

# **Network and Internetwork Security**

**Essay:**

**An Investigation into Trends and Techniques in Computer  
Intrusion**

<b>Name:</b>	<b>Cara Winterbottom</b>
<b>Student Number:</b>	<b>WNTCAR003</b>
<b>For:</b>	<b>Dr A Hutchison</b>
<b>Due:</b>	<b>15 April 2002</b>

## 1. Introduction

In a 2001 survey conducted by the Computer Security Institute and the FBI, American businesses and government organisations reported losses totalling \$378 million due to computer security breaches. The respondents who were willing to report losses numbered 186, which means that the actual figures in America and world-wide must be far greater than those reported.[24]

This essay examines the nature of these computer security breaches by examining different types of attacks that may be used. The attacks are all perpetrated by hackers, either personally or through writing programs. The term, hacker, is defined for the purposes of this essay as a person who uses computers to intrude on computer systems.

A hacker may be a curious teenager, a disgruntled employee, a government spy, a political activist, or many other things. There are lots of reasons for attacking computer systems. Some of these will be discussed later.

The kinds of attacks that are most commonly heard of involve programs, like viruses or worms. These programs spread through the Internet and other networks, causing such problems as denial-of-service and damage to software. The systems often have the potential to defend against attacks, however human error allows their propagation. For instance, patches for software weaknesses may be released and not installed by system administrators, default settings are not changed to increase security, users may not use secure passwords and users may open emails which are generally known to contain malicious code.

Organisations like the Computer Emergency Response Team (CERT) have been formed specifically to inform people about security issues and provide solutions. Much of the information about specific attacks that is used in this essay comes from the CERT web pages. However, there are so many software weaknesses that can be exploited, so many hackers attempting to attack systems and inform others about the weaknesses, and so many people for whom computer security is too costly, time-consuming and complicated to implement, that the number of attacks and the cost of the attacks keep rising.

This essay begins with a description of types of attacks on computer systems. This does not promise to be comprehensive, merely an exploration of some of the different attacks that exist. Software weaknesses, especially in Microsoft products, which have been exploited in recent attacks are then discussed. This is followed by a description of the differences between worms, viruses and Trojan horses. That section forms a prelude to a comprehensive examination of the Nimda worm. This program is interesting and relevant because it acts as a worm, a virus and a Trojan horse at different stages of its existence. It also exploits software weaknesses to gain entry into a system. It is an excellent example of new attacks that combine multiple existing attacks and have multiple points of entry into a system.

Finally, trends in software intrusion are discussed, such as the changes in techniques and types of attacks over the years and different views of hackers.

## 2. Types of Attacks

All hacker attacks seem to be about intruding on systems. There are many different ways of intruding, and of finding out information which makes intrusion easier. The most obvious kind of intrusion occurs when the hacker personally enters a system and performs actions, like stealing files or information, copying information or changing information. Other kinds of intrusion are viruses, worms and Trojan horses. These are remote intrusion techniques, where the hacker creates a program to intrude on the system and then releases it. See *Worms, viruses and Trojan horses* (section 2.2) for more information on these. Intrusion is made much easier by weaknesses in software. These are usually very quickly patched or worked around when they are discovered. However, new weaknesses are constantly discovered or patches are not installed by users. For more information on software weaknesses, see *Attacking weaknesses in software* (section 2.1).

Often viruses and worms carry a payload (and Trojan horses perform an action), which makes personal intrusion by a hacker easier. For instance, installing a back door, sniffing out passwords or installing guest accounts with administrative privileges on the infected system. Other ways of finding out information to facilitate intrusion are described briefly below.

One of the most common ways of finding out information for its own ends or to facilitate intrusion is social engineering. It is also one of the ways that has been used the longest, from the early days of hacking telephone lines before computer networks existed.[39] Social engineering is used when attackers trick users into giving them information or bluff their ways onto systems. For example, an attacker might send a bogus email, or construct a bogus web page which requires users to submit their passwords or credit card numbers. [28, 1, 20] Social engineering requires lax security in an organization or on the part of the user.

Another way of finding information is to place a sniffer program on a network. A sniffer generally works by placing the NIC (Network Interface Card) of a network into promiscuous mode. This gives the program the ability to capture and examine packets on the network, before sending them to their destinations. Sniffers are also known as network analysers and were originally used by security companies to protect against packet-based attacks. Sniffers are used by hackers to discover passwords and other confidential information passing through a network, including an idea of the structure of the network (this can be used for future attacks). Sniffers can often be detected on a system by increased CPU usage and missing disk space. Sniffers write the information they discover to log files, which use up space. The more connections a sniffer monitors, the more disk space its log file will use and the more CPU power it will use. In a large network a sniffer can generate as much as 10 megabytes a day.[1]

A very common type of attack that is used currently is denial-of-service. This kind of attack attempts to prevent users from accessing a particular resource. Examples include flooding a network with packets and so preventing normal network traffic and disrupting services to a specific system, computer or user. Many viruses and worms cause denial-of-service conditions either deliberately, or through their own usage of a system. Denial-of-service can be caused in a number of different ways, such as consumption of CPU time, network bandwidth or disk space or physically disabling connections.[4]

The most common kind of denial-of-service attacks are against networks. An example of this is the “SYN flood” attack, where the attacker begins to establish a connection with a server by sending SYN messages. The usual process is that the server responds with a SYN-ACK message and the client (in this case, the attacker or an IP address that the attacker has given) replies with an ACK message, whereupon the connection is established. However, in the case of a denial-of-service attack, the client does not respond (because the attacker has specified a return address which is unable to respond). The server is left with a half-open connection on which it must keep information. The server eventually times out and the connection is closed. However, in this case, the attacker sends multiple SYN messages, never giving the server time to recover. The server will be frozen and unable to respond to legitimate requests.[2]

In the case of network denial-of-service, it is often relatively simple to discover the address where the attack originated. It is also difficult to obtain enough bandwidth to create denial-of-service conditions from one address. Therefore, distributed denial-of-service is being used more and more. This works very similarly to denial-of-service, except that the attacks come from multiple machines. The attacker often gains control of these machines for this purpose by using a virus or a worm.

## **2.1 Attacking weaknesses in software**

There are always avenues of entry into a system for an attacker. Even when there are no specific weaknesses in the software, security settings may not be set up correctly. For instance, users often have passwords that are easy to break. However, often software has specific weaknesses that hackers use to gain entry into systems. This section will describe a few weaknesses that have been identified in Microsoft systems. Patches have been created for most of these weaknesses, however often these are not installed by system administrators because of the time and effort involved. Microsoft weaknesses are often found and exploited largely because of the ubiquity of the system. If a hacker exploits weaknesses in Microsoft software, then he or she has the potential to gain access to a large number of computers.[38]

### **2.1.1 Macros**

Macros have recently been used for attacks. This is because they are relatively easy to write and can take actions as if they were the user of a system. They also infect documents, which are exchanged more often than executables or disks. Macros run on different platforms (whichever platforms the programs in which they are used run on),

making malicious code very difficult to control. They are also present in many Microsoft products and very easy to trigger accidentally. The *Melissa* virus which spread in March 1999 was a Word macro virus which was sent as an email attachment.[13,1] Microsoft PowerPoint and Excel have a weakness called *Automatic Execution of Macros* by CERT.[10] This allows an attacker to include a macro in Excel or PowerPoint documents that will not be detected and will run automatically. Normally, when Excel or PowerPoint find macros, the user is asked whether the macros should be run. However, macros which are formed in a specific way may not be detected and so execute automatically. This would allow the attacker to act on the system with the same privileges as the user, e.g. sending emails, changing information, formatting the hard drive or changing security settings.[32]

### 2.1.2 Buffer Overflows

Buffer overflows are a common weakness in many software systems, that allow an attacker to gain entry. A buffer overflow occurs when too much data is packed into a contiguous block of memory. This causes other blocks of memory to be overwritten, leading to unexpected actions by the program. Attackers can manipulate this flaw by writing code that will cause overflow of appropriate assembly language code into specific places in memory. A common type of buffer overflow attacks involves causing a buffer that is allocated space on the stack to overflow. This places the attacker's code on the stack. The return address of the original code is also changed to point to the attacker's code, which then executes. C programs running on Unix systems have frequently been attacked in this way, often because of dynamically allocated variables. In this case, a program will not allocate memory to variables until runtime. If too much data is inserted into a dynamically allocated buffer at this point (in other words, the buffer cannot be increased in size because of memory constraints), it will overflow.[1]

There have been several buffer overflow weaknesses identified in Microsoft software some of which will be described below. In June 2001, CERT released information on the *Buffer Overflow in IIS Indexing Service DLL*. [7] This weakness was exploited by the *Code Red* worm to spread. It is an unchecked buffer in the IDQ.DLL ISAPI (Internet/Indexing Service Application Programming Interface) extension which overflows and allows entry by an attacker. This occurs before indexing is required, so the actual Indexing Services do not need to be active. Script mapping for Internet Data Administration and Internet Data Query files must be active so that the attacker can establish a web session. At this point the attacker can execute code with the privileges of the user.[25] There are various other buffer overrun problems in Microsoft Internet Information Services software, which have been discovered recently.[35]

In December 2001, eEye Digital Security discovered a buffer overflow problem in the Universal Plug and Play (UPnP) Microsoft service. [13,11,33] This is a protocol for allowing hardware and software to interface, for instance to find the correct drivers for a mouse or digital camera. The buffer overflow may allow attackers to execute code on computers with System privileges (in Windows XP, System privileges are the highest used), or launch denial-of-service attacks. The buffer overflow is in the UPnP code to

process NOTIFY directives, which is part of the device discovery subsystem. When a device that supports UPnP boots, it broadcasts a NOTIFY directive announcing its presence to all computers nearby. An attacker could send a NOTIFY directive crafted to cause this overflow to the IP address of any computer and thus run code with full System privileges. This is because UPnP runs as part of the operating system and can thus be used to take control of the system. The attacker could send the directive as a multicast or broadcast message (as if from a booting device), thus gaining access to multiple computers and possibly causing denial of service conditions.

More recently, in February 2002, CERT released an advisory about *Buffer Overflow in Microsoft Internet Explorer*. [34, 30] This occurs in code where Internet Explorer handles embedded objects in HTML documents. It allows an attacker to infect a computer and execute code with user privileges when the user visits a web page or views and HTML email message. Code as follows is used to embed objects (in this case an audio file) in HTML:

```
<EMBED TYPE="audio/midi" SRC="/path/sound.mid" AUTOSTART="true">
```

The SRC attribute specifies the path to the sound file. However, the code routine to parse this attribute for a location is not properly handled by Internet Explorer and can cause a buffer overflow.

### 2.1.3 Automatic execution of MIME types

A weakness which is exploited by the Nimda worm discussed below is described by CERT as *Automatic Execution of Embedded MIME Types*. [5,31] MIME stands for Multipurpose Internet Mail Extensions. It is an Internet standard for encoding binary files as email attachments. When Microsoft Internet Explorer (IE) parses the MIME parts of a document (when a user is rendering HTML by browsing a file system, reading email or visiting a web page), it uses a table to determine how to handle the MIME types. The weakness is in this table and causes IE to open the MIME part of the document automatically. An attacker can use this weakness by creating a document containing malicious code, specifying the document as a specific unusual MIME type and sending it as an email attachment. This code could be executed with user privileges, without the user opening the attachment.

### 2.1.4 Filename decode weakness in Internet Information Server (IIS)

A second weakness exploited by the Nimda worm is described by CERT as *Directory Traversal Vulnerability*. [6,36]

URIs can be encoded according to RFC 2396. This provides encoding for text using the percent sign and hexadecimal characters. When it receives a URI, IIS will decode it to remove this encoding. First the filename is decoded to check if it is an executable. Then the URI is decoded again. However, instead of only decoding the other parameters of the URI, IIS decodes the decoded filename as well. This is the error. If an attacker creates a

filename in a specific way, he or she can get around some of the IIS filename security checks, which are only performed on the first decoding. An example from the NSFOCUS web site is as follows:

*A character '\' will be encoded to "%5c". And the corresponding code of these 3 characters is:*

*'%' = %25*

*'5' = %35*

*'c' = %63*

*encode this 3 characters for another time, we can get many results such as:*

*%255c*

*%%35c*

*%%35%63*

*%25%35%63*

*....*

*Thereby, '..\' can be represented by '..%255c' and '..%%35c', etc. After first decoding, '..%255c' is turned into '..%5c'. IIS will take it as a legal character string that can pass security check-up. But after a second decode process, it will be reverted to '..\''. Hence, attacker can use '..\' to carry out directory traversal and run arbitrary programs outside of Web directory.*

Normally, IIS does not allow files to be executed outside the Web directory.

This allows an attacker to construct a URL with specific unicode characters, providing a relative reference to an executable in a directory other than that of the web server. Usually IIS would prevent access, but because of the weakness this is allowed through. The attacker is given the privileges of the IUSR\_machinename account and can install and run code, or access and change existing files on the server.

## **2.2 Worms, viruses and Trojan horses**

Worms, viruses and Trojan horses are created by hackers to intrude on systems. The type of damage that is done while on those systems depends on the payload of these programs. Concrete definitions exist for these three types of programs (and are given below), however the differences between them (especially between viruses and worms) are slight. The issue is complicated by the fact that the press tends to call every malicious program that spreads a virus, e.g. the Love Bug was a worm, but has been named a virus. Also, some programs seem to act as a worm, a virus and a Trojan horse at different stages of their existence.

### **2.2.1 Viruses**

A virus is a self-replicating program containing code that copies itself and can infect other programs by changing them, so that when they are executed the virus is also

executed. Apart from infecting other programs, viruses usually perform other actions, such as erasing files, printing messages or sending emails.[17,12,15]

Most viruses contain code to check if they have already infected programs with which they come into contact. This is to help avoid detection, as each time a virus infects a program, it increases its length.

### **2.2.2 Worms**

A worm is a self-replicating program that spreads copies of itself from program to program or from computer to computer (via network connections). Once the worm has infected a machine, it does not require any user interaction to spread. Unlike viruses, worms can run independently. They also do not change other programs.[17,12]

The first publicised worm was released in November 1988 by Robert Morris. It spread very fast and within hours had infected thousands of computers on the ARPANET. The worm infected computers running Berkely UNIX and spread using TCP/IP network protocols. The worm did not actually do any damage to computers, but it caused severe bandwidth problems because it replicated itself so easily.[12,39]

### **2.2.3 Trojan horses**

A Trojan horse is a program that pretends to be something it isn't. It performs a useful action that the user expects, but also performs hidden actions which are not expected (they can only perform actions that the user has security privileges to perform). Trojan horses are often used to introduce viruses onto systems. A Trojan horse that destroys or erases files is also called a logic bomb.[17,12,3]

Trojan horses must be installed. They can be installed by viruses or worms, or by the user (who may think that the program is benign). Social engineering is often used to persuade a user to install a Trojan horse, for instance it might be disguised as a software patch or a computer game. In order to perform actions, Trojan horses must be activated by the user. Therefore, they usually masquerade as programs that a user is likely to execute.

## **3 The Nimda worm/virus/Trojan horse**

The W32/Nimda (admin spelled backwards) worm is unusual and worth examining in detail for a number of reasons. First of all, at different stages of its cycle it acts as a worm, a virus and a Trojan horse. In other words, it cannot easily be classified. It also has multiple possible ways of spreading, using various software vulnerabilities and backdoors left by previous worms. It infects web sites and email, and has the potential to cause denial of service. Nimda attacks home computers as well as servers and routers.[26, 21]

### 3.1 Email propagation

The worm propagates through email by acting as a virus. It takes the form of an email which is a MIME “multipart/alternative” message with two sections. The message itself is a MIME type “text/html”. It has variable subject lines and no content. There is an attachment titled *readme.txt* or *admin.dll*, which is disguised as a MIME type “audio/x-wav”. However, it is actually a binary executable. Once this attachment is executed, the machine is infected with the worm. The matter is complicated by a weakness in Microsoft Internet Explorer 5.5 SP1 or earlier. This weakness is described above in section 2.1.3. Any mail software that runs on an Intel chip based PC and that runs Internet Explorer, will use Internet Explorer to render the mail (because it is an HTML mail). If the version of Internet Explorer is one of those mentioned above and it has not been patched, it will automatically execute the attachment when the mail is opened or previewed, without any user interaction. [9,40]

The worm contains a routine that attempts to send emails (and therefore infect other computers) every 10 days. It will search for email addresses in two ways on the local system. First, it searches all .htm or .html files in the web cache folder. Then, it uses the Messaging API to search through all messages in MAPI-compliant email programs (e.g. Microsoft Outlook and Outlook Express). The messages are passed through a pattern matcher, which finds strings that look like email addresses. These addresses are used both for the sending and receiving addresses of the emails that are then sent. This means that the mails will seem to have been sent by multiple users and not the infected computer. In order to do this, the worm uses the infected computer’s DNS (Domain Name System) entry to create and register a mail server record, so that it can act as an SMTP (Simple Mail Transfer Protocol) server.

### 3.2 Web propagation

The Nimda worm attempts to capitalise on a number of weaknesses in the Microsoft Internet Information Server (IIS), and backdoors left by the Code Red II and sadmind/IIS worms. If the IIS weaknesses have not been patched, then it will be able to spread using the web. These weaknesses are described above in section 2.1.4. They are very similar and are caused because IIS decodes URI input twice, unnecessarily. Security checks are only performed on the first decoding and not the second.

The Nimda worm uses this weakness to copy itself onto the web server as a file called *admin.dll*, using TFTP. This is performed from an already infected machine, which creates a TFTP server to listen at port 69. This infected machine finds web servers by scanning for randomly generating IP addresses. If any servers found do not have patches for the weaknesses explained above, then they are infected. The worm also looks for servers that have previously been infected by the Code Red II or sadmind/IIS worms. These worms left backdoors with which Nimda can infect a server, using the *root.exe* or the *cmd.exe* files.

The *admin.dll* file is then executed and copied to multiple locations on the server. Once on the server, the worm looks for files with the extensions .htm, .html or .asp. It appends the following Javascript code to every such file that it finds:

```
<script language="JavaScript">
window.open("readme.eml", null, "resizable=no,top=6000,left=6000")
</script>
```

The consequence of this is that anyone who opens one of these web pages will be presented with the *readme.eml* file. This file is an Outlook Express email file with the worm attached (as described above under email propagation). If the computer of the person browsing the web page has the Internet Explorer weakness described above, it will be infected automatically.

### 3.3 Nimda actions

When the Nimda worm is executed on a computer, it replaces the file *Riched20.dll* (This is part of its Trojan horse functionality). This file is used by Windows programs like Microsoft Word. As a result the worm is executed every time one of these programs is executed, which further increases its influence. The worm also copies itself as *Riched20.dll* into all folders containing .doc files. This means that every time a .doc file is opened, the corrupted version of *Riched20.dll* will be used, as it is in the local directory.

Nimda also attempts to infect all .exe files, after checking whether they have already been infected. If a .exe is not infected, Nimda copies itself into the Temp directory and embeds the .exe into this copy. This new file (another Trojan horse) is copied over the old .exe. Nimda creates multiple copies of itself that are MIME-encoded in all directories that are accessible by the user of the infected computer. These files have .eml and .nws extensions. If any of them are opened or previewed by other users (i.e. versions which are in shared network drives), they can spread the worm to the computers and systems of those users.

Nimda convinces the system to start it up whenever the computer is started by modifying the System.ini file as follows:

```
Shell = explorer.exe load.exe -dontrunold
```

It also registers itself as a service process so that it can continue to run even when there are no users logged onto the system.

Nimda opens the infected computer up to outside users and possible intrusion in two ways. First, it changes the registry key *HKLM\Software\Microsoft\Windows\Current\Version\Network\LanMan\[c\$ ->Z\$]* so that all of the drives on the computer become open network shares.

Secondly, it creates a Guest account on Windows and gives this account administrative privileges.[9,40]

### **3.4 Effects of Nimda**

One effect of the worm are to decrease security on an infected machine, in that an intruder can use the Guest account that has been created with administrative privileges and all drives are now open network shares.

The worm can also cause denial of service conditions on networks, due to the bandwidth required by infected machines for scanning IP addresses and sending emails.

## **4. Trends**

The activities associated with hacking have changed over time in some ways, and in other ways remained the same. The principles behind attacks are generally similar to what they were originally (see below for a discussion of this). The same attacks are also used, for instance worms and viruses as concepts have existed for over twenty years and have been used for most of that time. The changes have occurred because of the way technology has changed. Programs have become more and more flexible and complex, allowing increasingly flexible and complex hacks to occur. For instance, the MIME weakness discussed in section 2.1.3. Also, the Internet has grown enormously, allowing attacks to be much more wide-spread and damaging than they would otherwise be. [39]

The Internet allows information about almost anything to be accessed very easily. This means that hackers can share tools, such as virus kits, denial-of-service tools, sniffer programs, etc. Hackers can also share information about weaknesses in software and the security status of specific web sites and companies. On the other hand, the Internet also allows security patches for software can be distributed easily and quickly, cutting off entry points into systems for hackers. Security personnel can share information about hackers and their tools as easily as hackers can share attack information. Organisations like the Computer Emergency Response Team (CERT) provide a reliable source of security information and are able to do this easily through the Internet.

Hacking has become a household concern, as computers have become a household item. Everyday users are concerned about being infected with viruses and worms, and companies have even more to worry about. They have to secure multiple users connected to each other and the world through networks. CERT keeps a list every year of the number of hacking attacks (including attacks by worms and viruses) reported to them. In 1989, 132 incidents were listed. By 2000, the list had almost 20 000 entries. This number doubled in 2001. These figures are mostly due to the fact that computers all over the world are connected by networks. The *Love Bug* virus that spread recently originated in Pakistan and rapidly infected computers all over the world. [21,23 ]

Beginning, some would say, with the Morris Internet worm in 1988, media attention has focussed on specific hacking incidents, making them part of general public knowledge. Reactions by authorities have also increased and become more documented. Computer intrusion acts have been passed by most governments to allow hackers to be prosecuted relatively easily when they are caught. The process of tracking down and catching hackers has also become more efficient. Special computer security divisions have been set up in many countries, specifically for tracking down and collecting evidence against hackers.[39,28]

The reasons for hacking have become more varied. The hackers who attack systems in order to “see what they can do” still exist, as do those who believe that all information should be public. There will always be those who hack for monetary gain, for instance hacking banks and creating accounts or stealing credit card numbers. However, more and more, hacking is used to accomplish specific political goals. Following the September 11 bombings, there was a spate of “patriotic hacking” by American citizens against Arab targets. There have also been accusations of governments hacking systems of other governments. For instance, the CIA has been accused by Japanese and French governments and the European Commission of hacking into their systems to obtain trade secrets. [28,16,22,29]. Hackers can be criminals, terrorists, spies, freedom fighters or just marginalised members of society.

## **5. Conclusion**

As mentioned in the Introduction, computer intrusions are becoming more frequent. It used to be the case that companies and individuals would protect their systems like they would protect their homes against fire. In other words, attacks were fairly unusual. However, with the growth of the Internet, attacks are increasing in frequency. The cost of attacks is also increasing, because so many more systems can now be affected.

In order to combat these attacks, systems need to be protected more carefully and users must be better educated about the consequences of security breaches. Recently, Microsoft has stated that it will focus on improving security of existing software instead of adding new features.[37] This is a step in the right direction, but it is very late in coming. Also, ultimately systems need to incorporate security as they are being built.

This essay has examined techniques and trends in computer intrusion and shown how both security and intrusion issues are evolving. Hopefully, it has given some insight into the wide variety of attacks and possible entry points into systems, indicating the severity of the situation. This should indicate that computer security is a very important part of software development and business planning. It should be considered when software or businesses are first being designed and not as an afterthought.

## 6. References

1. Buzzard, K. Computer security – What should you spend your money on?. In: *Computers & Security*, Vol. 18, No. 4. Elsevier Science Ltd (1999).
2. Computer Emergency Response Team. *CERT® Advisory CA-1996-21 TCP SYN Flooding and IP Spoofing Attacks*. Available: <http://www.cert.org/advisories/CA-1996-21.html>. Accessed: 8 April 2002.
3. Computer Emergency Response Team. *CERT® Advisory CA-1999-02 Trojan Horses*. Available: <http://www.cert.org/advisories/CA-1999-02.html>. Accessed: 8 April 2002.
4. Computer Emergency Response Team. *Denial of Service Attacks*. Available: [http://www.cert.org/tech\\_tips/denial\\_of\\_service.html](http://www.cert.org/tech_tips/denial_of_service.html). Accessed: 8 April 2002.
5. Computer Emergency Response Team. *CERT® Advisory CA-2001-06 Automatic Execution of Embedded MIME Types*. Available: <http://www.cert.org/advisories/CA-2001-06.html>. Accessed: 8 April 2002.
6. Computer Emergency Response Team. *CERT® Advisory CA-2001-12 Superfluous Decoding Vulnerability in IIS*. Available: <http://www.cert.org/advisories/CA-2001-12.html>. Accessed: 8 April 2002.
7. Computer Emergency Response Team. *CERT® Advisory CA-2001-13 Buffer Overflow in IIS Indexing Service DLL*. Available: <http://www.cert.org/advisories/CA-2001-13.html>. Accessed: 8 April 2002.
8. Computer Emergency Response Team. *CERT® Advisory CA-2001-19 Code Red Worm Exploiting Buffer Overflow in IIS Indexing Service*. Available: <http://www.cert.org/advisories/CA-2001-19.html>. Accessed: 8 April 2002.
9. Computer Emergency Response Team. *CERT® Advisory CA-2001-26 Nimda Worm*. Available: <http://www.cert.org/advisories/CA-2001-26.html>. Accessed: 8 April 2002.
10. Computer Emergency Response Team. *CERT® Advisory CA-2001-28 Automatic Execution of Macros*. Available: <http://www.cert.org/advisories/CA-2001-28.html>. Accessed: 8 April 2002.
11. Computer Emergency Response Team. *CERT® Advisory CA-2001-37 Buffer Overflow in UPnP Service on Microsoft Windows*. Available: <http://www.cert.org/advisories/CA-2001-37.html>. Accessed: 8 April 2002.
12. Denning, P.J. (ed) *Computers Under Attack: Intruders, Worms and Viruses*. ACM Press, New York (1990).
13. Docherty, P. and Simpson, P. Macro attacks: What next after Melissa?. In: *Computers & Security*, Vol. 18, No. 5. Elsevier Science Ltd (1999).
14. Eeye Digital Security. *UPNP - Multiple Remote Windows XP/ME/98 Vulnerabilities*. Available: <http://www.eeye.com/html/Research/Advisories/AD20011220.html>. Accessed: 28 March 2002.
15. Fites, P., Johnston, P., Kratz, M. *The Computer Virus Crisis*. Van Nostrand Reinhold, New York (1989).
16. Furnell, S.M. and Warren, M.J. Computer hacking and cyber terrorism: The real threats in the new millenium? In: *Computers & Security*, Vol. 18, No. 1. Elsevier Science Ltd (1999).
17. Ford, Dr. R. Malware briefing. In: *Computers & Security*, Vol. 17, No. 2. Elsevier Science Ltd (1998).

18. Ford, Dr. R. Malware: Troy revisited. In: *Computers & Security*, Vol. 18, No. 2. Elsevier Science Ltd (1999).
19. Ford, Dr. R. No surprises in Melissa Land. In: *Computers & Security*, Vol. 18, No. 4. Elsevier Science Ltd (1999).
20. Hancock, Dr. B. Security Views. In: *Computers & Security*, Vol. 17, No. 2. Elsevier Science Ltd (2001).
21. Hancock, Dr. B. Security Views. In: *Computers & Security*, Vol. 20, No. 8. Elsevier Science Ltd (2001).
22. Hinde, S. Cyber wars and other threats. In: *Computers & Security*, Vol. 17, No. 2. Elsevier Science Ltd (1998).
23. Hinde, S. Love Conquers All?. In: *Computers & Security*, Vol. 19, No. 5. Elsevier Science Ltd (2000).
24. Hinde, S. Cyberthreats: Perceptions, reality and protection. In: *Computers & Security*, Vol. 20, No. 5. Elsevier Science Ltd (2001).
25. Hinde, S. Omnia te adversum spectantia, nulla retorsum. In: *Computers & Security*, Vol. 20, No. 6. Elsevier Science Ltd (2001).
26. Hinde, S. Incalculable potential for damage by cyber-terrorism. In: *Computers & Security*, Vol. 20, No. 7. Elsevier Science Ltd (2001).
27. Kovacich, Dr. G. Electronic-Internet business and security. In: *Computers & Security*, Vol. 17, No. 2. Elsevier Science Ltd (1998).
28. Kovacich, Dr. G. I-Way robbery: Crime on the Internet. In: *Computers & Security*, Vol. 18, No. 3. Elsevier Science Ltd (1999).
29. Kovacich, Dr. G. Hackers: Freedom fighters of the 21<sup>st</sup> century. In: *Computers & Security*, Vol. 18, No. 7. Elsevier Science Ltd (1999).
30. Internet Security Systems. Buffer Overflow in Microsoft Internet Explorer. Available: [http://www.iss.net/security\\_centre/alerts/advisories.html](http://www.iss.net/security_centre/alerts/advisories.html). Accessed: 8 April 2002.
31. Microsoft. Microsoft Security Bulletin (MS01-020). Available: <http://www.microsoft.com/technet/security/bulletin/MS01-020.asp>. Accessed: 10 April 2002.
32. Microsoft. Microsoft Security Bulletin (MS01-050). Available: <http://www.microsoft.com/technet/security/bulletin/MS01-050.asp>. Accessed: 10 April 2002.
33. Microsoft. Microsoft Security Bulletin (MS01-059). Available: <http://www.microsoft.com/technet/security/bulletin/MS01-059.asp>. Accessed: 10 April 2002.
34. Microsoft. Microsoft Security Bulletin (MS01-005). Available: <http://www.microsoft.com/technet/security/bulletin/MS01-005.asp>. Accessed: 10 April 2002.
35. Microsoft. Microsoft Security Bulletin (MS01-018). Available: <http://www.microsoft.com/technet/security/bulletin/MS01-018.asp>. Accessed: 10 April 2002.
36. NSFOCUS (Network Security Focus). Microsoft IIS CGI Filename Decode Error Vulnerability (SA2001-02). Available: <http://www.nsfocus.com/english/homepage/SA2001-02.htm>. Accessed: 9 April 2002.

37. Schneier, R. Microsoft and security engineering. In: *Cryptogram*. Counterpane Internet Security, Inc. (February 2002).
38. Schultz, Dr. E.E. Windows NT Security: Kudos, concerns, and prescriptions. In: *Computers & Security*, Vol. 18, No. 3. Elsevier Science Ltd (1999).
39. Sterling, B. *The Hacker Crackdown*. Bantam Books, New York (1992).
40. Symantec Security Response. W32.Nimda.A@mm. Available: <http://securityresponse.symantec.com/W32.Nimda.A@mm.htm>. Accessed: 8 April 2002.
41. Welch, D., Buchheit, N. Ruocco, A. Strike-back: Offensive actions in information warfare. In: *Proceedings of the New Security Paradigm Workshop Sept 22 – 24, 1999*. ACM, Inc, New York (2000).

This document was created with Win2PDF available at <http://www.daneprairie.com>.  
The unregistered version of Win2PDF is for evaluation or non-commercial use only.