FREE RADICALS IN CYBERSPACE

COMPLEX LIABILITY ISSUES IN INFORMATION WARFARE

MEIRING de VILLIERS
John Landerer Faculty Fellow
University of New South Wales, School of Law
Sydney, NSW 2052
AUSTRALIA

mdv@unsw.edu.au

Abstract

During the weeks leading up to September 11, 2001, successive versions of a particularly destructive and complex Internet worm, named W32/CodeRed, took information warfare to a new level of complexity and danger. W32/CodeRed was the forerunner of a new wave of malevolent software, known as a blended attack. A blended cyber attack exploits security vulnerabilities in a target computer system in combination with computer viruses carrying multiple destructive properties. The blended attack exploits synergies between the virus and the security vulnerability to enhance the effectiveness and destructiveness of the virus payload. Blended attacks present complex liability issues, including apportionment of liability among the original tortfeasor (the creator of the security vulnerability) and the second tortfeasor (the exploiter of the vulnerability.) The liability of an original tortfeasor is usually cut off by an intervening crime or intentional tort. The pattern of common law decisions suggests, however, that the liability of an original tortfeasor will be preserved if he or she created an opportunity for free radicals. Free radicals are individuals who are not deterred by the threat of liability. The analysis in the article suggests that virus authors and distributors have properties commonly associated with free radicals. The analysis informs, furthermore, that the factors that influence courts in holding a defendant liable for encouraging free radicals are present in a typical blended attack. We conclude that liability will be preserved against a primary tortfeasor whose negligence was responsible for a vulnerability intentionally exploited by a free radical cyber attacker. The primary tortfeasor is likely a solvent commercial entity, while the attacker is often judgment-proof, or otherwise shielded from liability. The result is therefore significant, especially for the victim of a blended attack seeking to recover damages related to the attack.

Acknowledgements

Mark Grady
Peter Szor
Audience at Georgia Tech

Table of Contents

1. Introduction

During the weeks leading up to September 11, 2001, successive versions of a particularly destructive and complex Internet worm, named W32/CodeRed, took information warfare to a new level of complexity and danger. By exploiting a common network vulnerability, the rapidly spreading CodeRed slowed down and compromised the security of the Internet, and attempted to launch denial of service attacks on the official White House Web page.[1] The first version of the worm, which we shall refer to as CodeRed-I,

[1] CodeRed infected over 300,000 machines within 24 hours. Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 98.

exploited a security vulnerability in Microsoft's Internet Information Services (IIS) web servers.[2]

Once CodeRed-I infected a machine, it checked whether the current date was between the first and nineteenth of the month. If that were the case, the worm probed a randomly generated list of machines for a vulnerability and continued the infection cycle. Between the twentieth and twenty eighth of every month, the worm turned its attention from other machines and proceeded to launch a denial of service attack on the official White House web page, www.whitehouse.gov. The worm remained dormant between the twenty eighth and the end of the month.

Due to a programming flaw, CodeRed-I spread slower than intended, yet infected enough hosts to cause a significant denial of service slowdown in the infected systems. Its attempted attack on the White House Web page failed, because the site was moved to a new IP address, following an intelligence alert.[3] The worm code continued to target the old address, while legitimate traffic was redirected to the new address.

A more destructive sequel, CodeRed-II, followed soon. CodeRed-II was similar to its predecessor, but had a greater impact on the global information infrastructure and did more harm, in part due to its more efficient propagation algorithm.[4] The new version spread multiple times faster and also created a back door[5] on infected systems. The backdoor installed by CodeRed-II enabled a hacker to gain access to confidential files

---

[2] The attacks occurred shortly after Microsoft had discovered the vulnerability and issued a patch to fix it. Microsoft, *A Very Real and Present Threat to the Internet*. http://www.microsoft.com/technet/treeview/default.asp?url=/technet/security/topics/codealrt.asp. Section xxx discusses the principles of the buffer overflow vulnerability.

[3] Jeremy D. Baca, *Windows Remote Buffer Overflow Vulnerability and the Code Red Worm*. SANS Institute White Paper (September 10, 2001). ("With plenty of warning about the coming attack, the site's IP address was moved from 198.137.240.91 to 198.137.240.92.")

[4] CodeRedII infected more than 359,000 computers in fourteen hours. See, e.g., Silicon Defense, *Code Red Analysis Page*. http://www.silicondefense.com/cr/.

[5] A back door is a method of gaining remote access to a computer, and is usually not detectable by casual inspection. See, e.g., http://en.wikipedia.org/wiki/Back_door. A backdoor may, for instance, consist of a password recognition routine installed in the computer, perhaps as a modification of a legitimate program. The routine would enable a hacker who provided the right input to gain access to confidential files and programs on the computer.

and programs on the compromised computer.[6] CodeRed-II exploited the same vulnerability as its predecessors.[7]

On August 4, 2001, a new worm, CodeRed-III, appeared, which exploited the same vulnerability as its predecessors.[8] CodeRed-III would infect its target system, initiate its propagation mechanism, and set up a back door into the infected machine. After installing the backdoor, it remained dormant for a day, rebooted the machine, and began to spread. The back door allowed remote, administrator-level access to the infected machine. This enabled the compromised system to be used as a launching pad for future denial of service attacks, among other hazards.[9]

The CodeRed family of worms was promptly followed by the fast spreading and complex threat, named Nimda. Nimda struck on 18 September 2001, within days of warnings issued by government agencies, including the Federal Bureau of Investigation and the National Infrastructure Protection Center. Nimda was described as "the most complicated malicious application to strike the Internet to date," and the office of the US Attorney-General predicted that Nimda would be more harmful than CodeRed.[10]

Nimda was a worm, but, like CodeRed, differentiated itself from other Internet worms in its exploitation of network security flaws, its use of multiple vectors of infection and propagation, and the resulting efficiency and speed by which it spread. Among many exploits, Nimda searched automatically for vulnerable Microsoft IIS Web servers to infect, and it used backdoors created by CodeRed. The different attack vectors resulted in multiple points of damage, which made clean-up particularly difficult after a

---

[6] Jeremy D. Baca, *Windows Remote Buffer Overflow Vulnerability and the Code Red Worm*. SANS Institute White Paper (September 10, 2001), at 5.

[7] Attacks occurred despite the fact that a patch had been issued for the vulnerability by Microsoft before the first attack. The Code Red-I as well as Code Red II worms could, for instance, not infect a system that had the MS01-033 patch installed. See, e.g., Baca, at 6 ("There was so much publicity and so many published articles by the time Code Red II hit, that any competent server manager would have had ample opportunity to patch their systems in time.") CodeRed-II also compromised devices with web interfaces, such as routers, switches, DSL modems, and printers. See, e.g., Cisco Systems, Inc., *Cisco Security Advisory: Code Red Worm - Customer Impact*. http://www.cisco.com/warp/public/707/cisco-code-red-worm-pub.shtml.

[8] eEye Digital Security, *CodeRedII Worm Analysis*, August 2001. http://www.eeye.com/html/Research/Advisories/AL20010804.html.

[9] A denial-of-service attack (also, DoS attack) is an attack on a computer system or network that causes a loss of service to users, typically the loss of network connectivity and services by consuming the bandwidth of the victim network or overloading the computational resources of the victim system.

[10] George V. Hulme, Nimda's Biography, InformationWeek, September 19, 2001.

Nimda attack.[11] Nimda's infection and attack vectors were not novel, individually, but their combination constituted a new level of complexity in malevolent code.

CodeRed and Nimda were the forerunners of a new genre of modern malevolent software, known as blended attacks. Information security scholars have described the succession of blended attacks ushered in by CodeRed and Nimda as the "fourth wave of modern worms." The fourth wave followed the initial experimental wave of the 1980s, the second wave of polymorphic viruses and virus toolkits, and the third wave of mass e-mail viruses, such as Melissa, of the late 1990s.[12]

Blended threats are diverse, but they have two main characteristics in common, namely (i) exploitation of one or more security vulnerabilities, and (ii) multivector malevolent code with multiple destructive properties. The combination creates synergies that make blended threats significantly more hazardous than their predecessors.[13]

Blended attacks create complex liability issues. In a negligence action[14] involving a blended attack, the two most likely defendants are (i) the person responsible for the security vulnerability, and (ii) the person who programmed and distributed the virus or worm to exploit the vulnerability. The former, the original tortfeasor, may be a software designer or commercial vendor and as such, likely to be solvent and able to pay a tort judgment. The second tortfeasor, the hacker or virus distributor, on the other hand, is often judgment-proof.

---

[11] Chen and Robert, *The Evolution of Viruses and Worms*, at 10 ("Even if found, the worm [Nimda] was very difficult to remove because it makes numerous changes to Registry and System files. It creates an adminstrative share on the C drive, and creates a guest account in the administrator group allowing anyone to remote login as guest with a blank password.")

[12] The first wave consisted of the experimental viruses and worms of the time period, 1979 through 1990; the second wave introduced polymorphism and virus toolkits; the third wave, roughly spanning 1999 through late 2000 brought the mass e-mail viruses, such as Melissa. The wave of blended attacks introduced by CodeRed and Nimda is commonly described as the fourth wave of modern worms. See, e.g., Chen and Robert, *The Evolution of Viruses and Worms*.

[13] Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 366 ("Security exploits, commonly used by malicious hackers, are being combined with computer viruses, resulting in very complex attacks that sometimes go beyond the general scope of antivirus software."); and Id, at 542 ("[H]ighly infectious worms [such as CodeRed and Slammer] jump ... over the Internet using buffer overflow attacks on networked services. Because the files need not be created on disk and the code is injected into the address space of the vulnerable processes, even file integrity systems remain challenged by this particular type of attack.")

[14] Negligence is the most widely used theory of liability in the law of torts. See, e.g., James A. Henderson, *Why Negligence Law Dominates Tort*, 50 UCLA L. REV. 377 (2003). See, also, Gary T. Schwartz, *The Vitality of Negligence and the Ethics of Strict Liability*, 15 GA. L. REV. 963 (1981); Gary T. Schwartz, *The Beginning and the Possible End of Modern American Tort Law*, 26 GA. L. REV. 601 (1992).

The liability of an original tortfeasor is usually cut off by an intervening crime or intentional tort. Suppose, for instance, a technician negligently fails to fasten the wheels of plaintiff's car properly. A wheel comes off, leaving the plaintiff stranded on a busy highway. The stranded plaintiff is subsequently struck by a passing driver who failed to pay attention. The technician and the inattentive driver were both negligent and would both be held liable for the plaintiff's harm. The inattentive driver's inadvertent negligence would not cut off the liability of the mechanic. Suppose, alternatively, a passing driver is in a mood to kill someone, and seeing the stranded motorist as a convenient target, shoots him. In this scenario, the passing driver's intentional or criminal intervention would cut off the liability of the negligent mechanic, and shift liability exclusively to the second wrongdoer. Such a liability shift from a solvent original tortfeasor to a judgment-proof cyber attacker may leave a victim without recourse. The original tortfeasor's liability may be preserved, however, under the Encourage Free Radical (EFR) doctrine.

The EFR doctrine preserves the liability of an original tortfeasor who has encouraged individuals who are shielded from liability by anonymity, insufficient assets, lack of mental capacity, or lack of good judgment.[15] Such trouble-prone individuals are termed "free radicals," because of their tendency to bond with trouble. Examples of free radicals include children, anonymous crowds, criminals, terrorists, and mentally incompetent individuals.[16]

The EFR doctrine recognizes that the prospect of negligence liability is ineffective against defendants who are shielded from or otherwise undeterred by the prospect of liability. The deterrence rationale of negligence law would be defeated if responsible people who encourage free radicals were allowed to escape judgment by shifting liability to undeterrable free radicals. Common law negligence rules therefore impose liability on the first tortfeasor, the encourager of the free radicals, even when intentional or criminal behavior by a free radical intervenes.

---

[15] For a discussion of the effects of the judgment-proof problem on the deterrence and insurance goals of torts, see, e.g., Steven Shavell, *The Judgment Proof Problem*, 6 INT'L REV. LAW & ECON, 45 (1986); Henry Hansmann and Reinier Kraakman, *Toward Unlimited Shareholder Liability for Corporate Torts*, 100 YALE L J 1879, 1882-83; John G. Fleming, *Report to the Joint Committee of the California Legislature on Tort Liability of the Problems Associated with American Motorcycle Association v Superior Court*, 30 HASTINGS L J 1465, 1470 (1979); William R. Keeton and Evan Kwerel, *Externalities in Automobile Insurance and the Underinsured Driver Problem*, 27 J LAW & ECON 149, 149-50 (1984); John Summers, Comment, *The Case of the Disappearing Defendant: An Economic Analysis*, 132 U PA L REV, 145, 145 (1983).

[16] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 306-312 (2002).

The analysis in the article suggests that virus authors and distributors have properties commonly associated with free radicals. The analysis informs, furthermore, that the factors that influence courts in holding a defendant liable for encouraging free radicals are present in a typical blended attack. We conclude that liability will be preserved against a primary tortfeasor whose negligence was responsible for a vulnerability intentionally exploited by a free radical cyber attacker. The primary tortfeasor is likely a solvent commercial entity, while the attacker is often judgment-proof, or otherwise shielded from liability. The result is therefore significant, especially for the victim of a blended attack seeking to recover damages related to the attack.

Section 2 introduces the principles of malevolent software, blended attacks, and the (currently) most commonly exploited security vulnerability, the buffer overflow. Section 3 discusses liability issues in blended attacks. Section 4 analyzes the EFR doctrine in the context of blended attacks. A final section discusses and concludes.

2. Malevolent Software

Background

Malevolent software is a term for code that is intended to cause damage to or disrupt the operation of a computer system. The most common of these rogue programs is the computer virus, and its common variant, the worm. Other forms of malicious software include so-called logic bombs, Trojan horses, and trap doors.[17]

The term "virus," Latin for "poison," was first formally defined by Dr. Fred Cohen in 1983,[18] even though the concept goes back to John von Neumann's studies of self-replicating mathematical automata in the 1940s.[19] Dr. Cohen describes a computer virus as a series of instructions, in other words, a program, that (i) infects other computer programs and systems by attaching itself to a host program in the target system, (ii) executes when the host is executed, and (iii) spreads by cloning itself, or part of itself, and attaching the copies to other host programs on the system or network. In addition,

---

[17] See, e.g., DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 75-78.

[18] FRED COHEN, COMPUTER VIRUSES. PhD dissertation, University of Southern California (1985).

[19] Jeffrey O. Kephart et al., *Fighting Computer Viruses*, SCIENTIFIC AMERICAN, November 1997. Dr. Gregory Benford published the idea of a computer virus as "unwanted code." Benford apparently wrote actual "viral" code, capable of replication. DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 74.

many viruses have a so-called payload capable of harmful side-effects, such as data corruption.[20]

A worm[21] is a special type of virus. It is similar to a virus in most ways, except that it is self-replicating. A worm does not need to attach itself to a host program to replicate and spread. Like viruses, worms often carry payloads capable of destructive behavior, such as deleting files on the system through which it propagates. Worms without a destructive payload can nevertheless slow down a system significantly through the network traffic it generates with its prolific replication and spreading.[22]

The first worm was implemented by scientists at Xerox PARC, in 1978.[23] The so-called Morris Worm, created by Cornell University graduate student, Robert T. Morris, was the first worm to become a household name. The 1989 Morris worm used a security flaw in a UNIX program to invade and shut down much of the Internet. By some accounts, this event first woke the world up to the dangers of the computer vulnerability known as the buffer overflow.[24]

As the definition suggests, computer viruses consist of three basic modules or mechanisms, namely an infection mechanism, a payload trigger, and the payload. The infection mechanism allows the virus to replicate and spread, analogously to a biological virus. This is the most salient property of a computer virus.[25] The infection module first

---

[20] JOHN MACAFEE AND COLIN HAYNES, COMPUTER VIRUSES, WORMS, DATA DIDLERS, KILLER PROGRAMS, AND OTHER THREATS TO YOUR SYSTEM, at 26; FREDERICK B. COHEN, A SHORT COURSE ON COMPUTER VIRUSES (Wiley, 1994, 2d ed.), at 1-2.

In his PhD dissertation, Dr. Cohen defined a virus simply as any program capable of self-reproduction. This definition appears overly general. A literal interpretation of the definition would classify even programs such as compilers and editors as viral. DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 75.

[21] The Concise Oxford English Dictionary (Rev. 10th ed.) defines a worm as "[a] self-replicating program able to propagate itself across a network, typically having a detrimental effect."

[22] See, generally, John F. Schoch and Jon A. Hupp, *The "Worm" Programs - Early Experience with a Distributed Computation*, COMM. ACM, Vol 25, No 3, March 1982, 172.

[23] http://www.parc.xerox.com/about/history/default.html.

[24] Takanen et al., *Running Malicious Code By Buffer Overflows: A Survey of Publicly Available Exploits*, 162. EICAR 2000 Best Paper Proceedings. ("The day when the world finally acknowledged the risk entailed in overflow vulnerabilities and started coordinating a response to them was the day when the Internet Worm was introduced, spread and brought the Internet to its knees.") Available at http://www.papers.weburb.dk.

[25] LANCE J. HOFFMAN (ed.), ROGUE PROGRAMS: VIRUSES, WORMS, TROJAN HORSES (Van Nostrand Reinhold, 1990), at 247 ("The ability to propagate is essential to a virus program"); DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 73-75.

searches for an appropriate executable host program to infect. It then installs a copy of the virus into the host, provided the host has not yet been infected.

When the host program executes, the virus is also executed. Upon execution, the virus typically performs the following sequence of actions. It replicates ("clones") by copying itself to other executable programs on the computer.[26] During execution, the virus program also checks whether a triggering condition is satisfied. When the condition is satisfied, the virus executes its harmful component, the so-called payload module. Triggering events come in a variety of forms, such as a certain number of infections, Michelangelo's birthday, or the occurrence of a particular date. The Friday-the-13th virus, for instance, only activated its payload on dates with the cursed designation.[27] More recently, the first CodeRed worm alternated between continuing its infection cycle, remaining dormant, and attacking the official White House web page, depending on the day of the month.

Execution of the payload may produce harmful side-effects, such as destruction or corruption of data in spreadsheets, word processing documents and data bases, and theft of passwords.[28] Some effects are particularly pernicious because they are subtle and undetectable until substantial harm has been done. Subtle harmful viral effects include transposing numbers, moving decimal places, stealing passwords and other sensitive information.[29] Payloads are not necessarily destructive, and may involve no more than displaying a humorous message.[30] Some virus strains do not destroy or corrupt

---

[26] Potential target hosts include application and system programs, and the master boot record of the hard disks or floppy disks in the computer.

[27] See, e.g., Eric J. Sinrod and William P. Reilly, Cyber Crimes A Practical Approach to the Application of Federal Computer Crime Laws, 16 Santa Clara Computer & High Tech. L. J. 177 (2000), at 217, n. 176.

[28] JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE, (Ellis Horwood Ltd., 1990), at 17, 18. (In addition to self-replicating code, viruses often also contain a payload. The payload is capable of producing malicious side-effects.) See, also, FREDERICK B. COHEN, A SHORT COURSE ON COMPUTER VIRUSES (Wiley, 1994, 2d ed.), at 8-15 (examples of malignant viruses and what they do.); JOHN MACAFEE AND COLIN HAYNES, COMPUTER VIRUSES, WORMS, DATA DIDLERS, KILLER PROGRAMS, AND OTHER THREATS TO YOUR SYSTEM, at 61.

[29] JOHN MACAFEE AND COLIN HAYNES, COMPUTER VIRUSES, WORMS, DATA DIDLERS, KILLER PROGRAMS, AND OTHER THREATS TO YOUR SYSTEM, at 61.

[30] Sinrod & Reilly, at 218 (describing the W95.LoveSong.998 virus, designed to trigger a love song on a particular date.)

information, but consume valuable computing resources.[31] Viruses and worms used in blended attacks, however, are harmful by design.

A virus may infect a computer or a network through several possible points of entry, including via an infected file downloaded from the Internet, through web browsing, via an infected e-mail attachment, or even through infected commercial shrinkwrapped software.[32] Fast-spreading worms, such as CodeRed and Blaster, can only infect new hosts that contain one or more exploitable vulnerabilities.[33] The recent trend in virus transmission has been a decrease in infected diskettes, and an increase in infection through e-mail attachments. In a 1996 national survey, for instance, approximately 9 percent of respondents listed e-mail attachments as the means of infection of their most recent virus incident, while 71 percent put the blame on infected diskettes. In 2003, the corresponding numbers were 88 percent for e-mail attachments, and zero for diskettes.[34]

It was once believed that viruses could not be transmitted by data files, such as e-mail attachments. Viruses such as the infamous Melissa taught us otherwise. Melissa typically arrived in the e-mail inbox of its victim, disguised as an e-mail message with a

---

[31] Viruses can cause economic losses, e.g. by filling up available memory space, slowing down the execution of important programs, locking keyboards, adding messages to printer output, and effectively disabling a computer system by altering its boot sector. The Melissa virus, for instance, mailed copies of itself to everyone in the victim's e-mail address book, resulting in clogged e-mail servers and even system crashes. See, e.g., PHILIP FRITES, PETER JOHNSTON AND MARTIN KRATZ, THE COMPUTER VIRUS CRISIS (Van Nostrand Reinhold, New York, 2d ed., 1992), 23-4 ("The Christmas card [virus] stopped a major international mail system just by filling up all available storage capacity."); Sinrod & Reilly, at 218 (describing the Melissa virus.)

See Section 6 for an analysis of damages from computer virus infection. For examples of benign viruses and how they operate, see, e.g., FREDERICK B. COHEN, A SHORT COURSE ON COMPUTER VIRUSES (Wiley, 1994, 2d ed.), at 15-21.

[32] There are three mechanisms through which a virus can infect a program. A virus may attach itself to its host as a shell, an add-on, or as intrusive code. A shell virus forms a shell around the host code, so that the latter effectively becomes an internal subroutine of the virus. The host program is replaced by a functionally equivalent program that includes the virus. The virus executes first, and then allows the host code to begin executing. Boot program viruses are typically shell viruses. Most viruses are of the add-on variety. They become part of the host by appending their code to the host code, without altering the host code. The viral code alters the order of execution, by executing itself first and then the host code. Macro viruses are typically add-on viruses. Intrusive viruses, in contrast, overwrite some or all of the host code, replacing it with its own code. See, e.g., DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 81; PHILIP FRITES, PETER JOHNSTON AND MARTIN KRATZ, THE COMPUTER VIRUS CRISIS (Van Nostrand Reinhold, New York, 2d ed., 1992), at 73-75.

[33] Peter Szor, The Art of Computer Virus Research and Defense (2005), chapter 10 (extensive discussion of viruses that use exploits to spread themselves.) Blended attacks that depend on vulnerabilities to spread, are discussed in this article, section 2.

[34] ICSA Labs 9th Annual Computer Virus Prevalence Survey 2003, Table 10, p. 14.

Microsoft Word attachment. When the recipient opened the attachment, Melissa executed. First, it checked whether the recipient had the Microsoft Outlook e-mail program on its computer. If Outlook were present, Melissa would mail a copy of itself to the first fifty names in Outlook's address book, creating the appearance to the fifty new recipients that the infected person had sent them a personal e-mail message. Melissa would then repeat the process with each of the fifty recipients of the infected e-mail message (provided they had Outlook), by automatically transmitting clones of itself to fifty more people. A Melissa attack frequently escalated and resulted in clogged e-mail servers and system crashes.[35]

We now turn to a discussion of defenses against malevolent software.

Technical defenses against malevolent code

Anti-virus technology comes in two broad categories, namely "virus-specific" and "generic." Virus-specific technology, such as signature scanners, detect known viruses by indentifying patterns that are unique to each virus strain. These identifying patterns, known as signatures, are analogous to human fingerprints. Generic anti-virus technology, on the other hand, detects the presence of a virus by recognizing generic virus-like behavior, usually without identifying the particular strain.

A virus-specific scanner typically makes a specific announcement, such as, "the operating system is infected with the Cascade virus," while its generic counterpart may simply state, "the operating system is (or may be) infected with an (unidentified) virus." Virus-specific technology is more accurate with known strains and produces fewer false positives, but generic technology is better at detecting unknown viruses.

Technical anti-virus defenses come in four varieties, namely scanners, activity monitors, integrity checkers, and heuristic techniques.[36] Scanners are virus-specific, while activity monitors and integrity checkers are generic. Activity monitors look out for suspicious, virus-like activity in the computer. Integrity checkers sound an alarm when detecting suspicious modifications to computer files. Heuristic techniques combine virus-

---

[35] David Harley et al., VIRUSES REVEALED UNDERSTAND AND COUNTER MALICIOUS SOFTWARE (Osborne/McGraw-Hill, 2001), 406-410.

[36] See, e.g., DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 90-93; KEN DUNHAM, BIGELOW'S VIRUS TROUBLESHOOTING POCKET REFERENCE, (McGraw-Hill 2000), at 78-83 and 102-108.

specific scanning with generic detection, providing a significantly broadened range of detection.

Scanners are the most widely used anti-virus defense. A scanner reads executable files and searches for known virus patterns. These patterns, or "signatures," are the most reliable technical indicator of the presence of a file-resident virus in a computer system. A virus signature consists of patterns of hexadecimal digits embedded in the viral code, that are unique to the strain.[37] These signatures are created by human experts, such as researchers at IBM's High Integrity Computing Laboratory, who scrutinize viral code and extract sections of code with unusual patterns. The selected byte patterns then constitute the signature of the virus.[38] The scanner announces a match with its database of known viral signatures as a possible virus.

The virus signature pattern is selected to be a reliable indicator of the presence of a virus. An ideal virus signature gives neither false negatives nor false positives.[39] In other words, it should ideally always identify the virus when present and never give a false alarm when it is not.[40] The IBM High Integrity Computing Laboratory has developed an optimal statistical signature extraction technique that examines all sections of code in a virus, and selects the byte strings that minimize the incidence of false positives and negatives.[41]

Scanners are easy to use, but they are limited to detecting known virus signatures. A scanner's signature database has to be continually updated, a burdensome requirement in an environment where new viruses appear rapidly. Use of scanners is further

---

[37] JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE (Ellis Horwood, Ltd., 1990), at 42.

[38] JEFFREY O. KEPHART ET AL., *Automatic Extraction of Computer Virus Signatures*, Proceedings of the 4th Virus Bulletin International Conference, R. Ford, ed., Virus Bulletin Ltd., Abingdon, England, 1994, pp. 179-194, at 2.

[39] A false positive is an erroneous report of the activity or presence of a virus where there is none. A false negative is the failure to report the presence of a virus when a virus is in fact present.

[40] JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE (Ellis Horwood, Ltd., 1990), at 42. For short descriptions and hexadecimal patterns of selected known viruses, see HRUSKA at 43-52; JEFFREY O. KEPHART ET AL., *Blueprint for a Computer Immune System*, IBM Thomas J. Watson Research Center Report, at 11 ("[A] signature extractor must select a virus signature carefully to avoid both false negatives and false positives. That is, the signature must be found in every instance of the virus, and must almost never occur in uninfected programs.") False positives have reportedly triggered a lawsuit by a software vendor, who felt falsely accused, against an anti-virus software vendor. JEFFREY O. KEPHART ET AL., *Blueprint for a Computer Immune System*, IBM Thomas J. Watson Research Center Report, at 11.

[41] Jeffrey O. Kephart et al., Automatic Extraction of Computer Virus Signatures, Proceedings of the 4th Virus Bulletin International Conferenc e, R. Ford, ed., Virus Bulletin Ltd., Abingdon, England, 1994, pp. 179-194.

complicated by the occurrence of false positives. This occurs when a viral pattern in the database matches code that is in reality a harmless component of otherwise legitimate data. A short and simple signature pattern will be found too often in innocent software, and produce many false positives. Viruses with longer and more complex patterns will less often give a false positive, but at the expense of more false negatives.[42] Finally, as the number of known viruses grows, the scanning process will inevitably slow down as a larger set of possibilities has to be evaluated.[43]

Activity monitors are resident programs that monitor activities in the computer for behavior commonly associated with viruses. Suspicious activities include operations such as attempts to rewrite the boot sector, format a disk, mass mail multiple copies of itself, or modify parts of main memory. When suspicious activity is detected, the monitor may simply halt execution and issue a warning to alert the user, or take definite action to neutralize the activity.[44] Activity monitors, unlike scanners, do not need to know the signature of a virus to detect it. It works for all viruses, known as well as unknown. It's function is to recognize suspicious behavior, regardless of the identity of the culprit.

The greatest strength of activity monitors is their ability to detect unknown virus strains, but they also have significant weaknesses. They can only detect viruses that are actually being executed, possibly after substantial harm has been done. A virus may, furthermore, become activated before the monitor code, and escape detection until well after execution has begun. A virus may also be programmed to alter monitor code on machines that do not have protection against such modification. A further disadvantage of activity monitors is the lack of unambiguous and foolproof rules governing what constitutes "suspicious" activity. This may result in false alarms when legitimate activities resemble virus-like behavior. Recurrent false alarms may ultimately lead users to ignore warnings from the monitor. Conversely, not all "illegitimate" activity may be recognized as such, leading to false negatives.[45]

---

[42] KEN DUNHAM, BIGELOW'S VIRUS TROUBLESHOOTING POCKET REFERENCE, (McGraw-Hill 2000), at 78-83; Jeffrey O. Kephart et al., *Fighting Computer Viruses*, SCIENTIFIC AMERICAN, November 1997. See, also, Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++*, Technical report CSD-TR-92-062, Dept. of Computer Science, Indiana University, at 6-8.

[43] See, e.g., Pete Lindstrom, The Hidden Costs of Virus Protection, Spire Research Report, June 2003, at 5 ("In this day of 80,000+ known viruses and frequent discovery of new ones, the size of the signature file can be large, particularly if the updates are sent out as cumulative ones. Large updates can clog the network pipelines ... and reduce the frequency that an administrator will push them out to the end users.")

[44] Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++*, Technical report CSD-TR-92-062, Dept. of Computer Science, Indiana University, at 3-4.

[45] JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE, (Ellis Horwood Ltd., 1990), at 75.

Integrity checkers look for evidence of file tampering, such as "unauthorized" changes in system areas and files. The typical integrity checker is a program that generates a code, known as a "checksum," for files that are to be protected from viral infection. A file checksum may, for instance, be some arithmetic calculation based on the total number of bytes in the file, the numerical value of the file size and creation date. The checksum effectively operates as a "signature" of the file. These checkcodes are periodically recomputed and compared to the original checksum. Tampering with a file will change its checksum. Hence, if the recomputed values do not match the original checksum, the file has presumably been modified since the previous check, and a warning is issued. Since viruses modify and change the contents of the files they infect, a change in the checksum may be a sign of viral infection.[46]

The advantage of integrity checking is that it detects most instances of viral infection, as infection must alter the target file. The main drawback is that it tends to generate many false alarms, as a file can change for "legitimate" reasons unrelated to virus infection.[47] On some systems, for instance, files change whenever they are executed. A relatively large number of false alarms may trigger compliance lapses, as users may ignore warnings or simply not use the utility. Integrity checking works best on static files, such as system utilities, but is, of course, inadequate for files that naturally change frequently, such as Word documents.

A fourth category of virus detectors uses *heuristic* detection methods. Heuristic rules are rules that solve complex problems "fairly well" and "fairly quickly," but less than perfectly. Virus detection is an example of a complex problem that is amenable to heuristic solution. It has been proven mathematically that it is impossible to write a program that is capable of determining with 100 percent accuracy whether a particular program is infected with a virus, from the set of all possible viruses, known as well as unknown.[48] Heuristic virus detection methods accept such limitations and attempt to

---

[46] PHILIP FRITES, PETER JOHNSTON AND MARTIN KRATZ, THE COMPUTER VIRUS CRISIS (Van Nostrand Reinhold, New York, 2d ed., 1992), Figures 5.2-5.5, at 69-76; KEN DUNHAM, BIGELOW'S VIRUS TROUBLESHOOTING POCKET REFERENCE, (McGraw-Hill 2000), at 79. See, also, Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++*, Technical report CSD-TR-92-062, Dept. of Computer Science, Indiana University, at 5-6.

[47] PHILIP FRITES, PETER JOHNSTON AND MARTIN KRATZ, THE COMPUTER VIRUS CRISIS (Van Nostrand Reinhold, New York, 2d ed., 1992), at 125.

[48] Diomidis Spinellis, *Reliable Identification of Bounded-Length Viruses is NP-Complete*, IEEE TRANSACTIONS ON INFORMATION THEORY, 49(1), 280, 282 (January 2003) (Stating that theoretically perfect detection is in the general case undecidable, and for known viruses, NP-complete.); Carey Nachenberg, *Future Imperfect*, VIRUS BULLETIN, August 1997, 6. See, also, Francisco Fernandez,

achieve a solution, namely a detection rate that is "pretty good," albeit below the (unachievable) perfect rate.

Heuristic virus detection methods examine executable code and scrutinize its structure, logic and instructions for evidence of "virus-like" behavior. Based on this examination, the program makes an assessment of the likelihood that the scrutinized program is a virus, by tallying up a score. Instructions to send an e-mail message with an attachment to everyone in an address book, for instance, would add significantly to the score. Other high-scoring routines include capabilities to replicate, to hide from detection, and to execute some kind of payload. When a certain threshold score is reached, the code is classified as malevolent, and the user so notified.

The assessment is necessarily less than perfect and occasionally provides false positives and negatives. Many legitimate programs, including even some anti-virus programs, perform operations that resemble virus-like behavior.[49] Nevertheless, state-of-the-art heuristic scanners typically achieve a 70-80 percent success rate at detecting *unknown* viruses.[50]

A heuristic scanner typically operates in two phases. The scanning algorithm first narrows the search by, for instance, identifying the location most likely to contain a virus. It then analyzes the code from that location to determine its likely behavior upon execution. A static heuristic scanner typically compares the code from the "most likely" location to a database of byte sequences commonly associated with virus-like behavior.[51] The algorithm then decides whether to classify the code as viral.[52]

Heuristic Engines, Proceedings of the 11th International Virus Bulletin Conference, September 2001, Virus Bulletin Ltd., Abingdon, England, 1994, pp. 407-444; Chess & White, *Undetectable Computer Virus*, at http://www.research.ibm.com/antivirus/SciPapers/VB2000DC.htm.

[49] Francisco Fernandez, *Heuristic Engines*, Proceedings of the 11th International Virus Bulletin Conference, September 2001, Virus Bulletin Ltd., Abingdon, England, 1994, at 409 ("Many genuine programs use sequences of instructions that resemble those used by viruses. Programs that use low-level disk access methods, TSRs, encryption utilities, and even anti-virus packages can all, at times, carry out tasks that are performed by viruses.")

[50] Carey Nachenberg, *Future Imperfect*, VIRUS BULLETIN, August 1997, at 7.

[51] Certain byte sequences are, for instance, associated with decryption loops to unscramble a polymorphic virus when an infected routine is executed. If it finds a match, e.g. the scanner detects the presence of a decryption loop typical of a polymorphic virus, it catalogues this behavior.

[52] Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++*, Technical report CSD-TR-92-062, Dept. of Computer Science, Indiana University, at 4-5 ("Detection by static analysis/policy adherence.")

A dynamic heuristic scanner uses CPU emulation.[53] It typically loads suspect code into a virtual computer, emulates its execution and observes its behavior. Because it is only a virtual computer, virus-like behavior can safely be observed in what is essentially a laboratory setting, with no need to be concerned about real damage. The program is monitored for suspicious behavior while it runs.[54]

Although dynamic heuristics can be time-consuming due to the relatively slow CPU emulation process, they are sometimes superior to static heuristics. This will be the case when the suspect code (i) is obscure and not easily recognizable as viral in its static state, but (ii) clearly reveals its viral nature in a dynamic state.

A major advantage of heuristic scanning is its ability to detect viruses, including unknown strains, before they execute and cause damage. Other generic anti-virus technologies, such as behavior monitoring and integrity checking, can only detect and eliminate a virus after exhibition of suspicious behavior, usually after execution. Heuristic scanning is also capable of detecting novel and unknown virus strains, the signatures of which have not yet been catalogued. Such strains cannot be detected by conventional scanners, which only recognize known signatures. Heuristic scanners are capable of detecting even polymorphic viruses, a complex virus family which complicate detection by changing their signatures from infection to infection.[55]

The explosive growth in new virus strains has made reliable detection and identification of individual strains very costly, making heuristics more important and increasingly prevalent.[56] Commercial heuristic scanners include IBM's AntiVirus boot scanner and Symantec's Bloodhound technology.

Blended Attacks

---

[53] The CPU, or central processing unit, of a computer is responsible for data processing and computation. See, e.g., JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE, (Ellis Horwood Ltd., 1990), at 115; D. BENDER, COMPUTER LAW: EVIDENCE AND PROCEDURE (1982), §2.02, at 2-7, -9.

[54] Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++*, Technical report CSD-TR-92-062, Dept. of Computer Science, Indiana University, at 4.

[55] Polymorphic viruses have the ability to "mutate" by varying the code sequences written to target files. To detect such viruses requires a more complex algorithm than simple pattern matching. See, e.g., DOROTHY E. DENNING and PETER J. DENNING, INTERNET BESIEGED (ACM Press, New York, 1998), at 89.

[56] Carey Nachenberg, *Future Imperfect*, VIRUS BULLETIN, August 1997, at 9.

CodeRed and Nimda were the forerunners of a new wave of modern malevolent software, the blended attack.[57] Blended attacks are more sophisticated, complex, faster, and dangerous than their predecessors. They exploit computer security vulnerabilities, and often create new vulnerabilities, to enhance their destructiveness. The earlier generation of viruses, such as LoveLetter, Melissa and Michelangelo, in contrast, exploited only the regular functionality of the systems they targeted.

Blended threats are diverse, but they have two main characteristics in common, namely (i) exploitation of security vulnerabilities, and (ii) malevolent code with multiple destructive properties.

A. Blended threats exploit network vulnerabilities.

Blended threats are designed to take advantage of security vulnerabilities to gain access to and compromise a system.[58] The buffer overflow is currently (and has for over a decade been) the most commonly exploited vulnerability to get unauthorized access to a system.[59] A buffer overflow vulnerability allows executable malevolent code to be copied into memory of a target computer. A skilfull attacker can then exploit the vulnerability to manipulate the computer to remotely execute the malevolent code.[60]

Other security flaws, such as input validation vulnerabilities, are also frequently exploited by blended threats. A web page exhibits an input vulnerability, for instance, if it asks for user input, such as an e-mail address, without verifying that the user-provided address is in the proper form. Such a flaw may enable a hacker to manipulate the system by providing a specially formatted input. The uncensored input may cause the system to perform in a way that compromises its security.

---

[57] The wave of blended attacks introduced by CodeRed and Nimda is commonly described as the fourth wave of modern worms. The first wave consisted of the experimental viruses and worms of the time period, 1979 through 1990; the second wave introduced polymorphism and virus toolkits; the third wave, roughly spanning 1999 through late 2000 brought the mass e-mail viruses, such as Melissa. Chen and Robert, *The Evolution of Viruses and Worms*.

[58] *Symantec Internet Security Threat Report*, Volume III, February 2003, at 34, 35 (By exploiting IT vulnerabilities, blended threats are frequently able to bypass conventional security practices such as requiring strong, non-default passwords, as long as systems have the type of vulnerability exploited by the attack.)

[59] Eric Chien and Peter Szor, *Blended Attack Exploits, Vulnerabilities and Buffer-Overflow Techniques in Computer Viruses*. Symantec White Paper. Originally appeared in Virus Bulletin, 2002. The buffer overflow is discussed in the next subsection.

[60] See subsection, "Buffer overflow", infra, p. xxx.

Vendors are usually quick to issue patches to fix vulnerabilities as soon as they are discovered, but users tend to be slow in implementing them, and even if several vulnerabilities are patched, some may remain that can be exploited. By some estimates, even if 90 percent of the users of a particular technology with a newly discovered vulnerability could be trusted to implement the security patch issued by the vendor, the remaining unpatched systems could still allow enough hijackings to launch a denial of service attack on millions of other systems and networks.[61] Successive generations of CodeRed plagued the Internet despite the fact that each attack and the role played by the vulnerability were widely publicized, and that a security patch to fix the vulnerability had been made available even before the first CodeRed attack.

B. Blended threats employ malevolent software with multiple destructive properties.

Blended attacks employ viruses and worms with multiple destructive properties. The properties are usually not individually novel, but their combination in one virus or worm is unique. The payloads of blended threats are multidimensional and harmful by design.

1. Blended threats are harmful by design. Many conventional virus strains do little besides being a mild nuisance. The earlier Italian PingPong virus, for instance, merely displayed a bouncing ball, and the W95/LoveSong/998 virus was designed to trigger a love song on a particular date.[62] Blended threats, in contrast, are destructive by design. Blended threats carry a variety of payloads, including mechanisms capable of triggering DoS agents, deleting data files, and setting up backdoors in infected systems.[63] The CodeRed blended attack attempted to launch a full-scale denial of service attack on the official White House web page. The Slammer worm infected more than 90 percent of computers with a particular buffer overflow vulnerability withinh 10 minutes, and caused

---

[61] George V. Hulme, *One Step Ahead*, InformationWeek (May 20, 2002).

[62] Some earlier viruses were, of course, destructive. The 1987 South African Friday the 13th virus, for instance, was programmed to delete its host program, if invoked on Friday the 13th.

[63] *Blended Threats: Case Study and Countermeasures*. Symantec White Paper, at 2. See, also, George V. Hulme, *One Step Ahead*, InformationWeek (May 20, 2002) (The destruction capabilities of a blended threat include destruction of files, create backdoors, leave Trojan horses, and so-called zombie programs that can later be used to launch denial of service attacks.) See, also, eEye Digital Security, CodeRed Worm Analysis, August 2001. Available at http://www.eeye.com/Research/Advisories/AL20010804.html.

significant disruption to financial, transporation and government institutions, including widespread ATM failures, canceled airline flights and interference with elections.[64]

2. Blended threats propagate by multiple methods, attack from multiple points, and spread without human intervention. The typical blended threat attacks its target via multiple attack methods and attack points, which enable them to spread more rapidly and efficiently, and consume more computational resources and network bandwidth, and in a shorter time period. The Nimda worm, for instance, attacked via five vectors, including E-mail propagation using its own SMTP engine, and attack via backdoors left by worms such as CodeRed.[65]

Blended threats attack from multiple points, including injecting malicious code into executable files on a system and targeting and infecting visitors to compromised Web sites, often through innovative use of mass e-mail. Mass-mailing worms in blended attacks frequently bypass existing e-mail applications by using their own e-mail servers to spread. Such a worm could infect a computer with Microsoft Outlook, for instance, and spread via e-mail without using the Outlook application.[66]

Blended attacks do not require user intervention to trigger and spread, whereas traditional viruses depend on such intervention. Melissa, for instance, required users to actually open an e-mail attachment before the virus could execute and continue its infection cycle. Blended attacks exploit vulnerabilities that allow them to dispense with such interaction. A buffer overflow vulnerability in the e-mail servers Microsoft Outlook and Outlook Express, for instance, enabled an e-mail worm to spread automatically. The malicious code in the infected e-mail message could be executed merely by reading an HTML message, without opening an attachment. The recipient could therefore not protect herself by declining to open any attached files.

---

[64] David Moore et al., *Inside the Slammer Worm*, IEEE SECURITY AND PRIVACY, July/August 2003, 33.

[65] Nimda's other attack vectors were infection of Microsoft IIS web servers via a buffer overflow exploit; infection of network shares; and infection via Javascript added to web pages. See, e.g., Thomas Chen, *Trends in Viruses and Worms*, presentation at SMU Dept. of EE. Other attack points frequently used in blended attacks include injecting malicious code into .exe files on a target system, creating world readable network shares, making multiple registry changes, and adding script code to html files.

[66] Symantec Internet Security Threat Report, Volume III, February 2003, at 36. [Describing the operation of the typical mass-mailing worm: First, they exploit a known IT vulnerability to infect the system. Then, they collect e-mail addresses from the infected system. Finally, they spread via their own e-mail system, which is independent of the client e-mail system. This methodology enables the worm or virus to spread and propagate without user intervention. Users whose systems have been hijacked in this manner are often unaware that they are being used as launching pads for infected e-mails. In addition, these viruses frequently spoof the "From" address on e-mails, obscuring the origin of the infected e-mail.]

Blended threats are programmed to automatically search for and exploit new vulnerabilities. Such vulnerabilities are often found in new and emerging technologies, such as instant messaging technology, wireless local area networks, personal digital assistants, peer to peer networks, and networked cellular telephones.[67] Corporations and government departments and agencies, which rely increasingly on such vulnerable new technologies to conduct business, are particularly at risk. Many of these organizations and agencies are crucial elements of the national critical information infrastructure, including banking, transportation, communications and energy provision systems.

We now turn to a discussion of the buffer overflow, the most commonly exploited security vulnerability.

The Buffer Overflow

Buffers are data storage areas in memory with a limited capacity. Buffers often function as temporary storage for data to be transferred between two devices that are not operating at the same speed. The purpose of the temporary storage is to coordinate speed differentials between the adjacent devices. A printer, for instance, is not capable of printing data at the speed that it receives the data from the computer. A buffer in the interface between the computer and printer typically resolves this bottleneck. Instead of feeding the printer directly, the computer sends the data to the buffer. While the buffer relays the information to the printer, at the printer's speed, the computer is freed up to carry on with other tasks.[68]

A buffer overflow occurs when a program attempts to fill a buffer with more data than it was designed to hold. A buffer overflow is analogous to pouring ten ounces of water into a glass designed to hold eight ounces. The water must obviously overflow somewhere and create a mess. The glass represents a buffer and the water the application or user data.[69] The excess data typically overflow into adjacent memory locations where

---

[67] Grey, M., *Instant Messaging in the Enterprise Will Remain a Puzzle*. Gartner Research Report COM-18-7979 (22 November 2002). http://www.gartner.com; K. Dulaney and B. Clark, *E-Mail/PIM Is Still No. 1*. Gartner Research Report SPA-18-5839 (20 November 2002). http://www.gartner.com. ["The 'always on' nature of the connectivity, remote access to critical sensitive data, and the increasingly computational nature of mobile devices, set the stage for a potential virus or worm of significance."] For a discussion of vulnerabilities in instant messaging technology, see, e.g., http://securityresponse.symantec.com/avcenter/reference/secure.instant.messaging.pdf.

[68] William S. Davis, OPERATING SYSTEMS: A SYSTEMATIC VIEW, at 27, 28.

[69] Mark E. Donaldson, *Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention*, SANS Institute White Paper, at 3.

it can corrupt existing data, possibly changing the instructions, resulting in unintended executions.

The unintended executions could be harmless, but could also be malicious by design. In the most benign scenario, the buffer overflow will cause the program to abort, but without much further harm.[70] In a darker scenario, a buffer overflow could allow a hacker to remotely inject executable malicious code into the memory of a target computer, and execute it.

Suppose, for instance, the adjacent area ("overflow area") contained an instruction pointer, which defines the instruction to be executed next. By overwriting this pointer, the attacker can influence the program's next execution. The attacker may, for instance, fill the buffer with malicious code, such as a virus or worm, and overwrite the pointer with the address of the buffer. This would cause the execution path to change and cause the program to execute the viral code in the buffer.[71]

The most basic elements of a buffer overflow attack may be summarized as follows:

1. Data are copied into the buffer.
2. The data overflow the buffer.
3. The overflow data overwrite the original procedure return address.
4. The new return address now points to the new data in the buffer, which may be malevolent instructions.
5. These instructions trigger execution of the virus.

Schematically,[72]

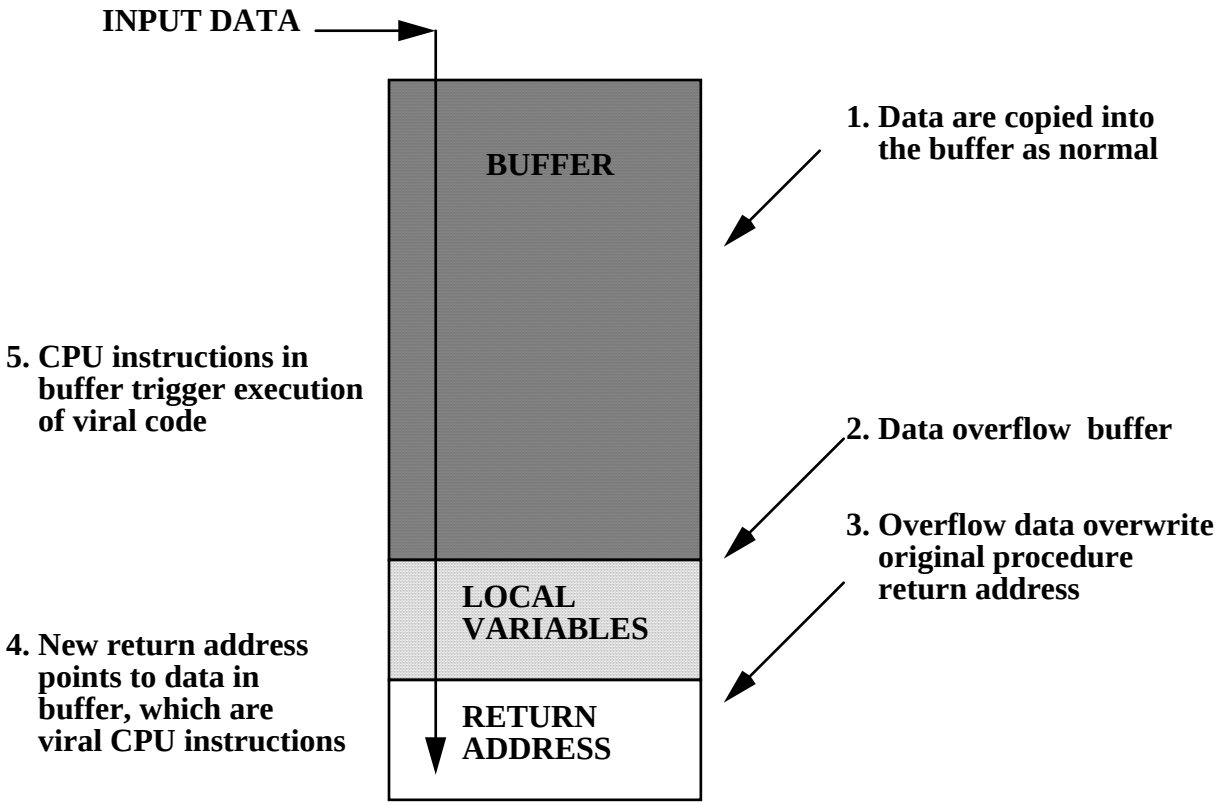
---

[70] The effect of a buffer overflow would be to abort the application program, resulting in a segmentation fault and terminating with a core dump.

[71] Microsoft Corporation defines a buffer overflow attack as follows:
"A buffer overflow attack is an attack in which a malicious user exploits an unchecked buffer in a program and overwrites the program code with their own data. If the program code is overwritten with new executable code, the effect is to change the program's operation as dictated by the attacker. If overwritten with other data, the likely effect is to cause the program to crash." Mark E. Donaldson, *Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention*, SANS Institute White Paper, at 3.

[72] The diagram is adapted from R. Enderle and J. Noel, *The New Approach to Windows Security*, at 7. Enderle Group White Paper (2004).

Basic Buffer Overflow Mechanism



In 1989, the so-called Morris Worm, created by Cornell University graduate student, Robert T. Morris, used a buffer overflow vulnerability in a UNIX program to invade and shut down much of the Internet. It was the first worm of its kind to become a household name, and, by some accounts, brought the destructive potential of the buffer overflow to the attention to the computer community.[73]

3. LIABILITY ISSUES IN BLENDED ATTACKS

3.1 Negligence concepts

[73] Takanen et al., *Running Malicious Code By Buffer Overflows: A Survey of Publicly Available Exploits*, 162. EICAR 2000 Best Paper Proceedings. ("The day when the world finally acknowledged the risk entailed in overflow vulnerabilities and started coordinating a response to them was the day when the Internet Worm was introduced, spread and brought the Internet to its knees.") Available at http://www.papers.weburb.dk.

Introduction

A civil action involving a blended attack would most likely be pursued under a negligence theory, the most widely used theory of liability in the law of torts.[74]

Negligence is generally defined as a breach of the duty not to impose an unreasonable risk on society.[75] It applies to any risk that can be characterized as unreasonable, including the risks associated with malevolent software. A victim of a blended attack may therefore bring legal action under a negligence theory against anyone who contributed to the risks associated with the attack, as well as those who failed in their duty to reduce or eliminate the risk.[76]

Blended threats are diverse, but they have two main characteristics in common, namely (i) exploitation of security vulnerabilities, and (ii) malevolent code with multiple destructive properties. This suggests the most likely defendants in a blended attack, namely (i) the original tortfeasor responsible for the security flaw, usually a solvent commercial vendor, and (ii) the virus distributor, the intervening party who programmed and distributed the virus or worm to exploit the vulnerability. The virus distributor is in practice often judgment-proof and shielded by the anonymity of cyberspace. The liability of the original tortfeasor is therefore likely of greater interest to a prospective plaintiff.

The plaintiff in a negligence action has to prove the following elements to establish her claim.

1. A legal *duty* on the part of the defendant not to expose the plaintiff to unreasonable risks.
2. A *breach* of the duty, namely a failure on the part of the defendant to conform to the norm of reasonableness.
3. A *causal* connection between defendant's conduct and plaintiff's harm. This element includes actual as well as proximate cause. Defendant's negligence is the actual cause

---

[74] See, e.g., James A. Henderson, *Why Negligence Law Dominates Tort*, 50 UCLA L. REV. 377 (2003). See, also, Gary T. Schwartz, *The Vitality of Negligence and the Ethics of Strict Liability*, 15 GA. L. REV. 963 (1981); Gary T. Schwartz, *The Beginning and the Possible End of Modern American Tort Law*, 26 GA. L. REV. 601 (1992).

[75] PROSSER AND KEETON ON THE LAW OF TORTS (5th ed., West Publ. Co., 1984), § 31. Second Restatement of Torts, § 282 (Describing negligence as conduct "which falls below the standard established by law for the protection of others against unreasonable risk of harm.")

[76] Dan B. Dobbs, The Law of Torts, at 258 (The plaintiff can assert that *any* conduct counts as negligence.)

of the plaintiff's harm if, but for the negligence, the harm would not have occurred. Proximate causation means that the defendant's conduct must be reasonably closely related to the plaintiff's harm.

4. Actual *damage* resulting from the defendant's negligence.

Generally, a duty exists (i) where someone sells a product; (ii) where someone has committed an affirmative act; (iii) when a special relationship exists; (iv) when a special kind of contract exists that benefits the plaintiff; and (v) where there is an undertaking by the defendant. Duty is also not an impediment to the plaintiff when a defendant has acted maliciously to destroy property.[77]

Courts require a plaintiff to prove breach of duty by identifying an untaken precaution that would have prevented the harm, and showing that the untaken precaution would have yielded greater benefits in accident reduction than its cost. The issue of breach in the context of a blended attack is discussed and analyzed in section 4.3 ("Encouragement of free radicals must be negligent.") The issue of damages in a virus context, including the economic loss rule, has been analyzed in related articles.[78]

We now turn to proximate causality, which is the most complex and interesting liability issue in blended attacks.

3.2 Proximate causality

Proximate cause applies to two broad categories of cases, namely those involving (i) multiple risks, and (ii) concurrent efficient causes.[79] A Multiple Risks case typically involves two risks, both of which would have been reduced by the defendant's untaken precaution. The first is the primary risk, which was clearly foreseeable to a reasonable person, and the second an ancillary risk, which would not have been reasonably foreseeable. Suppose, for instance, a surgeon performs a vasectomy negligently, and a child is born. The child grows up and sets fire to a house. The owner of the house sues the doctor for negligence. This is clearly a multiple risks case. The primary risk consists

---

[77] Mark F. Grady and A. Farnsworth, TORTS: CASES AND QUESTIONS (2004).

[78] Meiring de Villiers, *Computer Viruses and Civil Liability: A Conceptual Framework*, TORT AND INSURANCE PRACTICE LAW JOURNAL Fall 2004 (40:1); Meiring de Villiers,*Virus ex Machina Res Ipsa Loquitur*, 1 STANFORD TECH. L. REV., 2003.

[79] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 296 (2002) ("Proximate cause is a dualism.")

of foreseeable medical complications due to the incompetent vasectomy, including an unwanted pregnancy. The ancillary risk is the (unforeseeable) risk that the conceived child may grow up to be a criminal.[80] The proximate cause issue is whether the defendant should be held liable for the harm due to the ancillary risk.

A Concurrent Efficient Causes case involves multiple causes, all of which are actual causes of the same harm.[81] In a typical Concurrent Efficient Causes case an original wrongdoer and a subsequent intervening party are both responsible for the plaintiff's harm. Suppose, for instance, a technician negligently fails to fasten the wheels of plaintiff's car properly. A wheel comes off, leaving the plaintiff stranded on a busy highway. The stranded plaintiff is subsequently struck by a passing driver who failed to pay attention. The technician and the inattentive driver were both negligent and are concurrent efficient causes of the plaintiff's harm. The proximate cause issue is whether the second tortfeasor's intervening act should cut off the liability of the first. We now show that proximate cause is analyzed best when viewed as a dualism, consisting of two separate doctrines.

Proximate cause as a dualism

Proximate cause is a dualism consisting of two separate doctrines or tests. One doctrine applies to Multiple Risks cases, and the other to Concurrent Efficient Causes cases. Some accidents involve purely multiple risks, while others involve purely concurrent causes. In some cases, however, both doctrines apply. When both situations, Multiple Risks as well as Concurrent Efficient Causes, are present in the same case, both proximate cause doctrines apply and the requirements for both have to be satisfied for proximate cause to exist.[82]

The Reasonable Foresight doctrine applies to cases of multiple risks, where a primary and ancillary risk both caused the plaintiff's harm. This doctrine establishes the conditions under which the tortfeasor who created the primary risk will he liable for actual harm that has resulted from the ancillary risk. The bungled vasectomy is a typical Reasonable Foresight case. The Reasonable Foresight doctrine determines whether the

---

[80] Based on hypothetical in Dan B. Dobbs, The Law of Torts, p. 444.

[81] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 299 (2002).

[82] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 298 (2002).

surgeon would be held liable for damage caused by the ancillary risk, namely the risk that an unwanted pregnancy may produce a future criminal.

The Direct Consequences doctrine of proximate cause applies to cases involving multiple efficient causes. The doctrine examines concurrent causes to determine whether the person responsible for the second cause has cut off the liability of the person responsible for the first cause. The "loose wheel" case is a typical Direct Consequences case. The Direct Consequences doctrine would determine whether the intervening tortfeasor (the inattentive driver who struck the stranded plaintiff) would cut off the liability of the original tortfeasor (the negligent automobile technician.)

The Direct Consequences doctrine applies to blended attacks. A blended attack has two efficient causes, namely the security vulnerability and the virus distributor who exploited the vulnerability to launch the attack. The vulnerability and the intervening hacker are both essential to, and but-for causes of, the attack. Fast-spreading worms, such as CodeRed or Nimda, could not infect a system without an exploitable vulnerability. A system without the vulnerability or with an effective patch properly installed, would be immune to these worms.[83]

In the proximate cause analysis of a blended attack, the buffer overflow vulnerability is the original cause for which one of the defendants, the software designer, is responsible. Subsequently, an intervening defendant committed a second tort, namely transmitting a virus programmed to exploit the vulnerability. The second tort is a possible supervening tort which may cut off the liability of the first tortfeasor.

The direct consequences doctrine of proximate cause determines when the second concurrent efficient cause, the virus distributor, would cut off the liability of the person responsible for the first, the buffer overflow vulnerability. The liability of the virus distributor is not an issue, as she will always be liable, as long as the elements of duty, breach and actual causation are satisfied. However, a plaintiff would usually be more interested in suing the solvent original tortfeasor, rather than the judgment-proof hacker.

Analysis of the Direct Consequences doctrine is simplified if we break it down into five mutually exclusive paradigms. If a case falls clearly within one of the paradigms, its proximate cause analysis is normally straightforward.

Paradigms in Direct Consequences doctrine

---

[83] Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (2005), at 98.

Any direct consequences case belongs to one of five mutually exclusive paradigms, namely (i) No Intervening Tort, (ii) Encourage Free Radicals, (iii) Dependent Compliance Error, (iv) No Corrective Precaution, and (v) Independent Intervening Tort.[84]

The No Intervening Tort paradigm is the default paradigm. It preserves proximate cause if no tort by anyone else has intervened between the original defendant's negligence and the plaintiff's harm, as long as the type of harm was foreseeable. In this paradigm the original tortfeasor is not only the direct cause of the harm, but also the only wrongdoer. A speeding and unobservant driver who strikes a pedestrian walking carefully in a crosswalk is a clear example of a case within the No Intervening Tort paradigm. The original wrongdoer is clearly liable under this paradigm, and is also the only wrongdoer. A blended attack does not fit into this paradigm because of the intervening tort of a second wrongdoer, the cyber attacker.

Under the Encourage Free Radicals (EFR) paradigm, proximate cause is preserved if the defendant's wrongdoing created a tempting opportunity for free radicals. Proximate cause is preserved under the Dependent Compliance Error (DCE) paradigm if the defendant's wrongdoing has increased the likelihood that the victim will be harmed by someone else's *inadvertent* negligence. A blended attack would not fall into the DCE paradigm if the second wrongdoer acted *intentionally*.

Proximate cause is broken under the No Corrective Precaution paradigm if a third party with an opportunity and duty to prevent the plaintiff's harm, intentionally fails to do so. If, for instance, the plaintiff intentionally fails to take a corrective precaution that would have prevented the harm, such failure would cut off the original tortfeasor's liability.

As the name suggests, the Independent Intervening Tort paradigm cuts off the original tortfeasor's liability if an independent intervening tort caused the plaintiff's harm. Under this paradigm the original tortfeasor's liability will be cut off if the relation between the original tortfeasor's negligence and the second defendant's subsequent negligence is coincidental.

The victim of a blended attack, as plaintiff in a negligence action, would be interested in preserving the liability of a solvent original tortfeasor. There are three direct consequences paradigms which, if applicable, may preserve the liability of the original tortfeasor. The paradigms are the No Intervening Tort (NIT), Dependent Compliance Error (DCE) and EFR paradigms. The NIT and DCE paradigms do not apply to the

---

[84] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 301-321 (2002).

typical blended attack case, which leaves the EFR doctrine. The EFR doctrine would preserve the liability of an original tortfeasor if she encouraged free radicals, and if the factors that influence courts in holding a defendant liable for encouraging free radicals are present in the case. If, however, the second tortfeasor is a responsible person who deliberately omitted a reasonable precaution or committed an intentional tort or crime, the original tortfeasor's liability will be cut off.

The Encourage Free Radicals paradigm, to which we now turn, is therefore the most relevant paradigm in the liability analysis of a blended cyber attack.

3.3 The Encourage Free Radicals Doctrine

Introduction

Courts hold rational and "irrational" defendants equally liable for their torts. Actors with a severe mental illness, for instance, are not exempted from liability. Mentally incompetent people are held to the standard of normal people, even though they could never achieve it. In *Polamtier v Russ*,[85] for instance, a legally insane paranoid schizophrenic defendant was held liable for shooting his father-in-law. The court reasoned that, in spite of his mental illness, he could nevertheless form the intent to commit his unlawful act.[86]

We observe the same pattern in negligence law. People with mental illnesses are held to the negligence standards of normal people. In *Breuning v American Family Insurance Co.*,[87] a person started experiencing delusions, but continued driving her car and caused an accident. The court reasoned that a reasonable person should have seen the delusions as a danger signal, and that continuing to drive therefore constituted negligence.

The courts do distinguish between rational and irrational actors when they are encouraged by a rational defendant. Courts hold a rational defendant liable for encouraging or provoking an irrational person, but cuts off the encourager's liability

---

[85] 537 A.2d 468 (Ct. 1988).

[86] Mark F. Grady, *The Free Radicals of Tort*. See, also, *McGuire v Almy*, 8 N.E.2d 760 (Mass. 1937) ("[W]here an insane person by his act does intentional damage to the person or property of another he is liable for that damage in the same cirucmstances in which a normal person would be liable."); *Ellis v D'Angelo*, 253 P.2d 675 (1953) (Child held liable on grounds that he could predict the consequences of his tortious act.)

[87] 173 N.W.2d 619 (Wis. 1970).

when the provoked actor is rational. The rationale for this distinction appears to be rooted in the deterrence and insurance goals of tort law.

Negligence law is the most basic form of safety regulation, but it is an ineffective deterrent against defendants who are shielded from liability by anonymity, insufficient assets, lack of mental capacity or lack of good judgment. Such trouble-prone individuals are termed "free radicals," because of their tendency to bond with trouble. Examples of free radicals include children, anonymous crowds, criminals, mentally incompetent individuals, and in the cyber realm, hackers and cyber rogues, such as computer virus authors and distributors.[88]

Free radicals are not deterred by the threat of tort liability. Judgment-proof free radicals have insufficient assets to pay for the harms they cause, while other free radicals simply lack the good judgment or mental capacity to to care about the consequences of their actions.[89] Terrorists may be blinded to the threat of liability by ideological or religious motivations. The deterrence rationale of negligence law would therefore be defeated if responsible people who foreseeably encourage free radicals to be negligent were allowed to escape judgment by shifting liability to the latter. Common law negligence rules have responded to this policy dilemma with the Encourage Free Radicals (EFR) doctrine. The EFR doctrine imposes liability on the encourager, even when intentional or criminal behavior by a free radical intervenes.[90]

Satcher v James H. Drew Shows, Inc.[91] illustrates the Free Radicals paradigm. In *Satcher*, the plaintiff bought a ticket for a ride on the bumper cars in an amusement park. A group of mental patients on an excursion joined the plaintiff's group. When the ride started, the patients converged on the defendant and repeatedly crashed into her from all angles, injuring her neck permanently. The plaintiff filed suit, alleging that the defendant owner and operator of the ride had been negligent in allowing the patients to target and

---

[88] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 306-312 (2002).

[89] The effects of the judgment-proof problem on the deterrence and insurance goals of torts are discussed in the following references: Steven Shavell, *The Judgment Proof Problem*, 6 INT'L REV. LAW & ECON, 45 (1986); Henry Hansmann and Reinier Kraakman, *Toward Unlimited Shareholder Liability for Corporate Torts*, 100 Yale L J 1879, 1882-83; John G. Fleming, Report to the Joint Committee of the California Legislature on Tort Liability of the Problems Associated with American Motorcycle Association v Superior Court, 30 Hastings L J 1465, 1470 (1979); William R. Keeton and Evan Kwerel, *Externalities in Automobile Insurance and the Underinsured Driver Problem*, 27 J Law & Econ 149, 149-50 (1984); John Summers, Comment, *The Case of the Disappearing Defendant: An Economic Analysis*, 132 U Pa L Rev, 145, 145 (1983).

[90] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 308 (2002).

[91] 177 S.E.2d 846 (Ga. Ct. App. 1970).

injure her. The appellate court reversed the trial court's decision for the defendant, on the grounds that the defendant had encouraged free radicals.

If the plaintiff had sued the mental patients she likely would have won. Their mental illness would not have been a defense, although they may not have had the assets to pay a judgment. Their mental illness is a critical factor in the liability of the owner of the defendants. If the bumper car drivers were rational individuals, instead of free radicals, the defendant would probably not have been held liable. Tort law focuses liability on responsible people, which is where its policy goals will be best promoted.

Another free radicals case is presented by *Weirum v RKO General, Inc*.[92] The defendant radio station broadcast a contest in which a disk jockey would drive throughout Los Angeles. He would stop occasionally and announce his location on the radio. Teenagers would race to meet the disk jockey and he would give a prize to the first one who reached him. Eventually, two overeager racing teenagers were involved in a road accident, killing the plaintiff's deceased. There were two concurrent efficient causes of the accident, namely the organizers of the contest and the reckless teenage drivers. The radio station negligently encouraged the free radical teenagers to drive recklessly. The wrongdoing of the teenagers did therefore not cut off the defendant radio station's liability. The radio station was held jointly liable with the teens and, as the deeper pocket, likely paid most of the damages.

Historical review of the EFR doctrine

The EFR doctrine is not a modern development, but has a long history. The doctrine developed a critical mass throughout the nineteenth century, as did negligence cases generally.

One of the earliest cases in which a court applied the EFR doctrine is the 1773 English case, *Scott v Shepherd*.[93] The defendant threw a lighted squib, made of gunpowder, into a crowded marketplace. The squib was picked up and thrown away successively by several people, until it landed elsewhere in the market where it exploded and injured the plaintiff. The verdict turned on whether the harm was direct (trespass *vi et armis*) or consequential (trespass on the case.)

---

[92] 539 P.2d 36 (Cal. 1975).

[93] 96 Eng. Rep. 525 (K.B. 1773).

If the original throwers had acted out of self-defense or necessity, the harm would be considered to have been a direct consequence of the defendant's first throw of the squib. If the intermediate throwers, on the other hand, had acted to "continue the sport" as true free radicals would, then the harm would be consequential or indirect. Justice Blackstone argued for the free radical interpretation, but was outvoted by his colleagues. The court held for the plaintiff, because he had decided to plead trespass *vi et armis*.

The English case, *Dixon v Bell*,[94] may have been the original EFR case.[95] The defendant sent his thirteen-year old servant to fetch a loaded gun he had kept in his apartment. Assuming the gun was unloaded, the servant playfully pointed it at the plaintiff's son and pulled the trigger. The gun went off, injuring the boy. The plaintiff's declaration based its claim of liability on the allegation that the defendant had encouraged a free radical. In particular, the allegation claimed that the defendant had wrongfully sent a juvenile servant to fetch a loaded gun, fully aware that it was inappropriate and dangerous.

Lynch v Nurdin[96] was decided in 1841, in the full swing of the Industrial Revolution. The defendant had left his horse and cart unattended on a street that was usually thronged. On this day, the street was even busier than usual. The defendant knew that groups of children would be coming down the street and that they would be interested in his horse and cart. The plaintiff, a young child, was injured when another boy, who was playing on the cart, caused it to move and run across the plaintiff's leg. The Queen's Bench held the defendant liable for providing an opportunity and encouragement to free radicals.[97]

Guille v Swan[98] was possibly the original EFR case in the United States. In *Guille*, the defendant descended in a balloon over New York City into plaintiff's garden in a manner that attracted a crowd. The defendant's balloon dragged over the plaintiff's garden, but the crowd did much more damage to the garden. The defendant argued that he should be responsible only for his share of the damages, and not for that caused by the

---

[94] 105 Eng. Rep. 1023 (K.B. 1816).

[95] Mark F. Grady, *The Free Radicals of Tort*, at 113 ("[I]n 1816, the English Court of King's Bench had already decided the first indisputable EFR case, which was Dixon v Bell."]

[96] 113 Eng. Rep. 1041 (Q.B. 1841).

[97] See, also, *Lane v Atlantic Works*, 111 Mass. 136 (1872) (Defendant held liable for encouraging free radicals when carelessly laid iron bars fell off when jostled by a child, and injured the plaintiff.)

[98] 19 Johns. 381 (N.Y. 1822).

crowd, but the court held him responsible for all the damages. The crowd were free radicals in that particular situation. People who are otherwise perfectly rational may behave differently when they are shielded by the anonymity and diminished accountability of a crowd. Chief Justice Spencer stated that the defendant's manner of descent would foreseeably draw a crowd with predictable consequences, for which he should be held responsible, a classic description of the EFR doctrine.[99]

4. FREE RADICALS, THE BUFFER OVERFLOW AND BLENDED ATTACKS

4.1 Introduction

In a negligence action, liability of an original tortfeasor for encouraging a second tortfeasor will be preserved under the EFR doctrine, if (i) the second tortfeasor is in fact a free radical, and (ii) the case exhibits the factors that influence courts in holding a defendant liable for encouraging free radicals. We now turn our analysis to these two issues in the context of a blended attack.

The EFR doctrine only applies when a free radical is involved. If the encouraged person is not a free radical, and if the defendant's encouragement is insufficient to make him a co-actor with the immediate wrongdoer, the defendant is immune to liability. A defendant would, for instance, not be held liable for encouraging a responsible citizen. If Bill Gates had responded to the *Weirum* radio broadcast by racing to collect the prize, his intervening conduct would almost certainly have cut off the radio station's liability.[100] Likewise, in the unlikely event that Bill Gates would use a virus kit to create a virus that exploits a security flaw in Windows, the creator of the kit would escape liability. If, however, a free radical, such as a judgment-proof hacker did the same, proximate causality would likely not be broken by the hacker's intervention.

Seith v Commonwealth Electric Co.[101] presents a case where a non-free radical intervened, cutting off the liability of the defendant. In *Seith*, because of negligent maintenance, a live electric wire broke and fell on a sidewalk. Two police officers came to investigate and one of them flipped the wire with his club towards the plaintiff, a bystander. The plaintiff caught it reflexively, and suffere a severe electric shock. The trial

---

[99] Mark F. Grady, *The Free Radicals of Tort*, at 113.

[100] Weirum v RKO General, Inc, 539 P.2d 36 (Cal. 1975).

[101] 89 N.E. 425 (Ill. 1909).

court found for the plaintiff, but the Illinois Supreme Court reversed on the grounds that the police officer, as a model of propriety and responsibility, was not a free radical. If a free radical, such as a child or mentally incompetent person had flipped the wire to the plaintiff, the defendant would likely have been held liable.[102]

The second tortfeasors in a blended attack are typically virus authors and distributors who have exploited a security vulnerability to launch a cyber attack. The liability of the original tortfeasor will be preserved if the exploiters of the vulnerability are free radicals. We now turn to an analysis of virus authors and distributors as free radicals.

4.2 Virus authors and distributors as free radicals

Virus authors and distributors have properties commonly associated with free radicals. They are often judgment-proof and shielded by the anonymity of cyberspace. Furthermore, virus attacks are under-reported, under-prosecuted and the probability of catching a hacker or virus author is comparatively low. Virus authors appear undeterred by the threat of legal liability and often seem unconcerned about the problems caused by their creations. Most virus authors would either be unaffected or, perversely, actually encouraged by stricter anti-virus legislation. All these factors are consistent with a free radical profile.[103]

Anonymity

---

[102] *Travell v Bannerman*, 75 N.Y.S. 866 (App. Div. 1902) presents an analogous situation, where the harm was caused by an intervening free radical. (Children played with discarded explosive material, causing injury.)

[103] Paul A. Strassman and William Marlow, *Risk-Free Access Into the Global Information Infrastructure Via Anonymous Re-Mailers*, Symposium on the Global Information Infrastructure: Information, Policy & Information Infrastructure, Cambridge, MA, January 28-30, 1996 ("Information terrorism ... is a unique phenomenon in the history of warfare and crime. For the last two hundred years the theory of warfare has been guided by 'force-exchange' equations in which the outcome was determined by the rate of attrition of each opposing force. In information attacks these equations do not apply because the attacker remains hidden and cannot be retaliated against. Since Biblical times, crimes have been deterred by the prospects of punishment. For that, the criminal had to be apprehended. Yet information crimes have the unique characteristic that apprehension is impossible, since even identification of the criminal is not feasible. Information crimes can be committed easily without leaving any telltale evidence such as fingerprints, traces of poison or bullets.")

The Internet provides users with a degree of anonymity which has emboldened cybercriminals to commit crimes they would not otherwise consider.[104] The anonymity of cyberspace complicates the task of detecting computer crimes and tracking down offenders. It also makes it harder to obtain evidence against a wrongdoer such as a virus author or distributor.[105] Cyberspace provides the technology and opportunity to a skilled operator to assume different identities, erase his digital footprints, and transfer incriminating evidence electronically to innocent computers, often without leaving a trace.[106]

Suppose, for instance, a virus were transmitted from the e-mail account of someone named Jill Smith, and a copy of an identical virus were tracked down in the same account. This evidence may look like the proverbial smoking gun, but would likely not prove by a preponderance that the owner of the account is the actual culprit. Someone may have hacked into the Smith account, used it to launch a virus and stored incriminating files in the account.[107] Perpetrators of denial of service (DoS) attacks employ similar tactics to hide their identities. The most common form of distributed

---

[104] Mark D. Rasch, *Criminal Law and the Internet*, Chapter 11 in The Internet and Business: A Lawyer's Guide to the Emerging Legal Issues. Available on-line at http://www.cla.org/RuhBook/chp11.htm. (The anonymity of cyberspace encourages network users to commit offenses that they would not otherwise attempt. This is exacerbated by the fact that the bounds of acceptable behavior is not yet clearly defined in cyberspace.)

[105] Sarah Gordon, *Virus Writers: The End of Innocence* ("[T]racing a virus author is extremely difficult if the virus writer takes adequate precautions against a possible investigation."); Ian C. Ballon, *Alternative Corporate Responses to Internet Data Theft*, 471 PLI/Pat. 737, 739 (1997); M. Calkins, *They Shoot Trojan Horses, Don't They? An Economic Analysis of Anti-Hacking Regulatory Models*, 89 GEO. L.J. 171, November 2000; Jelena Mirkovic et al, INTERNET DENIAL OF SERVICE: ATTACK AND DEFENSE MECHANISMS (2005), 14 ([V]ery few attackers have been caught and prosecuted. ... [One] factor is the ease of performing a DoS attack without leaving many traces for investigators to follow. ... Another type of DoS perpetrator is a sophisticated hacker who uses several means to obscure her identity and create subtle variations in traffic patterns to bypass defenses.")

[106] See, e.g., Ted Bridis, *Microsoft Offers Huge Cash Rewards for Catching Virus Writers*, at http://www.securityfocus.com/news/7371. ("Police around the world have been frustrated in their efforts to trace some of the most damaging attacks across the Internet. Hackers easily can erase their digital footprints, crisscross electronic borders and falsify trails to point at innocent computers.")

[107] M.D. Rasch, Criminal Law and the Internet, Chapter 11 in The Internet and Business: A Lawyer's Guide to the Emerging Legal Issues (Computer Law Association). On-line version available at http://www.cla.org/RuhBook/chp11.htm. See, also, BizReport News, September 12, 2003 ("There are many ways for virus writers to disguise themselves, including spreading the programs through unwittingly infected e-mail accounts. The anonymity of the Internet allows you to use any vulnerable machine to launder your identity.") Report available at http://www.bizreport.com/print.php?art_id=4917.

denial of service attack[108] consists of flooding a network with bogus information packets, thereby preventing legitimate network traffic. The source addresses of this illegitimate network traffic is usually spoofed to hide the true origin of the attack, making it difficult to identify the attacker. This is especially true with distributed attacks.[109]

As the number of machines connected to the Internet increases, the ability of hackers to elude detection is enhanced. Subverting multiple machines makes it difficult to trace the source of an attack. An attacker can take a circuitous route and hide his tracks in the adulterated log files of multiple machines, which would reduce the likelihood of detection and allow the attacker to remain hidden from law enforcement.[110]

The anonymity of cyberspace has contributed to virus authors' graduation from cyber-vandalism to organized crime. Virus writers are increasingly cooperating with spammers and hackers to create viruses to hack into computers to steal confidential information, often hiding their identity by spoofing the identity of the legitimate owner. Spammers are using viruses, for instance, to mass distribute junk mail, by sending out viruses to take over computers and email accounts and using them to mass-distribute spam messages.[111] The owner of the hijacked computer usually does not know it has been hijacked, although there are often subtle indications, such as slower Internet connection.[112]

---

[108] A denial-of-service attack (also, DoS attack) is an attack on a computer system or network that causes a loss of service to users, typically the loss of network connectivity and services by consuming the bandwidth of the victim network or overloading the computational resources of the victim system.

[109] See, e.g., Rik Farrow, *Distributed Denial of Service Attacks (DDoS)* ("In an ordinary network-based denial of service attack, an attacker uses a tool to send packets to the target system. These packets are designed to disable or overwhelm the target system, often forcing a reboot. Often the source address of these packets is spoofed, making it difficult to locate the real source of the attack.")

[110] L. Jean Camp and Catherine Wolfram, *Pricing Security*, in L. Jean Camp and Stephen Lewis (eds.), ECONOMICS OF INFORMATION SECURITY (2004).

[111] The virus named "Sobig F," for instance, is programmed to turn a computer into a host which sends out spam e-mail messages, often without the knowledge of the owner. It is widely believed that half a million copies of the virus named AVF were sent by a spammer. Unlike Melissa, the AVF virus does not mail copies of itself out to everyone in the infected computer's address book. Instead, AVF makes the infected computer an intermediary, by opening a backdoor in the infected machine through which spammers can distribute their junk mail.

[112] *Spam Virus Hijacks Computers*, BBC News, at http://news.bbc.co.uk/1/hi/technology/3172967.stm; Jo Twist, *Why People Write Computer Viruses*, BBC News. http://news.bbc.co.uk/1/hi/technology/3172967.stm. See, also, Chen and Robert, *The Evolution of Viruses and Worms*, at 14 ("[A] sociological reason [for continued pervasiveness of worms and viruses] is the lack of accountability for worm writers. The long-held general perception has been that worms and viruses are low risk crimes. It is notoriously difficult to trace a virus or worm to its creator from analysis of the code, unless there are inadvertent clues left in the code.")

Role of e-mail

E-mail plays a prominent role in computer security. E-mail is currently the most widely used mechanism for virus transmission,[113] as well as a prime means to install backdoors and other malicious programs in target systems.[114] E-mail is a popular mechanism for transmitting viruses embedded in Word macros (such as Melissa), infected attachments (such as Love Bug), and viruses embedded in HTML mail. Technology that enables anonymous e-mail transmission would therefore be a significant tool in the hands of cyber rogues.

E-mail anonymity is substantially enhanced by the use of anonymous remailers. Remailers are servers which forward electronic mail to network addresses on behalf of an original sender who wishes to remain anonymous. An e-mail message usually carries a header with information about its starting point, its destination and some information about the route it has taken. This information makes the true source of the message traceable. The purpose of a remailer service is to disguise the true source by delivering an e-mail message without its original header and with a fictitious return address. This ensures almost total anonymity for the original sender.[115]

The remailer typically receives a message from A, intended to be transmitted to B. The remailer then transmits the message to B, but in such a way that the true source

---

[113] ICSA Labs 9th Annual Computer Virus Prevalence Survey 2003, at 15. (Showing e-mail attachments, as a source of infection, increasing from 9 percent in 1996, to 88 percent in 2003. In contrast, infection by diskette has decreased from 70 percent in 1996, to virtually zero in 2003.)

[114] See, e.g., *Why Anti-Virus Software is Not Enough: The Urgent Need for Server-Based Email Content Checking* (April 23, 2003). Available at http://www.secinf.net/anti_virus/
See, also, Lynne Munro, *Protecting Your Computer from e-Mail viruses and Worms*, Oxford University Computing Services White Paper (November 2001, revised January 2004), at 1 ("currently you are more likely to infect your computer with a virus/worm by reading an infected e-mail message than by any other route.") Available at http://www.oucs.ox.ac.uk/viruses/avdocs/emailvirs/index.xml?style=text.

[115] *Spammers and Viruses Unite*, BBC News, at http://news.bbc.co.uk/1/hi/technology/2988209.stm. (Describing the hijacking program named Proxy-Guzu, which would typically arrive as a spam message with an attachment. Opening the attachment triggers it to forward information about the hijacked account to a Hotmail account. This information then enables a would-be spammer to route mail through the hijacked computer. The source of this spam would be very hard if not impossible to trace, especially if the spammer and the sender of the hijacking program employed anonymity-preserving techniques, such as a remailer.)

See, also, Jay Lyman, Authorities Investigate Romanian Virus Writer, at http://www.linuxinsider.com/perl/story/31500.html, referring to "the difficulty of tracking down virus writers, particularly when they are skilled enough to cover their digital tracks, [so that] few offenders are ever caught."; Noah Levine, *Note: Establishing Legal Accountability for Anonymous Communication in Cyberspace*, 96 COLUM. L. REV. 1526, Section I.A.

(A) is obfuscated.[116] Some remailer services enable the recipient to reply to the true source, but without revealing the identity of the source. A virus may be circulated anonymously in this manner, by remailing an e-mail message with an attachment containing the virus.[117]

Remailers come in different varieties and levels of anonymity.[118] Some remailers maintain an internal list of the true identities of their clients. Any client of the remailer is in principle identifiable by someone with access to the internal master list. The former anonymous remailer, penet.fi, operated in this way, which ultimately led to its demise when a court ordered that the client list be made available to a plaintiff in a lawsuit. Pseudonymous remailers, generally termed nym servers, use cryptography to provide the same service but with a greater degree of confidentiality.

The purpose of keeping a list of client identities is to facilitate two-way interaction. When the remailer receives a message intended for one of its clients, the remailer consults the list and forwards the message to the client. If e-mail users are willing to forego two-way interaction, such a master list is no longer necessary and greater confidentiality can be achieved. When a message is remailed anonymously under such an arrangement, it leaves no information behind that can be used to trace it to the original sender. A determined sender can use "chained remailing" as an additional line of defense, namely a combination of anonymous remailers and encryption techniques, to make it virtually impossible to trace her communications.

Remailers fulfill an important function in attacks that rely on e-mail as propagation mechanism, as is often the case in blended attacks. A buffer overflow vulnerability in the e-mail servers Microsoft Outlook and Outlook Express, for instance, enabled an attacker to invade a target computer by sending an infected e-mail message. The malicious code could be executed merely by reading the transmitted HTML message, without opening an attachment. As soon as the recipient downloaded the

---

[116] A remailed message may still be traceable if, for instance, the anonymous remailer administrators keep a log of the identities of their clients. Not all remailers keep such information, though. See, e.g., Noah Levine, supra, at nn. 24-28.

[117] See, e.g., Noah Levine, *Note: Establishing Legal Accountability for Anonymous Communication in Cyberspace*, 96 COLUM. L. REV. 1526, at n. 50.

[118] See, e.g., *Anonymous Remailer*, at http://en.wikipedia.org/wiki/Anonymous_remailer ("An anonymous remailer is a server computer which receives messages with embedded instructions on where to send them next, and which forwards them without revealing where they originally came from. There are Cypherpunk anonymous remailers, Mixmaster anonymous remailers, and nym servers, among others, which differ in how they work, in the policies they adopt, and in the type of attack on anonymity of email they can (are intended to) resist.")

infected message from the server, Outlook would crash and the viral code activated. The infected e-mail message would then be sent to all contacts in the address book of the victim. The process would repeat itself, repeatedly causing e-mail clients to crash, and occasionally ending up paralyzing Internet traffic.[119]

Anonymous e-mail would protect and encourage the perpetrators of this exploit and countless others. The anonymity provided by remailing drastically reduces accountability and deterrence on the Internet, and is increasingly a hindrance to law enforcement efforts.[120]

Lack of deterrability

Perpetrators of virus attacks appear to be undeterred by the threat of legal action. In a leading study on the subject, Dr. Sarah Gordon examined the correlation between the number of new viruses in the wild and high profile prosecutions of virus authors, as a measure of the deterrence value of prosecution. Dr. Gordon reports that high profile prosecutions have had a limited deterrent effect.[121]

Dr. Gordon's conclusions were corroborated by a survey by the same author, in which virus authors and anti-virus researchers were asked whether the arrest and prospective sentencing of the Melissa author would have any impact on the virus writing

---

[119] Microsoft has since created a patch to fix this vulnerability. See details of vulnerability in Microsoft advisory VU#842160. The malicious code is commonly referred to as MyDoom{AG, AH, AI} or Bofra. When a user clicked on a malicious e-mail message, Internet Explorer opens and displays an HTML document that exploits the vulnerability to execute the virus. For details, see University of Cambridge, Technical User Support Computing Service. http://www-tus.csx.cam.ac.uk/virus/alerts.html.

[120] See, e.g., Robert Rossney, *How to Keep Your ID a Secret on Usenet*, S.F. CHRONICLE, Mar. 9, 1995, at D3 ("Freed from accountability, people can and do issue all kinds of vileness and stupidity."); Noah Levine, at 1537 ("In each case, anonymity serves to remove, or at least significantly decrease, the deterrence effect of the law against civil or criminal violations."); Tal Z. Zarsky, *Thinking Outside the Box: Considering Transparency, Anonymity, and Pseudonymity as Overall Solutions to the Problems of Privacy in the Internet Society*, 58 U. MIAMI L. REV. 991, 1028 (Anonymity adversely affects society by by causing the loss of accountability."); Paul A. Strassman and William Marlow, *Risk-Free Access Into the Global Information Infrastracture Via Anonymous Re-Mailers*, Symposium on the Global Information Infrastructure: Information, Policy & Information Infrastructure, Cambridge, MA, January 28-30, 1996 (The anonymous remailing service, anon.penet.fi., was frequently used by the Russian (ex-KGB) criminal element. The "double-blind" method of communication it offered is favored for engaging services of cybercriminals and for authorizing payments for criminal acts through a third party.)

[121] Sarah Gordon, *Virus Writers: The End of Innocence*. (Finding no evidence that such prosecutions have alleviated the virus problem, as measured by the rate of creation of new viruses in the wild subsequent to high profile porosecutions.) See, also, R. Lemos (1999), *'Tis the Season for Computer Viruses*. http://www..zdnet.co.uk/news/1999/49/ns-12098.html. (It is well-known that even after the author of the Melissa virus had been apprehended (and expected to be sentenced to a multi-year prison term), the appearance of new viruses on the Internet continued to proliferate, and at an increasing rate.)

community. All virus authors interviewed stated that there would be no impact, immediate or long-term, while the anti-virus researchers were evenly split between whether the arrest would or would not have any impact. These results are consistent with those of comparable surveys by other researchers.[122]

The results of a subsequent survey on the impact of anti-virus legislation on virus authors, suggest that new laws may, perversely, result in more viruses than before. According to the survey results, a majority of virus authors would either be unaffected or actually encouraged by anti-virus legislation. A significant number of the virus authors interviewed claimed that criminalization of virus writing would actually encourage them to create computer viruses, perhaps as a form of protest or civil disobedience.[123]

Laws against virus authors cannot be effective unless virus incidents are reported and perpetrators prosecuted There is evidence that virus crimes are seriously under-reported and as a consequence, under-prosecuted.[124] Companies tend to be reluctant to report security breaches, such as virus attacks, perhaps to avoid negative publicity.[125] Firms seem particularly reluctant to report and prosecute cybercrimes that originate from overseas.[126]

Commenting on the ineffectiveness of the law to combat computer viruses, Grable writes, "[b]oth the federal and New York state criminal statutes aimed at virus terror are

---

[122] Sarah Gordon, *Virus Writers: The End of Innocence* (Reference to a survey by A. Briney.)

[123] Sarah Gordon, *Virus Writers: The End of Innocence*, reference to DefCon survey.

[124] Sarah Gordon, *Virus Writers: The End of Innocence*. IBM White Paper, http://www.research.ibm.com/antivirus/SciPapers/VB2000SG.htm. ("Minnesota statute §§ 609.87 to .89 presents an amendment which clearly defines a destructive computer program, and which designates a maximum (prison term of) ten years; however, no cases have been reported. Should we conclude there are no virus problems in Minnesota?) See, also, Michael K. Block Joseph G. Sidak, *The Cost of Antitrust Deterrence: Why not Hang a Price-Fixer Now and Then?*, 68 GEO. L.J. 1131, 1131-32 (1980); Mitchell and Banker, *Private Intrusion Response*, 11 HARV. J.L. & TECH. 699, 704; Andy McCue, *IT Crime Still Going Unreported*, IT Week, 23 May 2002.
Available at: http://wwwinformaticsonline.co.uk/analysis/1132021. ("What we are seeing is an increase in actual and attempted crimes using technology and particularly the Internet. The number of security breaches reported is only the tip of the iceberg. For every one admitted there might be 100 held within companies.")

[125] Madeline Bennett, *Crime Laws Lack Coherence*, IT Week 20 May 2002 (citing Graham Cluley of anti-virus firm, Sophos, "[t]o ensure that virus authors receive sentences that reflect the gravity of their offenses, businesses should play their part. Viruses can cause great damage, yet businesses are ashamed to report infections. They must take a two-pronged stance: improve protection on their systems and be prepared to take action against the authors of malicious code.")

[126] Andy McCue, *IT Crime Still Going Unreported*, IT Week, 23 May 2002 (Security consulting firm reports that 90 percent of its client companies take no action when attack originates from overseas.)

ineffective because ... [t]he combination of the lack of reporting plus the inherent difficulties in apprehending virus creators leads to the present situation: unseen and unpunished virus originators doing their damages unencumbered and unafraid."[127]

We conclude that virus authors have the deterrence-teflon properties commonly associated with free radicals. They are often judgment-proof and shielded by the anonymity of cyberspace, are increasingly motivated by crime, and appear unconcerned about the problems caused by their creations. Currently, most virus and blended attacks depend on e-mail. Such attacks are aided by numerous technologies that enable anonymous e-mail transmission. Furthermore, virus attacks are under-reported, under-prosecuted and virus authors, to a significant degree, appear to be unconcerned and, perversely, often encouraged by the threat of legal liability and tougher laws.

4.3 EFR Factors

There are additional factors, besides the requirement that the second tortfeasor be a free radical, that courts look at before they hold a primary tortfeasor liable for encouraging free radicals. As a threshold requirement, the defendant's encouragement of free radicals must have been negligent before liability will be imposed.

Encouragement of free radicals must be negligent

A defendant will not be held liable for encouraging free radicals unless the encouragement was negligent. The encouragement must therefore have been a breach of duty to the plaintiff. Courts require a plaintiff to prove breach of duty by identifying an untaken precaution that would have prevented the harm, and showing that the untaken precaution would have yielded greater benefits in accident reduction than its cost. The CodeRed attack presents an illustrative example.

The Windows IIS vulnerability that enabled the CodeRed attack sequence was discovered on June 18, 2001.[128] A security patch to fix the vulnerability was promptly

---

[127] J. Grable, *Treating Smallpox with Leeches: Criminal Culpability of Virus Writers and Better Ways to Beat Them at Their Own Game*. Computers & The Law Project. University of Buffalo School of Law. See, also, Sarah Gordon, *Virus Writers: The End of Innocence*. ("[G]iven the small number of virus writers who have been arrested and tried ... this lack of arrests is one of the primary indicators used by some to argue that laws are not a good deterrent.")

See, also, BizReport News, Virus Writers Difficult to Find in Cyberspace, September 2003. (Reporting that it took 18 days to track down the author of the Blaster worm, even though the author left a clear trail behind, including his alias stitched into the virus code, and references to a Web site registered in his name.) Report available at http://www.bizreport.com/print.php?art_id=4917.

issued by Microsoft. The first version of CodeRed that exploited the vulnerability appeared approximately one month later.[129] Due to a programming flaw, the first version of CodeRed did not spread as fast and widely and do as much harm as its creator had apparently hoped for.

At this stage, after the first exploitation, the existence of the vulnerability was common knowledge in the IT community, a patch to fix it had been made available, and the first CodeRed attack, at the very least, alerted the IT community to the exploitability of the vulnerability and the harm it could cause. The IT community was also aware of the programming flaw in CodeRed that limited its effectiveness, and that the flaw could easily be fixed. The damage that a debugged version of CodeRed could do with the assistance of the Windows IIS vulnerability was therefore foreseeable.

A second, more virulent version of CodeRed appeared on July 19, 2001. The programming flaw that plagued its predecessor was fixed in this version, and the second version, predictably, caused substantially more harm than its predecessor.

An IT manager who failed to implement a security patch to fix the Windows IIS vulnerability, after the first CodeRed attack, may be held liable for negligently encouraging free radicals who subsequently exploited the flaw in his system to cause harm to other users. A breach of duty analysis would begin by considering an untaken precaution that would have avoided the second CodeRed infection. The most logical and probably most effective precaution would be implementation of the security patch provided by Microsoft. The Code Red worms could, for instance, not infect a system that had the Microsoft MS01-033 patch installed.[130] A commentator opined, "[t]here was so much publicity and so many published articles by the time Code Red II hit, that any competent server manager would have had ample opportunity to patch their systems in time."[131]

The plaintiff would be required to show that implementing the patch would have yielded greater benefits in accident reduction than its cost. The benefits would include avoidance of the foreseeable harm from further exploitation of the vulnerability. After the

---

[128] R. Lemos, *Microsoft reveals Web Server Hole.*
Available at: http://news.com/2100-1001-268608.html.

[129] eEye Digital Security Advisory. Available at:
http://www.eeye.com/html/Research/Advisories/AL20010804.html.

[130] Jeremy D. Baca, *Windows Remote Buffer Overflow Vulnerability and the Code Red Worm*. SANS Institute White Paper (September 10, 2001).

[131] Id., at 6.

appearance of the first version of CodeRed, a reasonably competent IT professional knew or should have been able to infer the potential harm from further and more efficient exploitation of the vulnerability.

The expected harm avoided must be weighed against the cost of implementing the patch. Although security patches are usually made available for free to users, implementing them may be costly and difficult, especially in large corporations with complex systems. Patches also tend to intereact with and affect the systems to which they are applied, sometimes impairing their performance.[132]

Although a conclusive resolution of the cost-benefit tradeoff would require a detailed numerical analysis of the costs and benefits involved, it appears that the defendant in this hypothetical was likely negligent in failing to implement the security patch. Implementing the security patch and dealing with and fixing bugs introduced by interaction between the patch and the regular system appear minor compared to fixing the harm from an attack by a blended threat, such as CodeRed and its potential successors.

Other EFR factors

In addition to the negligence requirement, the following factors influence courts in holding a defendant liable for damage caused by encouraging free radicals.[133]

1. The defendant's encouragement of the free radical was substantial.
2. The defendant created a scarce opportunity for the free radical.
3. The free radical's behaviour was foreseeable.
4. The free radical harmed a third party, as opposed to himself.
5. The foreseeable harm was serious.
6. The fact that the defendant's encouraging behaviour was deliberate, as opposed to inadvertent, was considered important in some cases.
7. The defendant had a special relationship with the free radical, the victim, or both.

A. Substantial encouragement

---

[132] Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 367, 410.

[133] Mark F. Grady, *The Free Radicals of Tort*, at 119.

The defendant's encouragement of the free radical must have been substantial for liability to be imposed. Courts have interpreted "substantial encouragement" in terms of the likelihood of provoking harmful behavior by free radicals.

Contrast, for instance, *Segerman v Jones*[134] with *Home Office v Dorset Yacht Co*.[135] In *Dorset*, seven boys who had been sentenced to working in a boot camp for juvenile offenders were working under supervision of three Home Office guards. One evening, in breach of their instructions to watch the boys, the guards simply went to bed, leaving the boys unsupervised. The boys swam out to an unattended yacht moored nearby and managed to set it in motion. They collided with another yacht owned by the plaintiffs, who sued the Home Office for the resulting damage. The trial court ruled in favor of the plaintiff, and the Court of Appeal affirmed.

Lord Reid stated that the Home Office would be liable if it appeared*very likely*, ex ante, that the boys would damage property if they were to escape from supervision. It is plausible to assume a high foreseeable likelihood of escape and harmful behavior by the delinquents. The boys were juvenile offenders, with records including convictions for breaking and entering, larcency and grand theft auto. Given, in addition, that five of the seven had a record of previous escapes from boot camp, the inference of substantial encouragement appears justified.

In *Segerman*, the defendant teacher left her classroom for a few minutes. During her brief absence, one student kicked out the teeth of one of his classmates. The Maryland Supreme Court held that the teacher was not liable. The extent of her encouragement of the children was leaving them to their own devices for a few minutes. This encouragement was too slight to impose liability. The free radicals in this case were ordinary school children, obviously not in the league of the juvenile offenders of *Dorset*.

Similar cases have denied liability for leaving a stake at a construction site,[136] for leaving a screwdriver out in a yard,[137] and leaving a load of dirt clods out in a backyard.[138] In these cases, the court apparently considered the likelihood of harm from the opportunity to be insignificant, and the encouragement therefore insubstantial.

---

[134] 259 A.2d 794 (Md. 1970).

[135] [1970] 2 A.C. 1004 (appeal taken from Eng.)

[136] Cole v Housing Authority, 385 N.E.2d 382 (Ill. 1979).

[137] Dennis ex rel. Evans v Timmons, 313 S.C. 338, 437 S.E.2d 138 (1993).

[138] Donehue v Duvall, 243 N.E.2d 222 (Ill. 1969). The defendants had hauled loads of dirt into their backyard. Children from the neighborhood frequented the pile and threw clods of dirt at each other, and the

In a blended attack, the original tortfeasor typically encourages free radical cyber rogues by making a tempting and exploitable security vulnerability available. Does a network security vulnerability constitute substantial encouragement to cyber rogues? The metric of the substantiality of a security vulnerability, such as a buffer overflow, is the likelihood that the vulnerability, when created, would be found and exploited to perpetrate a cyber attack. We argue that buffer overflows are likely to be discovered rapidly when they become available, and, once identified, promptly exploited. This conclusion is supported by empirical evidence.

It is unlikely that a valuable and exploitable computer security vulnerability, such as a buffer overflow, will remain undiscovered for long. Worms and viruses employed in blended threats are programmed to automatically search for and locate exploitable vulnerabilities. Furthermore, technologies are available to assist software designers in identifying security vulnerabilities in their products.[139] Although such vulnerability identififying technologies are intended to assist designers of "legitimate" software in troubleshooting and debugging, the technologies are, of course, equally available to designers of malevolent code.

Once an appropriate vulnerability is identified, it will likely be exploited. In his recent treatise on buffer overflow attacks, James Foster comments, "[i]t's no coincidence that once a good exploit is identified, a worm is created. Given today's security community, there's a high likelihood that an Internet worm will start proliferating immediately. Microsoft's LSASS vulnerability turned into one of the Internet's most deadly, costly and quickly proliferating network-based automated threats in history. Multiple variants were created and released within days."[140] In fact, current trends and patterns of infection suggest that the time lag between discovery of a vulnerability and its exploitation is shrinking.[141]

---

defendants knew about this. One of the children threw a dirt clod at the five-year old defendant, and injured his eye. The trial court dismissed the complaint, and the Illinois Supreme Court affirmed.

[139] See, e.g., James C. Foster et al., BUFFER OVERFLOW ATTACKS (2005), at 424 [Describing the CodeAssure Vulnerability Knowledgebase troubleshooting system, which is capable of reliably identifying flaws such as buffer and integer overflows in software.] See, also, Joel McNamara, Secrets of Computer Espionage: Tactics and Countermeasures (2003), at 235 [Discussing commercial, sometimes free, scanners, that can be used to probe a system for vulnerabilities.]

[140] James C. Foster et al., BUFFER OVERFLOW ATTACKS (2005), at 8.

[141] See, e.g., Chen & Robert, *The Evolution of Viruses and Worms*, at 13 ["Blaster suggests a trend that the time between discovery of a vulnerability and the appearance of a worm to exploit it is shrinking (to one month in the case of Blaster.)"]

Factors that contribute to the prompt exploitation of buffer overflow vulnerabilities include their ease of exploitation and their convenient properties that give cyber rogues exactly what they need. These properties include a buffer overflow's convenient configuration as a gateway to inject and execute attack code, and assume unauthenticated remote control of a system or network, including root control.[142]

A vulnerability may be considered easy to exploit if no special programming skills are necessary to take advantage of it, or if the necessary exploit code is publicly available.[143] Writing a successful buffer overflow exploit takes considerable programming skill, but buffer overflow exploit code is often publicly available and accessible, even to individuals without technical sophistication. As new buffer overflow vulnerabilities are discovered, exploits are habitually published shortly after the discovery.[144] Technical articles continuously appear, describing vulnerabilities and how to exploit them, often in substantial detail.[145]

Multiple buffer overflow vulnerabilities have been reported in advisories that were either trivial to exploit or for which exploit code was publicly available. A vulnerability in the Solaris KCMS Library Service System, for instance, was easy to exploit. Exploitation of this vulnerability could be accomplished by drawing on a standard and widely available software tool and basic computer literacy.[146]

A buffer overflow vulnerability in a version of the commercial program Hypermail was, likewise, easily exploitable.[147] Hypermail is an open-source program that converts e-mail messages into cross-linked HTML pages. The program contained a

---

[142] Cowan et al., at 1 ("Buffer overflow vulnerabilities particularly dominate in the class of remote penetration attacks because a buffer overflow vulnerability presents the attacker with exactly what they need: The ability to inject and execute attack code. The injected code runs with the privileges of the vulnerable program, and allows the attacker to bootstrap whatever other functionality is needed to control ("own") the host computer.")

[143] *Symantec Internet Security Threat Report*, Volume III, February 2003, at 47.

[144] For a review of publicly available exploits, see Takanen et al., *Running Malicious Code By Buffer Overflows: A Survey of Publicly Available Exploits.* EICAR 2000 Best Paper Proceedings. http://www.papers.weburb.dk.

[145] See, e.g., Smith, N.P., *Stack Smashing Vulnerabilities in the UNIX Operating System*. Southern Connecticut State University (1997). Available at http://destroy.net/machines/security/ (Thorough academic survey, covering history and terminology, various vulnerabilities and related technologies, as well as solutions).; Litchfield, D., Exploiting Windows NT 4 Buffer Overruns (1999).
Available at: http://wwwinfowar.co.uk/mnemonix/ntbufferoverruns.htm.

[146] SUN Advisory: http://sunsolve.sun.com/pub-cgi/retrieve.pl?doc=fsalert/50104.

[147] Vulnerability advisory: http://archives.neohapsis.com/archives/vulnwatch/2003-q1/0042.html.

vulnerability that could be exploited simply by sending malicious e-mail with an over-long attachment name. A detailed sample e-mail message that would trigger the overflow and control Hypermail's execution had been posted on the Internet.[148]

Not all buffer overflow vulnerabilities are easy to exploit. A recent advisory describes, for instance, the SpamAssassin buffer overflow as "challenging to exploit," depending on the target computing environment. This particular vulnerability is not exploitable on all platforms. To succeed, a would-be attacker would have to identify and target victims who are using a vulnerable spam filter.[149]

In summary, a security vulnerability such as a buffer overflow, likely constitutes substantial encouragement to perpetrators of cyber crimes such as blended attacks. A valuable and easily exploitable vulnerability, such as the buffer overflow, is likely to be promptly discovered and exploited. The discovery of vulnerabilities is facilitated by technology: Viruses used in blended attacks are often programmed to search for new vulnerabilities, and specialty software designed to identify security weaknesses in computer systems and networks is freely available. Attackers have a strong incentive to find and exploit vulnerabilities such as buffer overflows, because buffer overflows are easy to exploit and give attackers exactly what they need to launch a blended attack. Empirically, exploitation of buffer overflows is pervasive, both in an absolute sense, as well as measured as a percentage of all blended attacks.[150]

A buffer overflow vulnerability as substantial encouragement of free radical cyber rogues is therefore closer to *Dorset* (encouragement likely to incite free radical juvenile delinquents) than *Segerman* (encouragement insufficient to incite normal school children).

B. Scarce opportunity for wrongdoing

Courts are more likely to impose liability when the defendant has created a tempting opportunity that does not normally exist for the free radical. If a free radical already has

---

[148] http://packetstormsecurity.org/filedesc/hypermail.html.

[149] The SpamAssassin spamc daemon contains a buffer overflow vulnerability when running in Batched SMTP (BSMTP) mode. It is described in SANS Critical Vulnerability Analysis Vol 2 No 04. See, also, Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 402 (Describing the OpenSSL buffer overflow vulnerability as challenging to exploit.); Id, at 547 ("Some vulnerabilities are easily exploited by the attackers, while others take months to develop.")

[150] For statistics and a discussion of the pervasiveness of bufer overflow exploitation, see, infra, subsection C ("Predictable free radical behavior.")

several opportunities available for harmful behavior, the defendant's encouragement does not amount to a scarce opportunity.

A person flashing a wad of $100 bills, for instance, would probably not be liable for the harm caused by a fleeing thief who runs into and injures someone. Because of the availability to the thief of many other similar opportunities, the flash of money was not an unusually tempting opportunity to the free radical. If the person had not flashed the money, a determined thief would have found another equally attractive opportunity.[151]

In *Stansbie v Troman*,[152] the defendant, an interior decorator, neglected to lock the door of the house of a client. A burglar entered through the open door and stole the plaintiff's jewelery. The court held the defendant liable for the loss. The defendant had created an opportunity for the thief that does not normally exist - valuables are normally kept under lock and key.

In a similar case, the defendant put a scaffold in place next to the plaintiff's apartment building. Armed robbers used the scaffold to gain entry to the plaintiff's apartment and stole his goods. The New York Supreme Court denied the defendant's petition for summary judgment. The defendant had encouraged free radicals by making a scarce and tempting opportunity available to them.[153]

In a contrasting case, the defendant sold free radicals five gallons of gasoline into an open pail, in violation of a municipal ordinance which prohibited sales of gasoline in open containers and in excess of two gallons. The free radicals subsequently used the gasoline to commit arson. The defendant was held not liable. Providing the gasoline to the free radicals did not constitute a rare opportunity, as they could have siphoned the gasoline they needed from a car.[154]

The pattern of case law suggests that a scarce opportunity is one that (i) is not part of the free radical's normal opportunity set, and (ii) is more tempting than existing opportunities, perhaps because it lowers the transactions cost of the free radical's harmful behavior. A thief can always use brute force to break into an apartment to do his business. However, an unlocked door or conveniently placed scaffold would be an

---

[151] Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293, 310 (2002) ("The defendant, in order to be liable, must negligently provide some special encouragement of wrongdoing that does not exist in the normal background of incitements and opportunities.")

[152] [1948] 2 K.B. 48.

[153] Russo v Grace Institute, 546 N.Y.S.2d 509 (Sup. Ct. 1989).

[154] Gonsalez v Derrington, 363 P.2d 1 ((Cal. 1961).

unusual opportunity to lower his transactions cost: Less physical exertion, faster results, less likely to attract attention than a more forceful entry. The unlocked door and scaffold therefore fit the common law profile of a scarce opportunity. Selling gasoline to a free radical in an open pail, on the other hand, does not constitute a scarce opportunity. The alternative, siphoning the gasoline from a car, is not significantly more burdensome or costly.

Computer security vulnerabilities appear to be scarce opportunities. A buffer overflow vulnerability is analogous to the unlocked door in *Stansbie*. The unlocked door provides the convenient access normally reserved for someone with valid authentication, such as possession of a key. Similarly, a buffer overflow yields remote, unauthenticated and root access to a target computer system, as well as the opportunity to inject malicious code into the system. This kind of privileged access is normally reserved for the system administrator.[155]

The unlocked door also lowers the transactions costs of the intruder. Analogously, security vulnerabilities lower the cyber attacker's transactions cost by making attacks faster and more efficient, remotely executable, and the attack less likely to be blocked. Skilful exploitation of security vulnerabilities allow attackers to achieve more destruction, in less time and with lower computational expenditure.[156] Fast-spreading worms, such as Nimda, are also less likely to be blocked before their task is accomplished. The combination of security exploits with computer viruses enable complex attacks that are difficult to detect with conventional antivirus software.[157]

Security vulnerabilities conform to the common law pattern of scarce opportunities in free radical cases. They present opportunities not available in the normal functionality of computers and, analogously to an unlocked door or scaffold, offer

---

[155] See section xxx.

[156] Vulnerability exploits allow malevolent code to operate with greater speed and efficiency. Nimda relied, for instance, on an input validation exploit known as MIME header parsing, as well as auto-execution capability to spread rapidly. See, e.g., Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE, at 415 ("The ability to infect IIS Web servers via an input validation exploit and autoexecute upon users' reading or previewing e-mail played a large role in Nimda's ability to spread so rapidly.")

[157] Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 366 ("Security exploits, commonly used by malicious hackers, are being combined with computer viruses, resulting in very complex attacks that sometimes go beyond the general scope of antivirus software."), and Id, at 542 ("[H]ighly infectious worms [such as CodeRed and Slammer] jump ... over the Internet using buffer overflow attacks on networked services. Because the files need not be created on disk and the code is injected into the address space of the vulnerable processes, even file integrity systems remain challenged by this particular type of attack.")

privileged access and convenience to an attacker. Security vulnerabilities also lower the transactions costs of a cyber vandal: Greater potential harm, in less time, and with lower likelihood of timely detection. The opportunities presented by a buffer overflow therefore appear to be closer to *Russo* (scaffold) or *Stansbie* (unlocked door), than *Derrington* (pail of gasoline).

C. Predictable free radical behavior

Foreseeability is a touchstone of proximate cause. A crisp formulation of the proximate cause requirement is that the realized harm must be within the scope of risk foreseeably created by the defendant, and the plaintiff must belong to the class of persons foreseeably put at risk by the defendant's conduct.[158]

Consistently with the spirit of proximate cause, the behavior of free radicals must be foreseeable, or predictable, to hold their encouragers liable. If a radio station organizes a contest that encourages teenagers to race to catch up with a roving disk jockey, and they in fact race and cause an accident, the organizers of the contest will likely be held liable.[159] If the free radical goes too far or otherwise acts in an unpredictable manner, the defendant will escape liability. If one of the contestants had shot the other, for instance, the radio station would not be held liable.

In *Bansadine v Bodell*,[160] the defendant provoked a driver known for his aggression. The provoked driver shot the plaintiff's deceased. In spite of the driver's known aggressive tendencies, the defendant was found not liable. The court stressed that the defendant could not foresee that a driver would fire a gun at him for shining his high beams on the driver. The free radical driver's reaction, even for a person known for his fiery temperament, went beyond the encouragement of the defendant.

The exploitation by cyber attackers of vulnerabilities, such as the buffer overflow, is foreseeable. We have argued in subsection A ("Substantial encouragement") that buffer overflows are likely to be discovered rapidly when they become available, and, once identified, promptly exploited.

---

[158] Dan B. Dobbs, THE LAW OF TORTS, at 444. See, also, Judge Learned Hand in *Sinram v Pennsylvania RR. Co.*, 61 F.2d 767, 771 (2d Cir. 1932) ("[T]he usual test is ... whether the damage could be foreseen by the actor when he acted; not indeed the precise train of events, but similar damage to the same class of persons.")

[159] Weirum v RKO General, Inc, 539 P.2d 36 (Cal. 1975).

[160] 927 P.2d 675 (Utah App. 1996).

Foreseeability of exploitation of buffer overflows is confirmed by the empirical pervasiveness of such exploits, and the IT community's awareness of it. Buffer overflows are currently, and have been for a decade or so, the most commonly exploited security vulnerability and the most common way for an attacker outside of a target system to gain unauthorized access to the target system.[161] If buffer overflows were eliminated, the incidence of security breaches would be drastically reduced.[162] The computer security community is indeed aware of the exploitability of and hazards associated with buffer overflows. James Foster opines, "[b]uffer overflow vulnerabilities are the most feared of vulnerabilities from a software vendor's perspective. They commonly lead to internet worms, automated tools to assist in exploitation, and intrusion attempts."[163]

Statistics reported in security advisories confirm the dominance of the buffer overflow as exploit of choice. A third of investigating advisories spanning September 2002 through March 2004 were related to buffer overflows (224 out of 659).[164] In the year 2003, alone, approximately 75 percent of all CERT advisories were related to buffer

---

[161] Rupert Goodwins, *Playing Silly Buffers: How Bad Programming Lets Viruses In*, ZDNet White Paper (January 2004). ("The buffer overflow is the mechanism of choice for the discerning malware merchant. New hardware and software techniques are reducing the incidence of this perennial problem, but it's unlikely ever to go away completely.")
Available at http://insight.zdnet.co.uk/internet/security/0.39020457.39119117.00.htm.
See, also, Crispin Cowan et al., *Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade*, Working Paper, Dept. of Computer Science and Engineering, Oregon Graduate Institute of Science & Technology. (Describing buffer overflows as not the "most common form of security vulnerability" over the last decade, but also the dominant penetration mechanism for anonymous Internet users to gain remote control of a computer or network. ... Because buffer overflow attacks enable anyone to take total control of a host, they represent one of the most serious classes of security threats.")
Available at: http://www.cse.ogi.edu/DISC/projects/immunix.

[162] Crispin Cowan et al., *Buffer Overflows: Attacks and Defenses for the Vulnerability of the Decade*, Working Paper, Dept. of Computer Science and Engineering, Oregon Graduate Institute of Science & Technology. http://www.cse.ogi.edu/DISC/projects/immunix. ("If the buffer overflow vulnerability could be effectively eliminated, a very large portion of the most serious security threats would also be eliminated.")

[163] James C. Foster et al., Buffer Overflow Attacks (2005), at 20. See, also, Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 538 ["Every month critical vulnerabilities are reported in a wide variety of operating systems and applications. Similarly, the number of computer worms that exploit system vulnerabilities is growing at an alarming rate."]

[164] See, also, Mark E. Donaldson, *Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention* (April 2002). SANS White Paper ["Despite its lengthy history and simple preventitive methods, the buffer overflow continues to be a significant and prominent computer security concern even today. For example, buffer overflow problems are implicated in five of the SANS Top 20 vulnerabilities. The SuSE Linux website lists 22 buffer overflow vulnerabilities that require patching. Of the 44 CERT advisories published between 1997 and 1999, 24 were related to buffer overflow issues."]

overflows.[165] A Symantec security bulletin reported that blended attacks accounted for 60 percent of malicious code submissions during the first half of 2003, most employing a buffer overflow.[166] Other advisories and security reports exhibit a similar pattern.[167]

Serious buffer overflow problems are not limited to small and resource-deprived companies, but have also plagued the products of large and well-known software vendors. Advisories of companies such as Apple and Oracle were nearly all related to buffer overflows, while half the advisories of Cisco, Microsoft and Sun were related to overflows.[168]

Exploitation of buffer overflow vulnerabilities is foreseeable, because it is well known that new vulnerabilities are likely to be discovered, exploited when discovered, and that actual exploitation is in fact pervasive.

D. Third parties threatened

The courts do not usually allow free radicals to recover for injuries they have caused to themselves. The courts only hold encouragers liable when free radicals injure third parties.

In *Gilmore v Shell Oil Co.*,[169] the defendant's employee left a loaded gun within easy reach of a teenager. The teenager took the gun and shot and killed himself. Although the teenager had shot himself intentionally, it was clear from the circumstances of the case that, but for his ready access to the gun, he would not have done so. The trial court

---

[165] CERT Advisories, available at http://www.cert.org/advisories/. See, also, Mark Shaneck, *An Overview of Buffer Overflow Vulnerabilities and Internet Worms*, CSCI, December 2003, 1. Available at http://www-users.cs.umn.edu/~shaneck/MarkShaneck_BufferOverflows.pdf.

[166] John Leyden, *Worms Spread Faster, Blended Threats Grow*. The Register, 1 October 2003. http://www.theregister.co.uk/2003/10/01/worms_spread_faster_blended_threats/

[167] Among five attacks used in the 1998 Lincoln Labs intrusion detection evaluation, three were essentially social engineering attacks that snooped user credentials, and two were buffer overflows. Out of 13 CERT advisories from 1998, 9 involved buffer overflows. A least half of 1999 CERT advisories involved buffer overflows. See, e.g., Fred B. Schneider et al., TRUST IN CYBERSPACE. National Academy Press, 1999; Steve Bellovin, Buffer Overflows and Remote Root Exploits. Personal communication, cited in Crispin Cowan et al. (supra), ref. [5]. According to the (informal) Bugtraq security vulnerability survey, approximately two thirds of respondents felt that buffer overflows are the leading cause of security vulnerability. Schneider at al., Trust in Cyberspace (National Academy Press, 1999); Bugtraq mailing list: http://geek-girl.com/bugtraq/.

[168] Edmund X. DeJesus, *Drowning in Buffer Overflow Vulnerabilities*, SECURITY WIRE PERSPECTIVES (24 May 2004).

[169] 613 So. 2d 1272 (Ala. 1993).

nevertheless entered summary judgment for the defendant, and the Alabama Supreme Court affirmed. Deterrence of this type of behavior is outside the policy objectives of tort law. The case may have been decided differently if the deceased had been a young child, as the EFR doctrine protects children against themselves.[170]

E. Serious harm

Courts are more likely to hold a defendant liable if the foreseeably encouraged harm is serious. Someone who has left explosives around children,[171] for instance, is more likely to face liability than someone who has left a pile of dirt clods,[172] and someone who fails to supervise juvenile delinquents is more likely to face liability than a school teacher who leaves ordinary school children momentarily to their own devices.[173]

The severity of a computer security breach resulting from a system vulnerability is a function of the following factors.

A. The *degree of control* over the affected system or network given to an exploiter of the vulnerability. The degree of control afforded by the vulnerability depends on (i) the level of access it confers, (ii) the degree of remote exploitability it allows; (iii) its ease of exploitation; and (iv) the degree to which it enables circumnavigation of authentication requirements.[174]

B. Once the attacker has gained control, the *kind and degree of harm* such control allows the attacker to unleash.

Degree of control given by buffer overflow

Access. The ultimate gift to a cyber attacker would be the most privileged level of access, namely full root-level access to the target system. "Root" is the conventional name of the superuser who has all rights in all modes on the computer system. This is usually the system administrator's account. The superuser has privileges that an ordinary user does

---

[170] Mark F. Grady, *The Free Radicals of Tort*, at 127.

[171] Travell v Bannerman, 75 N.Y.S. 866 (App. Div. 1902).

[172] Donehue v Duvall, 243 N.E.2d 222 (Ill. 1969).

[173] Segerman v Jones, 259 A.2d 794 (Md. 1970).

[174] *Symantec Internet Security Threat Report*, at 47.

not have, such as authority to change the ownership of files, install and run programs, change Web Server databases, add, change, or delete system files or data, and change or replace web pages.[175] An attacker who gains system-level access inherits these privileges. Hence, if a program is already running with root privileges, a buffer overflow could hijack the program and transfer root control to the attacker.[176] The attacker would then effectively become the administrator of the system. Exploitation of buffer overflows commonly yield root access to the attacker. The Linux application, DosEMU, for instance, had a buffer overflow vulnerability that assisted an attacker in gaining root access.[177]

Remote exploitability. A vulnerability allows remote exploitability when it enables a user to access and execute commands on a remote system, as if the user were connected to a direct terminal on the system. Buffer overflow vulnerabilities are the perfect springboard to gain remote access to a target system, because it allows an attacker to inject malevolent code, such as an e-mail worm, directly into the execution path of the remote system.[178] The viral code could then create further opportunities for other remote attackers. The Nimda worm, for instance, attacked via backdoors left by worms such as CodeRed.[179]

Remote exploitation of buffer overflows has recently been reported in well known products, such as Sendmail, various Microsoft products, and, ironically, PGP.[180] A

---

[175] *Root (computing)*, at http://en.wikipedia.org/wiki/Root_access.

[176] Dorothy E. Denning, INFORMATION WARFARE AND SECURITY (1999), 214 (Referring to malcicious code executed via a buffer overflow as executing "with the privileges of the program it exploits, which is often root."

[177] See, e.g., Securiteam advisory, *DosEMU Buffer Overflow Assists in Gaining Root*, at http://www.securiteam.com/exploits/2GUPVSAQO0.html.

[178] Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 368.

[179] Nimda's other attack vectors were infection of Microsoft IIS web servers via a buffer overflow exploit; infection of network shares; and infection via Javascript added to web pages. See, e.g., Thomas Chen, *Trends in Viruses and Worms*, presentation at SMU Dept. of EE. Other attack points frequently used in blended attacks include injecting malicious code into .exe files on a target system, creating world readable network shares, making multiple registry changes, and adding script code to html files.

[180] See, e.g., Symantec Security Response (3-3-2003) ("A remotely exploitable vulnerability has been discovered in Sendmail. This vulnerability is due to a buffer overflow condition in the SMTP headrer parsing component. Remote attackers may exploit this vulnerability by connecting to target SMTP servers and transmitting them malformed data.")
http://securityresponse.symantec.com/avcenter/security/Content/3.3.2003.html.
See, also, *Technical Cyber Security Alert TA04-260A*, United States Computer Emergency Readiness Team (13 February 2005) (Describing a vulnerability in Microsoft's Graphic Device Intrface Plus, which may

vulnerability in Microsoft's Internet Explorer browser, for instance, allowed a properly formatted HTML document to cause a buffer overflow. This flaw could be exploited to allow an attacker to execute arbitrary code on the affected system, including malicious code, with the privileges of the user running Internet Explorer. The vulnerability was remotely exploitable.[181]

Ease of exploitation. As discussed in subsection A ("Substantial encouragement"), a vulnerability is easily exploited if a would-be attacker does not need technical sophistication or a complex exploit to use it, or if a suitable exploit is publicly available. We have argued that although not all buffer overflow vulnerabilities are necessarily easy to exploit, many are, and even for difficult to exploit vulnerabilities, exploits are frequently publicly available soon after the vulnerability is discovered.[182]

Authentication requirements. The term "authentication" refers to the procedures by which a computer system verfies the identity of a party from whom it has received a communication. The login procedure is probably the best-known example of an authentication procedure. A login prompt asks the user to identify herself, followed by a request for a password. The system then authenticates the stated identity of the user by validating the password, if the password and identity match. If they do not match, the user is restricted from accessing the system. Other examples of authentication include the requirement of confirmation e-mail to activate an on-line account, ATM access,

---

allow remote executability of malicious code on an affected system, by reading an HTML-rendered e-mail message or opening a crafted JPEG image. The attacker may execute their own code on the affected system with privileges of the user running the software component being attacked.)
http://www.us_cert.gov/cas/techalerts/TA04-260A.html.
See, also, Foundstone Labs Advisory - 090502-PCRO (advisory on remotely exploitable buffer overflow vulnerability in PGP.) http://www.foundstone.com/advisories.;
DosEMU Buffer Overflow Assists in Gaining Root (Describing a buffer overflow vulnerability in a Linux application that enables root access to a malicious attacker.) Available at:
http://www.securiteam.com/exploits/2GUPVSAQ0O.html.
Samba Buffer Overflow, IntruShield Security Advisory SA20 (Describing buffer overflow vulnerability in the Samba server, a widely used, open-source UNIX compatible server. Advising that vulnerability allows remote attackers to gain root access.) Available at:
http://www.networkassociates.com/us/security/resources/sv_20.htm.

Software vendors, notably Microsoft, typically issue a patch to fix a vulnerability as soon as it is discovered. Users are not always diligent in implementing the patch, hence the continuing exploitability of some vulnerabilities even after corrective action by the vendor.

[181] *Buffer Overflow in Microsoft Internet Explorer*, Computing Services Security Information, UC Davis Computing Services (2004-11-10). This document also advises on other remotely exploitable vulnerabilities in Microsoft Internet Explorer.
Available at http://www.ucd.ie/computing/support/security.html.

[182] See, supra, subsection A ("Substantial encouragement.")

cryptographic authentication of a digitally signed contract, and biometric identification in applications such as Internet banking.

Authentication provides a line of defense against unauthorized access to a resttricted system. A vulnerability that allows unauthenticated access may allow an attacker to bypass this line of defense. Network vulnerabilities, including buffer overflows, allow unauthenticated remote access to attackers without authentication.[183]

A remotely exploitable buffer overflow in Microsoft Data Access Components (MDAC), a system that provides database access for Windows platforms, was recently reported. The vulnerability enabled an attacker to run unauthenticated arbitrary code on an affected system.[184] The unauthenticated arbitrary code could, of course, be malicious.[185]

A vulnerability in Fusion News, a news management program for web servers, allowed remote unauthenticated attackers to create arbitrary user accounts on the Fusion News server by sending a specially crafted request to the server. If properly structured, the request could also be used to gain administrative access. Exploitation of this vulnerability was trivial. A ready-to-use sample server request was, for instance, available on the Internet.[186] This vulnerability contained all the critical elements

---

[183] See, e.g., *How Safe Is Your Firewall? Microsoft NetDDE Service Unauthenticated Remote Buffer Overflow (MS04-031)* (23 Jan 2005). Securiteam Advisory. (Reporting a vulnerability in the Microsoft DDE service that allows a remote attacker to execute arbitrary code on a system without authentication.)
Available at: http://www.ngssoftware.com/advisories/netddefull.txt. Microsoft has released an update addressing the vulnerability: http://www.microsoft.com/technet/security/bulletin/MS04-031.mspx. See, also, Dorothy E. Denning, INFORMATION WARFARE AND SECURITY (1999), 214 (Hackers execute malcicious code remotely, "without logging in through an account or password," by exploiting buffer overflow vulnerabilities.)

[184] The vulnerability offered several opportunities to an attacker, including exploiting clients via a malicious web server. If a user were to browse an infected web site, the server would invoke a new session, and then, unbeknownst to the user, bring the user back to the web site but via the new session. At this point a buffer overflow within Internet Explorer would allow the server to run unauthenticated arbitrary code on the client system. *Remotely Exploitable Buffer Overflow in Microsoft MDAC*. Available at: http://security-protocols.com/modules.php?name=News&file=article&sid=1391.

[185] *Remotely Exploitable Buffer Overflow in Microsoft MDAC*. Available at: http://security-protocols.com/modules.php?name=News&file=article&sid=1391.
See, also, *Buffer Overflow in System V Derived Logic*, Symantec Security Response (14 December 2001) (Reporting vulnerability that allows root access to systems with programs that use a vulnerable login for authentication.)
Available at: http://securityresponse.symantec.com/avcenter/security/Content/2001.12.14b.html.

[186] Posting by DarkKnight: http://archives.neohapsis.com/archives/bugtraq/2003-08/0201.html. See, also, Peter Szor, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE (Symantec Press, 2005), at 410.

favorable to a cyber attacker: no authentication barriers, system administration-level (root) access, ease of exploitation, and allowing execution of malicious code.

Economic impact

The ultimate measure of the severity of a cyber attack is its economic impact. By this measure, blended threats, aided by buffer overflow vulnerabilities, are capable of considerable harm. The CodeRed family of blended attacks, although not the first of its kind, woke us up to the risks of remotely launched buffer overflow attacks.[187] The first CodeRed worm caused billions of dollars of damage in just a few days, despite corporate firewalls and other defensive efforts. Worldwide harm caused by Code Red is estimated at $2.62 billion.[188] Subsequent blended attacks, such as Nimda, continued the trend. According to an estimate by consulting firm, Computer Economics, Nimda infected more than 2.2 million servers and PCs in a 24-hour period during September 2001, causing damage of more than $590 million worldwide.[189] A study by computer and communications consulting firm, Aberdeen Group, reports that annual productivity loss due to viruses and blended threats averages more than $200 per employee in the financial industry.[190] Buffer overflow vulnerabilities are well represented in the SANS top 20 list of CERT security vulnerabilities,[191] and is ranked fifth in the Top Ten Vulnerabilities by Orthus Information Security Solutions.[192]

In conclusion, security vulnerabilities, especially the buffer overflow, present opportunities to free radicals to do serious harm. The severity of the harmful behavior encouraged by a buffer overflow vulnerability is due to (i) the degree and level of control over the affected system or network it affords an attacker, and (ii) once the attacker has control, the potential harm such control allows the attacker to unleash. Empirical data

---

[187] See, e.g., http://www.cert.org/advisories/CA-2001-19.html.

[188] 2002 CSI/FBI Survey, Computer Security Institute.
See, also, http://www.idg.net/english/crd_code_red_665795.html.

[189] See, e.g, http://www.idg.net/english/crd_code_red_665795.html.

[190] *The Financial Value of Symantec's Security Solutions*, Aberdeen Group Executive White Paper, March 2003. http://www.aberdeen.com.

[191] *SANS Top 20 Internet Security Vulnerabilities.* http://www.sans.org/top20/ 10/08/2003.

[192] *Orthus Top Ten Vulnerabilities*. http://www.orthus.com/ttvuln.html 2004.

suggest that blended attacks do in fact exploit vulnerabilities to do considerable economic damage.

F. Deliberate encouragement

Negligence law distinguishes between deliberate and inadvertent failure to use a reasonable precaution, in EFR cases. A defendant will more likely face liability if he deliberately encouraged a free radical to do harm, although even inadvertent encouragement will yield liability when the threatened harm is sufficiently serious and probable.

In *Mills v Central of Georgia Ry.*, [193] the defendant had left a signal torpedo on its tracks. A signal topedo is an explosive device which would blows up upon impact, such as when hit by an oncoming train. Its purpose was to warn crews working on railroad tracks of an approaching train. If a torpedo were not detonated, it was supposed to be picked up and put away. Contrary to this precaution, however, the torpedo in question was left inadvertently on the tracks. The plaintiff's sons found the torpedo, played with it and injured themselves when it exploded. The Georgia Supreme Court ultimately found for the plaintiff. Although the defendant created the opportunity inadvertently, the harm threatened was sufficiently serious and probable to justify imposing liability.

G. Special relationship

Liability is more likely when the defendant had a special relationship with the victim, the free radical, or both. This is consistent with general principles of negligence law. A hotel or common carrier, for instance, has a special duty to guard the best interests of its customers. An airline would likely be held liable for negligently maintaining a highly disorganized baggage claim area that leads to injury of a passenger.[194]

A defendant who has encouraged free radicals through a nonfeasance as opposed to a misfeasance will not be liable, unless there was a special relationship. An individual who has advance knowledge of a cyber attack and who fails to warn an unrelated plaintiff, will not be liable to the unrelated plaintiff for any harm from the attack.

---

[193] 78 S.E. 816 (Ga. 1913).

[194] *Stagl v Delta Airlines*, 52 F.3d 463 (2d Cir. 1995) (per Calabresi, J.)

6. DISCUSSION AND CONCLUSION

Information security threats are diversifying and evolving into multi-threat weapons that combine a variety of attack technologies and exploitation of security vulnerabilities. The blended attack exploits synergies between a multi-vector virus or worm and a computer security vulnerability, such as the buffer overflow, to enhance the effectiveness and destructiveness of its payload.

Blended attacks vary in complexity and technology, but they have two elements in common, namely (i) a multi-vector worm or virus, and (ii) exploitation of a security vulnerability. Skillful combination of the two elements creates synergies that make such attacks more hazardous than previous generations of malevolent code. The two salient elements of a blended attack focus the spotlight on the two most likely defendants in a civil action involving a blended attack: (i) The original tortfeasor responsible for the security vulnerability, and (ii) the second tortfeasor responsible for malevolent code that exploited the vulnerability. The tortfeasors are concurrent efficient causes of the harm of the victim of a blended attack.

The direct consequences doctrine of proximate cause examines concurrent efficient causes to determine whether the second tortfeasor (the virus distributor) has cut off the liability of the first (the software vendor). An intervening crime or intentional tort, as is often the case in a cyber attack, normally cuts off the liability of the first tortfeasor. This is significant, because the second tortfeasor, the exploiter of the vulnerability, is often judgment-proof or otherwise immune to liability, in contrast to the original tortfeasor. If liability were fixed exclusively on the second tortfeasor, it would leave the victim of a blended attack without recourse.

The Encourage Free Radicals (EFR) paradigm of the direct consequences doctrine creates an exception if the second tortfeasor is a free radical. It fixes liability on the primary tortfeasor if she created an opportunity for free radicals to do harm. The policy objective of the EFR doctrine is to preserve the liability of individuals who are deterred by the threat of liability, by preventing a solvent defendant from shifting liability to a judgment-proof individual who is not so deterred.

The analysis in the article shows that virus authors and distributors who exploit security vulnerabilities to launch blended attacks have properties commonly associated with free radicals. An analysis of the technology and mechanism of blended attacks suggests that the factors that influence courts in holding a defendant liable for encouraging free radicals are present in a typical blended attack. We conclude that software designers and commercial vendors who are negligently responsible for security

vulnerabilities in their products would likely be held liable for the harm caused by cyber rogues who exploit such vulnerabilities. This result is especially significant to plaintiffs who have suffered harm in a blended attack.