighted materials, so they exchange locally developed or public domain software. It is more convenient to exchange source code, since differences in hardware architecture may preclude exchanging executables.

The advent of remote disk protocols, such as NFS (Network File System) and RFS (Remote File System), have resulted in the creation of many small populations of multi-user systems which freely exchange executables. Even so, there is little exchange of executables between different "clusters" of systems.

The following additional conclusions can be made:

- To spread, viruses require a large population of homogeneous systems and exchange of executable software.

2.1.2 Current Protection Against Viruses

Although many anti-virus tools and products are now available, personal and administrative practices and institutional policies, particularly with regard to shared or external software usage, should form the first line of defense against the threat of virus attack. Users should also consider the variety of anti-virus products currently available.

There are three classes of anti-virus products: detection tools, identification tools, and removal tools. Scanners are an example of both detection and identification tools. Vulnerability monitors and modification detection programs are both examples of detection tools. Disinfectors are examples of a removal tools. A detailed description of the tools is provided below.

Scanners and disinfectors, the most popular classes of anti-virus software, rely on a great deal of *a priori* knowledge about the viral code. Scanners search for "signature strings" or use algorithmic detection methods to identify known viruses. Disinfectors rely on substantial information regarding the size of a virus and the type of modifications to restore the infected file's contents.

Vulnerability monitors, which attempt to prevent modification or access to particularly sensitive parts of the system, may block a virus from hooking sensitive interrupts. This requires

a lot of information about "normal" system use, since personal computer viruses do not actually circumvent any security features. This type of software also requires decisions from the user.

Modification detection is a very general method, and requires no information about the virus to detect its presence. Modification detection programs, which are usually checksum based, are used to detect virus infection or trojan horses. This process begins with the creation of a baseline, where checksums for clean executables are computed and saved. Each following iteration consists of checksum computation and comparison with the stored value. It should be noted that simple checksums are easy to defeat; cyclical redundancy checks (CRC) are better, but can still be defeated; cryptographic checksums provide the highest level of security.

2.2 Worms

The following are necessary characteristics of a worm:

- replication
- self-contained; does not require a host
- activated by creating process (needs a multi-tasking system)
- for network worms, replication occurs across communication links

A worm is not a trojan horse; it is a program designed to replicate. The program may perform any variety of additional tasks as well. The first network worms were intended to perform useful network management functions [SH82]. They took advantage of system properties to perform useful action. However, a malicious worm takes advantage of the same system properties. The facilities that allow such programs to replicate do not always discriminate between malicious and good code.

2.2.1 History of Worms

Worms were first used as a legitimate mechanism for performing tasks in a distributed environment. Network worms were considered promising for the performance of network management tasks in a series of experiments at the Xerox Palo Alto Research Center in 1982. The key problem noted was "worm management;" controlling the number of copies executing at a single time. This would be experienced later by authors of malicious worms.

Worms were first noticed as a potential computer security threat when the *Christmas Tree Exec* [Den90] attacked IBM mainframes in December 1987. It brought down both the world-wide IBM network and BITNET. The *Christmas Tree Exec* wasn't a true worm. It was a

trojan horse with a replicating mechanism. A user would receive an e-mail Christmas card that included executable (REXX) code. If executed the program claimed to draw a Xmas tree on the display. That much was true, but it also sent a copy to everyone on the user's address lists.

The *Internet Worm* [Spa89] was a true worm. It was released on November 2, 1988. It attacked Sun and DEC UNIX systems attached to the Internet[2] (it included two sets of binaries, one for each system). It utilized the TCP/IP protocols, common application layer protocols, operating system bugs, and a variety of system administration flaws to propagate. Various problems with worm management resulted in extremely poor system performance and a denial of network service.

The *Father Christmas* worm was also a true worm. It was first released onto the worldwide DECnet Internet in December of 1988. This worm attacked VAX/VMS systems on SPAN and HEPNET. It utilized the DECnet protocols and a variety of system administration flaws to propagate. The worm exploited TASK0, which allows outsiders to perform tasks on the system. This worm added an additional feature; it reported successful system penetration to a specific site.

This worm made no attempt at secrecy; it was not encrypted and sent mail to every user on the system. About a month later another worm, apparently a variant of *Father Christmas*, was released on a private network. This variant searched for accounts with "industry standard" or "easily guessed" passwords.

The history of worms displays the same increasing complexity found in the development of PC viruses. The *Christmas Tree Exec* wasn't a true worm. It was a trojan horse with a replicating mechanism. The *Internet Worm* was a true worm; it exploited both operating system flaws and common system management problems. The DECnet worms attacked system management problems, and reported information about successful system penetration to a central site.

Several conclusions can be drawn from this information:

- worms exploit flaws (i.e, bugs) in the operating system or inadequate system management to replicate.
- release of a worm usually results in brief but spectacular outbreaks, shutting down entire networks.

[2] a large "network of networks" connecting smaller networks around the world

2.2.2 Current Protection Against Worms

Protecting a system against a worm requires a combination of basic system security and good network security. There are a variety of procedures and tools which can be applied to protect the system.

In basic system security, the most important means of defense against worms is the **identification & authentication** (I&A) controls, which are usually integrated into the system. If poorly managed, these controls become a vulnerability which is easily exploited. Worms are especially adept at exploiting such vulnerabilities; both the Internet and DECnet worms targeted I&A controls.

Add-on tools include configuration review tools (such as COPS [GS91] for UNIX systems) and checksum-based change detection tools. Design of configuration review tools requires intimate knowledge of the system, but no knowledge of the worm code.

Another class of add-on tools is the intrusion detection tool. This is somewhat analogous to the PC monitoring software, but is usually more complex. This tool reviews series of commands to determine if the user is doing something suspicious. If so, the system manager is notified.

One type of network security tool is the wrapper program. Wrapper programs can be used to "filter" network connections, rejecting or allowing certain types of connections (or connections from a pre-determined set of systems). This can prevent worm infections by "untrusted" systems. Overlaps in trust may still allow infection to occur (A trusts B but not C; B trusts C; C infects B which infects A) but the rate of propagation will be limited.[3]

These tools do not protect a system against the exploitation of flaws in the operating system. This issue must be dealt with at the time of procurement. After procurement, it becomes a procedural issue. Resources are available to system managers to keep them abreast of security bugs and bug fixes, such as the CERT[4] computer security advisories.[5]

Another class of security tools can be employed to protect a *network* against worms. The **firewall system** [GS91] protects an organizational network from systems in the larger network world. Firewall systems are found in two forms: simple or intelligent. An intelligent firewall filters all connections between hosts on the organizational network and the world-at-large. A simple firewall disallows all connections with the outside world, essentially splitting the network into two different networks. To transfer information between hosts on the different networks, an account on the firewall system is required.

[3]Unlimited trust was a vulnerability exploited by the Internet Worm.

[4]Computer Emergency Response Team

[5]System managers without network access can obtain this information from the NIST Computer Security BBS at (301)948-5717.

2.3 Trends for the Future

Personal computers have been immune to worms because they are single task systems. The increasing functionality of personal computer operating systems will soon change this. Personal computers will become true multi-tasking systems, and will inherit both the functionality and security vulnerabilities that those systems have exhibited.

Multi-user systems have never been attractive virus targets, due to limited population, low software interchange rates, and because they use some form of access control. The advent of 486-class PCs is likely to change this. In addition to the increased performance of PC based machines, the UNIX workstation market is growing rapidly, producing high-performance machines at extremely affordable prices. Multi-user systems will be gaining market share, increasing their attractiveness to virus authors.

This large homogeneous population of multi-user systems will be an attractive target for both virus authors and worm developers. Personal computer worms or virus/worm hybrids may become the new threat the 90s. With a large homogeneous population of systems available, it is conceivable that authors of malicious code will combine the previously disjoint attacks of viruses and worms. An attack consisting of a worm traversing a network and dropping viruses on the individual hosts becomes a startling possibility.

As the functionality of personal computers continues to grow, new types of tools will be required to achieve the same degree of security. Scanners must be supplemented with configuration review tools. Identification & authentication tools (non-existent or neglected on most PCs) will become an important security tool on personal computers. Intrusion detection tools may become applicable to personal computers. Change detection will also play an increased role.

Administrators of personal computer networks must become familiar with a new set of practices, tools, and techniques, such as firewalls. They will need to draw upon the world of multi-user systems for this knowledge.

As the differences between PC and multi-user environments decreases, the likelihood of these environments facing similar threats will increase. Viruses will be more likely in the multi-user world; worms will become a threat in personal computer networks.

3 Human Threats

Insiders, hackers and "phone phreaks" are the main components of the human threat factor. Insiders are legitimate users of a system. When they use that access to circumvent security, that is known as an insider attack. Hackers are the most widely known human threat. Hackers are people who enjoy the challenge of breaking into systems. "Phreakers" are hackers whose main interest is in telephone systems.

3.1 Insider Attacks

The primary threat to computer systems has traditionally been the *insider attack*. Insiders are likely to have specific goals and objectives, and have legitimate access to the system. Insiders can plant trojan horses or browse through the file system. This type of attack can be extremely difficult to detect or protect against.

The insider attack can affect all components of computer security. Browsing attacks the confidentiality of information on the system. Trojan horses are a threat to both the integrity and confidentiality of the system. Insiders can affect availability by overloading the system's processing or storage capacity, or by causing the system to crash.

These attacks are possible for a variety of reasons. On many systems, the access control settings for security-relevant objects do not reflect the organization's security policy. This allows the insider to browse through sensitive data or plant that trojan horse. The insider exploits operating system bugs to cause the system to crash. The actions are undetected because audit trails are inadequate or ignored.

3.2 Hackers

The definition of the term "hacker" has changed over the years. A hacker was once thought of as any individual who enjoyed getting the most out of the the system he was using. A hacker would use a system extensively and study the system until he became proficient in all its nuances. This individual was respected as a source of information for local computer users; someone referred to as a "guru" or "wizard." Now, however, the term hacker is used to refer to people who either break into systems for which they have no authorization or intentionally overstep their bounds on systems for which they do have legitimate access.

Methods used by hackers to gain unauthorized access to systems include:

- Password cracking
- Exploiting known security weaknesses
- Network spoofing
- "Social engineering"

The most common techniques used to gain unauthorized system access involve password cracking and the exploitation of known security weaknesses. Password cracking is a technique used to surreptitiously gain system access by using another users account. Users often select weak password. The two major sources of weakness in passwords are easily guessed passwords based on knowledge of the user (e.g. wife's maiden name) and passwords that are susceptible to dictionary attacks (i.e. brute-force guessing of passwords using a dictionary as the source of guesses).

Another method used to gain unauthorized system access is the exploitation of known security weaknesses. Two type of security weaknesses exist: configuration errors, and security bugs. There continues to be an increasing concern over configuration errors. Configuration errors occur when a the system is set up in such a way that unwanted exposure is allowed. Then, according to the configuration, the system is at risk from even legitimate actions. An example of this would be that if a system "exports" a file system to the world (makes the contents of a file system available to all other systems on the network) , then any other machine can have full access to that file system (one major vendor ships systems with this configuration). Security bugs occur when unexpected actions are allowed on the system due to a loophole in some application program. An example would be sending a very long string of keystrokes to a screen locking program, thus causing the program to crash and leaving the system inaccessible.

A third method of gaining unauthorized access is network spoofing. In network spoofing a system presents itself to the network as though it were a different system (system A impersonates system B by sending B's address instead of its own). The reason for doing this is that systems tend to operate within a group of other "trusted" systems. Trust is imparted in a one-to-one fashion; system A trusts system B (this does not imply that system B trusts system A). Implied with this trust, is that the system administrator of the trusted system is performing his job properly and maintaining an appropriate level of security for his system. Network spoofing occurs in the following manner: if system A trusts system B and system C spoofs (impersonates) system B, then system C can gain otherwise denied access to system A.

"Social engineering" is the final method of gaining unauthorized system access. People have been known to call a system operator, pretending to be some authority figure, and demand that a password be changed to allow them access. One could also say that using personal data to guess a user's password is social engineering.

3.3 Phone Phreaks

The "phone phreak" (phreak for short) is a specific breed of hacker. A phreak is someone who displays most of the characteristics of a hacker, but also has a specific interest in the phone system and the systems that support its operations. Additionally, most of the machines on the Internet, itself a piece of the Public Switched Network, are linked together through dedicated, commercial phone lines. A talented phreak is a threat to not only the phone system, but to the computer networks it supports.

There are two advantages of attacking systems through the phone system. The first advantage is that, phone system attack are hard to trace. It is possible to make connections through multiple switching units or to use unlisted or unused phone numbers to confound a tracing effort. Also by being in the phone system, it is sometimes possible to monitor the phone company to see if a trace is initiated.

The second advantage to using the phone system is that a sophisticated host machine is not needed to originate an attack nor is direct access to the network to which the target system is attached. A simple dumb terminal connected to a modem can be used to initiate an attack. Often, an attack consists of several hops, a procedure whereby one system is broken into and from that system another system is broken into, etc. This again makes tracing more difficult.

3.4 Trends for the Future

3.4.1 Configuration Errors and Passwords

Today, desktop workstations are becoming the tool of more and more scientists and professionals. Without proper time and training to administer these systems, vulnerability to both internal and external attacks will increase. Workstations are usually administered by individuals whose primary job description is not the administration of the workstation. The workstation is merely a tool to assist in the performance of the actual job tasks. As a result, if the workstation is up and running, the individual is satisfied.

This neglectful and permissive attitude toward computer security can be very dangerous. This user attitude has resulted in poor usage of controls and selection of easily guessed passwords. As these users become, in effect, workstation administrators, this will be compounded by configuration errors and a lax attitude towards security bug fixes. To correct this, systems should be designed so that security is the default and personnel should be equipped with adequate tools to verify that their systems are secure.

Of course, even with proper training and adequate tools threats will remain. New security bugs and attack mechanisms will be employed. Proper channels do not currently exist in most organizations for the dissemination of security related information. If organizations do not place a high enough priority on computer security, the average system will continue to be at risk from external threats.

3.4.2 Internal Threats

System controls are not well matched to the average organization's security policy. As a direct result, the typical user is permitted to circumvent that policy on a frequent basis. The administrator is unable to enforce the policy because of the weak access controls, and cannot detect the violation of policy because of weak audit mechanisms. Even if the audit mechanisms are in place, the daunting volume of data produced makes it unlikely that the administrator will detect policy violations.

Ongoing research in integrity and intrusion detection promise to fill some of this gap. Until these research projects become available as products, systems will remain vulnerable to internal threats.

3.4.3 Connectivity

Connectivity allows the hacker unlimited, virtually untraceable access to computer systems. Registering a network host is akin to listing the system's modem phone numbers in the telephone directory. No one should do that without securing their modem lines (with dial-back modems or encryption units). Yet, most network hosts take no special security precautions for network access. They do not attempt to detect spoofing of systems; they do not limit the hosts that may access specific services.

A number of partial solutions to network security problems do exist. Examples include Kerberos, Secure NFS [GS91], RFC 931 authentication tools [Joh85] and "tcp wrapper" programs (access controls for network services with host granularity). However, these tools are not widely used because they are partial solutions or because they severely reduce functionality.

New solutions for organizations are becoming available, such as the Distributed Intrusion Detection System (DIDS) [L+92] or filtering network gateways. DIDS monitors activities on a subnet. The filtering gateways are designed to enforce an organization's network policy at the interface to the outside network. Such solutions may allow the organization to enjoy most (if not all) of the benefits of network access but limit the hackers' access.

3.4.4 Information Dissemination

The Forum of Incident Response and Security Teams (FIRST), an organization whose members work together voluntarily to deal with computer security problems and their prevention, has established valuable channels for the dissemination of security information. It is now possible to obtain security bug fix information in a timely fashion. The percentage of system administrators receiving this information is still low, but is improving daily.

Hackers continue to make better use of the information channels than the security community. Publications such as "Phrack" and "2600" are well established and move information effectively throughout the hacking community. Bulletin boards and Internet archive sites are available to disseminate virus code, hacking information, and hacking tools.

4 Summary

Poor administrative practices and the lack of education, tools, and controls combine to leave the average system vulnerable to attack. Research promises to alleviate the inadequate supply of tools and applicable controls. These controls, however, tend to be add-on controls. There is a need for the delivery of secure systems, rather than the ability to build one from parts. The average administrator has little inclination to perform these modifications, and no idea how to perform them.

The joint NIST/NSA Federal Criteria project holds the most promise to drive the creation of reasonably secure systems. By building upon the various criteria projects that precede it (the TCSEC, the ITSEC, and the Canadian criteria), this project intends to address security requirements for commercial systems in a meaningful way. The initial version, which will focus on criteria for operating systems, will include extensions/enhancements in integrity, communications, and other areas. Future versions will address criteria for distributed systems.

Extensive connectivity increases system access for hackers. Until standards become widely used, network security will continue to be handled on a system by system basis. The problem can be expected to increase if and when the Integrated Systems Digital Network (ISDN) is implemented without appropriate security capabilities.

A promising note for the future does exist. Multiple sets of tools do not need to be developed in order to solve each of the potential threats to a system. Many of the controls that will stop one type of attack on a system will be beneficial against many other forms of attack. The challenge is to determine what is the minimum set of controls necessary to protect a system with an acceptable degree of assurance.

References

[CON91] Computer security: Hackers penetrate DoD computer systems. Testimony before the Subcommittee on Government Information and Regulation, Committee on Government Affairs, United States Senate, Washington DC, November 1991.

[Den90] Peter Denning. *Computers Under Attack: Intruders, Worms, and Viruses.* ACM Press, 1990.

[GS91] Simon Garfinkel and Eugene Spafford. *Practical UNIX Security.* O'Reilly & Associates, Inc., 1991.

[HM91] Katie Hafner and John Markoff. *Cyberpunk: Outlaws and Hackers on the Computer Frontier.* Simon and Schuster, 1991.

[Hof90] Lance Hoffman. *Rogue Programs: Viruses, Worms, and Trojan Horses.* Van Norstrand Reinhold, 1990.

[HP91] Benjamin Hsaio and W. Timothy Polk. Computer-assisted audit techniques for unix. In *14th Department of Energy Computer Security Group Conference*, 1991.

[Joh85] Mike St. Johns. Rfc 931: Authentication server, January 1985.

[L+92] Theresa Lunt et al. A real-time Intrusion-Detection Expert System (IDES). In *Final Technical Report for SRI Project 6784.* SRI International, 1992.

[Qua90] John Quarterman. *The Matrix - Computer Networks and Conferencing Systems Worldwide.* Digital Press, 1990.

[S+89] Eugene Spafford et al. *Computer Viruses: Dealing with Electronic Vandalism and Programmed Threats.* ADAPSO, 1989.

[S+91] Steven R. Snapp et al. DIDS (Distributed Intrusion Detection System) - motivation, architecture, and an early prototype. In *Proceedings of the 14th National Computer Security Conference*, 1991.

[SH82] John F. Shoch and Jon A. Hupp. The "worm" programs - early experience with a distributed computation. *Association for Computing Machinery*, 25(3), March 1982.

[Spa89] Eugene Spafford. The internet worm program: An analysis. *Computer Communication Review*, 19(1), January 1989.

[Wac91] John Wack. Establishing a Computer Security Incident Response Capability (CSIRC). NIST Special Publication 800-3, National Institute of Standards and Technology, 1991.