Table of Contents

1 - Introduction

1.1 Background 3
1.2 Purpose 3
1.3 Scope 3
1.4 Relationship of Incident Handling Guidelines to CIAC Workshop 4
1.5 Revision of Incident Handling Guidelines 4

2 - Responding to Incidents

2.1 Definitions 5
2.2 Causes of Incidents 5
2.3 Avenues of Attacks 5
2.4 Effects of an Attack 6
2.5 Incentives for Efficient Incident Handling 6
2.6 Priorities in Incident Handling 7
2.7 Stages /Steps of Incident Handling 7
2.7.1 Protection 7
2.7.2 Identification 8
2.7.3 Containment 9
2.7.4 Eradication 9
2.7.5 Recovery 10
2.7.6 Follow-up 10
2.8 Division of Responsibility 10
2.9 Reporting of Incidents 11
2.10 CIAC's Charter and Role 15

3 - Responding to Viruses

3.1 Introduction 16
3.2 Definition of Computer Virus 16
3.3 Overview of Virus Actions 17
3.4 History of Viruses 17
3.5 Basic Mechanisms of Viruses 18
3.6 Common Viruses and their Symptoms 19
3.7 How Viruses Work 25
3.8 Virus Weaknesses 27
3.9 Virus Platforms 30
3.10 Responding to a Virus Infection 30
3.10.1 Protection 31
3.10.2 Identification 33
3.10.3 Containment 37
3.10.4 Eradication 37
3.10.5 Recovery 38
3.10.6 Follow-up 38

4 - Responding to Worm Attacks

4.1 Introduction 39
4.2 Definition of Worm 39
4.3 A Brief History of Worms 39
4.4 Considerations in Worm Attacks 39

4.5 Basic Mechanisms of Worms 41
4.6 Recommended Steps for Responding 42
4.6.1 Protection 42
4.6.2 Identification 43
4.6.3 Containment 43
4.6.4 Eradication 44
4.6.5 Recovery 44
4.6.6 Follow-up 45
4.7 Final Words about Worm Attacks 46

5 - Responding to Hacker/Cracker Attacks

5.1 Introduction 47
5.2 Division of Responsibility 47
5.3 Recommended Steps for Responding 47
5.3.1 Protection 47
5.3.2 Identification 49
5.3.3 Containment 50
5.3.4 Eradication 51
5.3.5 Recovery 52
5.3.6 Follow-up 52
5.4 Automated Hacker/Cracker Attacks 52
5.5 More on Passwords 53

6 - Vulnerabilities

6.1 Introduction 55
6.2 Types of Vulnerabilities 55
6.3 UNIX Vulnerabilities 56
6.4 VM Vulnerabilities 56
6.5 VMS Vulnerabilities 57
6.6 MVS Vulnerabilities 57
6.7 File Protections 57
6.8 Concluding Comments 57

7 - Incident Handling Tools

7.1 Types of Tools Available 59
7.2 System Security Tools 60
7.3 Authentication Tools 60
7.4 Intrusion Detection Tools 61

8 - Exercises and Scenarios 62

References 67

1 - Introduction

1.1 Background

Responding to computer security incidents is generally not a simple matter. This activity requires technical knowledge, communication and coordination among staff responding to the incident, and adherence to applicable Department of Energy (DOE) orders on computer security (i.e., DOE Orders 1360.2A and 5637.1 for unclassified and classified computing, respectively). Incidents over the last few years indicate that, if anything, responding to incidents is increasingly more complex. There have been, for example, not one, but three largescale worm attacks since 1988. Intrusions into machines are a serious concern, and increasing sophistication and collaboration among network attackers poses considerable threat to the integrity of computing resources. Viruses continue to infect MS-DOS and Macintosh computers, despite widespread availability of virus detection and eradication software.

1.2 Purpose

These incident handling guidelines repeatedly mention the importance of establishing and following written procedures for handling incidents. There are several reasons for developing such guidelines:

1. The procedures in these guidelines will help site computer security personnel develop more complete and effective contingency response procedures.

2. There is usually some degree of confusion during an incident. Written procedures for handling incidents minimize confusion.

3. During times of increased stress (e.g., during a fast-moving incident), people's attention usually becomes more narrow, and memory is more prone to failure. Without a written procedure, the likelihood that someone responding to an incident will respond ineffectively increases.

4. Written procedures can help computer security personnel understand and consider the legal aspects of responding to attacks on computer systems.

1.3 Scope

These guidelines do not comprise an exhaustive set of incident handling procedures. A lengthy set of guidelines would be too intimidating to read and to incorporate into site contingency response plans. The discipline of responding to incidents is also very much in its infancy. Because so much is yet to be learned about handling incidents, this version of these guidelines will necessarily lack some degree of sophistication and detail. In addition, it is impossible to specify specific technical procedures for responding to the many types and versions of computer systems within DOE. This guidelines document contains basic information about responding to incidents which can be used, regardless of hardware platform or operating system. For specific technical details necessary to implement many of the recommendations in these guidelines, consult your system administrator or vendor.

NOTE: These guidelines are written for someone who has a working knowledge of the field of computer science. The beginner in computer science will do well to refer to an introductory text in this area to understand many of the computer science concepts (e.g., functionality of operating systems) mentioned but not explained in these guidelines.

1.4 Relationship of Incident Handling Guidelines to CIAC Workshop

Since its inception in early 1989, the DOE's Computer Incident Advisory Capability (CIAC) team from Lawrence Livermore National Laboratory has assisted many sites in dealing with a large variety of incidents. These guidelines reflect the corporate experience of the CIAC team. The content of these guidelines also parallels the content of the two-day CIAC workshop, "Responding to Incidents," and are designed to help those unable to attend the workshop become familiar with the content of the workshop and participate in workshop exercises. For more information about the CIAC workshop, contact:

Thomas A. Longstaff
University of California
Lawrence Livermore National Laboratory
P.O. Box 808, L-619
Livermore, CA 94550
(415) 423-4416 or FTS 543-4416
longstaf@pantera.llnl.gov

1.5 Revision of Incident Handling Guidelines

These guidelines will periodically be revised and upgraded. Send comments and feedback to:

E. Eugene Schultz
University of California
Lawrence Livermore National Laboratory
P.O. Box 808, L-619
Livermore, CA 94550
(415) 422-8193
gschultz@pantera.llnl.gov

UNIX is a registered trademark of AT&T.
VMS and DECNET are registered trademarks of Digital Equipment Corporation.
Macintosh and Virus Rx are registered trademarks for Apple Computer Inc.
Amiga is a registered trademark for Commodore Business Machines.
PC and IBM Scan are registered trademarks for International Business Machines.
PC tools is a registered trademark for Central Point Software.
Sidekick is a registered trademark for Borland International.
Anti-Viral is a registered trademark for Softhansa of Berlin.
Anti-Virus is a registered trademark for OS Software.
Code Safe is a registered trademark of ChrisWare.
Data Physician, VIRHUNT and RESSCAN are registered trademarks for Digital Dispatch, Inc.
Ingres is a registered trademark for Ingres Corporation.
Oracle is a registered trademark for Oracle Corporation.
Port of Entry is a registered trademark for Sector Technologies.
Virex is a registered trademark for CE Software.
VirusSafe is a registered trademark for ComNetCo.

2 - Introduction

2.1 Definitions

The title of this document includes the term incident. In its most broad sense, incident refers to an adverse event in a computer system or the threat of such an event occurring. Penetrations into computer systems or hoaxes, therefore, fall within the definition of incident. However, not all incidents are of interest to the computer security community. Damage to systems due to power outages, for example, is not an incident within the domain of computer security. For the purpose of this guidelines document, therefore, the term incident implies an incident related to computer security. The term incident encompasses the following general categories of adverse events:

1. Compromise of integrity (e.g., when a virus infects a file)
2. Denial of resource (e.g., when an attacker has set a system to single user mode, locking out all other users)
3. Penetration (e.g., when a worm has breached a system)
4. Misuse (e.g., when an intruder makes unauthorized use of an account, or insider incidents)
5. Damage (e.g., when a virus scrambles a file or makes a hard disk inoperable)
6. Intrusions (e.g, hacker/cracker incidents)

In contrast, an event is any noticeable occurrence in a system. A system crash and an unusual graphic display are both events. Events may lead personnel to determine that an incident is occurring, although closer investigation of events may or may not indicate that something adverse from the standpoint of computer security is occurring.

2.2 Causes of Incidents

There are at least four generic causes of computer security incidents. The first is malicious code, such as viruses, worms, trojan horses, time bombs, and pests. (Viruses, worms, etc. will be explained in detail shortly.) The second is called procedural failures or improper acts. Examples include transmitting information from a classified to an unclassified system or revealing one's password to someone else. A third cause can be called intrusions or breakins. For example, an intruder might bypass a system's authentication procedure, might attack a system using provided services (e.g., using UNIX commands such as sendmail, finger, or rsh), or might "snoop" (e.g., wiretapping, network monitoring, or electronic monitoring of terminals). Finally, an incident might be caused by an insider attack.

2.3 Avenues of Attacks

Attacks originate through certain avenues or routes. If a system were locked in a vault with security personnel surrounding it, and if the system were not connected to any other system or network, there would be virtually no avenue of attack. More typically, however, there are numerous avenues of attack, including:

1. Local networks

2. Devices connected illegally, such as a non-approved connection to a local network

3. Gateways, which allow an attack to come from outside a network

4. Attacks through communications devices, e.g., modems, telebits, and integrated systems digital networks (ISDN's)

5. Shared disks (e.g., transmission of a virus from one machine to another)

6. Downloaded or pirated software (e.g., from an electronic bulletin board)

7. Others, including direct physical access by an attacker

2.4 Effects of an Attack

There are at least five effects of attacks which compromise computer security:

1. Denial of service--Examples include network jamming, introducing fraudulent packets, and system crashes and/or poor system performance, in which people are unable to effectively use computing resources.

2. Unauthorized use or misuse of computing systems--System resources are tied up by someone who does not have permission to use the system, or are used for non-approved purposes (e.g., playing games on a U.S. Government computer).

3. Loss or alteration of data or programs--An example would be an attacker who penetrates a UNIX system, then modifies the wtemp program so that the intrusion will not be detected. Another example would be an infection by the SCORES virus on Macintosh computers resulting in loss of data (the Scores virus may cause destruction of files).

4. Compromise of protected data--An example in a multi-level system would be sending a classified file to an uncleared user's account.

5. Loss of trust in computing systems--Users may become hesitant to use a computing system which is plagued by virus infections, for example.

2.5 Incentives for Efficient Incident Handling

Learning to respond efficiently to an incident is important for numerous reasons. The most important benefit is directly to human beings--preventing loss of human life. Some computing systems are life critical systems, systems on which human life depends (e.g., by controlling some aspects of surgery in a hospital or assisting air traffic controllers).

An important but often overlooked benefit is an economic one. Having both technical and managerial personnel respond to an incident requires considerable resources, resources which could be utilized more profitably if an incident did not require their services. If these personnel are trained to handle an incident efficiently, less of their time is required to deal with that incident.

A third benefit is protecting classified, sensitive, and/or proprietary information. One of the major dangers of a computer security incident is that information may be irrecoverably lost. The

loss of classified information jeopardizes our Nation's security. Efficient incident handling minimizes this danger.

A fourth benefit is related to public relations. News about computer security incidents tends to be damaging to an organization's stature among current or potential clients. Efficient incident handling minimizes the potential for negative exposure.

A final benefit of efficient incident handling is related to legal issues. It is possible that in the near future organizations and/or institutions may be sued because one of their nodes was used to launch a network attack. In a similar vein, people who develop patches or workarounds may be sued if the patches or workarounds are ineffective, resulting in damage to systems, or if the patches or workarounds themselves damage systems. Knowing about operating system vulnerabilities and patterns of attacks, then taking appropriate measures is critical to circumventing possible legal problems.

2.6 Priorities in Incident Handling

It is important to prioritize actions to be taken during an incident well in advance of the time an incident occurs. Sometimes an incident may be so complex that it is impossible to do everything at once to respond to it; priorities are essential. Although priorities will vary from organization-to-organization and institution-to-institution, the following suggested priorities serve as a starting point for defining an organization/institution's response:

- Priority one -- protect human life and people's safety; human life always has precedence over all other considerations
- Priority two -- protect classified and/or sensitive data; our National safety and security is second only to protecting human life
- Priority three -- protect other data, including proprietary, scientific, managerial and other data, because loss of data is costly in terms of resources
- Priority four --prevent damage to systems (e.g., loss or alteration of system files, damage to disk drives, etc.); damage to systems can result in costly down time and recovery
- Priority five -- minimize disruption of computing resources; it is better in many cases to shut a system down or disconnect from a network than to risk damage to data and/or systems

An important implication for defining priorities is that once human life and National security considerations have been addressed, it is generally more important to save data than system software and/or hardware. Although it is undesirable to have any damage and/or loss during an incident, systems can be replaced; the loss or compromise of data (especially classified data), however, is usually not an acceptable outcome under any circumstances.

2.7 Stages /Steps of Incident Handling

There are at least six identifiable stages of response to a computer security incident. Knowing about each stage can help you respond more methodically (and thus efficiently), and can also help you develop a more complete contingency response plan for your organization.

2.7.1 Protection

Part of handling an incident is being prepared to respond before the incident occurs. This includes establishing a suitable level of protections, so that if the incident

becomes severe, the damage which can occur is limited. Protection includes preparing incident handling guidelines or a contingency response plan for your organization or site. Having written plans eliminates much of the ambiguity which occurs during an incident, and will lead to a more appropriate and thorough set of responses. Second, part of protection is preparing a method of notification, so you will know who to call and what everyone's phone number is. For example, every member of DOE's Computer Incident Advisory Capability (CIAC) Team carries a card with every other team member's work and home phone numbers, as well as beeper numbers. Third, your organization or site should establish backup procedures for every machine and system. Having backups eliminates much of the threat of even a severe incident, in that backups preclude data loss. Fourth, you should set up secure systems. This involves eliminating vulnerabilities, establishing an effective password policy, and other procedures, all of which will be explained later in this document. Finally, conducting training activities is part of protection. It is important, for example, to conduct "dry runs," in which your computer security personnel, system administrators, and managers simulate handling an incident.

2.7.2 Identification

This stage involves determining exactly what the problem is. Of course, many if not most signs often associated with virus infections, system intrusions, etc. are simply anomalies, such as hardware failures. To assist in identifying whether there really is an incident, it is usually helpful to obtain and use any detection software which may be available. For example, widely available software packages can greatly assist someone who thinks there may be a virus in a Macintosh computer. Audit information is also extremely useful, especially in determining whether there is a network attack. It is extremely important to obtain a system snapshot as soon as one suspects that something is wrong. Many incidents cause a dynamic chain of events to occur, and an initial system snapshot may do more good in identifying the problem and any source of attack than most other actions which can be taken at this stage. Finally, it is important to start a log book. Recording system events, telephone conversations, time stamps, etc. can lead to a more rapid and systematic identification of the problem, and is the basis for subsequent stages of incident handling.

There are certain indications or "symptoms" of an incident which deserve special attention:

- System crashes
- New user accounts (e.g., the account RUMPLESTILTSKIN has unexplainedly been created), or high activity on an account that has had virtually no activity for months
- New files (usually with novel or strange file names, such as data.xx or k)
- Accounting discrepancies (e.g., in a UNIX system you might notice that the accounting file called /usr/admin/lastlog has shrunk, something that should make you very suspicious that there may be an intruder)
- Changes in file lengths or dates (e.g., a user should be suspicious if s/he observes that the .EXE files in an MS DOS computer have unexplainedly grown by over 1800 bytes)
- Attempts to write to system (e.g., a system manager notices that a privileged user in a VMS system is attempting to alter RIGHTSLIST.DAT)

- Data modification or deletion (e.g., files start to disappear)

- Denial of service (e.g., a system manager and all other users become locked out of a UNIX system, which has been changed to single user mode)

- Unexplained, poor system performance (e.g., system response time becomes unusually slow)

- Anomalies (e.g., "GOTCHA" is displayed on a display terminal or there are frequent, unexplained "beeps")

- Suspicious probes (e.g., there are numerous unsuccessful login attempts from another node)

- Suspicious browsing (e.g., someone becomes a root user on a UNIX system and accesses file after file in one user's account, then another's)

None of these indications is absolute "proof" that an incident is occurring, nor are all of these indications normally observed when an incident occurs. If you observe any of these indications, however, it is important to suspect that an incident might be occurring, and act accordingly. There is no formula for determining with 100 percent accuracy that an incident is occurring (possible exception: when a virus detection package indicates that your machine has the nVIR virus and you confirm this by examining contents of the nVIR resource in your Macintosh computer, you can be very certain that your machine is infected). It is best at this point to collaborate with other technical and computer security personnel to make a corporate decision about whether an incident is occurring.

2.7.3 Containment

The third stage of incident handling, containment, should occur only if the indications observed during stage two conclusively show that an incident is occurring. The purpose of the containment stage is to limit the extent of an attack. For example, if there is a virus which can damage the disk drive after a fixed number of boots, it is important to take action to prevent this damage from occurring. Similarly, it is important to limit the spread of a worm attack on a network as quickly as possible. An essential part of containment is decision making, i.e., determining whether to shut a system down, to disconnect from a network, to monitor system or network activity, to set traps, to disable functions such as remote file transfer on a UNIX system, etc.). Sometimes this decision is trivial; shut the system down if the system is classified or sensitive, or if proprietary information is at risk! In other cases, it is worthwhile to risk having some damage to the system if keeping the system up might enable you to identify an intruder.

The third stage, containment, should involve carrying out predetermined procedures. Your organization or site should, for example, define acceptable risks in dealing with an incident, and should prescribe specific actions and strategies accordingly. Finally, notification of cognizant authorities should occur during this stage (see Section 2.9 of this document).

2.7.4 Eradication

Once an incident has been detected, it is important to first think about containing the incident. Once the incident has been contained, it is now time to eradicate the cause. Software may be available to help you in this effort. For example, eradication software is available to eliminate most viruses which infect small systems. If any bogus files have been created, it is time to delete them at this point. In the case of virus

infections, it is important to clean and reformat any disks containing infected files. (Important note: if a classified system has been infected with a virus, it is essential to follow guiadance issued by the DOE Office of Safeguards and Security. We also strongly advise in this case that a low-level format be performed to ensure the integrity of classified information.) Finally, ensure that all backups are clean. Many systems infected with viruses become periodically reinfected simply because people do not systematically eradicate the virus from backups.

5. Recovery. Once the cause of an incident has been eradicated, the recovery phase defines the next stage of action. The goal of recovery to to return the system to normal. In the case of a network-based attack, it is important to install patches for any operating system vulnerability which was exploited. Again, there is special guidance for recovery in classified computing systems from the DOE Office of Safeguards and Security.

6. Follow-up. One of the most important stages of responding to incidents is also the most often omitted---the follow-up stage. This stage is important because it helps those involved in handling the incident develop a set of "lessons learned" to improve future performance in such situations. This stage also provides information which justifies an organization's computer security effort to management, and yields information which may be essential in legal proceedings.

The most important element of the follow-up stage is performing a post-mortem analysis. Exactly what happened, and at what times? How well did the staff involved in dealing with the incident do? What kind of information did the staff need sooner, and how could they have gotten that information sooner? What would the staff do differently next time? A follow-up report is valuable in that it provides a reference to be used in case of other, similar incidents. Creating a formal chronology of events (including time stamps) is also important for legal reasons. Similarly, it is also important to as quickly as possible obtain a monetary estimate of the amount of damage the incident caused in terms of any loss of software and files, hardware damage, and manpower costs to restore altered files, reconfigure affected systems, and so forth. This estimate may become the basis for subsequent prosecution activity by the FBI, U.S. Attorney General's Office, etc.

2.8 Division of Responsibility

One of the most important strategies for responding to an incident is to divide responsibility among the people involved. This strategy, which should be incorporated into every organization/site's incident handling procedures, minimizes duplication of effort, and ensures that the most important tasks will become completed in a timely manner. In a similar vein, some responsibilities are more appropriate for some personnel than for others. At a minimum, for example, technical responsibilities should be assigned to technically qualified personnel. We recommend the following division of responsibilities:

1. End users. End users should notify site computer security authorities of problems they notice. Throughout this document end users will be called the "eyes and ears" of computer security--they frequently notice problems before anyone else does. End users also should be told to not shut down a system or disconnect from a network without first consulting site authorities. It many cases it is desirable to keep a system running to avoid subsequent damage or to stay on a network to attempt to localize attacks to that machine instead of risking more widespread attacks. End users should also monitor and log system activities (at least until more personnel are available to assist in handling the incident), and obtain a system snapshot or a copy of infected files. Finally, end users should follow their organization/site's reporting procedures (see Section 2.9).

There are also some "do not's" for end users. End users should not, for instance, modify the system or application software. Not only could modification produce damage, but it could also destroy evidence and/or important clues about the goals of an attack as well as to encourage an attacker to penetrate other machines. It is also important for end users to avoid talking to the media. Anything other than a well-deliberated and coordinated response to the media through one's public affairs office is likely to be a public relations catastrophe--avoid this mistake.

2. Technical personnel. Technical personnel have the responsibility of verifying that an incident is occurring. Once it is determined that an incident is in progress, technical personnel should follow the applicable recovery procedures which have been prepared in advance. It is important to divide work whenever possible--avoid duplication of effort, and ensure that the very high priority work gets done. Thus, it is important for technical personnel to communicate among each other frequently and systematically. Technical personnel should also log all activities (including recording time stamps), and gather information from system logs. If an incident in a classified system is occurring, it is the responsibility of technical personnel to protect classified data, and to follow containment, eradication and recovery procedures pertaining to a classified system. Finally, technical personnel should avoid talking to the media except through a public affairs office spokesperson.

3. Site managers. Site managers need to prepare everyone else for handling an incident. This process includes identifying key technical personnel, maintaining contact lists, and ensuring that effective written procedures are written, validated through training exercises, and widely available. Site managers also need to obtain and coordinate technical support during an incident. This may involve arranging for "shifts," since some incidents last for days and even weeks. Finally, site managers need to coordinate with the organization/site's public affairs office to determine what information needs to be released to the media.

2.9 Reporting of Incidents

DOE Orders 5637.1 and 1360.2A cover required reporting procedures for incidents in classified and unclassified computing systems, respectively. These Orders contain many details concerning time requirements for reporting and what must be reported. In overview, reporting of both classified and unclassified incidents must occur through a chain of cognizant individuals.

IMPORTANT NOTE: Information about incidents and incident handling that pertains to classified DOE systems or may have implications for such systems is subject to the guidance in CG-SS-2 , the soon to be released DOE Classification Guide on Safeguards and Security. Persons who handle incident reports at DOE sites should familiarize themselves with this guidance in order to avoid an inadvertent violation or compromise of classified information.

For a classified incident, an end user or system manager (if there is such a person) reports the incident to the Computer Security System Officer (CSSO). The CSSO then works with the Computer Security Site Manager (CSSM) on initial notification to DOE and to produce a historical report on the incident. The CSSM then notifies the Computer Security Operations Manager (CSOM) at the applicable DOE Operations Office. The CSOM then notifies the Computer Security Program Manager (CSPM) at DOE Headquarters.

Suppose, for example, that a virus incident occurs in a classified MS-DOS system. If there is a system manager, the end user would inform the system manager . The system manager (or end user, if there is no system manager) would assess the technical problem to determine whether there is an incident. If the MS-DOS computer is infected with a virus, the system manager would categorize the incident as a virus infection, then inform the CSSO. The CSSO would notify the CSSM, who, in turn, would notify the CSOM that an incident is in progress. The

CSSM and CSSO would jointly produce a historical report. The CSOM would inform the CSPM in a quarterly report of important and routine incidents.

For an unclassified incident, the end user notifies the site's Computer Protection Program Manager (CPPM). The CPPM in turn notifies the Computer Protection Program Coordinator (CPPC) at the DOE Operations Office for that site. The CPPC then notifies the head of unclassified computer security at the Office of Computer Security Information Resources Management (IRM) at DOE Headquarters.

For example, if a virus infected an unclassified MS-DOS system, the user would notify the system manager (if there were one). The user or system manager would then assess the technical problem, categorize the incident as a virus incident, and inform the CPPM of the situation. The CPPM would notify the CPPC, who would notify IRM that an incident is in progress. The CPPC and the head of unclassified computer security at IRM would jointly determine support needed to deal with the virus outbreak, and would determine how the incident would affect classified computing within DOE. The CPPC would also advise IRM of findings, measures taken, and a plan of action.

Example of Reporting Classified Incident within DOE

Example of Reporting Unclassified Incident within DOE

2.10 CIAC's Charter and Role

CIAC is a resource for personnel at DOE sites faced with computer security incidents. The CIAC team consists of computer scientists from the University of California's Lawrence Livermore National Laboratory who are available to assist on a 24-hour per day basis. This assistance might be in the form of direct technical assistance, information, or the names of technical personnel who can help with very specialized and complex problems. CIAC coordinates efforts by DOE sites, other Government agencies and teams, and vendors. CIAC also alerts and advises sites about vulnerabilities, potential attacks, and other threats. Team members have written these guidelines, and are developing databases of vulnerabilities, viruses, known incidents, etc. CIAC also engages in training and awareness activities, and has developed the two-day workshop on which this document is based.

3 - Responding to Viruses

3.1 Introduction

"It seems as if you can't trust anybody these days, at least when it comes to software for MS-DOS systems and Macintosh computers. " You have probably heard a friend relate how his or her computer became infected with a computer virus shortly after a new program that somebody else loaned him or her was run. Even your own machine may, in fact, have been affected. Worse yet, so-called shrink wrap software is sometimes infected with computer viruses, as are pre-configured computers from manufacturers. Did the salesperson who just demonstrated software on your machine also infect your machine? You may not have thought about this possibility, but vendor demos are becoming an increasing problem in terms of virus infections.

Fear of computer viruses will cause otherwise rational people to take wholly irrational actions. Their fear is understandable; there is a real, evident threat of computer virus infections. Viruses spread quickly and are often well hidden, and viruses can cause irreparable damage (although this is not usually the case). However, most people do not really understand what a computer virus is, and how to prevent, identify, contain and eradicate these fragments of harmful code. This chapter will define what a computer virus is and will describe how most viruses work. This chapter will also acquaint you with computer viruses on common platforms, and will apply our computer incident handling methodology to the problem of dealing with a computer infected by a virus.

3.2 Definition of Computer Virus

A computer virus is a segment of unwanted, self replicating code that (similar to its biological counterpart for which it is named) spreads from one computer to another by attaching itself to an application program or other executable system component. In this manner the virus survives and reproduces. A computer virus has several major characteristics, including:

- o Self-replication, that is, it can make exact copies of itself
- o Requires a host program to survive, and lives within the host program
- o Usurps control of the host programs functions for the virus' benefit

A computer virus spreads through an infected host, usually an infected application. The infection spreads to a new computer by executing the infected application on the new system. An infected program can be transmitted to the target system by disk or tape, modem or local network, or any other means and is executed on that system. The application then becomes infected, and the next time it is run, the virus will attempt to again replicate itself.

While the computer virus and its biological counterpart seem similar, there are specific differences. While a biological virus is a process of nature, a computer virus was created by some malefactor. The virus is simply a computer program written by somebody as an academic exercise, as a form a vandalism, or possibly to gain recognition among peers. We are familiar with the motives of conventional vandals (e.g., those who break windows at a school) and terrorists, but creating of viruses seems to motivate another kind of person. Virus writers appear to want a programming challenge, but misdirect their creative ability. Many virus writers also fail to anticipate consequences of their behavior--the disruption and even destruction their "prank" will cause.

3.3 Overview of Virus Actions

There is no such thing as a generic or prototypic type of virus. Every virus is to some degree unique. There is some commonality, however, in the way viruses work:

- o Infection of the host and its programs
- o Survival and propagation of the viral code
- o Transmission of the infection to other host applications
- o Possible destructive side effects caused by the virus.

Most often the virus code will search for other programs that thus far have not been infected. The virus will then select one of these programs and append itself to it as demonstrated below. The virus will next modify the program so that it will execute the virus part of the code before the actual program is executed. In this way the virus will "get a chance to live" before the actual program is run. With many viruses the user will probably not notice anything out of the ordinary initially because the requested program will run, and will cause no interference with the executing program.

At this point virus action diverges broadly. A virus may replicate virtually forever, causing no really harmful damage (as in the case of the WDEF virus on Macintosh computers). In a few cases, the virus may be downright malicious, damaging applications, data files, or even the operating system, as in the case of the Disk Killer virus on MS-DOS computers. Viruses have even been created to steal data and transmit it during non-working hours to another machine, where the virus author can peruse the information at his/her leisure.

3.4 History of Computer Viruses

The idea of self replicating code has been around since the late 1940's (see history shown below). John von Neuman's creating the first self-reproducing program stimulated considerable interest in this topic. At first developing self replicating code was an academic exercise. Then it expanded into a game in Bell Laboratory's "core wars." When main-frames were the only computing resource, the goal was to battle for core, the main memory of the computer. Two opponent programmers were each given a segment of memory and allowed to launch one program. The computer then divided compute time equally, first allowing one program to execute, then the other (in a time-sharing manner). Each program had two goals, the first to live, and the second to destroy the other program. Part of a successful survival strategy was for the program to make multiple copies of itself. During the 1960's and 1970's there was also a number of books on the topic of self-replicating code and the possibility of using this code maliciously. Interest in self-replicating code then waned temporarily. With the increased interconnectivity between systems and availability of powerful computers in the last decade, however, there has been increased interest in self- replicating code, and, in particular, viruses. As personal computers became popular, the idea naturally extended itself to creating life across separate disconnected machines. About the same time that personal computers were becoming numerous, the previously centralized computer was becoming part of a growing network of available hosts. The growth of the network allowed a third form of life to spawn, the worm (which will be covered in Chapter 5 of this document). Malicious code, both self-replicating and non self-replicating, began to be found. An issue of Scientific American even contained an article about viruses in 1984.

A Brief History of Computer Viruses

1949 - John von Neuman A self reproducing program in Theory and Organization of Complicated Automata
1960 - Bell Laboratory's Core Wars
1972 - Dave Gerrold, When Harley Was One, First edition only
1975 - John Brunner:,Shockwave Rider
1977 - Thomas J. Ryan, The Adolescence of P-1
1980 - IBM Trojan. All IBM 4341's stopped at 7:30 AM on 11 April 1980. Logic bomb planted by disgruntled employee
1984 - Scientific American, A.K. Dewdney. First article written on computer viruses
1987 - Pakistani or Brain Virus, Lehigh Virus, Jerusalem Virus, IBM Christmas Virus
1988 - Morris Worm
1990 - Many known viruses on microcomputers and mainframes.

3.5 Basic Mechanisms of Viruses

Viruses generally work through a four stage process. First, the virus must be transmitted to the host computer through an infected application or other executable program. Once the infected program has been executed on the host computer, the virus enters the second stage. The virus must now try to survive and propagate through the computer's file system. Frequently, the virus will infect both applications and the operating system. Sometimes the virus will also become resident in memory, so that it can infect disks when they are inserted (as in the case of WDEF) or cause side effects, such as characters falling to the bottom of the screen, as in the case of the Cascade virus on personal computers. The virus survives by disguising its presence (or at least not immediately announcing its presence), and by replicating as quickly as possible. As it replicates it also attempts to infect floppies or network mounted file systems, etc., so that it can continue to propagate between computer systems. This is the third stage of the virus' life, the attempt to infect other systems by retransmission of the viral code. At some point in the life of many viruses, they enter the fourth and final stage. Either by malicious or poor design practices, the virus causes interruptions in the processing of data or the corruption or destruction of data.

3.6

Common Macintosh viruses

Names	Resources Type ID Size	Comments
“Peace” , MacMag virus	INIT 6 unknown DR	First virus on the Macintosh. Displays “Peace on Earth” message on March 2, 1988 and removes itself the next day. Distributed via a HyperCard stack. An XCMD installed the INIT into the System. Its presence did cause problems with some programs. Remove the INIT Resource with ResEdit.
nVIR Family of viruses nVIRa, nVIRb, nVIRc, AIDS, Hpat, MEV#, FLU, Jude (J-nVIR)	See below for specifics	This virus is named after its resources. There are two general variations of this virus a and b. See below for specifics.
nVIRa	nVIR infected system file: INIT 32 366 nVIR 0 2 nVIR 1 378 nVIR 4 372 nVIR 5 8 nVIR 6 868 nVIR 7 1562 nVIR infected Application: CODE256 372 nVIR 1 378 nVIR 2 8 nVIR 3 366 nVIR 6 868 nVIR 7 1562	In the case of both nVIRa and nVIRb, the virus first copies the INIT 32 resource into the System file, there after any system restart causes the virus to become memory resident after which any application launched will become infected. After a set time period nVIRa will announce its presence by beeping or saying “Don’t Panic”, if MacInTalk is present
nVIRb, Hpat, MEV#, AIDS	nVIR infected system file: INIT 32 416 nVIR 0 2 nVIR 1 428 nVIR 4 422 nVIR 5 8 nVIR 6 66 nVIR 7 2106 nVIR infected Application: CODE256 422 nVIR 1 428 nVIR 2 8 nVIR 3 416 nVIR 6 66 nVIR 7 2106	nVIRb is similar to nVIRa, however it doesn’t use MacInTalk. To detect either strain open the suspected application with ResEdit and look for a CODE 256 resource of size 372 or 422 for nVIRa or nVIRb respectively. MEV# and AIDS are identical to nVIRb, except the resource type will read MEV# or AIDS instead of nVIR. Hpat is the same as nVIRb except it adds a code of 255 instead of 256 and has a resourc type of Hpat.
Dukakis		Written in HyperTalk on a HyperCard stack called "NEWAPP.STK". Adds itself to Home Card and other stacks. Flashes a message saying, "Dukakis for President in 88, Peace on Earth, and have a nice day." This virus can be eliminated by using the Hypertalk editor and removing the well commented virus code.

Scores, NASA	Scores infected system files: INIT 6 7721, 2, 3 INIT 10 1 020 0, 1, 4 INIT 17 4801, 3 atpl 128 2410 0, 1, 4 DATA4007026 0, 1 Corresponding system files are: 0 = Desktop, 1 = System 2 = Note Pad, 3 = Scrapbook Files 4 = Scores Scores infected applications: CODE n+1 7026 Where n = the id of the first unused CODE resource.	Most virulent of the known viruses. Written by disgruntled employee of EDS. Replicates rapidly and can cause unpredictable behavior. Looks for files of Creator VULT, TYPE ERIC. (Note the desktop is ERIK, not ERIC)These files were internal projects of EDS and not actual programs. Does not intentionally do damage, but causes problems in other programs by being present.
Sexy Ladies Trojan		Not a virus, but a Trojan Horse. Given away at 1988 San Fransisco MacWorld Expo;erased whatever hard disk or floppy disk it was on when it was launched.
INIT29	INIT29 infected system files: INIT 29 712 INIT29 infected applications: CODE n+1 712 Where n = the id of the first unused CODE resource.	Infects disks the moment they are inserted int a Mac with an infected system. Its sole purpose is to replicate itself , but has the unfortunate side effect of obliterating any legitimate INIT 29 in any file it infects. It will infect any file with resources. This makes it th only virus (so far) which will attack documents as well as applications.
WDEF-A, WDEF-B[1]	Infects the Desktop. Need not run an application to be infected, but can spread on disk insertion.	Causes both the Macintosh IIci and the portable to crash, severe performance problems on AppleTalk networks with AppleShare servers, frequent crashes when users try to save files i applications under MultiFinder, problems with the proper display of font styles (the outline style in particular), damage to disks, Macintoshes with 8 megabytes of memory to crash, Erratic system behavior due to incompatibility with the "Virtual" INIT from Connectix.

[1] Must use GateKeeper Aid, Disinfectant 2.0, or other software that specifically recognizes WDEF to eradicate this virus.

MDEF, Garfield virus	Virus Detective DA, search string: Resource MDEF & Name "Garfield" Resource MDEF & ID = 5378	MDEF or Garfield spreads through system and application files, causes serious damage to the menu system. Causes both the Macintosh 128K and 512K to crash. On the Macintosh IIci and IIfx, the MDEF virus spreads from infected applications to uninfected system files, but does not propagate from infected systems to uninfected applications. Vaccine will inform the user that the system file has been infected, but is only partially effective in preventing this virus from infecting the system file, this may damage the system.
Steroid trojan horse QuickDraw Accelerator	TYPE: INIT CREATOR: QDAC Code Size: 1080 Data Size: 267 ID: 148 Name: QuickDraw Accelerator File Name: " Steroid" (First 2 characters are ASCII 1)	The purported purpose of Steroid is to make QuickDraw run faster on computers with 9 inch screens. Steroid is actually an INIT that contains malicious code to check for the system date and to erase all mounted disks if this date is July 1, 1990 or afterwards. Examine system folder; if Steroid is there, save a copy and then drag the icon to the trash folder and empty trash, restart the Mac.

Common Macintosh Viruses

Common MS-DOS Viruses

Name	Strains	Mode of Infection, Attributes	Size	Potential Damage	Comments
Pakistani Brain, Ashar, Shoe	8	RES,FDB	Boot (7K)	BOT,RUN,DAT,FMT	Some variations corrupt File Allocation Table. Infects Floppies Only.
Merritt, Alameda, Yale, Golden Gate	8	RES,FDB	Boot (1K)	BOT	Golden Gate variation will reformat drive C: after n infections. Infects Floppies Only.
South African, Friday 13 the COM	2	COM	512	PRG	Deletes host program if run on Friday the 13th
Lehigh	1	RES,CC	OVER	PRG,FMT	
Vienna, 648, Lisbon, Vienna-B, Austrian, Dos-62, Unesco	12	COM	648	PRG	62-B variant deletes infected files. Program errors cause file fatalities.
Jerusalem, New Jerusalem, Jerusalem-B, Payday, Black hole, Blackbox	13	RES, COM,EXE,OVR	1808	RUN,PRG	
April-1-COM, Suriv-01	1	RES,COM,	897	RUN,PRG	Message: "APRIL 1ST HA HA HA HA YOU HAVE A VIRUS" , Deletes host file.
APRIL-1-EXE, Suriv-02	1	RES,EXE	1488	RUN,PRG	Same message, but causes system lock-up instead of deleting host file.
Suriv-03	1	RES,COM,EXE,OVR		RUN,PRG	
Ping-Pong,Bouncing Ball, Italian	3	RES,FDB	Boot	RUN,BOT	Bouncing dot appears on screen. No other intentiona damage.
Ping-Pong-B		RES,FDB,HDB	Boot	RUN,BOT	same as Ping-Pong
Marijuana, Stoned,New Zealand, Australian	2	RES,FDB,HDP	Boot (1K)	RUN,BOT,FAT	Message: "Your computer is now stoned. Legalize Marijuana." Affects partition record on hard disk. No intentional damage is done.
Agiplan, 1536, Zero Bug, Palette	1	RES,COM	1536	RUN,PRG	
1701,Cascade, Autumn, Blackjack, Falling Leaves	7	ENC,RES,COM	1701	RUN,PRG	Characters fall to the bottom of the screen. Both 1701 and 1704 versions are also known as: Autumn, Blackjack, Falling Leaves
1704,Cascade-B		ENC,RES,COM	1704	RUN,PRG	
1704 Format		ENC,RES,COM	1704	RUN,PRG,FMT	
Oropax, Music, Musician	1	RES,COM	2756+	RUN,PRG	Plays musical melodies repeatedly. File size varies from 2756 to 2860
DenZuk, Venezuelan, Search	6	RES, FDB	Boot	RUN,BOT	Displays a purple DENZUK graphic on a CGA, EGA or VGA screen. Overwrites FAT. SYS variation modifies the SYS command to preserve viral code against destruction. Infects Floppies Only
Dbase	1	RES, COM	1864	DAT,RUN,PRG	
DataCrime, 1280	4	COM,ENC	1280	PRG,FMT/FAT	Attempts to format the disk, on some systems this will fail, but the attempt to format will destroy the FAT or worse. Will require a low level format to repair.
DataCrime-B		COM,ENC	1168	PRG,FMT/FAT	Same as above
DataCrime II		COM,EXE,ENC	1514	PRG,FMT/FAT	Same as above
DataCrime II-B		ENC,CC,COM,EXE	1917	PRG,FMT	
405	1	COM	Over	PRG	
FuManchu	1	RES,COM,EXE,OVR	2086	RUN,PRG	
Ohio	1	RES,FDB	Boot	BOT	
Icelandic, Disk-eating virus	3	RES,EXE	642	RUN,PRG	Marks hard disk clusters as bad
Icelandic II		RES,EXE	632	RUN,PRG	
Saratoga		RES,EXE	661	RUN,PRG	

December 24th		RES,EXE	848-863	RUN,PRG	infects one out of every ten .EXE files run. If an infected file is run on December 24th it will stop any other program run later, displaying the message "Gledileg jol" (Merry Christmas)
Israeli Boot, Swap	1	RES,FDB	Boot	BOT	Reverses order of letters typed creating typographical errors.
Typo	1	RES,FDB,HDB	Boot	BOT,RUN	
Traceback, 3066	1	RES,COM,EXE	3066	PRG	
Disk Killer	1	RES,FDB,HDB	Boot	BOT,RUN,PRG,DAT,FMT	
Vacsina	1	RES,COM,EXE,OVR	1206	RUN,PRG	
Mix1	1	RES,EXE	1618	RUN,PRG	
Syslock, 3551	1	COM,EXE,ENC	3551	PRG,DAT	Some claim this is 3555 bytes in size.
Pentagon	1	FDB	Boot	BOT	
Dark Avenger	1	RES,CC,COM,EXEOVR	1800	FAT,RUN,PRG,	Infects every executable file that is opened.
AIDS	1	COM	Over	PRG	
2930	1	RES,COM,EXE	2930	PRG	
Yankee Doodle	1	RES,COM,EXE	2885	RUN,PRG	
Alabama	1	RES,EXE	1560	FAT,RUN,PRG	
Ghost	2	COM	2351	BOT,PRG	
Ghost - Boot		RES,FDB,HDB	Boot	BOT,RUN	
Typo/Fumble	1	RES,COM	867	RUN,PRG	
Sunday	1	RES,COM,EXE,OVR	1636	RUN,PRG	
Do-Nothing	1	COM	608	PRG	
Sylvia/Holland	1	RES,COM	1332	PRG	
Amstrad	1	COM	847	PRG	Adds code to front of any .COM file in the current directory. The virus contains an advertisement for Amstrad computers.
Devil's Dance	1	RES,COM	941	RUN,PRG,DAT,FAT	
4096	1	RES,CC,COM,EXE,OVR	4096	RUN,PRG,DAT,FAT	Infects every executable file that is opened.
Chaos	1	RES,FDB,HDB	Boot	BOT,RUN,PRG,FAT	
W13	2	RES,COM	534	PRG	
		RES,COM	507	PRG	A virus from Poland. Infected programs are padded so their length becomes a multiple of 512 bytes. Then the virus adds 637 bytes to the end of the file. It will also intercept any disk write and change it into a disk read.
Vcomm	1	RES,EXE	637	PRG,	
Perfume (alias 765 or "4711")		RES,COM,CC	765	PRG,RUN	May ask a question, won't run infected file unless the answer is "4711" (the name of a perfume). In a common variant, the questions have been overwritten with garbage.
Virus-90	1	RES,COM	857	PRG	

The Columns

Name: List the most common name of the virus and any aliases. Also lists direct mutations of the virus.

No. of known strains: List the number of known relatives in this family of viruses. Since people mutate viruses frequently this number is not guaranteed to be accurate

Size: The infected file will increase in size by this number of byte
Note some viruses are not as consistent as others. See the comm
Over: means that the virus overwrites the application and does nc its code to the application.

Comments: Notes on unique characteristics of the virus.

Mode of infection: Attributes: Describes the modus operandi of the virus:

EXE	Infects .EXE files
COM	Infects .COM files
OVR	Infects program overlay files
FDB	Infects Floppy disk boot sectors
HDB	Infects Hard disk boot sectors
HDP	Infects Hard disk partition table

Potential Damage: Describes what kind of damage the virus will attempt to do when it strikes.

BOT	Overwrites (corrupts) the boot sector
RUN	Interferes with application while running
PRG	Corrupts program or overlay files
DAT	Corrupts data files
FMT	Attempts to format the hard disk
FAT	Corrupts file linkage or the FAT table
RES	Memory resident

Comments: Notes on unique characteristics of the virus.

Common MS-DOS viruses

3.7 How Viruses Work

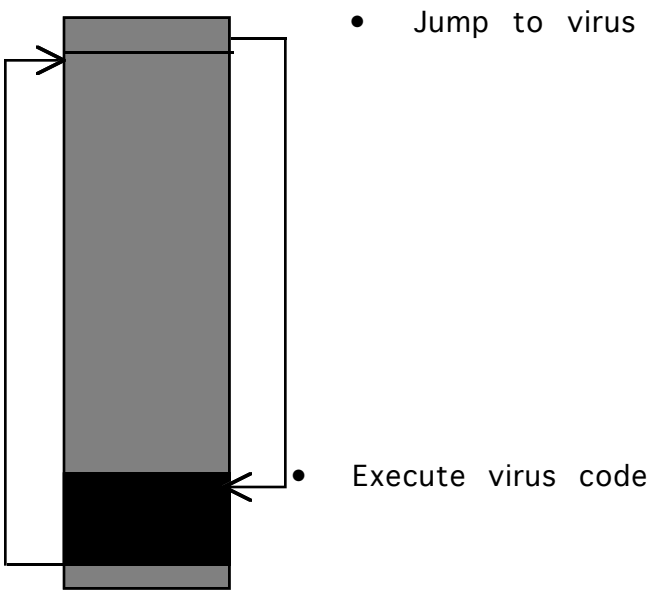
An overview of virus mechanisms has already been presented. In review, once a computer virus is transmitted to a new host via an infected application, it can then infect the new system. After infecting the new system, the virus will generally attempt to stay hidden by employing a variety of strategies. At some point the virus may activate and can cause damage to the host. This section is designed to help you understand in far greater detail how viruses work, including how virus actions are based on assumptions made about the host.

1. Transmission. Computer viruses are programs. They may enter a computer by any route used to transfer a computer program. The most commonly used route is via a floppy disk. Viruses can also be transmitted via a modem or a local area network. We are aware of at least one computer manufacturer from Taiwan that even shipped computers infected with the Disk Killer virus-- quite a time saver! Shrink wrap software is becoming increasingly unsafe in terms of risk of virus infection, and public domain and shareware software always carries a risk.

The source of infection can vary. Some people's machines become infected when someone borrows their computer for "just a minute to print a report," or "to show you something very quickly." Others become infected when they purchase shrink-wrap software. Even software houses are not immune from infection. These distributors can pass software on to unsuspecting buyers, or (more commonly)will take back used or demo'd software and repackage it, complete with a new shrink-wrap outer coating. The disk may look as good as new, but the previous user's machine may have been infected with a virus. You may innocently use software from a shared file server that has been infected, or use a shared computer connected to a central resource such as a laser-printer that has been infected.

2. Infection. Once an infected program is copied to the hard disk or onto one of the floppy disk drives, the target machine is usually not infected yet. In most cases the user must actually run the infected software to release the virus on that machine.

3. Execution. When an infected program is run, the computer virus interrupts the normal execution of the program. The virus quickly takes control, executes its viral code, then returns execution to the original program. Most viruses execute so quickly that a user will not notice the delay. Knowing about the structure of a program helps in understanding how a delay may not be detectable. A program consists of a linear list of machine op-codes that the computer will execute one at a time (see figure on next page) When a virus infects an executable, it often adds the viral code to the bottom of the program. This increases the program's length and changes its cyclic redundancy check (CRC), or in unused stack space, only changes the CRC. The virus must then patch the first line of the program with a jump statement to the first line of the viral code. Finally, the last line of the viral code is modified to return execution to the original program.



Computer virus infection

4. Masquerading. Once a virus infects a system it must try to survive both accidental encounters with the operating system or the user as well as purposeful search and destroy operations the user may attempt. To accomplish this the virus may use several tactics. The first is to lay dormant. A biological virus that kills its host immediately has little chance to spread; the same principle applies to computer viruses. A computer virus may wait for some specific trigger point, e.g., a designated number of boots on the infected machine to occur before it causes any destructive or disruptive effects to occur. This gives the virus time to spread throughout the host system as well as to other systems with which the host has had contact before someone is likely to notice the virus and eradicate it.

Viruses often employ some form of encryption to make them harder to discover. For example, the DataCrime (Columbus Day) virus is almost completely encrypted except for a small section of code that decrypts the rest of the program. A search program could easily miss this viral code. Besides trying to hide from detection programs, viruses can actively fight detection by modifying common detection programs. Viruses also find innovative ways to hide, such as in the stack space of other programs, or by marking disk sectors as bad and storing the virus code in these sectors.

5. Activation. The activation mechanism checks for the occurrence of a specific event. Not all viruses have an activation mechanism. If the virus has one and the specific event occurs, however, the virus will try to accomplish the objective of its author. A virus may activate based on two major types of events. If the activation mechanism checks for a specific date and time, the virus is called a time bomb. Examples of viruses which have time bomb mechanisms are DataCrime (the Columbus Day) virus and the Jerusalem virus on MS-DOS computers. If the virus checks for specific circumstances, such as the number of system boots, then the virus is called a logic bomb. An example of a logic bomb is the Lehigh-2 virus, which counts the number of infections that have occurred. Unfortunately, viruses can activate on the basis of either logic or a time bomb, or both, depending on the skill and motives of the

virus author. Other viruses are never intended to do intentional harm. An example is the WDEF virus for the Macintosh computers. However, even these programs can create real havoc such as slowing down or disrupting a computer system, or can even result in loss of data.

6. Damage. The actual damage a computer virus may do is extremely variable. Many viruses try to damage the data on the host's hard disk. Remember from Chapter 2 that losing data on a computer system generally is more of a loss than damage to the system itself and/or its hardware. If there are no backups, any data loss can be permanent. Other viruses are intended to damage specific targets. For example, the Scores virus was written by a disgruntled employee who was searching for a specific program used only internally within a particular company. The program looks for the files of creator VULT and of type ERIC. Theoretically, Scores should not affect other computers, but due to incompatibilities on systems infected by Scores, Scores can damage files. Other viruses accidently do damage due to bugs/programming errors. No matter what the damage, however, a computer virus will steal cycles, that is, it takes away time and memory that other programs should have.

3.8 Virus Weaknesses

Viruses may be a formidable challenge, but they are not undefeatable for three reasons. First, viruses must make assumptions about their hosts. If these assumptions fail, the computer will crash due to system incompatibilities, the virus may be easily detected, and/or the virus will fail to spread. Second, a virus may try to masquerade its presence, but it can never disguise itself perfectly, because viruses need a place to live and a time to run. Many viruses increase the original file size of the hosts they infect. In addition, they usually modify the CRC, a special code used to check the integrity of files, leaving viruses prone to detection. Finally, the third virus weakness is that because a small portion of virus code must remain constant, viruses leave fingerprints, i.e., observable characteristics that allow them to be identified and subsequently eradicated.

Most anti-viral software takes advantage of one of these three weaknesses. The most common scanning software looks for fingerprints within all system executables by searching all mounted media for a specific binary pattern for each known virus. The problem with scanners is that they only work for known viruses. Another class of anti-viral software uses the other two weaknesses of viruses to detect changes in the file size or file CRC. Even unknown viruses will probably have to modify one of these two file characteristics. If the viruses do, they can be detected.

NOTE: You can implement some anti-viral protection without using anti-viral software. Since most viruses affect file size, you can keep a record of original files sizes, then can compare current file size if you ever suspect that something is amiss.

3.9 Virus Platforms

We will look at two platforms that have been the most common target for computer viruses, MS-DOS based computers, and the Apple Macintosh computers. In addition we will look at the possibility of computer viruses infecting multi-user computers, such as UNIX- or VMS-based machines.

1. MS DOS computers. Computers running the MS-DOS platform present a tempting target to the virus author. One major weakness of the MS-DOS machine and its non-compatible cousins such as the Atari and Macintosh is that these types of machines run in a single state. Running in a single state means that the operating system is not protected from its applications by the hardware, as is the case with UNIX and other multi-user operating systems. In a single-state machine, for example, when an application gets control, as signified by the program counter pointing to the

application code, it never has to give the program counter back. In multi-user machines, a particular application may try to keep the program counter, but the operating system can take it back, giving the user or software a chance to kill the errant process.

Since any application can take control of the personal computer at any point, MS-DOS is susceptable to infection in any of the following:

- o boot record
- o system files
- o application files

Boot infectors attach viral code to the boot record or even replace the boot record altogether. The boot record exists on every single MS-DOS disk, and contains just enough information for the computer to load the rest of the operating system. If one of these boot sector viruses infects a machine, the virus will take control when the machine is initially booted, and will control the machine from that point on. These programs often stay in memory and infect other programs as well as possibly disrupting the system substantially.

A virus that stays in memory is very similar to a memory resident utility such as Sidekick or PCtools. This type of virus uses one of the many ways a program can terminate and stay resident (TSR) to stay active and control the program counter and the interrupts. A resident virus can monitor all interrupts and trap all system events. If the virus is monitoring a system, it can be programmed to take a number of sophisticated measures to accomplish its task. For example, it can hide if it appears that a detection program is being run. It can modify or destroy data, or create havoc with system activities and performance.

System infectors infect the operating system (which is actually a very special application). Examples in the PC arena include IBMDOS.COM, IBMBIOS.COM and in the case of the IBM version of PC-DOS, COMMAND.COM . Many are not familiar with the first two files mentioned. When you format a disk with the /s option, as in:

```
FORMAT A: /S
```

IBMDOS.COM, IBMBIOS.COM are created on the new disk as hidden files. Some viruses specialize in infecting these three special files. Most often, if the hidden system files or the equally hidden boot record is infected, the virus' effects will not be noticeable to the user. Because IBMDOS.COM and IBMBIOS.COM are hidden files, most users will not recognize if the size of these files grows. Just as with boot infectors, the system infectors can monitor all system activity, look for new disks to infect, and even take total control of the systems activities.

Application infectors can infect any application program. The first viruses infected only .COM files because the code was somewhat more simple to write. Later some viruses were modified (or mutated, to use the analogy of biological viruses) to infect only .EXE files. The next extension of this malicious work was evidenced by newer viruses that could infect both .EXE and .COM files. The code for infecting both .EXE and .COM files was even more complicated. An example of this is the DataCrime (Columbus Day) virus.

DataCrime, 1280	4	COM,ENC	1280	PRG,FMT/FAT	Attempts to format the disk, on some systems this will fail, but the attempt to format will destroy the FAT or worse. Will require a low level format to repair.
DataCrime-B		COM,ENC	1168	PRG,FMT/FAT	Same as above
DataCrime II		COM,EXE,ENC	1514	PRG,FMT/FAT	Same as above
DataCrime II-B		ENC,CC,COM,EXE	1917	PRG,FMT	

The DataCrime Virus Family

DataCrime was originally written as a .COM infector. Later it was mutated to become both a .COM and .EXE infector. We call slightly different viruses by the same family name if the code is substantially the same. In the case of the DataCrime virus, the major infectious and destructive portions of the code remained the same. Only the method of attaching the code was changed by determining if the file was a .EXE or .COM file, then branching to the appropriate routine to cause the infection. Because the code is fundamentally similar, if your computer were infected by DataCrime, the outward symptoms of the infection would be virtually indistinguishable. However, close analysis would reveal whether .EXE or .COM or both types of files were infected, and, thus, the particular family member which infected your machine.

2. Macintosh computers. The operating system of the Macintosh is fundamentally different in architecture from MS-DOS machines. Nevertheless, Macintosh computers are also especially vulnerable to virus infections because they also have single-state execution. In fact, the application structure of Macintosh programs and the event queue structure of the operating system greatly facilitate adding a virus to a Macintosh application. An application on the Macintosh can be infected by adding viral code to the resource bundle of the application. The virus can then have itself scheduled in the event queue. In this manner the virus can obtain a certain portion of central processing unit (CPU) time for its own malicious activities.

In the case of the nVIR virus, for example, the virus adds a CODE resource #256. The virus then patches the jump table so that it will get executed first. After infecting the computer, the virus returns control to the application, so the user is not aware of the infection. After the viral code is executed, nVIR first copies the INIT 32 resource into the system file. Now any system restart causes the virus to become memory resident, after which any application launched will become infected. After a set time period, nVIRa announces its presence by beeping or, if MacInTalk is present, saying “Don’t Panic.”

Similar to the boot record of a MS-DOS PC, the Macintosh has a hidden file called the desktop. However, the function of the desktop is to remember the position of all the files, and the windows of that disk. The desktop is a program and its run anytime something that affects the desktop is changed, e.g., opening a folder or disk, or moving a file. The WDEF virus takes advantage of this operation. Even if you only accept data-disks from other people, and even if you never run unsafe software, your machine can nevertheless get WDEF. WDEF spreads from sharing floppy disks, including any Macintosh disk containing data or applications. If an infected disk is inserted into a machine and the desktop is run, e.g., by copying a file, other desktops on the machine, e.g., the hard-disk desktop, will be infected.

3. Mainframe computers. Larger computers, such as those running UNIX or VMS, can also in theory be infected. However, although several claims of virus infections of mainframe computers have been made periodically, no mainframe viruses have as yet been detected and publically verified. If a mainframe virus were to be released, it could reproduce itself in three possible ways, namely by:

- o Modifying source code
- o Patching executable code
- o Infecting data files

The first method would involve modifying the source code of an application. Since programs are often shared as source, a virus could be written to modify the source of other source programs on the computer. An especially ripe target would be the compiler itself, such as cc (the C compiler).

Another method would be to infect executable applications, much as PC viruses do, by adding executable code to an application binary. Fortunately, some link editors in UNIX and VMS make code unrelocateable, making this method of infecting a mainframe computer nearly impossible.

Another possible avenue of attack would be to modify a data file. Many data files contain executable meta-commands that are interpreted by a pre-processor. If a preprocessor or screen formatting process such as curses is interpreting data, it could be vulnerable to attack.

Because mainframe viruses do not appear to exist outside of the closed, laboratory environment at this time, greater detail about these viruses is not warranted. Fortunately, because mainframes are not single-state machines and because their operating and file systems are more complex, mainframes make difficult targets for viruses. As mentioned above, executable code is often unrelocateable on a mainframe. In addition, code usually cannot be executed in a data segment. Finally, to be modified, files must have write access by the program . The built-in safeguards of these operating systems provides some protection against the ravages of potential viruses.

3.10 Responding to a Virus Infection

Some primary considerations for handling a computer virus infection include:

- o Determining that your machine has actually been infected
- o Containment and isolation of the infected host(s)
- o Evaluation of the actual virus, and forming an eradication plan
- o Cleaning up all media with which the infected host had contact
- o Notifying all computer users that have had contact with the system

Dealing with a virus will be greatly aided by your using a notebook. You should start your notebook by recording your current system configuration. List any network services you use as well. Then jot down symptoms that your computer experienced before and after you suspected a virus infection. Also create a list of media or network contact that you may have made recently. If you use scanning software, include printouts from the scanner with your notebook. If you are running more advanced, database-driven virus software, include any warning messages, or printouts displayed.

The first thing you should remember when you suspect that your computer has been infected by a virus is that errors in applications, misconfigurations, or interactions between hardware and

software can create pseudosymptoms of viruses. For example, you may load your applications from a shared file server for execution on your local machine. Often an advantage to this system is that the application can be upgraded for all users at one time. Unfortunately, the reverse side is that the upgrade may not be compatible with your particular machine. If you are not aware of software upgrades and new incompatibilities created by the software, you may be experiencing a psuedosymptom of a computer virus. Another psuedosymptom frequently observed is due to media, especially hard disks, approaching capacity. Some applications do not have a fail soft mechanism for a full hard disk. Defective media, again especially hard-disks, can also cause data corruption. Ironically, people have had to deal with scrambled file allocation tables (FATs) on hard-disks and floppies long before computer viruses were a problem!

We recommend that you maintain a curious and somewhat skeptical mind-set while exploring the symptoms you are experiencing. First, compare your symptoms to the known symptoms listed in the appendix of this document. Do the observed symptoms match those of any known virus? If not, your computer may still have a virus, because new viruses are being discovered almost every month. Contrawise, your machine may not have a virus. It may pay to pursue other strategies as well. This chapter will help you understand more about handling a computer virus incident, and how to apply these methods to arrive at a valid answer.

3.10.1 Protection

The first and most important stage of computer virus incident handling is to as much as possible avoid becoming infected in the first place. At the same time, it is important to be prepared if your computer does become infected by a virus.

Backups of all of your valuable programs and data are the best insurance you can purchase for a speedy recovery from a virus infection. Backups help in three ways. First, if a virus destroys the data on your hard-disk, you may lose many months if not years of work. Why risk your labor to a faulty mechanical drive, or a computer virus? If data or information is important to you, back-it up. Secondly, backups help you return your machine to an earlier state. If you suspect that a virus has erased your hard-disk, it is too late to make a determination after it happens . A backup (even if infected) will allow you to restore your machine for additional analysis. Finally, even if your backup is infected, all viruses known to date do not infect data, so you can still restore the infected back-up. Copy the data to clean media, then reformat the disk. (Note: with this procedure, you cannot save your applications).

Some programs can restore an application after the application has become infected. However, it can be risky trusting an application that has been repaired by a virus-tool, especially with PCs. A better approach is to protect (write-protect) your distribution disks, and make backup copies of every application when you take it out of its box. Except when there are certain copy protection schemes, their is absolutely no reason why you should insert a write-enabled disk into a computer to copy an application to your hard-disk. Every application that you insert in your machine should be write-protected. No known virus can circumvent the hardware that does write protection, so if it is write-protected, it is safe!

This suggests procedures that are similar to a good insurance plan. First, always write-protect your applications as soon as you get them. Then make a backup copy from the write-protected original. Next periodically backup your data. With your applications safely protected on write-protected media, you don't necessarily need to backup all of your applications. If you back-up only data with an incremental back-up, it will take very little time or effort.

Virus protection programs can also be very useful. A whole new software market has been created by the need for anti-viral software products. One of the first types of

products is a guardian program that warns users when an application tries to write to the hard disk. While useful in analysis, this kind of software is usually not good for day-to-day operation because it tends to generate a large number of false alarms (false detections). Consequently, the user may bypass this package with the result that there may be no protection against virus infections. The next class of anti-viral software to come on the market recently is the scanner. A virus scanner or hunter quickly examines each byte of every application on the media you are examining. The scanner has a database of virus signatures. If this type of product finds the signature (e.g., an identifying string) embedded in an application, it gives a warning, and (usually) some option to remove the virus. For example, a fictitious scanner might display the following:

Anti-vir 23.3 running.....

Black-hole virus found in the application runme
Do you want to remove this virus? (y/n) y

Black-hole removed

Scanning software can make your job easier. However, it is dangerous to depend on such a product, because scanning software has many pitfalls. One problem (possibly the most critical one) is that scanning software can find only the viruses it knows about. If a new virus or one of which the author of the scanning program was not aware infects your machine, the scanner will pass it by without a warning. A second class of software actually watches for changes to your file system. This kind of software will detect new viruses, since it does not depend on file signatures. Examples of commercially available software in this class includes Data Physician by Digital Dispatch Inc. and CodeSafe by ChrisWare.

Some Common Anti-Viral Software Packages

- PC Prevention Programs
  - flushot+
  - DDI Data Physician
  - Chris Ware CodeSafe

- MAC Prevention Programs
  - Virex
  - Dukakis Vaccine
  - GateKeeper/Gatekeeper Aid
  - RWatcher
  - Vaccine
  - VirusWarning INIT

When installing new applications, do not take unnecessary chances. Whether you download software or purchase it shrink-wrapped, you should now know that you should suspect this software. Examine the software before you load it. Look at the application package. Does it look professionally prepared, and is the documentation typeset? Read the license and at least some of the introductory text. Is this software suspicious?

For example here is a portion of a license agreement from a disk that purports to be an information package about the AIDS disease:

> ``In case of your breach of this license, PC Cyborg reserves the right to take any legal action necessary to recover any outstanding debts payable to the PC Cyborg Corp. and to use program mechanisms to insure termination of your use of these programs. The mechanisms will adversely affect other programs on your microcomputer.''

Does this look suspicious? It should. This is the license agreement that was enclosed with the PC Cyborg AIDS trojan. If loaded and executed, this trojan would hide all of your computer's files, and then would demand a payment to get them back. The message you would see after this malicious code had hidden the files would ask you to send a $387 fee to a Panama City address.

If you are an extra careful type of person, you might consider rigorously examining a working copy of a new application with something such as Norton or Mace Utilities. Look at the messages in the software. While you may not always be able to obtain conclusive evidence, this procedure will help you spot many viruses that use unencrypted message strings.

Before actually installing the program, you should use a virus scanning program to check the media. Another optional step is to use an isolated test machine, if you have one at your disposal. You should consider exercising the package on the test machine for at least 30 days before you install it on your operational computer. Before running the package, record the original file sizes and the CRCs, then compare them to current values.

To summarize, when you install new software onto your system you should take the following actions:

- o Back-up the computer
- o Write protect the original disks
- o Examine the software package
- o Run an anti-viral scanner on the distribution media
- o Install the software on a test machine
- o Install the package

3.10.2 Identification

Even if you follow the prevention strategy above, we cannot guarantee that you will never get a virus. If you suspect that your machine has become infected by a computer virus, the next step is to identify the problem. As discussed previously, not all problems (e.g., anomalies, poor performance) that you encounter with your computer will be a virus. In fact, polls in the industry suggest that the probability of becoming infected by a computer virus is only about four percent.

Before you proceed any further, you may want to create a fresh back-up of your system. This ensures that if you or the virus actually damage any data, you can return to a known state. Do not overwrite your old backup, but create a new one. Remember, even if your back-up is infected, you probably can recover the data files.

To identify the virus or problem,you should first record in your notebook the symptoms you have observed in the past and are observing now. Then you should look at the list of viruses and their symptoms (see Section 3.6) to determine if the symptoms you are currently observing in your computer match those of any given virus. If they do and you have eliminated other reasons for unusual system behavior, then you should proceed to the next step, containment.

Virus identification tools

Virus identification tools (see figure above) have become so useful that they are almost indispensable in the modern anti-viral arsenal. While tools are not absolutely necessary, they can make dealing with a virus considerably easier. Identification tools are not foolproof, since they can only identify the viruses that they are programmed to identify. For such viruses, however, these tools give almost positive proof that your machine is indeed infected. Many of the programs listed above can also help eliminate the virus and repair damage as well.

Let's look at a hypothetical example. Let's suppose you are about to use the shared MS-DOS computer in your office when you notice characters falling from their position on the screen, onto a heap at the bottom of the screen. You suspect something is amiss. Upon inspecting the tables in Section 3.6 of this document, you find the entry:

1701,Cascade, Autumn, Blackjack, Falling Leaves	7	ENC,RES,COM	1701	RUN,PRG	Characters fall to the bottom o the screen. Both 1701 and 1704 versions are also known as: Autumn, Blackjack, Falling Leaves

You now strongly suspect that your PC has the 1701 or Cascade virus. You also notice as you press the return key that the screen returns to normal, and you are in the middle of an application. You quit the application (e.g., Interleaf) and return to the DOS prompt. Just to be sure, you decide to try your trusty, if a little bit rusty, virus scanning program. You take the program and write-protect the disk so that it will not become infected. You now insert the disk into the A: drive and execute it. The drive whirls; you get reassuring words that its working; but upon completion the program indicates the machine is clean.

What can this mean? You look at the documentation for the scanning program. You find that the program detects the 1701 virus, so what could be wrong? Two things could be happening. First, this could be a mutant of the 1701 virus that your scanning tool is not programmed to detect. Second, you may not have a virus. Some screen-saving programs, such as the one in Interleaf, can have virtually the same behavior as some viruses. It pays to be careful, but in this case what appeared to be a virus infection was actually just the normal action of a program.

The previous example probably best illustrates why virus identification is still a "black-art." The best way to attack the problem of identification is with appropriate knowledge and procedures for dealing with viruses. You should know your platform, your applications, and how platforms and applications interact. Note strange behavior on your system and develop hypotheses to be tested.

Another example actually happened to a CIAC team member while he was getting an estimate to repair his car. The clerk was looking up some part numbers and availability of these parts on a Macintosh IIx. The CIAC team member noticed that the machine was slower than should be the case for a CPU such as the Macintosh IIx, and was also aware that the WDEF virus was becoming very prevalent in the San Francisco Bay Area. He asked the clerk if it would be o.k. to check the Macintosh for viruses. Using Disinfectant (a Macintosh virus detection tool), the CIAC team member found the WDEF-A virus on the machine, as well as all the rest of the Macintoshes in the immediate work area.

To repeat, software tools are very useful. Remember, however, that knowledge and good analysis procedures are at least as important in identifying and dealing with viruses.

3.10.3 Containment

A virus is not fundamentally like a worm that replicates independently of what users do. Responding to a virus, therefore, is not as dependent on the speed of your reaction (except, perhaps, when a virus attacks something such as a central file server machine). If you suspect your machine has a virus, do not automatically reach for the off switch on your computer. Turning your machine off is extremely unlikely to solve any problems. We recommend that unless you know it is better to turn your machine off, leave it on. If a virus has infected your computer and the virus counts the number of boots (waiting for the count to reach a certain number), your action might push the count over, causing it to attack your machine the next time you turn it on. For example, in the case of the Disk Killer virus it is actually better to let the virus finish its actions before you do anything. After the virus finishes damaging the data on your hard disk, Disk Killer actually writes information about what it affected . Some anti-viral utilities, therefore, can actually restore the disk. There are a few other viruses, however, for which the best response is to turn off your machine as fast as possible. This will give you a chance to recover the entire disk. Again, being familiar with viruses and their actions is a critical component of dealing with viruses.

When you suspect a computer virus has infected a computer, you should immediately establish a quarantine. Disconnect the infected machine from any network, then display a sign to caution people against using the computer. Once you have verified the infection, tighten the quarantine--do not let anyone use the computer until it is verifiably clean.

3.10.4 Eradication

Eradication is the process of removing a computer virus from your system. The surest way to recover from a computer virus is to start with a clean system again. A clean system is one in which the memory (both RAM and EEPROM) is clear, and the hard-disk has been low-level formatted. Since some viruses hide in the partition table or other specific portions of the disk, to fully recover it is important to erase the entire disk (including system- and hardware-specific areas such as the partition table).

Since a low-level format is completely destructive of all data on your disk, you will want to first make a full back-up of your data files. Some viruses, such as the Dukakis virus (which infects Hypercard stacks), actually infect data-files . This is, however, a rare case. In most cases data cannot be infected from currently known viruses. For the most part, therefore, you are safe from reinfection if you restore your data from an infected disk. However, the applications must come from clean source disks, not your hard disk if you wish to be safe.

It is important to be careful how you reformat your disk. You cannot simply issue the reformat command while the virus is in memory. You must power down the system, and then reboot from some trusted source, perhaps a diskette if the diskette boots first. Then issue the formatting instructions from that trusted diskette or other clean source.

Some anti-viral products will also remove known viruses from applications. Usually on Macintosh computers it is safe to allow these utilities to work. This is because the process of adding code in a Macintosh application is quite clean, and so is the removal. On the MS-DOS machines, however, automated recovery has mixed results. If you are unsure, check the documentation. Most anti-viral vendors are very straightforward in describing the particular viruses for which their products are effective.

NOTE: If a classified system has become infected by a virus, it is essential to follow guidance issued by the DOE Office of Safeguards and Security. In addition, we strongly recommend doing a low-level format on your classified system's hard disk.

3.10.5 Recovery

Recovery is the process of returning to normal operation. Once you have clean hardware, you can begin restoring your system. Take out your original write-protected applications disks and rebuild your file structure. If you have scanning software, scan your hard disk to ensure that none of the original disks were infected. Then carefully copy the data disk back to the hard disk. Don't allow any executable that might have found its way to the back-up disks to be copied onto the hard disk.

Scan one more time to make sure no infected applications made it back on the hard disk. You have now returned the file system to a clean state. If you practiced a good backup strategy, you should have all of your files and data as clean as new. If you are exhausted from all the work, you can take comfort that at least your hard disk is not fragmented. Loading all software over reduces hard disk fragmentation considerably.

There is one more recovery issue. Remember to check all backup floppy disks for virus infections! In many cases massive reinfections of a virus occur because someone forgets to disinfect backups. Any floppy disk which contains a virus should be erased and reformatted before it is used again. If this is not possible, use a dependable eradication program to remove or repair infected files.

NOTE: Again, if a classified system is involved, follow the guidance issued by the DOE Office of Safeguards and Security to ensure that classified data are not compromised.

3.10.6 Follow-up

Just because your system is now clean does not mean that all the work is over. What about all the other people with whom your system has had contact (e.g., vendors or clients)? People who may have mailed you an infected disk, and your co-workers should be told. If you don't practice this important follow-up, then the virus will probably spread more, and your system is very likely to become infected again. Inform others that you had a computer virus on your machine and that they should check theirs. Tell them how you discovered it, what the symptoms are, and how you recovered.

Now might be a good time to review your virus procedures and tighten scanning of all disks entering machines in the area, at least until the "coast is clear." How did the virus infection start? How adequate was the anti-virus software that you used? How many person-hours did this effort require? Was adequate technical assistance available? These questions must be answered and any open issues must be resolved before your follow-up activity is complete.

4 - Responding to Worm Attacks

4.1 Introduction

Imagine thousands of computers connected to a communication network suddenly all screeching to a halt. This actually happened in November, 1988, and the cause was a worm--the Internet (Morris) worm. Public awareness of a relatively unknown computer threat jumped to a high level, with national news coverage and front page newspaper articles. From this point in time onwards, computer worms have been a major concern of all computer users connected to networks.

This section will concentrate on computer worms. Worms will be defined, and a brief history of worm attacks will be given. The remainder of the section will outline a methodology for handling a worm event as it happens. The handling of a worm event follows the same pattern as other computer system attacks, and may be characterized by the six stages of incident handling described in the overview section above. This model will be used in describing the methodology for handling a worm attack.

4.2 Definition of Worm

A computer worm is a computer program in the category of malicious code with the following characteristics:

- o Is self-replicating
- o Infects hosts rather than programs (unlike a virus)
- o Attacks using automated hacking techniques to penetrate a host and begin execution
- o Exists as an autonomous process or set of processes

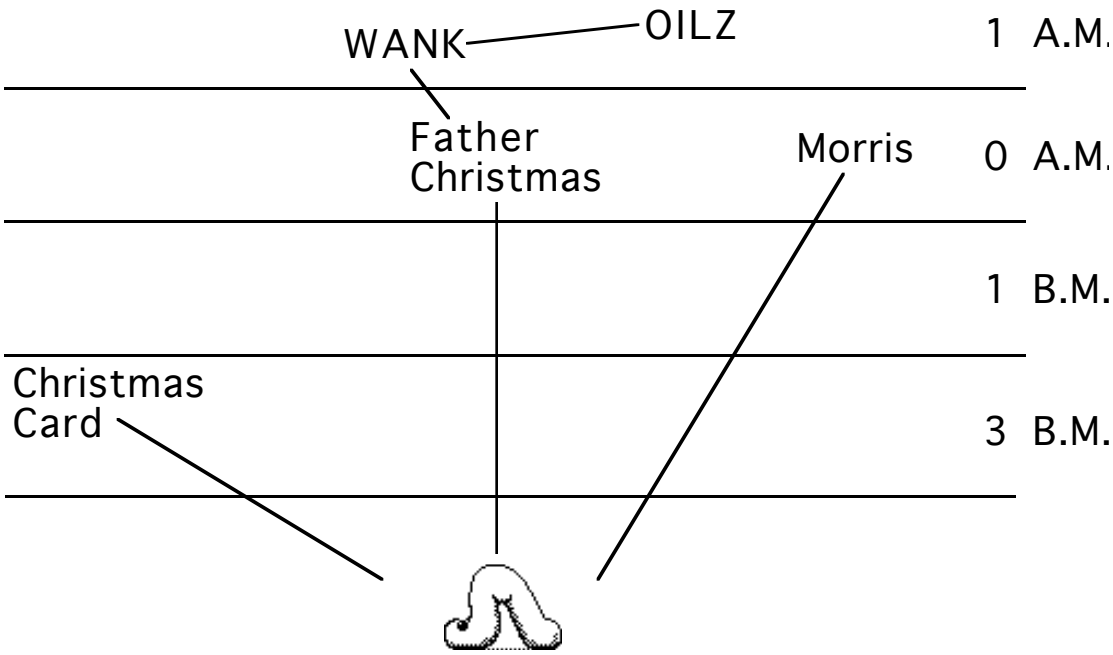
In general, a worm is very much like a virus, but with the primary distinction that a worm will infect a host rather than an application. A worm will spread using communication channels between hosts such as phone lines, networks, or gateways to access services on another host. The worm will then use these offered services to copy itself to the victim host and begin execution. The services used by the worm will vary, depending on the specific worm, but may include electronic mail, remote login, and remote file service.

The ability of worms to threaten thousands of computers is a direct result of the use of network communication between these computers. When computers were mostly isolated, worms could not easily jump between hosts. Today, the trend is for more and more computers and networks to be interconnected to provide easy access and distributed services. Worms take advantage of these advances in computer science to threaten the worldwide computing environment.

Once a worm has penetrated a system and begins execution, it may alter the operating system for its own use, cause damage, or simply access information and pass it back to the worm's creator. Previously known worms have stolen user accounts and passwords, installed trojan horse programs in the infected host, and caused denial of service by utilizing all of the available computing resources. To date, only the worms with relatively small disruptive potential have been released.

4.3 A Brief History of Worms

While the Morris Worm is the best known network based worm, it was not the first or only network worm. Because it is the best known, however, other worms are dated in relation to this one (see figure below).



A Chronology of Worms

The first recognized network-based worm was launched at Christmas time, 1987 and was known as the Christmas Card Worm. This worm attacked IBM mainframe computers running the VM/CMS operating system. It spread using the electronic mail system. When this "Christmas card" arrived at a person's account, it would exploit a feature that would automatically forward itself to everyone in the person's mailing list. This resulted in a denial of service over a large section of the internal BITNET of IBM due to the large number of these "Christmas cards" around the net. It did spread to the external BITNET, but was not successful due to the loose coupling outside of the IBM Corporation.

The Morris Worm has already been mentioned. It was released in November, 1988, and spread through a nationwide TCP/IP network known as Internet. Several operating systems, all variants of the UNIX operating system (such as BSD and Sun OS), were successfully attacked by this worm. Four vulnerabilities in UNIX services were exploited to spread the worm. One involved the electronic mail system (sendmail), two more involved an authentication bypass mechanism (.rhosts and hosts.equiv), and still another vulnerability involved a utility to find out who the users of a system are (finger). The worm would also attempt to guess passwords on a penetrated system and spread using the passwords it successfully guessed. The result of the Morris Worm was a tremendous slowdown of all successfully penetrated systems due to a bug in the code that caused the worm to replicate at a faster pace than intended. Thousands of hours of system administration nationwide was required to eradicate and recover from this worm.

Christmas, 1988 saw the first known VMS worm released. Named Father Christmas, this worm had many of the same characteristics as the Internet Worm, and was obviously influenced by the Internet Worm. The worm attacked a national DECNET, the Space Physics Analysis Network (SPAN), and used a few well-known exploitable features in VMS to copy itself between nodes and begin execution. This worm would send any user names and accounts to a known location for later retrieval by the author. This worm would first delete the file used to start the worm, thus leaving no trace in the file system of the worm's presence. The worm would send a message to each user on a system on Christmas Eve from "Father Christmas," the basis for this worm's name.

In 1989, another VMS worm attacked a number of nationwide DECNET systems. Known as the WANK/OILZ worm, this worm used many of the techniques found in other worms. WANK/OILZ was quite complex. It would guess passwords, install trojan horses in the penetrated system, add privileged accounts as back doors, and write annoying messages to users of remote systems. The first variant, the WANK worm, was quickly followed by the OILZ worm. The differences between the two was clearly intended to evade the common scripts used to eradicate and prevent against the WANK worm. The intended damage of this worm was severe. The worm would attempt to alter all of the executable DCL scripts in the system to add accounts, and changed the user passwords of all penetrated accounts.

4.4 Considerations in Worm Attacks

The major concerns in dealing with worm attacks are:

- o Speed of reaction
- o Indiscriminate spread
- o Bugs contained in the worm code
- o Reaction of worms to your incident response strategies
- o Coordination between sites

The most pressing concern in reacting to a worm attack is the speed of response. Because worms are automated attacks, they will proceed at the speed of the host computer. In order to respond to them effectively, the worm must be quickly identified and a correct reaction initiated to minimize the damage resulting from the worm.

Worms usually do not follow any particular boundaries when choosing a victim machine. Therefore, a worm will spread indiscriminately to any other host to which it can connect and from which it can request services. In order to contain a worm, all connections to other hosts must be discontinued. Note that simply because a connection is not usually used (i.e., between a company's computer and a Government lab) does not change the fact that a worm could use a common network to spread along this route.

It is difficult (if not impossible) to write error-free code, and worms are no exception to this principle. All of the worms detected to date have had programming errors (bugs) of one kind or another. In the most well- known case, the Morris worm, the error in the code caused the worm to replicate itself within a host so rapidly that no other processing on the host was possible. This caused an easy detection of the worm, and led to a significant denial of service.

When dealing with a worm attack on a particular system, there is no need to take into account the "feelings" of the worm. This is unlike a hacker attack, in which if you respond rudely to a hacker, you may increase the possibility of damage to your system. With a worm attack, the removal of the worm by any means at your disposal cannot foreseeably change the reaction of the worm. The worm can neither adapt nor react to your incident response strategies.

While it is always important to coordinate with other sites during an incident, the nature of worms makes this coordination essential to a favorable resolution to a worm incident. Because worms spread along network and other intra-site connections, failure to coordinate your response with other sites will prevent complete recovery to the worm attack. Worm attacks must be dealt with on the level of the interconnections between the hosts. In the case of a worm on the Internet, the entire Internet community must cooperate to completely eradicate the worm.

4.5 Basic Mechanisms of Worms

The basic mechanism of a worm is a two stage process. First, the worm must copy itself to the victim host, and secondly, the worm must begin an execution of the copy on the remote

machine. To copy itself to another machine, the worm makes use of services provided by the victim computer that are designed to accept information from the attacking machine. These are services that both have proved useful to users and are widespread (e.g., electronic mail, remote file service, "talk" or "phone" services, and "finger" services that will allow information about the users of a system to be provided).

In order to complete the process, the attacking worm must be able to start the execution of the worm on the victim machine. This is accomplished by exploiting other services that are provided for this purpose. Examples include rsh (remote shell execution for UNIX) or batch job submission. Other services not intended for remote execution may contain bugs or features that allow remote execution without authentication as well. Such bugs exist in many services such as sendmail, finger, and network file systems. Each operating system has its own set of services that may be exploited. This usually means that a worm is specific to a particular operating system, or a set of systems providing common services.

4.6 Recommended Steps for Responding

This section will apply the methodology introduced in Chapter 2 to handling a network-based worm attack. In review, this methodology includes the following actions:

- Protection
- Identification
- Containment
- Eradication
- Recovery
- Follow-up

Each of these will be discussed in detail below.

4.6.1 Protection

Protecting against worm attacks involves a number of issues, but the primary goal in protection is to assume that sometime you will experience a worm attack in which your system will be penetrated. Given this assumption, a worst-case scenario may be described, and countermeasures may be prepared. This is not to say that prevention is impossible. All of the known worm attacks to date could have been prevented if principles of good system administration were followed, including:

- Sound password management
- Closing known vulnerabilities and installing provided patches
- Running the latest version of the operating system
- Assuring that the systems had been properly configured

In addition, good system administration practice will aid in the process of recovery from a worm attack. Preparing for a worm attack includes:

- Preparing a set of procedures applicable to a worm attack
- Following a regular backup schedule
- Monitoring the system activity for unusual behavior

This methodology will aid in preparing a set of procedures for handling a worm attack. However, a set of procedures that take into account site specific considerations should be prepared ahead of time. These procedures should include the names and phone numbers of persons to contact in emergency situations involving worm attacks (as well as other incidents described in this document).

As in other types of incidents, procedures are more important than software solutions. For worms, this means that you should not rely on a software protection package to assure that you will not have to handle a worm event. Assume that a worm will get through your software defenses and determine the countermeasures from that point of view. Of course, this does not mean that you should not use software tools or install security patches to the operating system, but always consider the worst case scenario as the basis for your procedures.

4.6.2 Identification

Identifying a worm attack will require a careful analysis of the state of your system. Some of the things to look for are:

- Unusual names for processes on your system
- Newly created files in special directories such as system, user, and temporary directories. This is especially true if these directories are world writeable or in unused accounts
- New or altered accounts
- Unexplained performance problems on your systems

Be sure to document all observations. You may compare situations where no worm was found to exist with a new anomalous situation.

Some other important considerations in the identification of a worm event are to keep in touch with your local computer security department and CIAC. You should get an indication of a specific symptom for which to search from these sources if a wide-spread worm event is underway. CIAC and other coordinating groups will use the information you provide to aid other sites in getting the worm under control. If this is not done, the worm will be free to attack your systems again once you finish your recovery procedures.

In addition, be sure that your user community should be well aware of your policy concerning worm identification. Chances are that your users will detect anomalous behavior before you get a chance to monitor the system. User education will allow you to have a network of worm detectors available whenever a user is logged into the system.

4.6.3 Containment

Once it has been decided that a worm incident is under way, the most important goal of the containment stage is to stop the spread of the worm beyond the machines in which the worm has been detected. This is an exercise in getting the situation back under control. A controlled environment is required in order to effectively apply the subsequent stages of incident handling.

During the containment stage, you will have to make many important decisions and gather valuable information on the situation. Planning is essential since you will have to react quickly,but effectively. You will need to capture the baseline state of your system in order to save it for future analysis. It is probably not the case that you will be able to correctly analyze the worm before it is under your control, so you should plan to record as much information on the state of the system as you can. If practical, begin a full backup at this time (being careful to mark the backup tapes as infected with worm code). List and halt all nonessential processes on the system. If classified information is involved, be aware of and immediately deal with the threat involving disclosure of the information across any connected networks.

In order to gain control over your system, you may want to disconnect it from all forms of external networks. This is probably an effective way to prevent the spread of the worm, but realize that it may limit your ability to communicate information about the worm as well. During the attack of the Internet Worm, those systems that disconnected themselves from the Internet were not able to receive critical information concerning how to destroy the worm. If you do choose to disconnect from your normal communication networks, be sure to keep in touch via telephone, FAX, or other media to CIAC or some other coordinating group. CIAC is dedicated to providing solutions and technical advice during such an attack.

Finally, do not neglect your appropriate reporting procedures during a worm attack. Quite aside from technical issues that arise during a worm event, a widespread worm will attract the attention of the national media. Following the proper reporting procedures and allowing your public affairs group to handle the press will help you keep your situation under control.

4.6.4 Eradication

Once the situation is under control and contained, it is important to eradicate all traces of the worm from your affected systems. This may be partially accomplished by removing processes (as described in the preceding section), but there may also be other residue left from the worm. In particular, be sure to check for:

- New accounts added by the worm, especially privileged accounts (such as root accounts for UNIX systems, or privileged accounts on VMS systems)
- New files added by the worm--these files may be used during a reboot of the system to restart the worm, and should be removed
- Files for proper ownership and protections--it is common for worms to leave back doors in a system for the author to use later
- Trojan horses added by the worm--if possible, compare all executable files to files from a trusted source (e.g., backup tapes or distribution media)

Use any eradication scripts provided by CIAC or other trusted source to aid in the process of completely eradicating the worm code. Be sure to document all of your actions during this process. Not only will this allow you to describe precisely how your system has evolved, but gives you some additional confidence that you have left no steps out in the eradication process.

For some systems, especially those involving classified processing, it may be appropriate to completely erase the on-line storage and recover using clean backup tapes and the original distribution of the operating system. Be careful in this case to not only erase the disks, but shutdown the system to clear the internal dynamic memory. If your system uses static memory as part of its storage system, be sure that this is cleared as well.

4.6.5 Recovery

Before you can return to operational status, you must recover the state of the machine to before the worm attacked. During this recovery, you want to assure that the worm is eradicated from the system, and that it cannot reinfect the system. To accomplish this, you must plan to install security patches to close vulnerabilities exploited by the worm . (This may involve changing user passwords or disabling a system service.) Use your records from the eradication and detection stages to identify any modified or deleted

files, and restore these files from an uninfected backup copy. Restore whatever system services you can, and restart the normal user services unrelated to the worm infection.

After you have brought your system back to normal, you may consider placing it back on the network (assuming that it was taken off the network in the containment stage above). Before this step, assure that the worm infection has been eradicated from the network and that it is safe to restore network services. Coordinate with CIAC or other computer security groups to assure that the time has come to restore your network connection.

4.6.6 Follow-up

Once you have restored the normal operation of the system, you enter the follow-up stage of incident handling. This is an often neglected, but very important stage for a number of reasons described in Chapter 2. In addition, the substantial effort used to handle the worm incident will have to be justified to management, and information that could be used to prosecute the author of the worm must be properly handled.

An important goal in the follow-up stage is to analyze what happened. We recommend that you create a report detailing the chronology of the incident, actions taken, and lessons learned. This is the time for the detailed analysis of the specific occurrences and information taken during the incident. If the report is completed soon after the event, information that was not chronicled during the event may be reconstructed. This report may be used for a number of different purposes, so it is important that it be as complete as possible, and that it contain a true chronology of events.

If you have any indication of other specific systems attacked, be sure to notify the appropriate system managers as soon as possible. This may not only help save additional systems from the damage and/or inconvenience the worm may cause, but may also lead to reconstructing a path that the worm took through the network. This, in turn, could help trace the worm back to its point of origin. CIAC may be of great assistance during this activity to contact sites and coordinate between investigating agencies.

It is often the case that one of the lessons learned is that if a particular tool had been installed, the incident would have been easier to handle, or perhaps would not have occurred at all. The follow-up stage is the time to evaluate these tools and to decide which ones should be installed permanently on the system. Be aware, though, that too many invasive security tools tend to burden the user and will degrade the day-to-day operation of your system. Choose which tools you will install carefully, and be aware of the tradeoffs.

It is very important to remove any on-line copies of the worm from your system at this time. It is common for hackers to break into systems known to have been attacked by a worm with the sole purpose to find a copy of the worm and re-release it. There are many hackers who do not have sufficient knowledge to write a worm themselves, but may modify an existing worm to evade a common protection mechanism. This was the case with the OILZ variant of the WANK worm. Lock up or otherwise isolate any copies of malicious code you may obtain.

Finally, issue an “all clear” message to your user base. With all the "hype" associated with a worm incident, there may be many users who are confused whether they can get back to a normal usage schedule. If you are working to coordinate a number of systems, other system managers must be notified as well.

4.7 Final Words on Worm Attacks

Network-based worm attacks are a natural extension of malicious code to the current trend of interconnected, trusted systems. While it is currently impossible to prevent all future worm attacks, following an effective set of procedures will help you to deal with a worm event in a practical fashion and without panic. The national attention these incidents receive will make handling a worm event tricky, but following the procedures described in this chapter should keep the disruption of the system to a minimum.

5 - Hacker/Cracker Attacks

5.1 Introduction

The term "hacker" is in many ways a misnomer. The term "hacker" can be used to refer to someone who constantly plays with computers. A "hacker" can also imply a computer expert. In the context of this manuscript, "hacker" is used somewhat inappropriately to refer to someone who breaks into one or more computer systems. "Hacker" is nevertheless useful, in that it is a commonly used term within the world of computer security--few computer security experts fail to agree to some extent on what people mean when they say that a hacker broke into a system. On the other hand, "cracker" refers to someone who breaks into a system with malicious intent (e.g., someone who breaks in, then trashes a user's files).

Dealing with hacker and cracker attacks is a special challenge. Along with internal sabotage, hacker/cracker attacks are the most potentially destructive form of attack to a single system. A worm may attack a network, and it may virtually close that network. However, a worm is nothing more than an algorithm. Once the algorithm is understood, the worm attack can be terminated. In a hacker/cracker attack there is an intelligent human being back of a keyboard somewhere. Although the ethics of their activity must be seriously challenged, critics often admit that hackers/crackers are frequently resourceful and elusive. You can shut down the machine that is being attacked, but the attacker is likely to simply hit another machine. Furthermore, hacker/cracker attacks are far more frequent than people realize. Hackers/crackers are often very careful to "cover their traces," so that it may be difficult to notice the attack at all. Finally, there are international hacking clubs and hacker bulletin boards. Communication among hackers/crackers makes them better organized and increases information available to them, making them more capable of succeeding in whatever goals they have.

There are many misconceptions about hacker/cracker attacks. One misconception is that there is a single type of hacker or cracker, usually a frail teenage male with weak social skills. Another is that hackers/crackers usually destroy and/or steal files. Yet another misconception is that to break into a system requires considerable knowledge about computer systems. All of these misconceptions are far from reality! There is no single type of hacker or cracker--some hackers/crackers are young, and some are considerably older. Some are quite knowledgeable about computer science, whereas others know very little and mostly rely on bulletin boards and "word-of-mouth" to break into systems.

5.2 Division of Responsibility

Responding to a hacker/cracker attack generally requires a great deal of coordination from users and administrators of a system. Managers must implement and fund a security plan, and must ensure that this plan is followed. System administrators must fix security vulnerabilities and must take the lead in installing and maintaining security tools on their systems. Users must be aware of a security plan, and must operate their computer in a responsible manner. This may mean, among other things, not using all the available features/utilities of their system if some features introduce vulnerabilities. This may also mean furnishing the system manager with pertinent information, such as information about numerous failed login attempts on a user's account.

5.3 Recommended Steps for Responding

5.3.1 Protection

Protection against a hacker/cracker attack requires considerable effort. Taking protective measures can never provide absolute assurance that your system(s) will not be attacked, however. Additionally, some protective measures may be characterized as proactive measures to ensure that responding to a hacker/cracker attack will go more smoothly if

and when such an attack occurs. (Remember, it is essential to avoid viewing your system as invincible, and to devote a good portion of effort to ensure that things will go smoothly when the inevitable happens--your networked system is attacked by a hacker!) Protective or proactive measures include having effective password management, obtaining and installing special purpose hardware, creating backups, eliminating security holes, obtaining and installing monitoring tools, and developing effective procedures.

Effective password management is so important that there is a separate section devoted to this topic at the end of this chapter. Password management is, in fact the single most effective protection mechanism for systems connected to a network. At the most basic level, it is essential to require the use of passwords for logins, regardless of whether the login is coming from a terminal connected directly to a host machine or from a remote location. In the case of logins from a modem, it is important to require a separate password for using the modem! At the next level, it is critical to ensure that user accounts have effective (i.e., not easily guessed) passwords. For example, passwords such as TOYOTA, JENNIFER, and WEENIE are very easy-to-guess passwords--ask most any hacker!

Special purpose hardware is also useful in protecting systems against hacker/cracker attacks. This hardware may require the users to carry and use some hardware (e.g.,, a plastic card or authentication device) before the login sequence appears. When properly implemented, special purpose hardware can be effective because it can virtually eliminate the threat of intrusion from afar. Special purpose hardware, however has some drawbacks. The hardware can be expensive, and it is often inconvenient to use a special card or electronic device to gain access to a system. In addition, users with many accounts must keep track of many devices. For this reason, special purpose hardware is not widely used. We recommend such hardware when there is sensitive and/or proprietary information on a system, or when there is another special need for extra security. Another example of the wise employment of this hardware is when organizations and agencies require its use when there is an unusual threat, such as a recent rash of hacker/cracker attacks on the network to which that organization/agency's machines are connected, but not at other times.

File protections are important in protecting systems against hacker/cracker attacks. In general, files need to have protections such that only the users or programs needing access to a given file have access. In particular, assure that file protections for system binaries and home directories are not world-writeable, and that sensitive files are not world readable or writeable. Also, be aware of unexpected public files and directories with world read or write privileges, and files in a user's directory not owned by the user.

Backups are also essential. You can do many things wrong as far as computer security goes, but if you backup your system frequently, you will at least not lose data. If you have frequent backups off-line, any attack can damage at most only your on-line data. (Backups are good not only for handling hacker/cracker attacks, for also for other incidents which could lead to a loss of data.) You need an effective backup strategy to make backups work for you. As part of this strategy, make sure that you back up every important file, and that you perform the backup on non-volatile media. Use long backup cycles (e.g., backup once a month), since it may take a considerable amount to time to detect a mistake in backing up your data. Use separate media for each backup, and take the time to verify that your backups are correct.

Another important step in protecting against hacker/cracker attacks is eliminating security holes. Although this topic will be covered in more detail in Chapter 6, it is introduced here to emphasize the role of unpatched holes in hacker/cracker attacks. Many operating systems have numerous holes, whereas others present opportunities for hacker/cracker attacks only if they are improperly configured. In general, it is better to run the most recent

version of an operating system unless you know that the newest release introduces new vulnerabilities (which, unfortunately, sometimes happens). Sun OS 4.1 is considerably more secure than is Sun OS 3.4. For the sake of system security, upgrade to Sun OS 4.1 in this case!! Keep in touch with information about security holes distributed through user groups, emergency response teams such as CIAC, etc., and take appropriate action (e.g., install the patches that are available).

Obtaining and installing monitoring tools is another important protective measure. There is a variety of monitoring tools available, and it is wise to have a number of these tools. (Refer to Chapter 7 for more specific information.) Now is the time to obtain and install these tools--you may find that once a hacker/cracker attack starts, it is impossible to quickly obtain and install the tools you need!

Developing effective procedures for responding to hacker/cracker attacks is essential. These procedures should include call lists, lists of people to be contacted during an incident. These lists should include both work and home phone numbers, since emergencies often happen at inopportune times. (CIAC team members have a laminated card containing phone numbers of other team members, our site computer personnel, our operations office managers, and people at DOE Headquarters! We wear this card with our badges.) Procedures should reflect realistic time-distance considerations. If there is a hacker/cracker attack during non-working hours, some people who will help respond to the hacker/cracker attack may have to travel farther than others. Procedures should have those who are closest to the site of response coordination initially doing all the work or the most urgent tasks. Other considerations for developing effective procedures include specifying who users should notify, how to get information out to users, and what to tell users. Dealing with the media is also important. How should calls from the media be handled, and what kind of information, if any, should be given? Procedures should address shut-downs and when a system should be taken off a network. When to prosecute and when not to prosecute should also be covered, as well as the types of evidence that needs to be gathered if you decide to prosecute. Procedures should address the need for keeping timestamps, especially since hackers and crackers may change computer records. Who keeps the "master clock?" Finally, it is important to decide how to create, change and distribute your incident handling procedures, and to specify the tools or other computing resources needed to implement your procedures.

We also recommend another simple protective measure. If practical to implement, modify your system banner to delete any words or phrases which welcome users to your system. At the same time, warn intruders with a message such as the following:

> WARNING: Unauthorized access to this computer system is prohibited, and is subject to criminal and civil penalties.

Changing your system banner may not deter all or even most hackers, but it is likely to deter some hackers. In addition, removing welcome messages in your system banner may be important in that in a court of law welcome messages may be considered invitations to intrude into systems.

5.3.2 Identification

Identifying a hacker/cracker attack can be an immense challenge. Good hackers/crackers are incredibly good at avoiding detection, often by writing and planting routines to mask their presence in systems they attack. There are a few general clues that a hacker/cracker intrusion is occurring. A general clue is that hacker/cracker attacks often occur during the night or early morning when few users are on the system, or when the system manager is not on the system. Cooperative users are the "eyes and ears" of computer security--keep in touch with them, and encourage them to report unexplained failed login attempts and

other specific evidences of a hacker/cracker attack. Finally, detection tools (covered in Chapter 7) are available to help identify an intrusion.

There are also specific indications of a hacker/cracker attack. Look for combinations of the following:

- o New, unauthorized user accounts
- o Processes owned by unfamiliar users
- o New files or file modifications
- o Accounting discrepancies (e.g., charges for using your system do not coincide with compute time)
- o Changes in system files (e.g., password files)
- o Modified/deleted data
- o Denial of service (e.g., your system suddenly goes into single user mode, locking you out)
- o Poor system performance
- o Anomalies (e.g., compiling new programs in a temporary directory)
- o Suspicious probes (e.g., rlogin attempts) and/or browsing (e.g., file to file)

5.3.3 Containment

Once you have identified that there is (or probably is, in many cases) a hacker/cracker attack, the next step is to contain the attack. Containment may not exactly be the correct term to describe all of the goals and procedures necessary during this stage of response. Goals, for example, include identifying the intent of the hacker/crack attack. Is this a malicious attack, are system files being changed, or is the intruder just browsing? During this stage it is also appropriate to start trying to identify the intruder. Intruders have identifiable patterns of attack, and it is important to learn about other ongoing hacker/cracker attacks to compare the pattern of your intrusion with others. CIAC can be very useful to you at this stage, and you should also help CIAC help others by notifying CIAC about your current situation periodically. (CIAC will not reveal the name of your site to others.) It is important to gather evidence to be used against the attacker (e.g., logfiles with timestamps). Assessing damage incurred is also important both in gathering evidence for prosecution, but also for urgent decisions to be made (covered in next paragraph).

The containment stage involves a critical decision. You could pull the plug to the compromised machine, or disconnect that machine from the network. This would immediately terminate the attack, and would at least superficially relieve your anxiety. In some cases this is the preferred decision, especially when files are being destroyed or your machine is being used to launch many attacks on other systems. In many cases, however, it is not wise to turn off your machine or disconnect from the network. If you immediately terminate the attack, for example, you may never know what the hacker/cracker has really done to your system. Furthermore, the intruder will probably just find another machine to attack, perhaps even a machine in your organization or site. If you can keep the hacker on your machine without incurring substantial damage to that

machine, you can assess damage, gather evidence to identify and prosecute the intruder, and probably save a lot of other system managers and users a lot of grief.

When we recommend that it is generally better to monitor the intruder's activity, we assume that you will already have created system backups, and that you will protect sensitive files and users' data. Additionally, you should never risk compromising classified information. If there is any reasonable probability that classified information could be compromised, it is imperative to turn your system off or disconnect from the network. You will also have to determine the extent of user interruption that can be tolerated while monitoring occurs. You may need to remove computer service from the user community, and/or to inform users that the system is being attacked. In general, we recommend the following monitoring strategy.--try to keep the hacker on one machine. To do this you may have to plant "treasures," games and/or bogus but interesting files (e.g., " Highly protected data on nuclear arsenals"). Also, while you are monitoring be sure to check periodically for phone messages. The intruder may try to call you! Another consideration is that sophisticated intruders often get into protected parts of a system and redirect electronic mail (e-mail)---the intruder may be able to read the e-mail messages you send! Finally, if you notice that an intruder has gotten into your system, do not taunt the intruder or order that person off your machine. Do not turn a casual attacker into a dedicated opponent! Encourage the intruder to call you, and find out what the intruder wants. In several cases hackers have apparently permanently ceased hacking activity when they were politely asked to do so, and when they were informed of the extent of the problems they had caused. These now former hackers have on several occasions, in fact, helped several emergency response teams correctly determine that hacking attacks were occurring and which hackers were involved.

There are a few final considerations about monitoring activity. Be sure to document everything (keystrokes, timestamps, the actions you take, etc.). You will need the information you record to ensure that you communicate accurate information to others with whom you are coordinating your response to the hacker/cracker attack. (Remember, people under considerable stress are not particularly good at retrieving information from memory.) Your documentation will also be critical in subsequent stages of handling the incident. If your monitoring efforts are unsuccessful, consider contacting the hacker directly before you take any other measures such as turning your machine off. Finally, remember that you should deal with the media only through your public affairs office. Your taking on the media yourself may not only be against your organization or site's rules, but also may be an invitation for catastrophe unless you have specific training in dealing with the media.

5.3.4 Eradication

Eradication means ending the hacker/cracker attack. As mentioned previously, you can eradicate the attack by turning off the attacked machine or disconnecting it from the network. Hopefully, however, you can achieve a better solution---identification and apprehension of the attacker. A third set of outcomes is possible, too--the hacker/cracker might get bored or suspicious and terminate the attack. Similarly, the hacker/cracker might obtain the data or programs that he or she wants, and might then terminate the attack. Finally, the intruder might voluntarily agree to terminate the attack.

For eradication to be complete, you need to take additional actions. If applicable, remove the hacker from the system login files. Also, remove any processes left by the intruder. Backup your system again to preserve any trail the intruder may have left. (Again, remember that many hackers/crackers are often very adept at covering their traces.) If you have a current backup, you may seriously want to consider performing a complete restore.

5.3.5 Recovery

The recovery means that you or the system manager will restore the attacked system to its "normal" state after the attack has been eradicated. The first action is to perform a final damage assessment. You should now have more time than with the previous stages of incident handling, because the hacker/cracker attack has been eradicated. After determining the extent of the damage, close any security vulnerabilities that the intruder exploited. For example if the intruder entered your system through a sendmail hole, ensure that you obtain and install a patch for this hole. Next, restore data. An attacker may change the date and time stamp, so doing a full restore may require a full removal of all files and a full restore from backups. Next, restore all services, and inform users that the system is no longer being attacked and has been returned to normal status. Finally, determine what tools used to handle the incident you should leave in place. In general, discontinue using tools useful mostly or only in dealing with the particular attack that occurred, but leave in place tools which might be able to both help you detect intrusions and capture data early in the incident handling process without severely taxing system performance.

5.3.6 Follow-Up

The follow-up stage is critical. If law enforcement agencies are not pressing for prosecution, it is probably appropriate at this point to debrief the attacker (if possible). Again, it is important to politely inform the attacker of the effort handling the intrusion has required, and to learn any additional information which may help you (e.g., other systems in your organization/site which may have been attacked, methods and routes of attack, etc.) Be sure to record all of the information you learn. Alternatively, you may need to assist in prosecution efforts. In this case you should not talk to the attacker, but should submit your logfiles and documentation of your efforts in dealing with the incident to law enforcement agencies which ask for evidence. Regardless of prosecution efforts, you need to determine time and resources used as well as damage incurred, and should produce a monetary cost estimate. This estimate is critical in a court of law, but it is also critical in justifying the importance of adopting appropriate security measures and having an incident response team to your management. In addition, determine how the incident happened. You know have a valuable set of "lessons learned" which you can incorporate into a post-mortem report--something to which you and others can refer during training, when you revise your incident handling procedures, or when you have a similar incident in the future. Review and revise your protection program, including your procedures, password management policy, and arsenal of tools. Be sure among other things that you are not introducing new security measures purely in reaction to the recent hacker/cracker attack. Don't introduce unduly restrictive policies or install numerous tools which diminish system performance simply because an attack occurred !

5.4 Automated Hacker/Cracker Attacks

Some intruders have created tools to help them break into systems. For example, password cracker programs that attempt to match candidate passwords with actual system passwords are rather common. Some of these programs have very large dictionaries. The success of these programs is an important reason why effective password management is a must (see Section 5.5). Worms, discussed extensively in Chapter 4, are another form of automated hacker/cracker attack. There are also programs used to find system protection problems to enable hackers/crackers to break into that system. (Note: these tools may also be used by computer security personnel to locate vulnerable systems and make them more secure.) There are also network snooping programs to help intruders locate systems to attack, and programs to check for libraries that contain old bugs. These programs can be linked into other programs to break security.

5.5 More on Passwords

By now you have seen how important a role passwords play in allowing intruders to break into systems. The overwhelming problem is not really so much the individual user as much as failure on the part of management to develop and implement an effective password policy. Unfortunately, we have found that many organizations or sites have no password policy at all! An effective password policy should mandate that the following types of passwords and procedures are eliminated:

- o "Joe" Accounts, in which the account name and password are identical (e.g., WILLIAMS and WILLIAMS)

- o Same passwords for one user on different machines (e.g., WILLIAMS uses the password tsr80p on SAMVAX and on BAKVAX). Intruders into a system often realize that same passwords are frequently used on different machines, and capitalize on this knowledge to break into other machines once they have initially broken into a user's account. In general, using the same password on different machines is the greatest problem when the hosts are run by different organizations, the hosts have different levels of security, or the hosts have different operating systems.

- o Accessible password files. At a minimum, these files must be hidden. Unfortunately, some older systems had a publicly known reversal to the encrypting algorithm, thus making all accounts available once a hacker stole the password file. Remember, if the encrypting scheme is known and the password file is accessible, guesses can be made at the convenience of the attacker!

- o Easily guessed passwords, including words found in Webster's dictionary, car names, people's names, computer jargon and nicknames, "trendy" words, etc. Your password policy should disallow these classes of words. Make sure that your training and awareness program gets this message to users. Also, you should consider using software to restrict users' choices of passwords, so that users are forced to adopt more difficult-to-guess passwords. Remember, "brute force" attacks in which password after password is tried are now easy because of password cracker programs.

We also recommend that your password policy include the following:

- o Checking for easily guessed passwords. Checking can be done when a user chooses a password and/or at regular intervals (e.g., once every 30 days). You can have the system manager disallow weak passwords when users choose passwords. You can also obtain a password checking tool. (CIAC, for example, can help you obtain a password checker for either UNIX or VMS.) Your policy should include procedures for finding and changing easily guessed passwords.

- o Using machine-generated passwords. There are advantages and disadvantages to machine-generated passwords. These passwords, which may be letters, numbers, syllables, or phrases, are difficult to guess. On the other hand, they are more difficult to remember, especially when there are multiple accounts for one user on several machines. Users tend to write down machine-generated passwords (see below). If you are from a DOE site, DOE Order 5637.1 requires that machine-generated passwords be used with classified computing systems. Also, passwords for classified computing systems are classified at the same level as the system.

- o Obtaining and installing special purpose hardware (discussed previously).

- o Adopting a password aging policy. In general, it is important to require users to change passwords on a regular basis (e.g., every month in the case of sensitive/proprietary systems, and every four to six months otherwise). For one thing, if an attacker steals the password file, it will take time to crack the passwords. Changing passwords from time-to-time may foil crackers' plans. A password aging policy can also flag and lead to the removal of dormant accounts. One disadvantage, however, is that users who are forced to change passwords regularly may choose easier-to-guess passwords, because the user has to create and remember new passwords.

- o Developing a policy for writing down passwords. Sites and organizations often prohibit writing down passwords because writing down passwords leads to increased probability that passwords will be compromised. Slips of paper with passwords, for example, are often put in wallets, which may be lost. Also, it is important to remember that writing down a password for a classified system requires following your classification guidelines! On the unclassified side of the picture, there must be a policy for protecting written passwords if writing down passwords is allowed. Writing down passwords has one major advantage--it allows users to have passwords which are harder to remember and guess.

Selecting a good password requires some inconvenience on the part of users. It is more difficult to choose and remember a more difficult-to-guess password than an easy-to-guess password. We concur with Wood and Kochan's (1985) recommendation to use a four letter word, then a control character, then a four letter word which is unrelated to the first word (e.g., dust&book). Such a password is easy to remember, but is very difficult for an intruder to crack. Another sound technique for constructing a password is the first letter method, in which the password consists of the first letters of each word in an easily remembered sentence. For example, "my Country tis of thee...sweet land of liberty" becomes mctotslol, which is easy to remember but almost impossible to crack. Again, an effective awareness program which motivates users to choose appropriate passwords is critical.

The "bottom line" is that your organization or site needs to create a password policy. This policy will vary, depending on requirements of users and the particular system involved. We have found, however, that the following additional guidelines are applicable to most sites and organizations:

- o Be sure you have a policy for detecting and removing "Joe" and dormant accounts.

- o Explicitly state a policy covering the periodic, mandatory changing of passwords.

- o If your site or organization uses machine-generated accounts, have a policy for written password management.

- o Include a password policy for installing accounts and machines. Remove or disable all demo, default and guest accounts. When adding a new user to a system, do not allow system managers to furnish users with an initial password which can be easily guessed--users often do not change passwords!

- o Build user education into your site or organization's password policy. User education is time-consuming, but is critical in gaining user support and giving users sufficient knowledge to avoid giving attackers easy access to a system through weak passwords. Your user education plan should include actually helping users select proper words such as non-dictionary words, initial letters from phrases, etc.

6 - Vulnerabilities

6.1 Introduction

The terms "vulnerabilities" and "holes" have by now been used dozens of times in these guidelines. All computer systems have some features that can be exploited by an attacker. The emphasis of this chapter is upon these features. First, let's get our terms straight. A vulnerability is a feature or bug in a system or program which enables an attacker to bypass security measures. For example, suppose that there is an operating system which, unbeknown to the developers of this operating system, allows an unprivileged user to suddenly become privileged simply by hitting CONTROL/C at a certain point in using one of the operating system's utilities. We consider the difference between a vulnerability and hole to be an issue of semantics, not one of practical significance. We, therefore, define a hole as a vulnerability. A "Joe" account, previously covered in Chapter 5, is an account in which the account name and password are identical. A default account is an account shipped with the system that has a default password. (Unfortunately, system managers too infrequently change default account passwords.) A patch is a modification to a program (e.g., an operating system) to correct a bug or vulnerability. (Note: patches may have undesirable side effects, unfortunately, depending on how a system is configured.) A workaround is a "kluge," a makeshift, temporary solution to a vulnerability. For example, if there is a system utility which allows unprivileged users to read other users' files, a workaround might be to disable that utility (which, unfortunately, may inconvenience many users).

There is generally (but not always) a tradeoff between the amount of functionality that a system delivers and the number of exploitable features. For example, one of the reasons that UNIX has traditionally had more exploitable features than many other operating systems is that UNIX offers a great amount of functionality. Networking functions, which are built into UNIX, offer attackers opportunities to jump from UNIX machine to UNIX machine, to steal files, etc. It is essential that you know your system well. You need to learn about the exploitable features in your system, and to close these features within the constrains of user needs, manpower limitations, etc. For example, you would do well to disable a seldomly used utility with a well-known security hole because few, if any, users would miss that utility anyway. Disabling a frequently used utility with an obscure hole may not be as advisable. Remember, too, that one machine in a network which has unpatched vulnerabilities can become a "launching pad" for attacks throughout the network. Regardless of whether you are a user or system manager, you have a responsibility not only to protect your own machine but also to be a responsible citizen of the network!

There are many misconceptions about vulnerabilities and how a person's system can be attacked. One of the most common misconceptions is that if one's machine runs a particular operating system (e.g., VMS) but the network to which that machine connects runs different communications protocols (e.g., TCP/IP), the VMS machine cannot be exploited through that network. The truth is that common communications protocols and applications create common vulnerabilities. Thus, sites running TCP/IP may be exploited regardless of the particular operating system run in machines connecting to the TCP/IP network if there is a vulnerability in TCP/IP.

6.2 Types of Vulnerabilities

Needless to say, there are many types of security vulnerabilities in computing systems. Poor passwords (see Section 5.5) comprise by far the largest class of vulnerabilities. Dormant accounts and orphan machines (machines which are not currently maintained) comprise another significant class of vulnerabilities. If dormant accounts and orphan machines also have poor passwords, this constitutes an even greater security problem. Holes in operating systems are another type of vulnerability. Security holes in applications are another set of vulnerabilities.

For example, Stoll (1989) gives an interesting account of a hacker attack launched through a vulnerability in a word processing program. The main focus of this chapter is on operating system vulnerabilities because these vulnerabilities are so frequently exploited by attackers. (Please note that the following descriptions of vulnerabilities include only the outcomes of exploiting each vulnerability, rather than the actual mechanics of exploiting each vulnerability. We take this approach to avoid disseminating information that attackers might use.)

6.3 UNIX Vulnerabilities

As discussed previously, UNIX systems generally have numerous vulnerabilities. Because there are many flavors of UNIX (e.g., Sun OS, BSD UNIX, ULTRIX, System V, etc.) and usually many versions of each flavor, a particular UNIX vulnerability seldom is universal. In fact, it takes careful investigation to discover exactly which vulnerabilities are and are not present in any given UNIX system. We recommend that, at a minimum, you become aware of the following vulnerabilities frequently found in UNIX systems:

- o set-uid (user id) scripts - if set-uid scripts are not set set up properly, an intruder can obtain unauthorized root access
- o anonymous ftp (file transport protocol) - some versions of UNIX have bugs or have anonymous ftp set up incorrectly, which can allow unauthorized file modifications (e.g., unauthorized modification of /etc/passwd)
- o tftp (trivial file transport protocol) - some versions of UNIX have bugs in tftp, or tftp may not be set up correctly, allowing an attacker to steal any publicly readable file (including /etc/passwd)
- o login (the program that logs a user into a system) - susceptible to trojan horses
- o fingerd - there are several vulnerabilities, some of which enable attackers to obtain unauthorized root access. Intruders can also use fingerd to locate user accounts to attack.
- o sendmail (the program that delivers mail to the user) - new vulnerabilities in this function seem to be discovered frequently. Some of the vulnerabilities include allowing an attacker to corrupt non-root owned files, or, when sendmail is used in connection with .rhosts, allowing unauthorized root access
- o uudecode - vulnerabilities in this function allow an attacker to overwrite files (including .rhosts)
- o nis (network information system, formerly yp--yellow pages) - problems in this function make multiple attacks within a cluster of machines easier
- o nfs (network file system) - problems in nfs also make multiple attacks within a cluster of machines easier

Remember, too, that in UNIX systems there are demo, user, and guest accounts that may have no passwords or "Joe" passwords.

6.4 VM Vulnerabilities

VM is a relatively secure operating system, and the exploitable features in VM of which we are aware are not severe threats to system and/or network security. Some versions of VM have an option to allow the user name and password to be displayed in clear text on the login line. This allows someone else in the same room as the user to view the passwords entered during logins.

Some versions of VM may flag a bad userid without asking for a password. This enables an attacker to learn which accounts are and are not viable to attack. Finally, older releases of VM have several additional exploitable features, including echoed passwords as they are entered and a default password on the DURMAINT account.

6.5 VMS Vulnerabilities

Exploitable features in VMS are, for the most part, configuration problems; VMS is reasonably secure if configured properly. Default accounts such as the default DECNET account in pre-5.0 version releases of VMS as well as SYSTEM and SHUTDOWN accounts on some VMS systems present a problem. Applications such as INGRES, ORACLE and mail running on VMS systems also have default accounts which, if unchanged, allow an easy avenue of entry. The VMS phone object does not check authorizations before allowing someone to write to another terminal. This utility can, therefore, be used to trojan horse an account. (The phone object has also been used by DECNET worms to write offensive messages to users.) The Task 0 object (which allows, among other things, functions to run remotely) may be configured to allow users with no account on a machine to run applications. (Note: to misuse Task 0, an intruder must first gain access to a system. It is debatable, therefore, whether Task 0 is, in and of itself, the real problem.)

6.6 MVS Vulnerabilities

MVS systems are seldom attacked, probably because other operating systems such as UNIX are considerably more popular and thus well known. However, the MVS operating system is, according to many experts, very vulnerable to attack unless C2 features are added (RACF). In addition, DSMOM provides attackers with sensitive information from the authorized caller table. The encryption module, SYSL2PALIB (which does RACF encryption), can easily be broken. Finally, problems with the PARMLIB and EXAMINE functions within RACF need to be fixed in most MVS systems.

6.7 File Protections

File protections constitute a special set of security considerations. File protections determine who may read, write or execute a file. Simple file protection measures can prevent many opportunities for misusing a system! Publicly writeable binaries and system directories allow an attacker to modify the system. Protections should be set accordingly. Readable restricted system files allow an attacker to discover other trusted computers or other privileged information. Writeable home directories allow trojan horses to be planted.

6.8 Concluding Comments

The information presented in this chapter is not just some theory. Taking action on the information in this chapter will probably save you considerable time, frustration, embarrassment, and effort, including the effort to "pick up the pieces" after a major hacker/cracker or worm attack on your system. At any given hour of any day there are dozens of computing systems being attacked because people have not learned about vulnerabilities or have not done anything about them. Learning of the vulnerabilities in your operating system and applications, then patching them is the key to avoiding catastrophe. The CIAC team is especially anxious to lend assistance to help anyone in the DOE community who has a question about vulnerabilities, or who needs help obtaining and/or installing a patch. The CIAC team also needs to learn as rapidly as possible about any vulnerabilities you may discover so that we can start working with vendors to close these vulnerabilities as quickly as possible.

Scenario For The "Big Fall"

7 - Incident Handling Tools

Throughout this document, it has been stressed that procedures are more important than software. This does not, however, diminish the value of software when used as part of an incident handling procedure. There is an increasing number of prevention and incident handling tools available. In this section, some currently available tools will be discussed, where to find them, and their usefulness.

7.1 Types of Tools Available

Currently available tools can aid in dealing with incidents at each stage of the incident handling model outlined previously. These stages include:

- o Preventing
- o Detecting
- o Containing
- o Eradicating
- o Recovering
- o Following up

There are many different types of tools available. In general these fall into the categories of software solutions, hardware solutions, and integrated products. Software solutions are layered software products that do not require any additional hardware to function. These would include additional passwords for authentication, checksums for file integrity, etc. Hardware solutions are devices that attach to your system or network to provide some security function without affecting your software on the system. Examples of these are network monitors that snoop your network, or printers that attack to a dial-in port to record traffic. Integrated products are combinations of software and hardware on your system. Hand-held authenticators or encryption boxes are examples of integrated products.

Aside from the types of security tools that you can add to a system, there are systems that are more inherently secure than others. Often these systems are based on a trusted operating system (TOS). These systems are formally proven to be secure to a particular level. In the Department of Defense, these systems are rated according to criteria outlined in a standardized document on computer security (the Orange Book). For the DOE, no such document exists, but security considerations are outlined in DOE Orders 5637.1 and 1360.2A.

TOSs are useful in a number of situations including dedicated network controllers, data base or file servers, or human safety controllers. However, TOSs are not a panacea. TOSs are not in widespread use because of a few serious drawbacks including:

- o It can take an extensive amount of time to certify a TOS
- o Security is often gained at a restricted functionality
- o The assumptions used in certification cannot be applied in some situations
- o There is often a unique or archaic user interface
- o The application base is too small to provide off-the-shelf solutions

Anti-virus software is a category all in itself. There are many good anti-virus packages available (see the section on viruses for more details). These packages can save many hours of recovery time, and have proven to be useful. The main considerations for obtaining and using a virus tool are:

- o Be careful of your source
- o Run the most recent version

- o Never entirely trust the virus software (procedures are more important than software)
- o Maintain a good backup policy

Encryption is a useful tool for computer security. There are many tradeoffs in using encryption however. Be aware that encryption cannot be used to solve computer security problems in general, but may only be used in particular situations such as virus detection, communications, and authentication. Using software to perform the encryption can be slow and something must be trusted to do the encryption--if that something is corrupted, the entire system is corrupted. Dedicated hardware to perform the encryption helps, but this may be expensive and difficult to integrate into the system.

The remainder of this section will be devoted to a listing of tools available to aid in computer security incident handling for multi-user systems. These tools are categorized as either system security tools providing added security features to a system, authentication tools that specifically provide additional authentication support, and intrusion detection tools that provide detection and identification of a system penetration.

7.2 System Security Tools

SPI (Security Profile Inspector)

This tool aids system managers in maintaining the security and integrity of UNIX and VMS systems. It provides password checking, file integrity checks, and checks for correct file permissions. The VMS version adds, in addition, checks for using identifiers and for accessibility of files based on UIC, identifier, and privileged access.

For information on SPI/UNIX or SPI/VMS, call Doug Mansur, LLNL, (415) 422-0896 or (FTS) 532-0896, or send e-mail to mansur@tiger.llnl.gov

COPS (Computerized Oracle and Password System)

COPS is a UNIX security checking tool. It uses configuration information in special files to check for the read/write permissions of all important files. It will also check for superuser access and passwords, as well as other aspects of computer security. For information, contact Dan Farmer (412) 268-7080, or send e-mail to df@cert.sei.cmu.edu

SecurePak

SecurePak is a commercial package for VMS systems from DEMAX software. It will check many security features on VMS systems, and provides a much nicer interface to the authorize function. Contact DEMAX software for more details on this package.

Matt Bishop's password program

This is a tool that will search for easily guessed passwords based on a dictionary search. It also may replace the passwd function on UNIX systems to restrict the initial selection of an easily guessed password. It contains a fast DES implementation that will improve the performance of the password functions above the standard UNIX implementations. Contact Bill Kramer, NASA Ames, (415) 604-4600, for additional information on this package.

7.3 Authentication Tools

Kerberos

Developed within the Athena project at MIT, this is a UNIX tool (which has been ported to other operating systems as well) to provide authentication in a distributed environment. This tool's

goals are to provide network authentication without sending clear text passwords across the network, or storing them on the individual hosts. Every connection may be authenticated using kerberos, and it is designed to be used transparently with other network services. For additional information or to obtain the tool, contact geer@athena.mit.edu or info-kerberos@athena.mit.edu

DES Encryption and Secure nis

Both DES encryption and secure nis are provided with the Sun operating system. These systems aid in protecting files with a user-defined key, and protecting network communication by encrypting the information used in the distributed authentication provided by dns (domain name service). For more information on how to use this tool on Sun systems, see your Sun documentation.

7.4 Intrusion Detection Tools

Haystack

This is an intrusion detection tool that will run on an IBM PC to monitor Sperry-UNISYS mainframe computers for intrusions. It was developed by Steve Smaha of Haystack, Inc., under contract by the Air Force. For additional information, contact Capt. Tim Grance, AFCSC, (512) 925-2386.

Wisdom and Sense

This is an intrusion detection tool that will run on a Sun computer to monitor a VMS system. It will layer the appropriate software on the sun to monitor DECNET traffic and determine if there is a deviation from normal operation. For information on this tool, contact Hank Vaccaro, LANL (FTS) 843-3382.

CSM (Computer Security Monitor)

The CSM is a rule-based monitor that will detect penetrations based on statistical analysis of normal usage patterns. In addition to the statistical analysis, deterministic rules may also be applied to determine if a user is performing an unauthorized function. The rules are system dependent, and have been developed for Sun OS 4.0 and IBM MVS systems. Rules may be developed for other platforms as well, since the CSM is designed to be operating system generic. For information, contact Chip Hatfield, LLNL, (415) 422-8567 or (FTS) 532-8567.

Midas

Midas is an intrusion detection based on an expert system. The system runs on a Symbolics Lisp machine, and is used to monitor a MULTICS system. It was developed by the National Computer Security Center (NCSC), and is based on the IDES research into user profiles and events. It is widely recognized as one of the best intrusion detection systems available, and is likely to be ported to other systems (e.g., UNIX) to assist other vendors in the certification process. For additional information on how to obtain this product, call Eric Shellhouse at (301) 859-4513.

8 - Exercises and Scenarios

Chapter Two - Overview of Incident Handling

1. Certain stages of the six stages of responding to incidents might be more important for some types of incidents than others. Explain why.

2. Which of the six stages is most likely to be overlooked? Why is it important to not overlook this stage?

3. Suppose you were handed a disk and were told that it contained a new virus called SPREAD, but the routine used by the virus to replicate itself had been removed. a) would the malicious code left on the disk best be called a virus, worm, or trojan horse? b) How would you analyze the code on the disk, and what precautions would you take during this process?

4. Below is a list of symptoms of three types of incidents, virus infections, hacker attacks, and worm attacks. For each symptom determine the type of incident with which it is most closely associated by placing a check in the appropriate box. For example, if you think that the first symptom, system crashes, is most indicative of a virus infection, check the first box to the right of "system crashes" (under "virus"). If you think that system crashes are most indicative of a hacker attack, check the second box to the right of "system crashes" (under "hacker"), instead.

Chapter Three - Responding to Viruses

1. Suppose that your personal computer's disk crashes every third boot. How could you determine whether your machine has a virus?

2. a) How could you assure that your virus eradication tool is not itself infected? b) How could you know that it eradicates the virus which has infected your machine? c) How could you know that you have the latest version of this tool?

3. You have determined that your PC has been infected by "Disk Killer." Should you turn your machine off immediately? Explain.

4. Given a virus that masks all interrupts into the operating system, how would you write an anti-virus code to detect/eradicate this virus?

5. Can two or more viruses infect a single machine at one time? Explain.

6. While editing a classified file on a Macintosh you notice that the file is infected by nVIR. What procedures should you follow to recover?

Chapter 4 - Responding to Worm Attacks

1. A user who accesses a mainframe from his Macintosh discovers that his account has been penetrated by a worm. He immediately turns his Macintosh off and calls the system manager. What advice should his system manager give him? Now the system manager confirms that there is a worm and turns the attacked host off. What advice would you give the system manager?

2. Suppose that there is a worm in a classified network. What additional actions must be taken to deal with a worm that might not have to be taken in an unclassified network?

3. Given the PS list (below), what are the suspicious processes? Would this indicate a worm, hacker or virus?

Chapter Five - Responding to Hacker/Cracker Attacks (Scenarios)

1. A system programmer notices that at midnight each night, someone makes 25 attempts to guess a username-password combination. What should the programmer do next? What is your organization/site's procedure?

   Suppose that the programmer decides to do nothing. Two weeks later, the password guessing attempts continue. The programmer checks the log and finds that each night the same username-password combination is being used. What should you do, based on your procedures?

2. A system programmer gets a call reporting that a major underground cracker newsletter is being distributed from the administrative machine at his center to five thousand sites in the U.S. and Western Europe. What do your procedures say he should do? What should you do?

3. A user calls in to report that he cannot login to his account at 3 a.m., Saturday. The system staffer cannot login either. After rebooting to single user mode, he finds that the password file is empty. What should be done?

   By Monday morning your staff determines that a number of privileged file transfers took place between this machine and a local university. What should be done now?

   By Tuesday morning a copy of the stolen password file is found on the university machine along with password files for a dozen other machines. What has happened?

4. You receive a call saying that a break-in to a Government laboratory occurred from one of your center's machines. You are requested to provide accounting files to help track down the attacker. Do you have a policy for helping others? What should you do?

5. You hear reports of a computer virus that paints trains on CRTs. You login to a machine at your center and find such a train on your screen. You look in the log and find no notation of such a feature being added. You notice that five attempts were made to install it, each exactly one hour apart. What should you do?

6. You notice that your computer connected to the Internet has been broken into. You find that nothing is damaged. A high school student calls up and apologizes for doing it. What should you do?

7. A reporter calls up asking about the breakin at your center. You haven't heard of any such incident. Three days later you learn that there was an intrusion due to a poorly chosen password on one account. What should you do?

8. A change in system binaries is detected. Thanks to your good prevention policy, the binaries are changed back to the original. The next day the binaries are changed again. This repeats itself for several weeks. What should you do?

Answers to Exercises and Scenarios

Chapter 2

1. There will be a wide range of acceptable answers. For example, detection is extremely critical (and difficult) in many hacker/cracker attacks.

2. Follow-up. This stage provides lessons learned, justification of the incident handling activity to management, etc.

3. a) The code would best be called a trojan horse, since it is malicious code that is not self-replicating. b) The code should be run in an isolated, backed-up machine and its symptoms should be recorded. Tools to monitor memory or disassemble the code may be used. Precautions must be taken to isolate the code, since there may be more than one method of replication.

4.

Chapter 3

1. There is no 100 percent guaranteed detection procedure. Some possibilities include: 1) check file sizes against a write-protected backup (but remember, the backup may be infected), 2) use a virus detection package, and 3) watch for unusual activity, such as odd disk access or disks that do not exist, unusual graphic displays, etc.

2. a) Run the tool on itself (but be sure to use a write-protected disk) or check the tool against a trusted source. Some manufacturers have authentication tools to verify their product. b) Use your detection procedures after using the tool. c) Get the tool from a trusted source, or from the originator of the tool. You could also read Virus-L, contact CIAC, or find a knowledgeable person.

3. No. In the case of disk killer, there is information stored on completion of the virus that will aid in the recovery of the original files.

4. Many methods exist. The important thing to realize is that if you provide the correct vector addresses in your program, the virus could intercept them and substitute its own values. The correct values must be computed and compared to the actual vectors to assure that the virus is not present. Also, care must be taken to not use interrupt services in the program.

5. Yes. nVIR A and nVIR B (Macintosh viruses) may infect the same application on a host, and, in this case, would create a new variant, nVIR C.

6. You should save the infected application on a WELL LABELED floppy, then re-format the disk and re-install all your applications and data from a recent backup. Then check for the virus again.

Chapter 4

1. Turning off the Macintosh is ineffective. A worm invades the host machine, not terminals or terminal emulators used to access the host. Notifying the system manager is an important action, however--users are one of the most important "detection tools." The system manager's actions are good ones. There are many things which now should be done, among which are: a) follow the site's reporting procedures, c) call CIAC to report the attack and to obtain immunization/eradication scripts, c) locate or check the latest backup, d) disconnect the system from all networks, and e) bring the system up in a restricted (i.e., single user) mode.

2. DOE Order 5637.1 does not specifically address the problem of a worm attack. Actions depend on whether the worm has changed the accessibility of classified information. We generally recommend that you start from a clean backup (assured to be worm-free). If the worm has caused any unauthorized disclosure of classified information, all disks on a system should be purged.

3. The ftp processes should be examined. This could be a hacker or a worm.

Chapter 5

1. The fact that the same username-password combination is used repeatedly suggests that this is not an intrusion attempt.

2. In this case, eight weeks later the authorities called to inform that the information in one of these newsletters was used to disable 911 in a major city for five hours. (This is a true story.) Your procedures should require that management should be notified of the type of information in this scenario.

3. A week later you find that your system initialization files have been altered maliciously. Recovery will be complicated, but at a minimum, passwords for all users should be changed on any systems with stolen passwords.

4. A week later you are given a list of machines at your site which have been compromised. Once you authenticate the caller, determine whether the caller has a need-to-know. If so, it is important to supply the needed information.

5. This is a "false alarm." Three days later you learn that the unusual graphic display was put up by a system administrator locally who had heard nothing about the virus scare or about your concern.

6. There is no outcome to this scenario. We have found, however, that persuading the hacker to meet you in person (if possible), acquainting the hacker with the problems he or she caused, and getting him/her to promise to quit this activity and possibly to assist you in subsequent attacks is very worthwhile. In this case, in which the hacker initiated contact and apologizes, your following through with law enforcement agencies is not necessarily advisable unless there is reason to believe that the hacker has damaged other machines or caused considerable trouble elsewhere.

7. In this case there was a breakin due to a poorly chosen password. Do not talk directly to the reporter. It might be good to have your public relations office contact the reporter and find out how the reporter learned the information related to you, however.

8. This is a potentially complicated situation. You need to attempt to rule out plausible causes for this unusual system behavior other than the work of an intruder. In this case, there was no intrusion.

References

Stoll, C. (1989) Tracking a spy through the maze of computer espionage: The cuckoo's egg. New York: Doubleday.

Wood, P.H., & Kochan, S.G. (1985) UNIX system security. Hasbrouck Heights, NH: Hayden.

RESPONDING TO COMPUTER SECURITY INCIDENTS: GUIDELINES FOR INCIDENT HANDLING

E. Eugene Schultz, Jr.
David S. Brown
Thomas A. Longstaff

University of California
Lawrence Livermore National Laboratory

July 23, 1990