ENABLING TECHNOLOGIES OF CYBER CRIME: *WHY LAWYERS NEED TO UNDERSTAND IT*

By Meiring de Villiers

Volume XI – Spring 2011

ABSTRACT

This Article discusses the enabling technologies of cyber crime and analyzes their role in the resolution of related legal issues. It demonstrates the translation of traditional legal principles to a novel technological environment in a way that preserves their meaning and policy rationale. It concludes that lawyers who fail to understand the translation will likely pursue a suboptimal litigation strategy, face speculative recovery prospects, and may overlook effective and potentially powerful defenses.

ENABLING TECHNOLOGIES OF CYBER CRIME:
WHY LAWYERS NEED TO UNDERSTAND IT

By Meiring de Villiers [#]

Volume XI – Spring 2011

> "Just as the law and economics movement taught lawyers that understanding the interaction between rules and economic incentives was essential if regulations were to have their intended effect, so too does cyberlaw teach that an understanding of the interaction between rules and technical structures is essential if regulations are to have their intended effect."
>
> - Lawrence Lessig.[1]

1. INTRODUCTION

Information is the lifeblood of modern society, and its security is a defining issue of the Information Age.[2] Computerized information such as medical histories and financial records allows business to operate more efficiently, but also exposes the persons to whom the information relates to risks such as identity theft, monetary losses, and loss of privacy.[3]

Technology is at the core of information security. It enables crime, but also prevents it.[4] Cryptography is a prime example of a technology that stops some crimes but also creates new crimes. It preserves confidentiality, but also facilitates identification. Cryptography preserves

[#] Senior Lecturer at the University of New South Wales, School of Law, Sydney, Australia.

[1] Lawrence Lessig, *Law Regulating Code Regulating Law*, 35 LOYOLA U. CHI. L.J. 1, 1 (Fall 2003).

[2] *See e.g.* Danielle Keats Citron, *Reservoirs of Danger: The Evolution of Public and Private Law at the Dawn of the Information Age*, 80 S. CAL. L. REV. 241, 241 (2007) ("A defining problem of the Information Age is securing computer databases of ultrasensitive personal information").

[3] *See e.g.* Thomas M. Chen and Chris David, *An Overview of Electronic Attacks*, *in* INFORMATION SECURITY AND ETHICS: CONCEPTS, METHODOLOGIES, TOOLS AND APPLICATIONS 532 (2008) *available at* http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.128.146.

[4] *See* LAWRENCE LESSIG, CODE AND OTHER LAWS OF CYBERSPACE 31, 36 (1999) (Describing cryptography as a Janus-faced technology that stops some crimes but enables others.); BILL BLUNDEN, THE ROOTKIT ARSENAL: ESCAPE AND EVASION IN THE DARK CORNERS OF THE SYSTEM 13-15 (2009) (Discussing use of rootkit technology as a tool of crime as well as law enforcement).

confidentiality by keeping communications secret.[5] It facilitates identification by providing reliable digital signatures that uniquely identify the sender of a communication.[6] The confidentiality function of cryptography helps wrongdoers such as money launderers escape regulation. The identification function enables regulation by authenticating the identity of senders of information.[7]

Most information-related crime is enabled by computer code generically known as malevolent software, which is commonly referred to as "malware".[8] The most common of these rogue programs are the computer virus and its rapidly evolving variant, the so-called "worm."[9] Viruses can be programmed to access and steal confidential information, corrupt and delete electronic data, and monopolize computational resources that should be available to legitimate users.[10] Like their biological counterparts, computer viruses prey on weakness. Most virus and worm attacks on the Internet exploit security vulnerabilities in a target to penetrate and take control of the vulnerable system.[11] Technology also plays a defensive role in information

[5] *See* CHARLES P. PFLEEGER & SHARI LAWRENCE PFLEEGER, SECURITY IN COMPUTING (4th ed. 2007) (1989) 37-8.

[6] *See id*. at 82-3 (Stating that a digital signature is a protocol that provides the same effect as a realspace signature. A digital signature must be incapable of forging and authentic. It must be a mark that only the sender can make, and it must be easily recognizable as belonging to the sender).

[7] *Id.*

[8] *See* PETER SZOR, THE ART OF COMPUTER VIRUS RESEARCH AND DEFENSE 308-11 (2005); V. Skormin et al., *Prevention of Information Attacks by Run-Time Detection of Self-replication in Computer Codes* 15 J. COMPUTER SEC. 273, 273 (2007) ("Most information attacks are carried out via Internet transmission of files that contain the code of a computer virus or worm."); Verizon Business Risk Team, *2009 Data Breach Investigations Report (2009)* at 20, http://www.verizonbusiness.com/resources/security/reports/2009_databreach_rp.pdf (last visited March 27, 2011) (Estimating that 90 percent of security breaches in 2008 were enabled by malevolent code such as computer viruses and worms); Symantec, *Anatomy of a Data Breach: Why Breaches Happen and What to Do About It (2009)* at 4, http://eval.symantec.com/mktginfo/enterprise/white_papers/b-anatomy_of_a_data_breach_WP_20049424-1.en-us.pdf (last visited March 27, 2011).

[9] *See* FREDERICK B. COHEN, A SHORT COURSE ON COMPUTER VIRUSES 1-2 (2nd ed. 1994); DOROTHY E. DENNING & PETER J. DENNING, INTERNET BESEIGED: COUNTERING CYBERSPACE SCOFFLAWS 75 (Dorothy E. Denning et al. eds, 1998); John F. Schoch & Jon A. Hupp, *The "Worm" Programs - Early Experience with a Distributed Computation*, 25 COMM. ACM 172 (1982).

[10] *See* e.g. SZOR, *supra* note 8, at 296-7.

[11] *See* Rahul Telang & Sunil Wattal, *An Empirical Analysis of the Impact of Software Vulnerability Announcements on Firm Stock Price*, 33 IEEE TRANSATCIONS ON SOFTWARE ENGINEERING 544, 544 (Aug. 2007); MARCUS

security.[12] The escalation of cyber crime has spawned a burgeoning computer security industry and prompted the development of advanced anti-virus and other security technologies.[13]

Malevolent code is particularly dangerous in the hands of terrorists who can exploit it to disrupt elements of the national critical information infrastructure, such as banking, transportation, communications, and energy provision systems.[14] A recently discovered worm known as Stuxnet highlighted the fact that direct attacks on the critical infrastructure are a real possibility.[15] Stuxnet was programmed to infiltrate the industrial control systems of infrastructure targets such as gas pipelines, factories, and nuclear power plants, and reprogram them to sabotage their operation.[16] Stuxnet is a sophisticated and targeted technology.[17] It

JAKOBSSON AND ZULFIKAR RAMZAN, CRIMEWARE: UNDERSANDING NEW ATTACKS AND DEFENSES 458 (2008) ("Crimeware is beginning to combine characteristics of viruses, worms, and trojan horses with server and Internet vulnerabilities, fusing numerous methods for compromising end-user systems with multiple means of propagation to other networked machines").

[12] *See e.g.*, Samuel T. King et al., *SubVirt: Implementing Malware with Virtual Machines* 1 (2006), *available at* http://www.eecs.umich.edu/virtual/papers/king06.pdf (last visited March 27, 2011). ("A battle is taking place between attackers and defenders of computer systems. An attacker who manages to compromise a system seeks to carry out malicious activities on that system while remaining invisible to defenders. At the same time, defenders actively search for successful attackers by looking for signs of system compromise or malicious activities").

[13] *See generally*, SZOR, *supra* note 8, chs.11-16; JESSICA JOHNSTON, TECHNOLOGICAL TURF WARS: A CASE STUDY OF THE COMPUTER ANTIVIRUS INDUSTRY (2009); Carey Nachenberg, *Computer Virus-Antivirus Coevolution*, 40 COMM. ACM 46, 46 (Jan. 1997); Nancy R. Mead, *Who is Liable for Insecure Systems?*, COMPUTER 27, 27 (July 2004) (Stating that an entire industry has developed in response to security concerns. "Software vendors develop and announce patches for security holes. System administrators busy themselves applying security patches. Other vendors sell products such as virus detection tools, and consultants perform penetration testing. Network technicians install firewalls and configure them to block intrusions. Clearinghouses provide a continual stream of information about the latest viruses, worms and Trojan horses.").

[14] *See e.g.,* PFLEEGER & PFLEEGER, *supra* note 5, at 403; Thomas Claburn, *CIA Admits Cyber Attacks Blacked Out Cities*, INFO. WK. (Jan. 18, 2008), http://www.informationweek.com/news/internet/showArticle.jhtml?articleIC=25901631 (last visited March 27, 2011); CRIMEWARE: UNDERSTANDING NEW ATTACKS AND DEFENSES XVII (M. Jakobsson & Z. Ramzan, Crimeware eds., 2008) ("[M]alware ... is now a tool firmly placed in the hands of organized crime, terror organizations, and aggressive governments.").

[15] *See* Nicolas Falliere et al., *W32.Stuxnet Dossier*, SYMANTEC SECURITY RESPONSE (February 2011), http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf (last visited March 27, 2011).

[16] *Id.* at 1 (Describing the ultimate goal of Stuxnet as reprogramming industrial control systems).

[17] *Id.* ("Stuxnet is a large, complex piece of malware with many different components and functionalities"[,] including "zero-day exploits, a Windows rootkit, the first ever PLC rootkit, antivirus evasion techniques, complex process injection and hooking code, network infection routines, peer-to-peer updates, and a command and control interface."); *Kaspersky Lab Provides Insight on Stuxnet Worm*, KASPERSKY.COM (Sept. 24, 2010),

avoided detection by using a Windows rootkit[18] enabling it to hide its code from users of an infected machine.[19] Computer security firms such as Symantec have developed technical defenses against Stuxnet and its activities.[20]

This Article reviews the enabling technologies of cyber crime and analyzes their role in the resolution of legal issues related to cyber crime. The first part of the Article discusses the principles of the major enabling technologies, namely malevolent code, computer security vulnerabilities typically exploited, and technical defenses against malevolent attacks. The second part analyzes the interaction between the law and technology of information security.

The Article focuses on the paradigmatic case where a defendant such as a financial institution or medical service provider, enables the compromise of confidential information by failing to take reasonable security precautions. A database owner such as a financial institution may be under a statutory duty to exercise due care to protect sensitive data against intruders.[21] For instance, California's Security Breach Information Act (SBIA)[22] expressly creates a civil cause of action for failure to protect customers' personal information, and provides for recovery

http://www.kaspersky.com/news?id=207576183 (last visited March 27, 2011); C.P. PFLEEGER & S.L. PFLEEGER, *supra* note 5, at 141 (Describing targeted malicious code as "code written for a particular system, for a particular application, and for a particular purpose.").

[18] *See* PFLEEGER & PFLEEGER, *supra* note 5, at 145 (Describing a rootkit as "a piece of malicious code that goes to great lengths not to be discovered or, if discovered and removed, to reestablish itself whenever possible."); BLUNDEN, *supra* note 4, at 10 (2009) (Defining a rootkit as "a collection of tools ... that allow intruders to conceal their activity on a computer so that they can covertly monitor and control the system for an extended period.").

[19] Jarrad Shearer, *W32.Stuxnet*, SYMANTEC.COM (Sept. 17, 2010, 8:53 AM), http://www.symantec.com/security_response/writeup.jsp?docid=2010-071400-3123-99 (last visited March 27, 2011).

[20] *Id.*

[21] *See, e.g.* RESTATEMENT (THIRD) OF TORTS: LIABILITY FOR PHYSICAL HARM § 14 cmt. b (Proposed Final Draft No. 1, 2005) (Discussing express and implied statutory causes of action); VINCENT R. JOHNSON & ALAN GUNN, STUDIES IN AMERICAN TORT LAW, 305-06 (3d ed., 2005) ("In the one case, the court is saying that the legislation sets the standard because the legislature implicitly intended it to do so, and in the other case, the court acknowledges that the statute sets the standard because the court thinks that it is a good idea. Either way, if the statute does not expressly create a cause of action, the essential inquiry is the same: was the law intended to protect this class of persons from this type of harm.").

[22] Act effective July 1, 2003, ch. 915, 2002 Cal. Legis. Serv. (West), *available at* CA LEGIS 915 (2002) (Westlaw).

of civil damages.[23] The civil cause of action created by the SBIA is rooted in negligence, because the duty it imposes is based on reasonableness.[24] The essence of the negligence standard of care is a duty to avoid unreasonable risks of harm.[25]

Negligence law has been aptly described as "a creature of technology."[26] The elements of a negligence cause of action are defined in terms of technical specifications, and their analysis is shaped by technology. Common law negligence pleading rules require a plaintiff to identify and plead a precaution that the defendant had failed to take.[27] The untaken precaution must be technically feasible, must provide greater benefits in risk reduction than its cost, and must be capable of defeating the wrongdoer's attack technology.[28] The untaken precaution forms the basis of the plaintiff's case, and defines the standard of care that the defendant, the court hearing the case, and perhaps a subsequent appellate court, will use.[29]

Technology plays a role in other key issues in negligence law, such as reasonable foreseeability. The issue of foreseeability is usually resolved trivially in real space cases. A

[23] Cal. Civ. Code 1798.81.5(b) (West 2006) ("A business that owns or licenses personal information about a California resident shall implement and maintain reasonable security procedures and practices appropriate to the nature of the information, to protect the personal information from unauthorized access, destruction, use, modification, or disclosure."); Cal. Civ. Code 1798.84(b) (West 2010) ("Any customer injured by a violation of this title may institute a civil action to recover damages.").

[24] *See* Cal. Civ. Code 1798.81.5 (b), (c) (West 2006) (requiring "reasonable security procedures and practices" that are "appropriate to the nature of the information"); *See also* Michael L. Rustad & Thomas H. Koenig, *The Tort of Negligent Enablement of Cybercrime*, 20 BERKELEY TECH. L. J. 1553, 1592-93 (2005) ("Although courts vary in what impact a statutory violation has on the adjudication of negligence, they may employ civil statutes to set standards in negligent enablement lawsuits. An unexcused violation of a statute requiring reasonable security is itself negligence, that is, negligence per se").

[25] *See* RESTATEMENT (THIRD) OF TORTS: LIAB. PHYSICAL HARM § 3 (2005) ("A person acts negligently if the person does not exercise reasonable care under all the circumstances."). *See also* Richard W. Wright, *The Standards of Care in Negligence Law*, *in* PHILOSOPHICAL FOUNDATIONS OF TORT LAW 249, 250 (David G. Owen ed., 1995) ("Negligence is generally described as behavior that creates unreasonable foreseeable risks of injury"); PROSSER AND KEETON ON THE LAW OF TORTS § 31 (5th ed., 1984).

[26] *See* Mark F. Grady, *Why Are People Negligent? Technology, Nondurable Precautions, and the Medical Malpractice Explosion*, 82 NW. U.L. REV. 293, 293 (1988).

[27] Mark F. Grady, *Untaken Precautions*, 18 J. LEGAL STUD. 139 (1989).

[28] *See Id.* at 143.

[29] *Id.* at 144.

collision is clearly a foreseeable consequence of failing to maintain a proper lookout in a busy traffic intersection,[30] and the link between mesothelioma and protracted exposure to asbestos fibers is well established.[31] The issue is more complex in information security. Consider for instance a case where the plaintiff pleads as an untaken precaution the defendant's failure to remedy a security vulnerability exploited by attackers. The plaintiff must show that the vulnerability's exploitation was reasonably foreseeable. The analysis in the Article shows that the foreseeability of exploitation of a security vulnerability depends on complex technical features of the vulnerability such as authentication, access complexity, and root access. This may raise the following difficult question: is vulnerability A, which provides root access but with greater access complexity than vulnerability B, more foreseeably exploitable than vulnerability B, which provides greater ease of exploitation and less access complexity but no remote access? The Article argues that these and similar issues require a translation between law and technology that speaks the language of a novel technological environment, yet preserves the meaning and policy rationale of the traditional doctrines. It demonstrates how the translation must be done and why lawyers need to understand it.[32]

The Article is organized as follows: Section 2 discusses the principles of the major enabling technologies, namely malevolent code, technical defenses against malevolent attacks (subsection 2.1), and computer security vulnerabilities (subsection 2.2). Section 3 analyzes

[30] *See generally* Bunting v. Hogsett, 21 A. 31 (Pa. 1891).

[31] *See generally* Borel v. Fibreboard Paper Prod. Corp., 493 F.2d 1076, 1105 (5th Cir. 1973) (Scope of asbestos manufacturer's duty to warn includes foreseeable dangers related to exposure to asbestosis, mesothelioma, and other cancers); Jenny Steele & Nick Wikeley, *Dust on the Streets and Liability for Environmental Cancers*, 60 MODERN L. REV. 265, 268 (1997).

[32] *See* LESSIG, *supra* note 4, at 109 ("Just as a language translator constructs a text that is different from the source but has the same meaning as the source, so too does the constitutional translator construct an application that, though different from the original application, has the same meaning in the current context as the original did in its context.").

liability issues related to information security, focusing on the interaction between law and technology. Following Section 3, the Article concludes.

2. ENABLING TECHNOLOGIES

An attack on a target typically consists of the phases of reconnaissance, penetration, and cover-up.[33] During the reconnaissance phase, the attacker collects intelligence about the target system and its vulnerabilities. It may use a worm programmed to scan the Internet for a vulnerable remote system and penetrate it by injecting copies of itself directly into the memory of the system.[34] During and after the attack, the attacker may take action to avoid detection by, for instance, installing a program known as a rootkit.[35]

Technology plays a key role in all three phases of a cyber attack. The key technologies involved in a cyber attack can be illustrated by a description of the infamous Internet worm known as W32/CodeRed ("CodeRed").[36] CodeRed was released in July 2001 with devastating consequences. It spread faster than any prior worm[37] and infected over 250,000 systems in less than nine hours,[38] causing damage estimated at $2.6 billion.[39]

[33] *See* Chen and David, *supra* note 3 at 532.

[34] *See e.g.* SZOR, *supra* note 8, at 398-401.

[35] *See* PFLEEGER & PFLEEGER, *supra* note 5, at 145 (Describing a rootkit as "a piece of malicious code that goes to great lengths not to be discovered or, if discovered and removed, to reestablish itself whenever possible.").

[36] *See* JAKOBSSON & RAMZAN, *supra* note 11, at 458-60 (Describing the stages of a typical cyber attack.).

[37] *See* H. Berghal, *The Code Red Worm*, 44 COMM. ACM 15-19 (Dec. 2001) (A news release by the Federal Bureau of Investigation proclaimed that "on July 19, the CodeRed worm infected more than 250,000 systems in just nine hours. This spread has the potential to disrupt business and personal use of the Internet for applications such as e-commerce, e-mail and entertainment. [Indeed,] the CodeRed worm struck faster than any other worm in Internet history.").

[38] *See* ED SKOUDIS & LENNY ZELTSER, MALWARE: FIGHTING MALICIOUS CODE 78 (2004).

[39] *See* MALWARE DETECTION IX (Mihai Christodorescu et al. eds., 2007).

The CodeRed worm carried multiple destructive properties and propagated by exploiting a so-called "buffer overflow" vulnerability[40] in Microsoft's Internet Information Services (IIS) web servers to penetrate and launch attacks on vulnerable systems.[41] It located potential targets by scanning the Internet for vulnerable systems, to which it propagated automatically.[42] Once CodeRed infected a machine, it verified 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 an exploitable vulnerability and continued the infection cycle. Between the twentieth and twenty-eighth of every month, the worm proceeded to launch a denial of service attack on the official White House web page. The worm remained dormant between the twenty-eighth day and the end of the month.[43] It is estimated that CodeRed infected one in eight of the six million Web servers with the particular vulnerability it was designed to exploit. Microsoft issued software tools, known as security patches,[44] to remedy the vulnerability exploited by CodeRed, but many vulnerable servers remained unpatched.[45]

This section reviews the key technologies at work in a targeted attack such as CodeRed, namely malevolent code, the computer security vulnerabilities it typically exploits, and technical defenses against malevolent attacks.

[40] *See* infra § 2.2 (Discussion of the buffer overflow vulnerability.).

[41] *See* SZOR, *supra* note 8, at 98.

[42] See *id.* at 398-401.

[43] PFLEEGER & PFLEEGER, *supra* note 5, at 137-9.

[44] A "patch" is a software update overlaid on an existing program in order to fix a vulnerability in the program. A patch is usually a temporary remedy, to be used until a more permanent remediation of the vulnerability becomes available. *See* ROBERT SLADE, DICTIONARY OF INFORMATION SECURITY 139 (2006).

[45] *See* CERT, *A Very Real and Present Threat to the Internet: Resurgence in Code Red Scanning Activity* (August 1, 2001), http://www.cert.org/archive/html/coderedannounce.html (last visited March 27, 2011). The CodeRed attacks occurred shortly after Microsoft had discovered the vulnerability and issued a patch to fix it.

2.1 PRINCIPLES OF MALEVOLENT CODE

Malevolent software is a generic term for computer code that is designed to disrupt the operation of a computer system. The most common of these rogue programs are the computer virus and its variant, the worm. Other forms of malevolent software include so-called "logic bombs,"[46] Trojan horses,[47] and trap doors.[48]

COMPUTER VIRUSES

The term "virus," Latin for "poison," was first formally defined by Dr. Fred Cohen in 1983,[49] even though the concept originated in John von Neumann's studies of self-replicating mathematical automata in the 1940s.[50] A computer virus is a series of instructions (a program) that: (i) infects a host program by attaching itself to the host, (ii) executes when a triggering condition is satisfied, or when an executing host transfers control to the viral code, and (iii) spreads by cloning itself, or part of itself, and attaching the new versions to other host programs. In addition, many viruses have a so-called "payload" capable of harmful side effects, such as deleting, stealing, or modifying digital information.[51] As the definition suggests, a typical computer virus consists of three basic modules or mechanisms, namely an infection module, payload trigger, and payload.

[46] A logic bomb is "a section of code, preprogrammed into a larger program, that waits for a trigger event to perform a harmful function. Logic bombs do not reproduce and are therefore not viral, but a virus may contain a logic bomb as a payload." *See* SZOR, *supra* note 8, at 30.

[47] A Trojan horse is a program that appears to be beneficial, but contains a harmful payload. *See* SZOR, *supra* note 8, at 663; SLADE, *supra* note 44, at 188.

[48] A trapdoor, or backdoor, is a function built into a program or system to allow unauthorized access to the system. SZOR, *supra* note 8, at 643.

[49] Fred Cohen, Computer Viruses (1985) (unpublished Ph.D. dissertation, University of Southern California) (on file with the University of Southern California library).

[50] *See e.g.* RICK LEHTINEN et al., COMPUTER SECURITY BASICS 83 (2nd ed., 2006).

[51] *See* FREDERICK B. COHEN, A SHORT COURSE ON COMPUTER VIRUSES 1-2 (2nd ed., 1994); PFLEEGER & PFLEEGER, *supra* note 5, at 117-22.

INFECTION MODULE

The infection module is the most salient technical property of a computer virus.[52] It enables a virus to reproduce and spread, by locating a prospective host program and installing a copy of the virus onto it.[53] When the host program runs, control is eventually passed to the resident virus code, allowing it to execute. The executing virus repeats the infection cycle by automatically replicating itself and copying the newly created clones to other executable host files on the system or network, and even across networks.[54]

A virus may infect a computer or a network through several possible points of entry, including via a downloaded infected file, through web browsing, through removable media such as writable compact disks and DVDs, via infected files in shared directories, via an infected e-mail attachment, and even hidden in infected commercial shrink-wrapped software.[55]

E-mail is the most widely used medium of exchanging files and sharing information, but it has also become a convenient and efficient vehicle for virus and worm propagation.[56] Fast-spreading viruses such as ExploreZip and Melissa exploit automatic mailing programs to spread within and across networks.[57] Melissa would arrive in the e-mail inbox of its victim, disguised as an e-mail message with a Microsoft Word attachment. When the recipient opened the

[52] *See* LANCE J. HOFFMAN , ROGUE PROGRAMS: VIRUSES, WORMS, TROJAN HORSES 247 (1990); J.A. Morales et al., *Characterization of Virus Replication*, 4 J. COMPUTER VIROLOGY 221, 221-22 (2008) (Describing replication as the "only characteristic guaranteed to be consistently present in all viruses,") and ("A malware classified as a virus under the accepted definitions implies the virus has the ability to replicate."); DAVID HARLEY ET AL., VIRUSES REVEALED 87 (2001) ("The first and only necessary part of the structure [of a virus] is the infection mechanism. This is the code that allows the virus to reproduce, and thus to be a virus.").

[53] *See* DENNING, *supra* note 9, at 81.

[54] *See* SKOUDIS & ZELTSER, *supra* note 38, at 31-37.

[55] *See* DENNING, *supra* note 9, at 81.

[56] ICSA LABS, 10TH ANNUAL COMPUTER VIRUS PREVALENCE SURVEY 15 (2004), http://craigchamberlain.com/library/malware/ICSA%20Labs%2010th%20Annual%20Virus%20Prevalence%20Survey%202004.pdf (last visited March 27, 2011) (demonstrating popularity of e-mail as medium of virus attack).

[57] *See e.g.* A. Bisset and G. Shipton, *Some Human Dimensions of Computer Virus Creation and Infection*, 52 INT'L J. HUM. COMPUTER STUD 899 (2000); R. Ford, *No Surprises in Melissa Land*, 18 COMPUTERS AND SECURITY, 300-302 (1999).

attachment, Melissa executed. First, it verified 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 user of the infected system 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 it to fifty more computer users. Melissa attacks frequently escalated and resulted in clogged e-mail servers and system crashes.[58]

PAYLOAD

In addition to replicating and spreading, viruses may be programmed to perform specific harmful actions. The module that implements this functionality is known as the payload.[59] A payload can perform a wide range of functions, depending on the objectives of the virus author.[60] A payload can be programmed to perform destructive operations such as corrupting, deleting, and stealing information.[61] A payload may also create intrusion-enabling devices such as a backdoor[62] that allow unauthorized access to the infected machine.[63] Some payload effects are immediately obvious, such as a system crash, while others are subtle, such as transposition of numbers and alteration of decimal places.[64] Subtle effects tend to be dangerous because their

[58] HARLEY, ET AL., *supra* note 52, at 406-10.

[59] *See e.g.* JAN HRUSKA, COMPUTER VIRUSES AND ANTI-VIRUS WARFARE 17-18 (1990).

[60] *See* NICHOLAS WEAVER et al., *A Taxonomy of Computer Worms, in* PROCEEDINGS OF THE 2003 ACM WORKSHOP ON RAPID MALCODE 11 (2003) ("The payload is limited only by the imagination of the attacker.").

[61] *See* Meiring de Villiers, *Computer Viruses and Civil Liability: A Conceptual Framework*, 40 TORT TRIAL & INS. PRAC. L.J., 123, 172 (2004) (Discussion of damage due to virus infection).

[62] A backdoor is a method of gaining remote access to a computer without passing through normal security controls on a system. *See* SLADE, *supra* note 44, at 19.

[63] SKOUDIS & ZELTSER, *supra* note 38, at 27.

[64] SZOR, *supra* note 8, at 302-03 (Describing "data diddlers" as viruses that "do not destroy data all of a sudden in a very evident form, ... but slowly manipulate the data, such as the content of the hard disk").

presence may not be detected until substantial harm has been done. Payloads are often relatively harmless and do no more than entertain the user with a humorous message, musical tune, or graphical display.[65]

The payload is triggered when a specific condition is satisfied. Triggering conditions come in a variety of forms, such as a specified number of infections, a certain date, or specific time. The Friday-the-13th virus, for instance, only activated its payload on dates with the cursed designation.[66] The 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.[67] In the simplest case, a payload executes whenever the virus executes, without a special triggering event. Viruses do not always have a payload module, but even viruses without a payload may harm their environment by consuming valuable computing resources, by for instance, consuming available memory space and slowing the execution of important programs.

VIRUS DETECTION

Technical anti-virus defenses come in four varieties, namely signature scanners, activity monitors, integrity checkers, and heuristic techniques.[68] Scanners detect specific, known viruses by indentifying patterns that are unique to a particular virus strain. Activity monitors look out for virus-like activity in a computer. Integrity checkers sound an alarm when detecting suspicious

[65] *See* E.J. Sinrod & W.P. Reilly, *Cyber-Crimes: A Practical Approach to the Application of Federal Computer Crimes Laws*, 16 SANTA CLARA COMPUTER & HIGH TECH. L.J. 117, 218 (describing the W95.LoveSong.998 virus, designed to trigger a romantic song on a particular date).

[66] *Id.* at 217, n. 176.

[67] S*ee* Meiring de Villiers, *Free Radicals in Cyberspace: Complex Liability Issues in Information Warfare*, 4 NW. TECH & INTELL. PROP. L.J. 13, 13-14 (2005).

[68] Szor, *supra* note 8, Ch. 11.

modifications to computer files. Heuristic techniques detect viruses by analyzing a program's structure and logic.

SCANNERS

Scanners are the most widely used anti-virus defense. A scanner reads executable programs and searches for the presence of virus patterns, known as "signatures." A virus signature consists of patterns of hexadecimal digits embedded in viral code that are unique to a particular virus strain.[69] These signatures are created by human experts at institutions such as IBM's High Integrity Computing Laboratory, who scrutinize viral code and extract sections of code with unusual patterns.[70] The selected byte patterns are collected in a signature database and used in anti-virus scanners. A scanner detects a virus in a program by comparing the program to its database of signatures, and announces a match as a possible virus.[71] When a scanner detects a known virus signature, it informs the user and deactivates the virus.[72]

An ideal virus signature should generate neither false negatives nor false positives.[73] Although this ideal is unachievable in practice, anti-virus researchers pursue optimal solutions within practical constraints. The IBM High Integrity Computing Laboratory, for instance, has developed an optimal statistical signature extraction technique that examines all sections of code

[69] *See* HRUSKA, *supra* note 59, at 42; PFLEEGER & PFLEEGER, *supra* note 34, at 124 ("[A] virus executes in a particular way, using certain methods to spread. Each of these characteristics yields a telltale pattern, called a signature that can be found by a program that looks for it.").

[70] *See* SKOUDIS & ZELTSER*, supra* note 38, at 53-4.

[71] Jeffrey O. Kephart et al., *Automatic Extraction of Computer Virus Signatures*, *in* PROCEEDINGS OF THE 4TH VIRUS BULLETIN, INTERNATIONAL CONFERENCE (1994), http://www.research.ibm.com/antivirus/SciPapers/Kephart/VB94/vb94.html (last visited March 27, 2011).

[72] *See* PFLEEGER & PFLEEGER, *supra* note 34, at 124.

[73] *See* Jeffrey O. Kephart et al., *Blueprint for a Computer Immune System*, Virus Bulletin International Conference (1997) *reprinted in* IBM THOMAS J. WATSON RESEARCH CENTER REP., http://www.research.ibm.com/antivirus/SciPapers/Kephart/VB97/ (last visited March 27, 2011).

in a virus, and selects the byte strings that optimize the tradeoff between false positives and negatives.[74]

Scanners are easy to use, but they are limited to detecting known signatures. Furthermore, a scanner's signature database has to be continually updated as new viruses are discovered and their signatures catalogued, a burdensome requirement in an environment where new viruses appear daily. Modern antivirus vendors have attempted to lighten the burden on users by distributing signature updates directly to their customers via the Internet.[75] As the number of known viruses grows, the scanning process will inevitably slow down, as an ever-increasing set of possibilities has to be evaluated.

ACTIVITY MONITORS

Activity monitors are resident programs that monitor activities in a computer for behavior commonly associated with viruses. Suspicious activities include operations such as attempts by a program to delete information and mass mail copies of itself. When suspicious activity is detected, the monitor may simply halt execution and alert the user, or take definite action to neutralize the activity.[76] Activity monitors, unlike scanners, do not need to know the signature of a virus to detect it. Their function is to recognize generic suspicious behavior, not the precise identity of the culprit.

The greatest strength of activity monitors is their ability to detect unknown virus strains, but activity monitors also have significant weaknesses. They can only detect viruses that are actually executing, potentially after substantial harm has been done. A virus may, furthermore,

[74] *See supra* note 71.

[75] *See* SKOUDIS & ZELTSER, *supra* note 38, at 54.

[76] Sandeep Kumar and Eugene H. Spafford, *A Generic Virus Scanner in C++ in* PROCEEDINGS OF THE 8TH COMPUTER SECURITY APPLICATIONS CONFERENCE 3-4 (1992).

execute before the monitor code does, and do harm before it is detected. A virus may also be programmed to alter monitor code on machines that do not have protection against such modification.

The effectiveness of activity monitors is limited by the lack of unambiguous rules defining "suspicious" activity. This ambiguity may result in false alarms when an activity monitor picks up legitimate activities resembling virus-like behavior.[77] False negatives may result when an activity monitor fails to recognize viral activity not fitting the monitor's programmed definitions.[78] Recurrent false alarms may ultimately lead users to ignore warnings from the monitor.

INTEGRITY VERIFICATION

An integrity verifier applies the electronic equivalent of a tamper-proof seal to protected programs, and issues an alert when the seal has been broken, presumably due to virus intrusion. An integrity verification program generates a code, known as a "checksum," for protected files. A checksum may be an arithmetic calculation based on variables such as the total number of bytes in a file, the numerical value of the file size and its creation date. A checksum is periodically recomputed and compared to the original. When a virus infects a file, it usually modifies the contents, resulting in a change in the checksum. When the recomputed value does not match the original, it is presumed that the file has been modified since the previous inspection, and an alert is issued.[79]

[77] *See* ROBERT SLADE, ROBERT SLADE'S GUIDE TO COMPUTER VIRUSES 40-41 (2d ed., 1996).
[78] *See* HRUSKA, *supra* note 59, at 75.
[79] *See* SKOUDIS & ZELTSER, *supra* note 38, at 58.

The advantage of integrity checking is that it detects most instances of viral infection, as infection usually alters the target file. Its main drawback is its tendency to generate a false alarm when a file changes for "legitimate" reasons unrelated to virus infection.[80] Integrity checking software therefore presents a high likelihood of false positives because of the general difficulty of determining whether a program modification is legitimate or due to a virus.[81] Integrity checking works best on static files, such as system utilities, but it is, of course, an inappropriate technique for files that naturally change frequently, such as word processing documents.

HEURISTIC DETECTION

Modern virus detectors increasingly use heuristic methods. Heuristic rules solve complex problems "fairly well" and "fairly quickly," but less than perfectly.[82] Virus detection is an example of a complex problem that is amenable to heuristic solution. It has been proven that it is mathematically impossible to write a virus detection program that is capable of consistent perfect detection.[83] Heuristic virus detection methods accept such limitations and attempt to achieve a heuristic solution, namely a detection rate that is below the (unachievable) perfect rate, but representing an optimal tradeoff between detection accuracy, speed, and computational expense.[84]

[80] PHILIP FRITES et al., THE COMPUTER VIRUS CRISIS 125 (2d ed., 1992); SKOUDIS & ZELTSER, *supra* note 38, at 58.

[81] SLADE *supra* note 77, at 157.

[82] *See* SLADE, *supra* note 44, at 52 (Describing "heuristic" as relating to "shortcuts to solutions on the basis of a 'best guess'").

[83] Diomidis Spinellis, *Reliable Identification of Bounded-Length Viruses is NP-Complete*, 49 IEEE TRANSACTIONS ON INFORMATION THEORY 280, 282 (2003) (Stating that theoretically perfect detection is, in the general case undecidable, and for known viruses, NP-complete.); David Chess & Steve White, *An Undetectable Computer Virus*, IBM THOMAS J. WATSON RESEARCH CENTER REP., http://www.research.ibm.com/antivirus/SciPapers/VB2000DC.htm (last visited March 27, 2011).

[84] *See* Carey Nachenberg, *Future Imperfect*, VIRUS BULLETIN at 6 (August 1997).

Heuristics detect novel viruses by examining the structure and logic of executable code for evidence of virus-like behavior. The heuristic program then assesses the likelihood that a scrutinized program constitutes a virus by calculating a score based on the number and type of virus-like characteristics detected. If the score exceeds a certain threshold, the scanner classifies the program as malevolent code, and notifies the user. Instructions to send an e-mail message with an attachment to every listing in an address book, for instance, would add significantly to the score. Other high-scoring routines include the capability to replicate, hide from detection, and execute some kind of payload.[85]

A heuristic scanner typically operates in two phases. The scanning algorithm first narrows the search by identifying the location most likely to contain a virus. It then analyzes code from that location to determine its likely behavior upon execution. A static heuristic scanner compares the code from the "most likely" location to a database of byte sequences commonly associated with virus-like behavior. The algorithm then decides whether to classify the code as viral.[86] A dynamic heuristic scanner uses central processing unit ("CPU")[87] emulation.[88] It loads suspect code into a virtual computer, emulates its execution, and monitors its behavior. Because it is only a virtual computer, virus-like behavior can be safely observed in what is essentially a laboratory setting, with no need to be concerned about real damage.[89] Although dynamic heuristics can be time-consuming due to the computationally intensive CPU emulation process, they are sometimes superior to static heuristics. This will be the case when

[85] *See* SZOR, *supra* note 8, at 472-74.

[86] Kumar & Spafford, *supra* note 76, at 4-5.

[87] The CPU of a computer is responsible for data processing and computation.

[88] *See* HRUSKA, *supra* note 59, at 115; DAVID BENDER, COMPUTER LAW: EVIDENCE AND PROCEDURE § 2-7, 9 (1982).

[89] Kumar & Spafford, *supra* note 76, at 4; Nachenberg, *supra* note 13, at 50-51.

the suspect code is obscure and not easily recognizable as viral in its static state, but clearly reveals its viral nature in a dynamic state.

A heuristic assessment is less than perfect by design and will inevitably provide false positives and negatives. A low threshold will result in false positives. A scanner with a threshold that is set too high, on the other hand, will fail to detect viruses that are malicious but that do not exactly match the unrealistically tight specifications, resulting in false negatives.[90] As in the case of activity monitors, the term "suspicious" is ambiguous. Many legitimate programs, including even some anti-virus programs, perform operations that resemble virus-like behavior.[91] Nevertheless, state-of-the-art heuristic scanners have a 70-80 percent success rate of detecting unknown viruses.[92]

A major advantage of heuristic scanning over generic anti-virus technologies such as behavior monitoring and integrity checking is its ability to detect viruses before they execute and cause harm. Its advantage over conventional signature scanners lies in its capability to detect novel virus strains whose signatures have not yet been catalogued. Heuristic scanners are also capable of detecting polymorphic viruses, a complex virus family which complicates detection by changing their signatures from infection to infection.[93]

[90] SKOUDIS & ZELTSER, *supra* note 38, at 56; Symantec, *Understanding Heuristics: Symantec's Bloodhound Technology in* 34 SYMANTEC WHITE PAPER SERIES at 9 (1997), http://www.symantec.com/avcenter/reference/heuristc.pdf (last visited March 27, 2011) ("If [the behavior analysis component of a heuristic scanner] is too stringent in its requirements, it will have difficulty detecting a significant number of viruses. On the other hand, if the anti-virus programmer designs the analysis component to be too lenient in its behavioral requirements, the anti-virus product may be overly susceptible to false identifications.").

[91] Francisco Fernandez, *Heuristic Engines*, Proc. 11th Intl. Virus Bulletin Conference, September 2001, Virus Bulletin Ltd., Abingdon, England, 1994, at 409.

[92] Nachenberg, *supra* note 84, at 7; Symantec, *Understanding Heuristics: Symantec's Bloodhound Technology in* 34 SYMANTEC WHITE PAPER SERIES at 9 (1997), http://www.symantec.com/avcenter/reference/heuristc.pdf (last visited March 27, 2011).

[93] 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* DENNING, *supra* note 9, at 89.

The explosive growth in new virus strains has made reliable detection and identification of individual strains very difficult and costly, making heuristics more important and increasingly more prevalent.[94] Commercial heuristic scanners include IBM's AntiVirus boot scanner and Symantec's Bloodhound technology.[95]

COMPUTER WORMS

Worms are similar to viruses, but differ in two important respects. Worms propagate autonomously across networks without human intervention, and they replicate and spread without attaching themselves to a host program.[96] The CodeRed worm, for instance, propagated by injecting copies of itself directly into the memory of a remote system by exploiting a security vulnerability in the target system. It located potential targets by scanning the Internet for vulnerable systems, to which it propagated automatically.[97] The typical virus, in contrast, needs to attach itself to an executable file, and then relies on human interaction to propagate across networks. Like viruses, worms may carry destructive payloads, but even without a destructive payload a fast-spreading worm can do significant harm by slowing down a system through the prolific network traffic it generates.[98]

[94] Nachenberg, *supra* note 84, at 9.

[95] *See generally,* Symantec, *Understanding Heuristics: Symantec's Bloodhound Technology in* 34 SYMANTEC WHITE PAPER SERIES (1997), http://www.symantec.com/avcenter/reference/heuristc.pdf (last visited March 27, 2011).

[96] *See* United States v. Robert Tappan Morris, 928 F.2d 504 (1991) (Defining a “worm” as "a program that travels from one computer to another but does not attach itself to the operating system of the computer it infects. It differs from a virus, which is also a migrating program, but one that attaches itself to the operating system of any computer it enters and can infect any other computer that uses files from the infected computer."); Nicholas Weaver et al., *A Taxonomy of Computer Worms* 11-18, 2003 ACM WORKSHOP ON RAPID MALCODE, (Defining a worm as "a program that self-propagates across a network exploiting security or policy flaws in widely used services.").

[97] *See* SZOR, *supra* note 8, at 398-401.

[98] *See* John F. Schoch & Jon A. Hupp, *The "Worm" Programs - Early Experience with a Distributed Computation*, 25 COMM. ACM 172 (1982).

Scientists at Xerox PARC implemented the original worm in 1978,[99] but the so-called Morris Worm, created in 1989 by Cornell University graduate student, Robert T. Morris, was the first to become a household name.[100] The 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 computer security vulnerabilities, such as the buffer overflow flaw that enabled the Morris worm to paralyze the Internet.[101]

GENERIC STRUCTURE OF A WORM

A typical worm consists of the following basic components:

1. Activation mechanism,
2. Target selection algorithm and scanning engine,
3. Warhead,
4. Propagation engine, and
5. Payload. [102]

The activation mechanism triggers execution of the worm on the target computer. The target selection algorithm identifies new potential targets, and the scanning engine narrows down the selection by identifying the vulnerable subset. The warhead penetrates the target, paving the way for the propagation engine to move the worm body to the target. The payload, if present, is programmed to cause harm such as launching a denial of service attack.[103]

[99] Parc, *Xerox PARC History, Worm*, http://www.parc.com/about/ (last visited March 27, 2011).

[100] *See* HARLEY ET AL., *supra* note 52, at 347-52.

[101] Takanen et al., *Running Malicious Code By Buffer Overflows: A Survey of Publicly Available Exploits*, *in* 2000 EUROPEAN INSTITUTE FOR COMPUTER ANTI-VIRUS RESEARCH BEST PAPER PROCEEDINGS 162 (2000), http://www.itsec.gov.cn/docs/20090507160032879038.pdf (last visited March 27, 2011) ("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.").

[102] *See* SKOUDIS & ZELTSER, *supra* note 37, at 79-80.

[103] A DoS attack aims to deprive legitimate users of a resource or service provided by a system by overloading the system with a flood of data packets, thus preventing it from processing legitimate requests. *See generally* Meiring de Villiers, *Distributed Denial of Service: Law, Technology & Policy*, 39 WORLD JURIS LAW. TECH. J. 1 (2006).

ACTIVATION MECHANISM

A worm may be activated in a number of different ways. The Melissa worm relied on a tantalizing message to persuade a user to open an e-mail attachment that launched the worm.[104] Worms such as Iloveyou[105] and Benjamin[106] employed similar tactics. The CodeRed worm self-activated by automatically searching for and exploiting network vulnerabilities it used to inject itself into the memory of a target server.[107]

TARGET SELECTION ALGORITHM AND SCANNING ENGINE

A worm needs to locate new targets in order to continue spreading. The target selection algorithm of a worm selects Internet Protocol ("IP") addresses[108] of potential targets, and a scanning algorithm determines whether the computer at the selected address contains an exploitable vulnerability.[109]

The most basic target selection algorithm chooses an IP address at random. The fast-spreading Slammer worm, for instance, generated random IP addresses and sent a packet to each

[104] The user was tempted with a subject line such as "*Here is that document you asked for ... don't show anyone else ;-)*." When the recipient opened the attachment, Melissa executed. CERT, *Cert Advisory CA-1999-04 Melissa Macro Virus* (March 31, 1999), http://www.cert.org/advisories/CA-1999-04.html (last visited March 27, 2011). *See also* HARLEY ET AL., *supra* note 52 at 406-10. Other worm infection propagators include e-mail attachment inserters, instant messaging attacks, and SMTP attacks. *See* SZOR, *supra* note 8, at 331-38 (2005).

[105] Cert, *CERT Advisory CA-2000-04 Love Letter Worm Virus* (May 9, 2000), http://www.cert.org/advisories/CA-2000-04.html (last visited March 27, 2011).

[106] Symantec, W32.Benjamin.Worm (February 13, 2007), http://securityresponse.symantec.com/avcenter/venc/data/w32.benjamin.worm.html (last visited March 27, 2011).

[107] *See* SZOR, *supra* note 8, at 315.

[108] Each computer on the Internet is uniquely identified by its IP address. *See e.g.* FRED T. HOFSTETTER, INTERNET TECHNOLOGIES AT WORK 19 (2004) ("Every computer on the Internet has a unique Internet Protocol (IP) address. Each packet of information that gets transmitted over the Internet contains the IP address of the computer that sent it and the IP address of the computer to which it is being sent.").

[109] *See* SKOUDIS & ZELTSER, *supra* note 38, at 84-87; JOSE NAZARIO, DEFENSES AND STRATEGIES AGAINST INTERNET WORMS 14 (2004).

target, without first verifying the validity of the IP address.[110] More sophisticated worms, such as CodeRed, are programmed to scan a network for vulnerable IP addresses and "fingerprint" a remote system to ascertain its vulnerability.[111] E-mail worms such as W97M/Melissa@mm read e-mail addresses on a system and mail copies of themselves to each address.[112] Worms may also harvest e-mail addresses from a mail server or DNS server, or use search engines to harvest addresses on the Internet.[113]

WARHEAD

The first step towards taking over a target computer is gaining access to the target machine. A worm accomplishes this through its warhead, typically by exploiting a vulnerability in the target system. Commonly employed penetration techniques include buffer overflow exploits, e-mail penetration, file sharing, and backdoor attacks.

Buffer overflow: The buffer overflow is currently (and has been for over a decade) the most commonly exploited vulnerability to get unauthorized access to a system.[114] A buffer overflow vulnerability allows executable malevolent code to be copied into the memory of a target computer, and executed remotely.

E-mail penetration: E-mail is a popular penetration technique. E-mail worms transmit themselves to a target via an executable infected e-mail attachment. Sophisticated e-mail worms,

[110] David Moore et al., *Inside the Slammer Worm*, 2003 IEEE SECURITY & PRIVACY (4) 33 (July-Aug. 2003).

[111] *See* SZOR, *supra* note 8, at 315.

[112] The W97M/Melissa@mm worm propagated itself widely by exploiting the Microsoft Outlook e-mail program. SZOR, *supra* note 8, at 319, 334.

[113] *Id.* at 319-24.

[114] *See* Eric Chien & Peter Szor, *Blended Attacks Exploits, Vulnerabilities and Buffer-Overflow Techniques in Computer Viruses*, VIRUS BULLETIN (Sept. 2002) *reprinted in* Symantec Security Response, *Blended Attacks Exploits, Vulnerabilities and Buffer-Overflow Techniques in Computer Viruses (*2003) at 3, http://www.symantec.com/avcenter/reference/blended.attacks.pdf (last visited March 27, 2011). The buffer overflow is discussed in § 2.2.

such as W32/Nimda.A@mm, are programmed to activate automatically when an infected e-mail message is read or merely previewed.[115]

File sharing techniques: Some viruses and worms propagate through file-sharing mechanisms, such as the peer-to-peer (P2P) services, Gnutella and Kazaa.[116] Each member ("peer") of a P2P network maintains a shared folder with files made available to other peers for downloading. Files that are exchanged over a P2P network may be infected with malevolent code capable of infecting a user's computer when downloaded and opened.[117]

Propagation through backdoor interfaces: Some worms use backdoor interfaces to propagate themselves. A backdoor is a software or hardware mechanism that can be used to gain remote access to a computer without passing through normal security controls.[118] An attacker can exploit a backdoor to take control of a system and compromise sensitive information.[119] A backdoor may, for instance, consist of a password recognition routine installed in the computer as a modification of a legitimate program. The routine would enable a hacker who provided the correct password to gain access to confidential files and programs on the computer.[120] Worms that utilize backdoor interfaces include the Nimda worm, which took advantage of a backdoor

[115] *See* SZOR, *supra* note 8, at 414-15.

[116] *See* SKOUDIS, *supra* note 38, at 51; Kim Zetter, *Kazaa Delivers More Than Tunes*, WIRED (Jan., 2004), http://www.wired.com/techbiz/media/news/2004/01/61852 (last visited March 27, 2011) (Reporting that 44 percent of executable files downloaded through a Kazaa client application are infected by malicious code).

[117] Some authors prefer the term "virus" rather than "worm" for malevolent code that spreads via a P2P network, because of its reliance on human intervention. *See* Seungwon Shin et al., *Malware Prevalence in the KaZaA File-Sharing Network* 3 n.4 (Oct. 25, 2006) (unpublished), *available at* http://conferences.sigcomm.org/imc/2006/papers/p34-shin.pdf ("We use the term 'virus,' as opposed to 'worm,' to refer to a malicious program spreading in a P2P network since it requires human intervention.").

[118] *See* SLADE, *supra* note 44, at 19-20.

[119] *See* SKOUDIS, *supra* note 38, at 190.

[120] *See* SZOR, *supra* note 8, at 309-11. *See also* J. Nazario et al., *The Future of Internet Worms*, BLACKHAT (2001) at 6, http://www.blackhat.com/presentations/bh-usa-01/JoseNazario/bh-usa-01-Joes-Nazario.pdf (last visited March 27, 2011).

opened by CodeRed. The W32/Borm worm used network scanning and fingerprinting techniques to locate backdoor-compromised systems.[121]

Propagation Engine

The propagation engine moves the body of the worm to the target. The warhead may for instance, execute a program such as the File Transfer Protocol,[122] to move the worm's code. The transported worm then installs itself on the machine, loads its code into memory and prepares to run on the system. An efficient worm carries its entire body of code within its warhead. E-mail worms such as the SQL Slammer, usually carry the entire body in the e-mail attachment.[123]

Payload

The payload is an optional but common component of computer worms. It consists of special code designed to achieve a specific aim of the attacker. The payload may, for instance, display a simple one-time message or graphic image. Some payloads perform more destructive acts such as deleting, stealing, or corrupting information. Payloads may also install spyware, disable anti-virus software, and open up a backdoor to allow remote access to an attacker.[124]

[121] *See* SZOR, *supra* note 8, at 331.

[122] File Transfer Protocol, or FTP, is a popular file transfer program used to move files across networks. *See* SLADE, *supra* note 44, at 84-5.

[123] *See* SKOUDIS, *supra* note 38, at 82-83.

[124] *See e.g.* HARLEY, ET. AL, *supra* note 52, at 88-89.

2.2 SECURITY VULNERABILITIES

Viruses and worms gain unauthorized access to information systems by exploiting security lapses, such as unpatched security vulnerabilities.[125] A security vulnerability is an error in an information system that an intruder can exploit to violate the system's security policy.[126] A system's security policy protects the confidentiality, integrity, and availability of information contained in the system, by controlling access to the system.[127] For instance, the information system of a bank may allow a customer to access her own account through login authentication, but restrict access to any other information. A vulnerability facilitates violation of such a security policy. For instance, a vulnerability such as a backdoor in an automated bank teller program may allow a wrongdoer to enter a special number on the keypad and capture sensitive information, such as personal identity ("PIN") numbers of previous users.[128]

[125] *See e.g.* N. Hanebutte and P.W. Oman, *Software Vulnerability Mitigation as a Proper Subset of Software Maintenance*, 17 J. OF SOFTWARE MAINTENANCE & EVOLUTION RES. & PRACTICE 379, 381 (2005) ("Cyber attacks take advantage of one or more vulnerabilities and can encompass several coordinated intrusions."); Ron Brandis, *Common System Security Risks and Vulnerabilities and How Malicious Attackers Exploit Them*, BRIDGEPOINT (2001), at 1, http://bridgepoint.com.au (last visited March 27, 2011). ("A few software vulnerabilities account for the majority of successful attacks because attackers are opportunistic - taking the easiest and most convenient route. ... They rely on organizations not fixing the problems (applying patches), and they often attack indiscriminately, by scanning the Internet for vulnerable systems."); Symantec, *Symantec Internet Security Threat Report, Trends for July 05-December 05* (March 2006), at 25, http://eval.symantec.com/mktginfo/enterprise/white_papers/ent-whitepaper_symantec_internet_security_threat_report_ix.pdf (last visited March 27, 2011) ("Attributing the success of an attack to "the high volume of computers running vulnerable software.").

[126] W. L. Fithen et al., *Formal Modeling of Vulnerability*, 8 BELL LABS TECH. J. 173, 174 (2004) (Defining a vulnerability as "an unplanned system feature that an intruder may exploit, if he/she can establish certain preconditions, to achieve particular impacts on that system that violate its security policy."); Symantec, *supra* note 125, at 45. (Defining vulnerabilities as "design or implementation errors in information systems that can result in a compromise of the confidentiality, integrity, and/or availability of information stored upon or transmitted over the affected system.").

[127] Confidentiality refers to the prevention of unauthorized access to sensitive information. Integrity refers to the protection of digital data from unauthorized change, such as corruption or deletion. Availability refers to procedures and safeguards ensuring that authorized users have access to information, when needed and in convenient format. *See e.g.* R. LEHTINEN ET AL., *supra* note 50, at 9-11; PFLEEGER & PFLEEGER, *supra* note 5, at 17-20; Fithen *supra* note 126, at 174.

[128] PFLEEGER & PFLEEGER, *supra* note 5, at 116.

The CodeRed worm infected insecure Web servers by exploiting a buffer overflow vulnerability[129] to inject itself into the server's memory. It then executed and propagated further by searching IP addresses for new vulnerable servers to infect.[130] CodeRed depended on the presence of this specific vulnerability to replicate and spread. More versatile worms, such as W32/Welchia, are capable of exploiting multiple vulnerabilities to invade any system on which at least one of the exploitable vulnerabilities is present.[131]

Vendors usually issue software patches promptly to fix newly discovered vulnerabilities. Users tend to delay implementing these patches, and vulnerabilities often remain susceptible to exploitation.[132] Successive generations of CodeRed plagued the Internet despite the fact that the attacks and the role played by the exploited vulnerability were widely publicized, and that a security patch to fix the vulnerability had been made available even before the first CodeRed attack.[133] Significant security risks are created if even a small number of computer users fail to implement a patch.[134]

[129] The buffer overflow vulnerability is discussed in the next subsection.

[130] *See* PFLEEGER & PFLEEGER, *supra* note 5, at 137.

[131] *See* SZOR, *supra* note 8, at 98.

[132] *See e.g.* Chen & David, *supra* note 3 at 533 ("The most critical vulnerabilities are often published by vendors along with a software patch. In practice, organizations find it hard to dedicate the time and effort needed to keep up regularly with security bulletins and patches. The time between the publication of a security vulnerability and the installation of patches leaves a window of opportunity for attackers to exploit that vulnerability.").

[133] *See* NAZARIO, *supra* note 109, at 98; Stephen Henderson & Matthew E. Yarbrough, *Suing the Insecure?: A Duty of Care in Cyberspace*, 32 N.M. L. REV. 11, 16 ([M]any computer systems are knowingly insecure, in part on account of failure to install readily available software patches); *See* CERT, *A Very Real and Present Threat to the Internet: Resurgence in Code Red Scanning Activity* (August 1, 2001), http://www.cert.org/archive/html/coderedannounce.html (last visited March 27, 2011).

[134] *See* George V. Hulme, *One Step Ahead*, INFORMATION WEEK (May 20, 2002, 12:00 AM), http://www.informationweek.com/news/development/tools/showArticle.jhtml?articleID=6502396 (last visited March 27, 2011) (Stating that 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 a vendor, the remaining unpatched systems could still allow enough hijackings to launch a denial of service attack on millions of other systems and networks.).

THE BUFFER OVERFLOW VULNERABILITY

The so-called buffer overflow has been exploited by worms such as the Morris worm, CodeRed, and many others, and is (currently) the most commonly exploited security vulnerability.[135] A buffer overflow vulnerability allows executable malevolent code to be copied directly into the memory of a target computer. A skillful attacker can then manipulate the invaded computer to remotely execute the injected code.

WHAT IS A BUFFER?

Buffers are data storage areas in computer memory with limited capacity. Buffers often function as temporary storage for data in transit between two devices operating at different speeds. The purpose of the temporary storage is to coordinate speed differentials between the devices. For instance, a mechanical printer is not capable of printing data at the speed at which it receives the data from a computer. A buffer in the interface between the computer and printer resolves this bottleneck, by receiving the data from the computer and temporarily storing it. The buffer then relays the information to the printer, at the printer's speed, while the computer is freed up to carry on with other tasks.[136]

A buffer overflow occurs when a program attempts to fill a buffer with more data than it was designed to hold.[137] The excess data typically overflow into adjacent memory locations

[135] *See* B. A. Kuperman et al., *Detection and Prevention of Stack Buffer Overflow Attacks*, 48 COMM. ACM. 51, 51 (November 2005) (Describing the buffer overflow vulnerability as "a security vulnerability that has been discussed for 40 years yet remains one of the most frequently reported types of remote attack against computer systems."); *See generally* SZOR, *supra* note 8, ch.10; Chien, *supra* note 114.

[136] WILLIAM S. DAVIS & T,M. RAJKUMAR, OPERATING SYSTEMS: A SYSTEMATIC VIEW 27-28 (2005).

[137] Mark E. Donaldson, *Inside the Buffer Overflow Attack: Mechanism, Method, & Prevention*, SANS INSTITUTE (April 3, 2002) at 3, http://www.sans.org/reading_room/whitepapers/securecode/buffer-overflow-attack-mechanism-method-prevention_386 (last visited March 27, 2011) (A buffer overflow is analogous to pouring ten ounces of

where it can corrupt existing data, possibly changing the instructions, resulting in unintended consequences. The "unintended consequences" may be relatively innocuous, but could be malicious by design. In a relatively benign scenario, the buffer overflow will merely cause the corrupted program to abort, without much further harm.[138] In a darker scenario, a buffer overflow could allow a hacker to remotely execute malicious code in a target computer. The next subsection describes how an attacker can exploit a buffer overflow vulnerability to achieve this objective.

EXPLOITATION OF A BUFFER OVERFLOW

A computer program consists of a set of instructions and a set of data on which the instructions operate. In the case of the common login procedure, for instance, the data consist of the user provided identity and password. The instructions of the login program parse the input data and authenticate the stated identity of the user by validating the password, if the password and identity match. If they do match, the user is allowed to access the system.

A program occupies a memory buffer consisting of a text segment and a stack segment. The text segment contains the program instructions, and the stack segment contains data.[139] A final instruction in the buffer contains the return address specifying the instruction to be executed next.

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.).

[138] A buffer overflow may, for instance, abort the application program, resulting in a segmentation fault and core dump. *See, e.g.,* RANDAL E. BRYANT ET AL., COMPUTER SYSTEMS: A PROGRAMMER'S PERSPECTIVE (2003).

[139] In the case of a login program, the text segment contains the instructions that examine a user's input to determine its validity. The stack segment stores the user input.

A program such as the login procedure that accepts external input is a possible entry point for an attacker's malicious code.[140] Instead of providing a valid user name, an attacker could enter characters representing malicious code. The computer would read the attacker's input into the stack segment in order to determine its validity. The attacker has, at this point, injected his malicious code into a memory buffer allocated to the currently active program, the login procedure. The attacker now needs to instruct the computer to execute the (malicious) contents of the buffer. It is done as follows:.

A carefully crafted attack strategy would not only fill the stack with malicious code, but also overflow the stack and overwrite the adjacent return address. The return address contains an instruction pointer specifying the instruction to be executed next. By overwriting this pointer with the address of the stack segment, the attacker effectively instructs the computer to execute the malicious content of the stack.

The attacker has now injected malicious code into memory and ensured that control will pass to the malicious code, and furthermore, that the code will execute at the privilege level of the running program. If, for instance, the running program has control over confidential information such as passwords and financial data, the malicious code inherits the same privilege.

In summary, the most basic elements of a buffer overflow attack are as follows:[141]

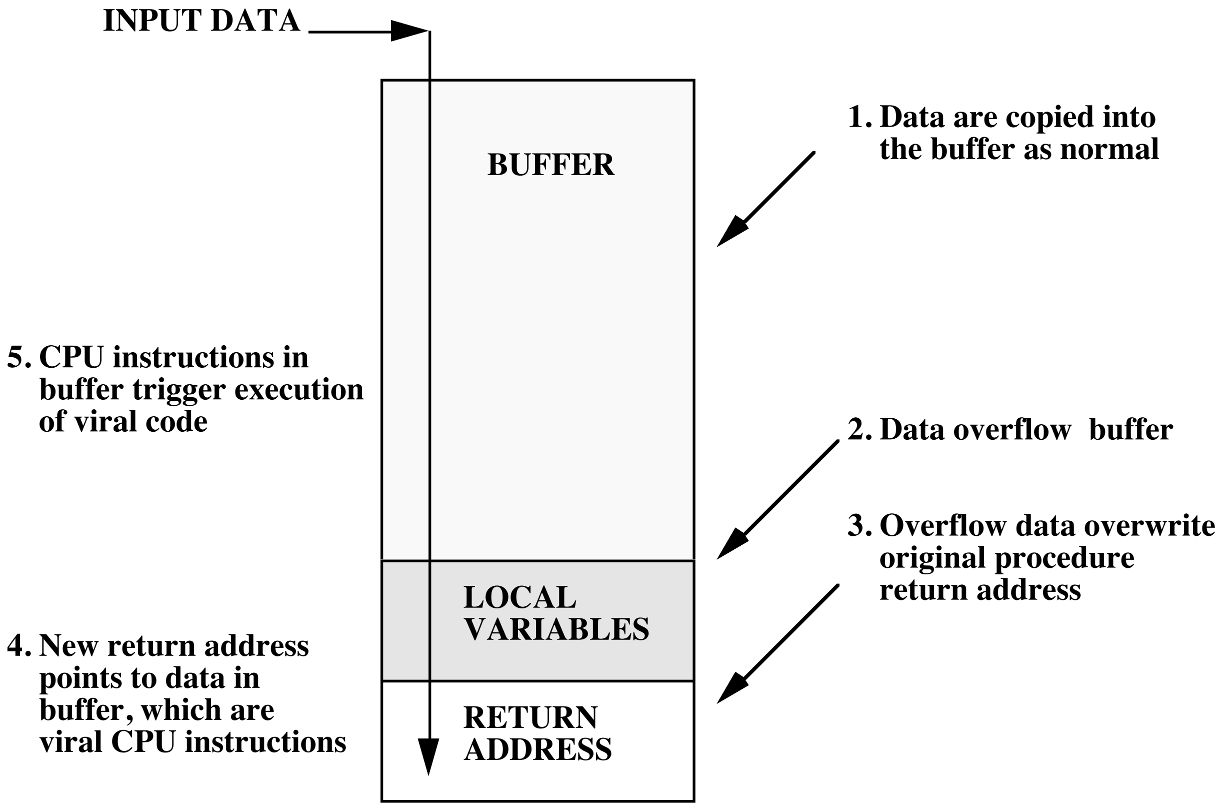
1. Data are copied into the buffer,
2. The data overflow the buffer,
3. The overflow data overwrite the original procedure return address,
4. The new return address now points to the new data in the buffer, which may be malevolent instructions, and
5. These instructions trigger execution of the virus.

[140] *See* SZOR, *supra* note 8, at 413. ("Computer worms typically attack service processes and daemon programs that are waiting to handle incoming requests by listening on various TCP/UDP ports. Any such communication service could potentially contain flaws, as did the Finger (in the case of the Morris worm), the BIND (in the case of the ADM worm), and the Microsoft IIS (in the case of the CodeRed worm).").

[141] This section has described the classic buffer overflow exploit mechanism. Buffer overflow exploits come in many varieties. *See* SZOR, *supra* note 8, at 365.

Schematically, the process is as follows:[142]

BASIC BUFFER OVERFLOW MECHANISM


INPUT DATA
BUFFER
LOCAL VARIABLES
RETURN ADDRESS
1. Data are copied into the buffer as normal
2. Data overflow buffer
3. Overflow data overwrite original procedure return address
4. New return address points to data in buffer, which are viral CPU instructions
5. CPU instructions in buffer trigger execution of viral code

3. Liability Issues

Businesses, non-profit organizations and government agencies may be held liable for failure to safeguard sensitive information in their possession.[143] A victim of an information security breach may pursue a civil action under a negligence theory against defendants who

[142] The diagram is adapted from Rob Enderle and Jasmine Noel, *The New Approach to Windows Security* Enderle Group (2004) at 7, http://www.amd.com/us/documents/new_approach_to_pc_securityfinal_(2).pdf (last visited March 27, 2011).

[143] *See* Meiring de Villiers, *supra* note 67, at § 3.3 (Analysis of civil liability for enablement of cyber crime and tort.); Michael L. Rustad and Thomas H. Koenig, *The Tort of Negligent Enablement of Cybercrime*, 20 BERKELEY TECH. L. J. 1553 (2005).

contributed to the risks associated with the breach, including those who failed in their duty to reduce or eliminate the risk.[144]

The plaintiff in a negligence action has to prove the following elements:[145]

1. A duty of care to prevent unreasonable risks of harm,
2. A breach of duty,
3. A causal connection between the defendant's conduct and the plaintiff's harm,[146] and
4. Actual damage resulting from the defendant's negligence.[147]

Negligence liability of a defendant depends first and foremost on the existence of a duty of care to the plaintiff.[148] A duty of care may be imposed by common law tort principles.[149] A

[144] *See* Michael L. Rustad, *Private Enforcement of Cybercrime on the Electronic Frontier*, 11 S. CAL. INTERDISC. L.J. 63, 66; Robin A. Brooks, *Deterring the Spread of Viruses Online: Can Tort Law Tighten the Net?*, 17 REV. LIT. 343 (1998); 18 U.S.C. 1030(g) (2006) (Provision in Computer Fraud and Abuse Act allowing civil action against wrongdoer "to obtain compensatory damages and injunctive relief or other equitable relief."); DAN B. DOBBS, THE LAW OF TORTS 258 (2000) (The plaintiff can assert that *any* conduct counts as negligence.); Prosser and Keeton, *supra* note 25, at § 31; RESTATEMENT (SECOND) OF TORTS § 282 (1965) (Defining negligence as a breach of the duty not to impose an unreasonable risk on society.); James A. Henderson, *Why Negligence Law Dominates Tort*, 50 UCLA L. REV. 377, 405 (2003) (Describing negligence as the dominant theory of liability in the law of torts).

[145] *See* David G. Owen, *Idea: The Five Elements of Negligence*, 35 HOFSTRA L. REV. 1671, 1671 (2007).

[146] This element includes actual, as well as proximate cause. A defendant's negligence is the actual cause of the plaintiff's harm if, but for the breach the harm would not have occurred. *See* de Villiers, *supra* note 61, at 141-42. (Analysis of causality in context of virus attack). The proximate causation element requires the defendant's conduct to be reasonably related to the plaintiff's harm. *See* Owen, *supra* note 145, at 1681. (Defining proximate cause as "a reasonably close connection between a defendant's wrong and the plaintiff's injury, a connection that is not remote.").

[147] *See* de Villiers, *supra* note 61, at 172-74. (Discussion of damage due to virus infection).

[148] *See* David G. Owen, *Figuring Foreseeability*, 44 WAKE FOREST L. REV. 1277, 1301 (2009) ("Every negligence claim must pass through the duty portal that bounds the scope of tort recovery for accidental harm.").

[149] *See e.g.* Johnson & Gunn, *supra* note 21, at 272-82; Michael J. Rustad & Thomas H. Koening, *Extending Learned Hand's Negligence Formula to Information Security Breaches*, 3 I/S: J.L. & POL'Y FOR INFO. SOC'Y 237, 239-40 (2007) ("[C]ompanies have a duty to provide reasonable information security practices under the common law of torts.").

duty may also be imposed by statute, either expressly,[150] or by legal precedent, if the statute does not expressly provide for civil liability.[151]

"Breach of duty" refers to a violation of the duty to avoid unreasonable risks of harm to others. Common law rules require the plaintiff to identify and plead an untaken precaution that would have prevented the accident, if taken. The defendant will then be held to be in breach if the benefits of risk reduction provided by the pleaded precaution exceeded its cost.[152] The plaintiff must further show that the precaution is technically feasible; namely that there were reasonable practical means by which it could have been implemented.[153] The victim of an information security breach may select among security precautions such as a better firewall, an intrusion detection system or superior anti-virus technology.

The breach calculus weighs the cost of the untaken precaution against the value of the reduction in all foreseeable risks that the precaution would have provided, not just the risk that actually materialized.[154] For instance, a buffer overflow vulnerability may enable a variety of cyber crimes, including identity theft, denial of service, and data corruption. A victim of identity theft enabled by a buffer overflow may assert that the defendant should have patched the

[150] *See e.g.* Cal. Civ. Code §1798.81.5 (West Supp. 2005) ("A business that owns or licenses personal information about a California resident shall implement and maintain reasonable security procedures and practices appropriate to the nature of the information, to protect the personal information from unauthorized access, destruction, use, modification, or disclosure.").

[151] *See e.g.* RESTATEMENT (THIRD) OF TORTS § 12 (Discussion Draft 1999) ("An actor is negligent if, without excuse, the actor violates a statute that is designed to protect against the type of accident the actor's conduct causes, and if the accident victim is within the class of persons the statute is designed to protect."); Johnson & Gunn, *supra* note 21, at 305-06.

[152] *See* Grady, *supra* note 27, at 143; Delisi v. St. Luke's Episcopal-Presbyterian Hosp., Inc., 701 S.W.2d 170 (Mo. Ct. App. 1985) (Plaintiff had to prove physician's breach of duty by specifying the treatment that should have been given, but was not).

[153] *See e.g.* Martin v. Michelin N. Am., Inc., 92 F.Supp. 2d 745, 754 (E.D. Tenn. 2000) (Defining "feasibility" in terms of technological capability and scientific state of the art at time of manufacture).

[154] *See* RESTATEMENT (SECOND) OF TORTS § 281(b), comment e (1965) ("Conduct is negligent because it tends to subject the interests of another to an unreasonable risk of harm. Such a risk may be made up of a number of different hazards, which frequently are of a more or less definite character. The actor's negligence lies in subjecting the other to the aggregate of such hazards."); Grady, *supra* note 27, at 146.

vulnerability in a timely fashion. To show breach, the plaintiff must balance the cost of eliminating the buffer overflow against the aggregate benefit of all risks so reduced, including risks related to denial of service and data corruption, not just the risk of identity theft.

The analysis of breach of duty can be illustrated by reference to the CodeRed attacks. CodeRed exploited a buffer overflow vulnerability in Windows IIS[155] for which a security patch had been issued by Microsoft, well before the first attack. An organization failing to implement the patch may be sued for negligently enabling the attack. A breach analysis would begin by considering an untaken precaution that would have avoided the CodeRed attack. A logical, and likely most effective precaution would be implementation of the Microsoft security patch. The precaution was feasible and, if implemented, would have prevented the attack.[156] After the appearance of the first version of CodeRed, a reasonably competent server manager knew, or should have been able to infer the potential harm from further exploitation of the vulnerability.[157]

The breach analysis weighs the cost of implementing the patch against its benefits of risk reduction. Although security patches are usually made available for free to users, implementing them may still be costly and technically challenging, especially in large corporations with complex systems. Patches interact with the systems to which they are applied, sometimes impairing their performance.[158] A proper cost-benefit analysis must also take into account the

[155] *See* R. Lemos, *Microsoft Reveals Web Server Hole,* CNETNEWS (June 18, 2001), http://news.com/2100-1001-268608.html (last visited March 27, 2011).

[156] *See generally,* Jeremy D. Baca, *Windows Remote Buffer Overflow Vulnerability and the Code Red Worm*, SANS INSTITUTE (September 10, 2001), http://www.sans.org/reading_room/whitepapers/malicious/windows-remote-buffer-overflow-vulnerability-code-red-worm_86 (last visited March 27, 2011) (Code Red could not infect a system that had the Microsoft MS01-033 patch installed).

[157] *See, id.* 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.").

[158] *See,* NAZARIO *supra* note 109, at 224 ("Patching puts the system at risk for downtime that can come from many sources. The first is that patches can change the software's behavior, on which other applications may depend. This can interrupt critical services. Secondly, patching takes time to implement, even if nothing goes wrong. Systems must be brought down and the patch applied."); SZOR, *supra* note 8, at 410.

fact that application of security patches must be optimally timed. Patching a vulnerability too soon may cause instability in the system due to a defective or incompatible patch. Delaying patching buys time to fix the defects, but leaves the system vulnerable to cyber attacks in the interim.[159] The defendant would be in breach of duty if the benefits of risk reduction of an optimally timed patch exceeded its total cost.[160]

The untaken precaution must satisfy all the elements of negligence. Failure to take the precaution must not only be a breach of duty, but also the actual as well as proximate cause of the plaintiff's harm. The untaken precaution therefore forms the basis of the plaintiff's case, and defines the standard of care that the defendant, the court hearing the case, and perhaps a subsequent appellate court will use.[161] The success of the untaken precaution as a pleading strategy depends on the technology involved in the case and the plaintiff's ability to translate traditional common law rules governing the untaken precaution into the language of its technological environment.

[159] *See,* Steve Beattie et al., *Timing the Application of Security Patches for Optimal Uptime*, 2002 LISA XVI 101, 101 (November 3-8 2002), *available at* http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBkQFjAA&url=http%3A%2F%2Fwww.homeport.org%2F~adam%2Ftime-to-patch-usenix-lisa02.pdf&rct=j&q=Steve%20Beattie%20et%20al.%2C%20Timing%20the%20Application%20of%20Security%20Patches%20for%20Optimal%20Uptime&ei=YCCRTYj5OMnA0QHhnLHKDg&usg=AFQjCNE1fIFEJ7yW8Rxq4RN3WN--46QlMQ&cad=rja (last visited March 27, 2011).

[160] For a detailed numerical example of the calculations involved in a cost/benefit analysis of an untaken precaution in information security see Meiring de Villiers, *Information Security Standards and Liability*, 13 J. INTERNET LAW 24, 28-29 (2010).

[161] *See* Grady, *supra* note 27, at 144.

FORESEEABILITY

Foreseeability is a key concept in negligence law.[162] Foreseeability defines whether the defendant owed a duty to the plaintiff,[163] whether the defendant breached a duty,[164] and whether the defendant's breach proximately caused the plaintiff's injury.[165] Courts have denied a duty based on absence of foreseeability, even where the defendant's conduct created a risk of harm.[166]

LEGAL MEANING OF FORESEEABILITY

An event is the foreseeable result of an action if the action created or heightened the risk of the event.[167] The basic test of foreseeability can also be described as "whether one can see a

[162] *See generally* PROSSER & KEETON *supra* note 25, at § 43 (discussing role of foreseeability in torts); W. J. Cardi, *Reconstructing Foreseeability*, 46 B.C. L. REV. 921, 921 (2005) ("The concept of foreseeability is fast devouring the negligence cause of action.").

[163] *See* W. Jonathan Cardi, *Purging Foreseeability: The New Vision of Duty and Judicial Power in the Proposed Restatement (Third) of Torts*, 58 VAND. L. REV. 739, 755 (2005) (Section 2. "Duty and Foreseeability"); Cardi, supra note 162, at 923 ("Foreseeability remains a pervasive consideration in many courts' duty analyses."); Arthur Ripstein, *Justice and Responsibility*, 17 CAN. J.L. & JURIS. 361, 374 (2004) ("Other factors may be relevant to the existence of a duty, but foreseeability provides an outer bound beyond which there can be no liability because there can be no duty."); Brennen v. City of Eugene, 591 P.2d 719, 723 (Or. 1979) (stating that a duty is created where the defendant "created a foreseeable risk of harm to others."); Greater Houston Transp. Co. v. Phillips, 801 S.W.2d 523, 525 (Tex. 1990) ("In determining whether the defendant was under a duty, the court will consider several interrelated factors . . . . Of all these factors, foreseeability of the risk is the foremost and dominant consideration.").

[164] *See* Cardi, *supra* note 162, at 921 ("Foreseeability of a risk has for centuries rested at the heart of court determinations of whether a defendant breached its duty of care."); Grady, *supra* note 27, at 146 ("The general rule is that any risk can be included in the [cost/benefit calculation of breach analysis] as long as it would be reduced by the untaken precaution in question and as long as it was foreseeable."); RESTATEMENT (THIRD) OF TORTS § 4 (Discussion Draft 1999) ("Primary factors to consider in ascertaining whether conduct lacks reasonable care are the foreseeable likelihood that it will result in harm, the foreseeable severity of the harm that may ensue, and the burden that would be borne by the actor and others if the actor takes precautions that eliminate or reduce the possibility of harm.").

[165] *See, e.g.*, MARC A. FRANKLIN & ROBERT L. RABIN, TORT LAW AND ALTERNATIVES 399 (7th ed. 2001); Mark F. Grady, *Proximate Cause Decoded*, 50 UCLA L. REV. 293 (2002); DOBBS, *supra* note 144, at 444 (stating that the proximate cause doctrine limits the plaintiff's recovery to harm reasonably related to the defendant's wrongdoing); Cardi, *supra* note 162, at 749 ("[A] plaintiff may fail to survive the proximate cause inquiry where the defendant's actions resulted in 1) an unforeseeable type of injury, 2) an injury occurring in an unforeseeable manner, or 3) injury to an unforeseeable plaintiff."); Walcott v. Total Petrl., Inc., 964 P.2d 609, 611 (Colo. App. 1998) ("[F]oreseeability is the touchstone of proximate cause.").

[166] *See, e.g.,* Herrera v. Quality Pontiac, 73 P.3d 181, 187 (N.M. 2003). *See also* Cardi, *supra* note 162, at 930 ("Foreseeability has become so central a concept in many courts' duty analyses that a ruling on foreseeability is outcome-determinative.").

[167] *See* Grady, *supra* note 165, at 323 (stating that, if plaintiff cannot show that the untaken precaution would have reduced the risk of the accident at issue, plaintiff fails on proximate cause grounds.); Stephen R. Perry, *Responsibility for Outcomes, Risk, and the Law of Torts*, *in* PHILOSOPHY AND THE LAW OF TORTS 72, 98 (Gerald

systematic relationship between the type of accident that the plaintiff suffered and ... the defendant's [wrongdoing]."[168]

In *Bunting v. Hogsett*,[169] the driver of a dinky train negligently failed to pay proper attention upon approaching a railroad intersection and, following a number of intervening events, collided with a passenger train at the intersection. Injured passengers on the passenger train filed suit against the owner of the dinky line, and the court held the defendant liable. The accident was a foreseeable consequence of the defendant's negligence and no tort or crime intervened between the defendant's negligent act and the plaintiff's injuries. The court correctly concluded that there is a systematic relationship between the driver's failure to keep a proper lookout and the type of accident and class of plaintiffs. The driver's wrongdoing created the exact risk that caused the plaintiff's injuries, namely the risk of a collision.[170]

Coincidental harm is, by definition, not systematically related to a defendant's wrongdoing, hence unforeseeable. Suppose, for instance, a motorist negligently exceeds the speed limit and arrives at a spot just in time to be struck by a falling tree. Although a plaintiff such as an injured passenger may argue credibly that falling trees are foreseeable, the accident is likely outside the scope of risk that is foreseeably created by the defendant's speeding. The defendant's speeding created risks of certain types of traffic accidents, but it did not create the risk that materialized. The accident was, therefore, unforeseeable.[171]

The outcome would likely have been different if instead a tree had fallen in front of the speeding driver, and the car collided with it. If it can be shown that the accident could have been

Postema ed., 2001) (explaining that proximate causation views the plaintiff's harm "from the standpoint of an appropriately general description of the risk created by the defendant.").

[168] *See* Grady, *supra* note 165, at 323.

[169] 21 A. 31 (Pa. 1891).

[170] *Id*., at 31. The case is much more interesting than this brief description reveals. It is discussed in more detail in Grady, *supra* note 165, at 304-5.

[171] See Berry v. Borough of Sugar Notch, 43 A. 240 (1899).

avoided had the driver travelled at a reasonable speed, the speeding driver's negligence may have been a proximate cause of the accident. Failure to stop within a short time window is a foreseeable risk of speeding.[172]

Foreseeability is not necessarily a reflection of the objective probability of an event. It is a reflection of what a reasonable person would foresee under the circumstances.[173] The reasonable person's foresight may be equal to the objective probability of the event, or it may be a fraction thereof.[174] This fraction may be zero if the defendant is reasonably ignorant of the systematic relationship between her wrongful act and the plaintiff's injury. This principle is formalized by the "Reasonable Ignorance of the Relationship" doctrine.[175] Under the doctrine, the defendant's liability is cut off when, even though ex post there is clearly a systematic relation between the untaken precaution and the harm, scientists could not have predicted the relationship ex ante. The following case illustrates the doctrine.

In *Doughty v. Turner Manufacturing Co.*,[176] foreseeability turned on scientific state of the art. A worker negligently knocked the cover of a vat containing molten sodium cyanide into the liquid in the vat. The plaintiffs were injured when a chemical reaction between the molten sodium cyanide and the cover, which was made of a combination of asbestos and cement known as sindayo, caused an eruption that resulted in injuries to the plaintiffs. The risk that the cover might splash the molten liquid onto bystanders was known and foreseeable, but the chemical reaction that actually caused the harm was unknown and unpredictable at the time of the

[172] See Grady, *supra* note 165, at 324.

[173] See Arthur Ripstein, EQUALITY, RESPONSIBILITY, AND THE LAW 94 (1999).

[174] *See* Stephen R. Perry, *Responsibility for Outcomes, Risk, and the Law of Torts*, *in* PHILOSOPHY AND THE LAW OF TORTS 72, 122 (Gerald Postema ed., 2001) ["[F]oreseeability is often referred to as epistemic (or knowable) probability."].

[175] *See* Grady, *supra* note 165, at 328.

[176] [1964] 1 Q.B. 518 (Eng.).

accident. Scientists later demonstrated that at sufficiently high temperatures the sindayo compound would undergo a chemical change creating steam. It was steam created in this manner that caused the eruption that injured the plaintiff in *Doughty*. None of this was known at the time of the accident. The court held for the defendant, stating that the defendant was reasonably ignorant of the chemical reaction that caused the injuries.[177] The defendant escaped liability under the Reasonable Ignorance doctrine.

The Reasonable Ignorance doctrine will likely play a significant role in the dynamic technological environment of information security law. The rapid development of virus technology has introduced an unpredictable element into the behavior of viruses. New virus creations often have the explicit goal of making detection more difficult and expensive.[178] Innovations with this goal in mind include stealth,[179] polymorphic, and metamorphic viruses.[180] Stealth viruses are designed to evade detection, and polymorphic and metamorphic viruses are programmed to change their nature and identity.

Unpredictable aspects of virus technology may cause a negligence action to fail on proximate cause grounds. In a particular virus incident, an ex post obvious systematic relation may exist between a defendant's wrongdoing and the harm caused by a novel virus. If, however, computer scientists could not ex ante predict this relation due to the novelty of the technology,

[177] Id., at 520, 525.

[178] *See*, Spinellis, *supra* note 83, at 281 ("Even early academic examples of viral code were cleverly engineered to hinder the detection of the virus."); Ken L. Thompson, *Reflections on Trusting Trust*, 27 COMMUNICATIONS OF THE ACM 761-763 (August 1984); Carey Nachenberg, *supra* note 13, at 46; CHRISTODORESCU *supra* note 39, at ix ("The goal of [a] malware writer (hacker) is to modify or morph their malware to evade detection by a malware detector.").

[179] Stealth virus strains are designed to evade detection by assuming the appearance of legitimate code when a scanner approaches. *See*, Kumar & Spafford *supra* note 76, at 78.

[180] Polymorphic viruses change their signature from infection to infection, making them harder to detect. Metamorphic viruses are capable of changing not only their identity, but also their entire nature and function. *See*, Carey Nachenberg, *Understanding and Managing Polymorphic Viruses*, XXX SYMANTEC ENTERPRISE PAPERS (1996), http://www.symantec.com/avcenter/reference/striker.pdf (last visited March 27, 2011); Spinellis, *supra* note 83, at 280 ("Viruses that employ these techniques, such as W32/Simile can be very difficult to identify.").

the plaintiff's case may fail under the Reasonable Ignorance doctrine. The following example illustrates such a situation:

Viruses can be roughly divided into two groups, namely those with a destructive payload, and those without a payload, or with a relatively harmless payload, such as display of a humorous message.[181] For the purposes of this example we refer to the two types as "harmful" and "harmless" viruses, respectively. Suppose a software vendor declines to scan for "harmless" viruses, perhaps to increase scanning speed and reduce costs, or because of a perceived low likelihood of liability exposure. The vendor purchases only signatures of new viruses that are known to be harmful at the time, for inclusion in its scanner database. The vendor then sells a software product containing a harmless virus strain, which by design was not detected. This virus infects the computer network of the purchaser of the infected program.

The virus happens to be a metamorphic virus with a capability of mutating into a different virus strain. In fact, it mutates into a strain with a malevolent payload capable of deleting data. The mutated strain, now a harmful virus, erases the hard disk of its host computer. In a negligence lawsuit by the purchaser of the infected software against the vendor, the vendor's liability would likely be cut off under the Reasonable Ignorance doctrine of proximate cause. Although it is clear after the incident that a systematic relation existed between the plaintiff's harm and the defendant's untaken precaution (failure to scan for harmless viruses), computer scientists were ex ante unaware of this relationship. The relationship originates from metamorphic technology and its ability to transform the nature and function of a virus. Scientists could not predict this relationship because the technology was unknown. The defendant's

[181] A. Bissett and G. Shipton, *Some Human Dimensions of Computer Virus Creation and Infection*, 52 INT. J. HUMAN-COMPUTER STUDIES 899, 903 (2000) ("Viruses may be classified as destructive or non-destructive in their primary effect. The least destructive ... simply print a ... message and then erase themselves. ... Destructive effects include halting a legitimate program. More destructive viruses erase or corrupt data or programs belonging to legitimate users of the computer.").

alleged wrongdoing was therefore not the proximate cause of the plaintiff's harm. As in *Doughty*, the risk that materialized was different in kind and degree of harm from the foreseeable risk of the defendant's conduct.[182]

FORESEEABILITY AND INFORMATION SECURITY

This subsection analyzes the doctrine of foreseeability in legal issues related to information security. It focuses on the paradigmatic case where a defendant, such as a financial institution or medical service provider, enabled the compromise of confidential information by failing to take reasonable security precautions against such risk. Professor Robert Rabin has termed wrongdoing of this kind an "enabling tort."[183] An enabling tort is a negligent act by a primary tortfeasor setting the stage for an intervening actor to commit a tort or crime.[184] For instance, a construction worker who negligently leaves an unattended scaffold beside an open window may be held liable for enabling the crime of a burglar who used the scaffold to break into the building.[185]

Professor Mark Grady has developed a theory explaining that a primary tortfeasor's liability will be preserved for conduct that foreseeably encouraged intervening actors who are so-called "free radicals."[186] Free radicals are individuals who are not deterred by the threat of liability, because they are judgment-proof, anonymous, immature, or strongly motivated by

[182] *See* Doughty v. Turner, [1964] 1 Q.B. 518, 526-27 (Lord Pearce) (Eng.). *See also id.* at 528-29 (Lord Justice Harman).

[183] *See* Robert L. Rabin, *Enabling Torts*, 49 DEPAUL L. REV. 435, 438 (1999).

[184] *See id.* at 438 (“Beyond the immediate perpetrator of harm, the victim perceives the individual, or more often, the enterprise, that set the stage for the suffering that unfolded. The Enabler.").

[185] *See* Stansbie v Troman [1948]2 K.B. 48. (Eng.).

[186] *See* Mark F. Grady, *The Free Radicals of Tort*, 11 SUP. CT. ECON. REV. 189 (2004). *See also* Rabin, *supra* note 183, at 439 ("The key factor counseling liability ... is that defendant paved the way for a truly reckless individual to be imposing serious risks of injury on the public at large.").

ideological considerations. Free radicals include persons such as the mentally incompetent, terrorists, and criminals.[187] The "Encourage Free Radicals" 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 individuals that cannot be deterred. Common law negligence rules therefore impose liability on a primary tortfeasor even when intentional or criminal behavior by a free radical intervenes.[188] Research has shown that cyber wrongdoers generally fit the profile of free radicals.[189]

"Reasonable foreseeability" is an essential element of enabling torts. A defendant cannot be held liable for enabling an intervening tort or crime unless the intervening wrongful behavior was foreseeable.[190] The common law pattern in enabling torts cases reveals particular factors that have convinced courts of the foreseeability of an intervening crime. In *Richardson v. Ham*,[191] the defendants left an unlocked bulldozer parked overnight in a public area. Three inebriated young men started one of the bulldozers, set it in motion, and then abandoned it, allowing the runaway bulldozer to plough through a residential area.[192] Plaintiffs who suffered injuries and property damage sued the owners of the bulldozer, alleging negligence in leaving the

[187] Grady, *supra* note 186, at 191.

[188] *See id.* at 196 ("When the encouraged people predictably lack exposure to tort law deterrence, the courts have concluded that more responsible people should be deterred from encouraging them."). *See also* Rabin, *supra* note 183, at 444 ("The main deterrence gap is the inability to effectively reach the putative wrongdoer himself, either through criminal or tort sanctions. This is the crux of the matter and the link to creating responsibility for enabling behavior.").

[189] *See* de Villiers, *supra* note 67, at 15-17.

[190] *See* Grady, *supra* note 186 at 214; Rabin, *supra* note 183 at 447. *See also* RESTATEMENT (THIRD) OF TORTS § 17 (Discussion Draft 1999) ("The conduct of a defendant can lack reasonable care insofar as it can foreseeably combine with or bring about the improper conduct of the plaintiff or a third party.").

[191] Richardson v. Ham, 285 P.2d 269 (Cal. Sup. Ct. 1955).

[192] *Id.* at 270.

machine unattended and unlocked.[193] In imposing a duty of care on the defendants, the court emphasized the factors that had made the intermeddling foreseeable.

The bulldozer was easily accessible to intermeddlers, as it had been left unlocked and unattended in a public area. A relatively unskilled person could start the machine and set the bulldozer in motion, even though he may not have the skill to stop it.[194] Once intermeddlers gained basic access to the bulldozer, nothing prevented them from starting the engine and setting it in motion. The bulldozer was fueled up, and the engine could be started with the bulldozer in gear by simply pushing in a lever and stepping on the starter. If so started, the bulldozer would be set in motion immediately. In addition, the bulldozer was left parked overnight, which provided intermeddlers with a significant time window of opportunity.[195] Bulldozers are relatively uncommon and present a greater attraction to intermeddlers than other vehicles. The public's fascination with bulldozers made unattended bulldozers a more attractive target than, for instance, an ordinary automobile left with its key in the ignition.[196]

The aim of this section is to identify and analyze the factors that make exploitation of a computer security vulnerability foreseeable. To do this, we must analyze the factors that make an opportunity foreseeably exploitable by criminals in real space cases such as *Richardson*, and develop analogies in information security.

Unauthenticated access: The intermeddlers had unauthenticated access to the bulldozer. "Unauthenticated access" refers to access without a valid key or identity validation.[197] In information security, authentication refers to procedures by which a computer system verifies the

[193] *Id.*

[194] *Id.* at 274.

[195] *Id.* at 270.

[196] *Id.*at 271.

[197] *See* Lessig, *supra* note 4, at 31 ("Authentication' is the process by which aspects of your identity becomes known.").

identity of a party from whom it has received a communication.[198] Examples of authentication procedures include cryptographic authentication of a digitally signed contract and biometric identification in applications such as Internet banking.[199] Network vulnerabilities, including some types of buffer overflow, may allow unauthorized individuals to access and remotely execute attack code in vulnerable systems.[200] A system with a security vulnerability that allows an authentication barrier to be bypassed is the cyber-analogue of an unlocked and unguarded vehicle in a public place, such as the bulldozer in *Richardson*.

Ease of exploitation: An opportunity that can be exploited conveniently and without significant effort and skill on the part of a wrongdoer may be characterized as "easy to exploit." The bulldozer in *Richardson* presented an easily exploitable opportunity, because an unskilled person could start the machine and set it in motion. A computer security vulnerability is considered easy to exploit if it requires minimal technical sophistication to leverage. This would be the case if ready-to-use exploit code to leverage the vulnerability were publicly available.[201]

[198] *See* Pfleeger, *supra* note 5, at 619.

[199] *See* Lessig, *supra* note 4, at 34-5; Pfleeger, *supra* note 5, at 219; Andy Jones and T. Martin*, Digital Forensics and the Issue of Identity*, 15 INFO. SEC. TECH. REP.67, 67 (May 2010).

[200] *See* Securiteam, *Microsoft NetDDE Service Unauthenticated Remote Buffer Overflow (MS04-031)* (Jan. 23, 2005), http://www.securiteam.com/windowsntfocus/5OP0K0UELY.html (last visited March 27, 2011) (reporting through a Securiteam Advisory, a vulnerability in the Microsoft DDE service that allows a remote attacker to execute arbitrary code on a system without authentication). Microsoft has released an update addressing the vulnerability. *See* Microsoft, *Microsoft Security Bulletin MS04-031* (October 12, 2004), http://www.microsoft.com/technet/security/bulletin/MS04-031.mspx (last visited March 27, 2011). *See also* DOROTHY E. DENNING, INFORMATION WARFARE AND SECURITY 214 (1999) (showing that hackers execute malicious code remotely, without logging in and providing a valid password, by exploiting buffer overflow vulnerabilities).

[201] *See* Symantec*, Symantec Internet Security Threat Report- Trends for July-December 06,* XI SYMANTEC ENTERPRISE SECURITY (March 2007) at 90, http://eval.symantec.com/mktginfo/enterprise/white_papers/ent-whitepaper_internet_security_threat_report_xi_03_2007.en-us.pdf (last visited March 27, 2011); JAMES C. FOSTER, ET AL, BUFFER OVERFLOW ATTACKS 10 (2005); Peter Mell et al., *A Complete Guide to Common Vulnerability Scoring System Version 2.0*, FORUM OF INCIDENT RESPONSE AND SECURITY TEAMS (June 2007) at 10, http://www.first.org/cvss/cvss-guide.pdf (last visited March 27, 2011) ("Public availability of easy-to-exploit code increases the number of potential attackers by including those who are unskilled ...").

Easily exploitable security vulnerabilities are attractive to attackers, thus foreseeably exploitable.[202]

Low access complexity: Opportunities that provide wrongdoers with basic access to their ultimate objectives but leave significant remaining barriers to their accomplishment are less attractive than opportunities that provide direct access. For instance, an unlocked door may provide basic access to valuables, but leave remaining impediments such as an alarm system that needs to be disabled, a safe that has to be cracked, or an inconveniently narrow time window of opportunity. Following the terminology of computer science, we define an access opportunity with significant remaining barriers as having "access complexity."[203]

The *Richardson* intermeddlers faced low access complexity. Once they were inside, achieving their ultimate goal was easy. The bulldozer was conveniently fueled up, and could be set in motion by simply pushing in a lever and stepping on a starter. In addition, the bulldozer was left parked overnight, which provided intermeddlers with a significant time window of opportunity.

A computer security vulnerability offers low access complexity if an attacker can gain basic access to a system and directly take control of the system at the desired level of privilege. Low access complexity is characterized by features such as a large time window of opportunity and an absence of technical complications beyond basic access. A vulnerability with high access

[202] *See* Symantec, *Symantec Security Update- June 2005 Worldwide and APAC* (June 2005) at 4, http://www.symantec.com/avcenter/reference/SSU_APAC_06_2005.pdf ("The threat of severe vulnerabilities is increased if and when an associated exploit is released publicly or if the vulnerability can be exploited trivially."); Mell *supra* note 201, at 10 ("Public availability of easy-to-exploit code increases the number of potential attackers by including those who are unskilled ..."); Chen and David, *supra* note 3, at 532 ("The ease of carrying out electronic attacks adds to the temptation for attackers.").

[203] *See* R. Chandramouli et al., *Common Vulnerability Scoring System*, 4 IEEE SECURITY AND PRIVACY 85 ,85-86 (Nov.-Dec. 2006).

complexity, in contrast, is characterized by specialized access conditions, such as a requirement of non-default technical conditions or special "cooperation" from the victim.[204]

Scarce opportunity: An opportunity may be considered "scarce" if it is aligned with the objectives of a wrongdoer, and if there are few equivalent alternative opportunities available. The unusual fascination of the bulldozer in *Richardson* made it a scarce opportunity to intermeddlers. A computer security vulnerability presents a scarce opportunity to an attacker who is using a virus or worm which is programmed to exploit unique features of the specific vulnerability.[205] The attractiveness of such a vulnerability over generic vulnerabilities is analogous to a bulldozer's superior fascination to the intermeddlers compared to ordinary vehicles. A vulnerability that presents a scarce opportunity is a relatively attractive target.

Root access: The objectives of cyber wrongdoers determine the degree of control and level of privilege they need in a target system. An intruder intent on stealing sensitive information from a financial institution, for instance, needs access to the institution's information

[204] *See id.*; Mike Schiffman, *The Common Vulnerability Scoring System*, RSA CONFERENCE (Feb. 2005), http://www.stanford.edu/~stinson/cs155/rdg/schiffman_cvss.pdf (last visited March 27, 2011). *See also* AUSCERT, *AUSCERT Security Bulletin ESB-2002.122, Microsoft Security Bulletin MS02-014, Unchecked Buffer in Windows Shell Could Lead to Code Execution,* AUSTRALIAN COMPUTER EMERGENCY RESPONSE TEAM (Mar. 11 2002), http://www.auscert.org.au/render.html?it=1779 (last visited March 27, 2011); Microsoft, Microsoft Security Bulletin, Unchecked Windows Shell Could Lead to Code Execution (MS02-014), Net-Security (March 7, 2002)http://www.net-security.org/advisory.php?id=593 (last visited March 27, 2011) (reporting a vulnerability in Microsoft Windows which was not remotely exploitable by default, but could be exploited via a Web page "if the user has installed an application with custom URL handlers and then uninstalled that application, and the uninstall failed to correctly remove the application completely." In addition to this fortuitous sequence of events, the attacker would have to create an HTML Web page according to narrow technical specifications in order to exploit the vulnerability. This vulnerability is therefore exploitable only under specific non-default conditions, a characteristic of high access complexity. It is therefore less foreseeably exploitable than an equivalent vulnerability without these complicating factors.).

[205] *See* Nicholas Weaver et al., *A Taxonomy of Computer Worms*, INTERNATIONAL COMPUTER SCIENCE INSTITUTE (2003) at 2, http://www.icir.org/vern/papers/taxonomy.pdf (last visited March 27, 2011) (certain worms are designed to exploit specific vulnerabilities, and are programmed to search and locate hosts that contain these vulnerabilities.); NAZARIO, *supra* note 109, at 39-41 (Describing the Morris worm and the vulnerabilities it exploited.); SZOR, *supra* note 8, at 395-97.

system at the privilege level of an administrator who has authority to access and transmit such information. This is usually the most privileged level of access, namely full root-level access.[206]

"Root" is the conventional name of the account of a superuser such as the system administrator. The superuser has privileges that an ordinary user does not have, such as authority to change the ownership of files, install and run programs, change Web Server databases, and modify system files.[207] An attacker who gains root access inherits these privileges. If a program is already running with root privileges, an appropriately configured vulnerability may be exploited to hijack the program and transfer root control to the attacker.[208] The attacker would then effectively become the administrator of the network or system. A vulnerability that allows this level of access is clearly attractive to a wrongdoer, thus foreseeably exploitable.

Remote access: A remotely exploitable vulnerability enables a user to execute commands on a target system across a network. A vulnerability that grants only local access requires physical access or a local account on the target system. An attacker with remote access may effectively take over a remote console and enter keystrokes and commands as if the attacker had local access. Some versions of the *rlogin* program for instance, contain a vulnerability that allows attackers to launch a remote attack on any vulnerable system connected to the Internet.[209] Vulnerabilities such as the buffer overflow are useful springboards to gain remote access to a

[206] *See* SLADE, *supra* note 44, at 158; BLUNDEN, *supra* note 4, at 8 ("[T]he system administrator's account (i.e., the user account with the least number of security restrictions) is often referred to as the root account.").

[207] *See* GREG HOGLUND & GARY MCGRAW, EXPLOITING SOFTWARE: HOW TO BREAK CODE 151-53 (2004).

[208] *See* DENNING, *supra* note 200, at 214 (Referring to malicious code executed via a buffer overflow as executing "with the privileges of the program it exploits, which is often root.").

[209] LAWRENCE R. ROGERS, *rlogin(1): The Untold Story*, CERT (1998) at 3 , 16, 23 http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBQQFjAA&url=http%3A%2F%2Fwww.cert.org%2Farchive%2Fpdf%2F98tr017.pdf&rct=j&q=LAWRENCE%20R.%20ROGERS%2C%20RLOGIN(1)%3A%20THE%20UNTOLD%20STORY%20&ei=0TCRTcbHJa680QHii-W8Dg&usg=AFQjCNHJTw1YVkJ9UnsGDnovF2QGQIc95w&cad=rja (last visited March 27, 2011); SZOR, *supra* note 8, at 341.

target system, because they allow an attacker to inject malevolent code directly into the execution path of a remote system.[210] Remotely exploitable buffer overflow vulnerabilities have recently been reported in well-known products, such as Sendmail, various Microsoft products, and ironically, PGP.[211]

A vulnerability which is not remotely exploitable offers limited scope for a virus or worm attack, and is therefore less likely to be exploited.[212] A recent Microsoft Security Bulletin advised of an image parsing vulnerability in Microsoft Office™ which allowed an attacker to install programs, create new accounts with full user rights, and change or delete data on a client workstation. This was a local vulnerability which was not remotely exploitable. The Security Bulletin predictably concluded that "we do not expect to see widespread exploitation of these vulnerabilities in current operating system versions."[213]

Remote access offers obvious advantages to an attacker, such as convenience, avoidance of the physical risks of on-site entry, and anonymity.[214] Courts have recognized that anonymity

[210] SZOR, *supra* note 8, at 309-11, 331, 368 ("Exploiting a buffer overflow makes it possible to inject arbitrary code into the execution path. This arbitrary code could allow remote system-level access, giving unauthorized access not only to malicious hackers, but also to replicating malware."); HARLEY, ET AL., *supra* note 52, at 74; Thomas Chen, *Trends in Viruses and Worms* (October 16, 2003), http://www.cisco.com/web/about/ac123/ac147/archived_issues/ipj_6-3/virus_trends.html (last visited March 27, 2011).

[211] *See* Symantec, *Sendmail Header Processing Buffer Overflow Vulnerability*, SYMANTEC SECURITY RESPONSE (2003), http://securityresponse.symantec.com/avcenter/security/Content/3.3.2003.html (last visited March 27, 2011) ("A remotely exploitable vulnerability has been discovered in Sendmail. This vulnerability is due to a buffer overflow condition in the SMTP header parsing component. Remote attackers may exploit this vulnerability by connecting to target SMTP servers and transmitting them malformed data.").

[212] *See* Scott Carpenter, *Vulnerability Management Solutions*, ITDEFENSE at 2 (April 2006). ("If the vulnerability is not remotely exploitable, the likelihood of a virus based on the vulnerability alone is minimized.").

[213] *See* Microsoft, *Microsoft Security Bulletin MS06-039* (July 11, 2006), http://www.microsoft.com/technet/security/bulletin/ms06-039.mspx (last visited March 27, 2011). *See also* Adobe, *Coldfusion Sandbox Security Vulnerability* (September 12, 2006), http://www.adobe.com/support/security/bulletins/apsb06-13.html (last visited March 27, 2011) (Reporting a vulnerability in Coldfusion software versions MX 7 and MX 7.01, which is not remotely exploitable.).

[214] Pfleeger, *supra* note 5, at 397 ("An attacker can mount an attack from thousands of miles away and never come into direct contact with the system, its administrators, or users. The potential attacker is thus safe behind an electronic shield."); Symantec, *supra* note 202, at 4 ("Remotely exploitable buffer overflow vulnerabilities are particularly dangerous, as skilled attackers can carry out exploitation without alerting a target user to the attack.").

enables wrongdoing,[215] a position supported by empirical research suggesting that cyber criminals are encouraged by anonymity-preserving technologies.[216] Geographic remoteness also complicates law enforcement efforts to establish jurisdiction, obtain extradition and procure evidence.[217]

A Foreseeability Metric

The analysis in this section has identified the technical features that make a computer security vulnerability foreseeably exploitable. To be useful to the legal analyst, the raw data must be processed into a usable format, ideally a single numerical metric that encapsulates the aggregate contribution of the features to foreseeability.

[215] *See* Guille v. Swan, 19 Johns. 381 (N.Y. 1822) (people who are otherwise perfectly rational may behave differently when they are shielded by the anonymity and diminished accountability of a crowd.); Religious Tech. Ctr. v. Netcom On-Line Comm. Servs., Inc., 923 F.Supp. 1231, 1255-57 (9th Cir. 1995) (cautioning that an anonymous or judgment proof defendant can do significant harm and leave the plaintiff without recourse).

[216] *See* Jelena Mirkovic et al, Internet Denial of Service: Attack and Defense Mechanisms 30 (2005). ("This disassociation and lack of physical proximity encourages people to participate in illegal activities in the Internet, such as hacking, denial of service, or collecting copyrighted material. They do not feel that in reality they are doing any serious harm."); J. R. Suler & W. Philips, *The Bad Boys of Cyberspace: Deviant behavior in Online Multimedia Communities and Strategies for Managing It*, Rider University (September 1997), http://www.rider.edu/~suler/psycyber/badboys.html (last visited March 27, 2011); J.P. Davis, *The Experience of Bad Behavior in Online Social Spaces: A Survey of Online Users*, Penn State University, http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.84.5025&rep=rep1&type=pdf (last visited March 27, 2011).

[217] *See* Stephen E. Henderson & Matthew E. Yarbrough, *Frontiers of Law: The Internet and Cyberspace: Suing the Insecure?: A Duty of Care in Cyberspace,* 32 N.M.L. Rev. 11, 17 (2002) ("In the case of a DDoS attack, the person who uses the weapon…is generally clearly liable, but that person is often either impossible to locate, is judgment-proof, or both."); Allen Householder et al., *Managing the Threat of Denial-of-Service Attacks*, CERT (October 2001) at 23, http://www.google.com/url?sa=t&source=web&cd=1&ved=0CBsQFjAA&url=http%3A%2F%2F98.15.203.119%2Fuploads%2FAzag%2FEbooks%2520Mega%2520Collection%2520-%2520Hacking-Coding-Exploiting-Phishing-HowTo%2FManaging%2520DoS.pdf&rct=j&q=A.%20Householder%2C%20Managing%20the%20Threat%20of%20Denial-of-Service%20Attacks&ei=7TSRTZizLpKK0QGl6q3FDg&usg=AFQjCNFeI4SGj2Xf4CrXRJh0afWGNWxpQg&cad=rja (last visited March 27, 2011) ("It is easy for attackers to avoid getting caught by hiding their identity. They command their attack network from stolen dial-up accounts and other compromised systems, and they use spoofed source addresses for attack traffic. Victim sites and law enforcement face a daunting and frequently unfeasible task to identify and prosecute attackers. Suffering few consequences - if any - for their actions, attackers continue their work.").

A recent article published in Hastings Communications and Entertainment Law Journal provides a metric that combines numerical values for each of the foreseeability attributes described in the previous section into a composite score that quantifies the foreseeability of exploitation of a vulnerability.[218] The numerical values chosen for the attributes are not arbitrary, but are consistent with industry standards developed by the U.S. National Infrastructure Assurance Council.[219] The values are also consistent with the legal objectives of the metric, in the sense that the numerical values associated with individual attributes increase with that attribute's contribution to foreseeability. A vulnerability for which there is no exploit code available, for instance, is assigned a relatively low value for "ease of exploitation." A vulnerability for which functional exploit code is publicly available for every exploitable incidence of the vulnerability is assigned the metric's maximum value for ease of exploitation.[220]

The following example illustrates the metric's ability to measure subtle differences in foreseeability between vulnerabilities. The example analyzes two vulnerabilities, both of which have been exploited in actual cyber attacks. They have very different foreseeability metrics, namely 3 and 10 out of 10, respectively. In a case where the former (less foreseeably exploitable) vulnerability was exploited, the plaintiff would be advised to select an untaken precaution other than patching this vulnerability. If the plaintiff were to plead this precaution, the defendant would have a powerful defense based on absence of foreseeability.

[218] *See* Meiring de Villiers, *Reasonable Foreseeability in Information Security Law: A Forensic Analysis*, 30 HASTINGS COMM. ENT. L. J. 419, 462, 472 (2008).

[219] *See* Mell, *supra* note 201, at 85. CVSS was developed by the U.S. National Infrastructure Assurance Council, a group of industry leaders who advise the US Department of Homeland Security on critical information infrastructure security. *See also* Chandramouli, *supra* note 203, at 85.

[220] *See* de Villers, *supra* note 218, at 462, 472.

This exercise demonstrates the power of a foreseeability metric that is both technically logical and consistent with common law rules. In a given case, application of such a metric may be outcome-determinative.

VULNERABILITY I: CVE:-2003-0818: Microsoft Windows ASN.1 Library Integer Handling Vulnerability.[221]

This is a buffer overflow vulnerability in Microsoft Windows which allows an attacker to execute arbitrary code with root privileges. It allows unauthenticated access and is remotely exploitable. The vulnerability offers the additional advantage of low access complexity, because no additional effort or special circumstances are necessary for a successful exploit. Functional exploit code is publicly available, making the vulnerability easy to exploit. Microsoft released a patch to remediate the vulnerability.

The vulnerability's multiple attractive properties (from an attacker's viewpoint) suggest that exploitation of this vulnerability is highly foreseeable. The foreseeability composite score based on these features, on a scale from 0 to 10, is calculated as 10/10.[222] The high foreseeability score is consistent with the vulnerability's high score on all the features that make it an attractive target, thus likely to be exploited.

VULNERABILITY II: CVE-2003-0062: Buffer Overflow in NOD32 Antivirus.[223]

In February 2003, this buffer overflow vulnerability was discovered in Linux and UNIX versions of NOD32, an antivirus application. The vulnerability allowed attackers to execute

[221] *See* National Vulnerability Database, *Vulnerability Summary for CVE-2003-0818*, (March 8, 2011), http://nvd.nist.gov/nvd.cfm?cvename=CVE-2003-0818 (last visited March 27, 2011).

[222] For details of the calculation, *see* de Villers, *supra* note 218, at 473-74.

[223] *See* National Vulnerability Database, *supra* note 221.

arbitrary code with the privileges of a user running NOD32. The vulnerability was not remotely exploitable and had significant access complexity. To execute malevolent code via the buffer overflow, an attacker had to wait for another user to scan a directory path of excessive length. If a user executed the scan, full compromise of the confidentiality, integrity, and availability of information on the target system was possible. Proof-of-concept level exploit code is available for the metric. Proof-of-concept code is a program with the sole purpose of verifying the existence of a vulnerability. It is not sufficiently functional to exploit the vulnerability.[224] The foreseeability score for this vulnerability is calculated as 3.0/10, a relatively low score.[225]

This vulnerability contains several features that make it unattractive to an attacker. In order to successfully exploit the vulnerability, a prospective attacker would have to develop exploit code beyond the available proof-of-concept. This requires a level of skill and effort that would limit the number of prospective exploiters. An attacker would furthermore have to contend with access complexity, such as having to wait for non-default conditions before a successful attack can be launched. An attacker would also have to work within the limitations of local access.

A cyber wrongdoer who contemplates designing a worm to exploit a specific vulnerability would likely prefer to target a "10" vulnerability such as Example I, rather than, for instance, a "3", such as Example II.

[224] *See* Ivan Arce, *The Quality of Exploit Code*, CSI Net Sec 2004 (2004) at 4, http://www.coresecurity.com (last visited March 27, 2011); Symantec *supra* note 201, at 90.

[225] For details of the calculation, *see* de Villers, *supra* note 218, at 473-74.

CONCLUSION

This article discusses the enabling technologies of cyber crime and analyzes their role in the resolution of legal issues related to cyber crime. Statutory regulation of information security has developed rapidly, but the common law remains influential in the interpretation of the statutes.[226] A statute such as California's Security Breach Information Act (SBIA),[227] for instance, provides that "[a] business that owns or licenses personal information about a California resident shall implement and maintain reasonable security procedures and practices appropriate to the nature of the information, to protect the personal information from unauthorized access, destruction, use, modification, or disclosure."[228] The provision is rooted in negligence law, because the essence of the negligence standard of care is a duty to avoid unreasonable risks of harm. Common law negligence doctrines such as reasonable foreseeability and proximate cause will therefore play a significant role in cases governed by statutes such as the SBIA.

Negligence law has been aptly described as a creature of technology. The elements of a negligence cause of action are defined in terms of technical specifications, and their analysis is shaped by technology. Common law rules require a plaintiff to plead an untaken precaution that is feasible, cost-effective, and capable of defeating the wrongdoer's attack technology. An outcome-determinative issue such as foreseeability of exploitation of a security vulnerability

[226] *See* Susan Freiwald, *Electronic Surveillance at the Virtual Border*, 78 MISS. L. J. 329, 330-31 (2008) (Stating that "lawmakers have drafted numerous laws designed specifically to address new problems arising in cyberspace ... [and] courts have interpreted pre-cyber statutes in new cyberspace contexts and have extended common law precedents into cyberspace settings."); Orin S. Kerr, *Cybercrime's Scope: Interpreting "Access" and "Authorization" in Computer Misuse Statutes*, 78 N.Y.U. L. REV. 1596, 1596 (2003) (Stating that "[t]he federal government, all fifty states, and dozens of foreign countries have enacted computer crime statutes that prohibit 'unauthorized access' to computers." The author offers "a normative proposal for interpreting 'access' and 'authorization'.").

[227] Act effective July 1, 2003, ch. 915, 2002 Cal. Legis. Serv. (West), available at CA LEGIS 915 (2002) (Westlaw).

[228] Cal. Civ. Code 1798.81.5 (West Supp. 2005).

depends on complex technical features of the vulnerability, such as authentication, access complexity, and root access.

Technological issues in negligence are trivially resolved in most real space cases. Did the building contractor leave a scaffold outside the building? Did the scaffold enable the burglary?[229] The technological issues in information security are more complex and require a deeper analysis. Would an optimally timed remedy to a security vulnerability have been cost-effective? Is vulnerability A, which provides root access but with greater access complexity than vulnerability B, more foreseeably exploitable than vulnerability B, which provides greater ease of exploitation and less access complexity, but no remote access? The resolution of these and similar issues requires a translation between law and technology that speaks the language of a novel technological environment, yet preserves the meaning and policy rationale of the traditional doctrines.

This translation, how it must be done, and why lawyers need to understand it is the focus of this Article. Lawyers who fail to understand it are doomed to pursue a suboptimal litigation strategy. A plaintiff who makes an uninformed selection of untaken precaution faces speculative recovery prospects, and a defendant who fails to analyze a key issue such as foreseeability at the proper level of detail, may overlook effective and potentially powerful defenses.

[229] *See* Stansbie v. Troman [1948] 2 K.B. 48 (Eng.).