Cyber Security: Of Heterogeneity and Autarky

Randal C. Picker[*]

The Internet is an almost-organic mix of actors and their machines, an eclectic scheme of government and private-decision-making, of non-profits and for-profits. As in any integrated system, my choices affect your life in a very direct way. So "Zombie PC Army Responsible for Big-Name Web Blackout" sounds like a headline from a bad Hollywood B-movie, when instead it means that computer users could not access the websites of Apple, Google, Microsoft and Yahoo because a Trojan horse program—which, by definition, had been placed surreptitiously on thousands of personal computers, turning those machines into zombie computers under the control of their cyber-master—launched a simultaneous attack on a key piece of the domain name system infrastructure.[1] Here we have perhaps one bad actor, thousands of sloppy computer users and externalities galore.

Taking down prominent websites is one way for a malicious computer programmer to seek fame (perhaps infamy), but spam provides perhaps a more meaningful way in which the day-to-day computer experience is degraded by our shared

---

 Paul and Theo Leffmann Professor of Commercial Law, The University of Chicago Law School and Senior Fellow, The Computation Institute of the University of Chicago and Argonne National Laboratory. This paper was given at the conference under the title "Raising Transaction Costs and Network Security: Of Heterogeneity and Autarky." I thank Ryan Foreman for able research assistance; Ira Rubinstein for comments; and the Paul Leffmann Fund, the Russell J. Parsons Faculty Research Fund and the John M. Olin Program in Law and Economics at The University of Chicago Law School for their generous research support, and through the Olin Program, Microsoft Corporation and Verizon.

[1] See Robert Lemos and Jim Hu, "Zombie PC army responsible for big name web blackout," CNET News.com, June 17, 2004 (available at http://software.silicon.com/malware/0,3800003100,39121439,00.htm).

network decisions. Some estimates suggest that 80% of spam arises from zombie machines.[2] Many of these are residential PCs with broadband hook ups. Why? This is the dark-side of Yochai Benkler's work on shareable goods.[3] From the consumer's perspective, both the PC and the broadband connection are shareable goods. Given the lumpiness of processing power, the average PC user has power to spare. This makes it easy for users to contribute computing cycles to seek extraterrestrial life and to other large-scale projects.[4] But, at the same time, excess cycles can be stolen with little obvious consequence to the computer owner. The consumer may experience no real loss when the evil cyber-master enslaves the consumer's PC to devote a chunk of the cycles and broadband connection to spam or a denial-of-service attack.[5]

But this is driven by more than excess cycles. The spam externality also has arisen from important changes in the way in which the network is organized. We moved from centralized processing power accessed through distributed dumb terminals (the 1970s) to distributed processing power in freestanding

---

[2] See "Trend analysis: Spam Trojans and their impact on broadband service providers," Sandvine Inc., June 2004 (available at http://www.sandvine.com/solutions/pdfs/spam_trojan_trend_analysis.pdf).

[3] See Yochai Benkler, "Sharing Nicely": On shareable goods and the emergence of sharing as a modality of economic production (forthcoming, Yale Law Journal, 2004); Yochai Benkler, Peer Production of Survivable Critical Infrastructures.

[4] Visit the SETI@home webpage to donate cycles (http://setiathome.ssl.berkeley.edu/).

[5] See The National Strategy to Secure Cyberspace, February, 2003, at 39 ("In recent years, with the spread of 'always on' connections for systems, such as cable modems, digital subscriber lines (DSL), and wireless and satellite systems, the security of home user and small business systems has become more important not only to the users themselves, but to others to which they are connected through the Internet.). See also "Scotland Yard and the case of the rent-a-zombies," CNET News.com, July 7, 2004 (describing rental of zombie networks—botnets—created by teenage hackers) (available at http://news.com.com/2102-7349_3-5260154.html).

PCs (the 1980s) to, with the rise of the Internet, highly-interconnected PCs (the 1990s). 1970s centralized processing was coupled with centralized control, a Soviet-style computer architecture. Users were eager to control their own destinies and the personal computer made that possible.

The freestanding PC world that supplanted centralized computing gave rise to few direct externalities, either positive or negative. Viruses could be spread through shared floppy disks, but the transactions costs of this type of inter-computer communication were sufficiently high that viruses didn't pose a particularly large problem. Plus a zombie PC wasn't even possible: even if the hacker could figure out how to get malicious software—malware—onto a floppy and from there to a particular PC, there was no easy way to get information or cycles back out. The hacker would have needed physical access to future floppies to get content out, a cumbersome arrangement.[6]

The rise of the networked PC changed this completely. Email and the web make the spread of viruses and bots easy, plus the hacker can initiate access at will to the infected machine. This has made the decentralized decisions of end-users much more salient. My failure to manage my computer appropriately puts you at risk.

All of this has made cyber-security increasingly important. The concept of cyber-security is sufficiently new that we should draw some lines of demarcation to understand what is at stake. Consider three categories that might be encompassed within the notion of cyber-security: cyber-vandalism, cyber-crime and cyber-terrorism. Offline vandals break windows and deface

[6] For lack of a generally accepted alternative, I use the term "hacker" to refer to a malicious computer programmer. I do realize that this use is a bone of contention in some parts of the computer community. For discussion, see Tony Bradley, "What Is In a Name?" (available at http://netsecurity.about.com/cs/generalsecurity/a/aa070303.htm).

buildings; online vandals—cyber-vandals—take down websites through denial-of-service attacks or deface websites by having an alternative webpage load. The Recording Industry Association of America is front-and-center in the record industry's effort to combat music downloading and that has made the RIAA's website a popular target.[7] Microsoft is frequently targeted as well.[8] Like its offline counterpart, cyber-vandalism can inflict real costs on its targets.

Cyber-crime is just crime over the Internet. So "phishing"—the cyber-criminal sends a fake email that appears to be from the recipient's financial institution seeking "reconfirmation" of financial information—is big business, with a 5 to 20% response rate that would make most marketers drool.[9] Congress recently made life harder for phishers in passing the Identity Theft Penalty Enhancement Act.[10] Other approaches to illicitly obtaining financial data seek to induce users to download software that sits in the background and records keystrokes, enabling the criminal to extract credit card information, passwords and the like.[11] The Computer Crime and Intellectual Property Section (CCIPS) of the Criminal Division of the U.S. Department of Justice focuses on these issues,

---

[7] See Declan McCullagh, "Recording industry site hit again," CNET News.com, September 3, 2002 (available at http://news.com.com/2102-1023_3-956398.html).

[8] See Jay Lyman, "Denial-of-Service Attack Brings Down Microsoft," TechNewsWorld, August 4, 2003 (available at http://www.technewsworld.com/story/31258.html).

[9] Christopher S. Stewart, Fighting Crime One Computer at a Time, The New York Times, June 10, 2004. For background, see U.S. Department of Justice, Criminal Division, Special Report on "Phishing" (available at http://www.usdoj.gov/criminal/fraud/Phishing.pdf).

[10] P.L. 108-275 (July 15, 2004).

[11] Kevin J. Delaney, "Web-Data Hackers Thwarted, But PCs Are Still Vulnerable," The Wall Street Journal, June 28, 2004.

though other parts of the federal government exercise authority as well.[12]

We might distinguish cyber-vandalism and cyber-crime from cyber-terrorism, even though these lines aren't particularly clean.[13] We should probably define terrorism before defining cyber-terrorism. The legislation creating the Department of Homeland Security defines separately both "terrorism" and "an act of terrorism." The "act of terrorism" definition focuses on any act that "uses or attempts to use instrumentalities, weapons or other methods designed or intended to cause mass destruction, injury or other loss to citizens of institutions of the United States."[14] A cyber version of that might overlap with notions of cyber-vandalism or cyber-crime. In contrast, the "terrorism" definition looks to acts directed at human life, critical infrastructure or key resources where the motive is political.[15] *The National Strategy to Secure Cyberspace*, issued by the White House in February, 2003, focuses on "threat[s] of organized cyber attacks capable of causing debilitating disruption to our

[12] Go to www.cybercrime.gov for info; see also U.S. Secret Service Press Release of September 11, 2003 (PUB 25-03), "United States Secret Service and Romanian Police Work Together to Solve Major Computer Fraud Investigation" (describing arrest of Romanian eBay phisher who defrauded Americans of $500,000) (available at http://www.secretservice.gov/press/pub2503.pdf).

[13] And might not even distinguish cyber-vandalism from cyber-crime. See Neal Katyal, The Dark Side of Private Ordering: The Network\Community Harm of Crime."

[14] The Homeland Security Act of 2002, P.L. 107-296 (Nov. 25, 2002), § 865; see also 6 CFR 25.9 (definition of "act of terrorism").

[15] Id at. § 2 ("appears to be intended—(i) to intimidate or coerce a civilian population; (ii) to influence the policy of a government by intimidation or coercion; or (iii) to affect the conduct of a government by mass destruction, assassination or kidnapping").

Nation's critical infrastructures, economy or national security."[16]

Within the U.S. Department of Homeland Security, the recently-created National Cyber Security Division focuses on the ways in which cyber-security implicates key infrastructure.[17] In January, 2004, the NCSD launched its National Cyber Alert System,[18] which builds on the prior work of the CERT Coordination Center at Carnegie-Mellon.[19] I grabbed at random a cyber-security bulletin: it opened with a 25-page list of software vulnerabilities identified between May 12, 2004 and May 25, 2004.[20] *Ten days, 25 pages*.

Who is on this list of infamy? To choose just a handful of prominent names: Apache, Apple, BEA Systems, Eudora, GNU, Hewlett Packard, KDE, the Linux kernel, Microsoft, Netscape, Novell, Opera, Sun and Symantec. This list covers both commercial and open-source software, companies with dominant and used-to-be dominant market positions, PCs and micro-computers. And it is not the sheer number of vulnerabilities alone that is problematic: the time between knowledge of the vulnerability and exploitation by a hacker is dropping, as hackers pursue the zero-day exploit (no gap between knowledge of the vulnerability and malware that exploits it).[21]

---

[16] The National Strategy to Secure Cyberspace, supra note 5, at viii.

[17] See "Ridge Creates New Division to Combat Cyber Threats," U.S. Dept. of Homeland Security, Press Release of June 6, 2003 (available at http://www.dhs.gov/dhspublic/display?theme=52&content=918).

[18] See "U.S. Department of Homeland Security Improves America's Cyber Security Preparedness – Unveils National Cyber Alert System," Press Release of Jan. 28, 2004 (available at http://www.us-cert.gov/press_room/cas-announced.html).

[19] See www.cert.org.

[20] US-CERT Cyber Security Bulletin, SB04-147, May 26, 2004 (available at http://www.us-cert.gov/cas/body/bulletins/SB04-147.pdf).

[21] David Rink, "Computer Worm is Turning Faster," Wall Street Journal, May 27,

We need to figure out how to deal with this systematic cyber-insecurity. The problem arises from underlying architecture of the system, as implemented in the joint decisions of hardware makers and software creators; from the malware creators themselves; and from the aggregate consequences of many individual decisions made by end-users. We have a number of possible targets and instruments to work with.

The hackers themselves are the most natural target, and we clearly will pursue them, but they can be quite elusive. We might consider end-users themselves. After all, their infected machines do much of the work of the system: take those machines off of the system and the hackers will be deprived of one of their most valuable resources. And end-users repeatedly create problems by clicking on executable email attachments. Think of end-users as engaging in negligent computer set-up or negligent computer use. In a parallel setting, the RIAA has sued consumers for copyright violations tied to uploading and downloading music. The RIAA switched to this approach after it was frustrated in its efforts to hold liable KaZaA and other creators of file-sharing software.[22]

Internet service providers are another natural target. As critical intermediaries in the network, they are operationally situated to intervene in the working of the network. The institutional structure matters too. The always-on, one-price all-you-can-eat structure for consumer broadband means that end-users pay no attention to how bandwidth is used. I have no reason to pay attention when a hacker turns my home broadband-enabled PC into a zombie.

2004.

[22] See John Borland, "RIAA sues 261 file swappers," CNET News.com, September 8, 2003 (available at http://news.com.com/2102-1023_3-5072564.html).

I will not consider the position of end-users or of internet service providers.[23] Instead, I want to consider two inquiries regarding how we manage cyber insecurity: (i) the monoculture argument, which favors forced heterogeneity in operating systems, and (ii) the ways in which liability rules influence software. First, the software adoption choices of individual users create a software infrastructure against which hackers operate. One prominent argument—dubbed the "monoculture" argument—suggests that the collective choice is flawed, even if the individual choices are perfectly sensible. These individual choices have led to a Microsoft Windows monopoly in personal computer operating systems. According to the claim, the Microsoft operating system monopoly creates a harmful monoculture—a common code base through which computer viruses spread easily putting the computing network at risk.[24]

I consider the monoculture argument's focus on forced heterogeneity as a means of creating redundancy in our integrated computer network. I believe that forced heterogeneity would be quite expensive and that we would be better suited to focus on autarky, meaning here the conditions under which individual computers or internal systems within a firm should be isolated from the rest of the public network. That is already a standard cyber-security practice, and one frequently associated with the management of critical assets. Heterogeneity and autarky are

---

[23] For views on the potential liability of the latter, see Doug Lichtman and Eric Posner, Holding Internet Service Providers Accountable.

[24] D. Geer et al, Cyber*In*security: The Cost of Monopoly (Sept. 24, 2003) (available at http://www.ccianet.org/papers/cyberinsecurity.pdf). For discussion, see Justin Pope, "Biology stirs Microsoft monoculture debate," Salon.com, Feb. 15, 2004; James A. Whittaker, "No Clear Answers on Monoculture Issues," IEEE Security & Privacy, Nov./Dec., 2003; "Warning: Microsoft 'Monoculture'", Associated Press, Feb. 15, 2004 (available at http://www.wired.com/news/privacy/0,1848,62307,00.html).

substitutes in pursuing redundancy, but I think that there is a decided advantage for autarky in protecting key assets.

Second, I consider the overall question of software quality, since the implicit (explicit?) premise of the monoculture work is not merely that we have *a* software monoculture, but that it is also a particularly bad one. I consider the way in which liability rules—understood generally to include as a family insurance (contractual liability), warranties (more contracts) and torts—might influence software quality. Full-blown liability would help solve a software adoption version of the prisoner's dilemma—each user wants the other user to adopt early and get the bugs out of the system—but would also introduce standard adverse selection problems. Voluntary contractual liability—a warranty to some customers—would mitigate those problems while permitting a natural improvement of software over time.

I. Redundancy: Heterogeneity vs. Autarky

Sometimes it seems it is almost impossible to pay too much attention to Microsoft. As perhaps the leading firm of the Information Age, Microsoft is everywhere, an unavoidable fact in the daily life of every computer user. Windows, Office and Internet Explorer are ubiquitous, with market shares to die for (and many competing products have done just that). Yes, Linux chips away on the desktop and cell phones grow more powerful each day, but for the foreseeable future—say the next decade—there is every reason to think that Microsoft will continue to define the computing experience of most users.

Monopoly inevitably brings attention. Good students of U.S. antitrust law understand that Judge Learned Hand's famous statement on monopoly—"The successful competitor, having been urged to compete, must not be turned upon when

he wins"[25]—is at best a half-truth. We will scrutinize winners to make absolutely sure that they dot their Sherman Act i's and cross their Clayton Act t's. I know less about European competition policy, but we all know that Microsoft has received the most exacting attention on both sides of the Atlantic for more than a decade.[26]

The governments have focused on the competition policy consequences of Microsoft's monopolies. These are the usual issues of antitrust: Are prices too high? Has competition on the merits been squelched? Has output been reduced? These inquiries have not focused on the security consequences of monopoly, but others have rushed in to fill the void. The most visible strain of this analysis is the "monoculture" argument, namely that the Microsoft operating system monopoly creates a harmful monoculture—a common code base through which

---

[25] United States v. Aluminum Co. of America, 148 F.2d 416, 430 (2nd Cir. 1945) ("A single producer may be the survivor out of a group of active competitors, merely by virtue of his superior skill, foresight and industry. In such cases a strong argument can be made that, although the result may expose the public to the evils of monopoly, the Act does not mean to condemn the resultant of those very forces which it is its prime object to foster: finis opus coronat. The successful competitor, having been urged to compete, must not be turned upon when he wins.")

[26] United States v. Microsoft Corp., 56 F.3d 1448 (D.C. Cir. 1995) (ordering federal district court to approve July 15, 1994 settlement between the United States and Microsoft regarding licensing practices for Windows and DOS); United States v. Microsoft Corp., 253 F.3d 34 (D.C. Cir. 2001) (en banc) (unanimously affirming district court finding of illegal monopoly maintenance under Section 2 of the Sherman Act); Commonwealth of Massachusetts v. Microsoft Corp., 2004 WL 1462298 (D.C. Cir. 2004) (affirming district court approval of settlement agreement among certain states, the United States and Microsoft); Commission of the European Communities, Commission Decision of 24.03.2004 (COMP/C-3/37.792 Microsoft) (finding an abuse of a dominant position in refusing to disclose certain interoperability information for servers and in condition acquisition of Windows on simultaneous acquisition of the Windows Media Player). For my views on the most recent U.S. antitrust case against Microsoft, see Randal C. Picker, Pursuing a Remedy in Microsoft: The Declining Need for Centralized Coordination in a Networked World, 158 Journal of Institutional and Theoretical Economics 113 (2002).

computer viruses spread easily putting the computing network at risk.[27] The National Science Foundation is pouring $750,000 into funding research on ways of creating cyber-diversity, one possible antidote to the monoculture.[28]

A. Monocultures: Supply v. Demand

Consider one formulation of the monoculture argument:

> Most of the world's computers run Microsoft's operating systems, thus most of the world's computers are vulnerable to the same viruses and worms at the same time. The only way to stop this is to avoid monoculture in computer operating systems, and for reasons just as reasonable and obvious as avoiding monoculture in farming. Microsoft exacerbates this problem via a wide range of practices that lock users to its platform. The impact on security of this lock-in is real and endangers society.[29]

This argument builds off of other work that draws out the analogy between farming—and in particular cotton growing—and computer software.[30] That work suggests that in the early 20th century U.S., a cotton "monoculture" had emerged, that only one strain of cotton was grown and that too much acreage was devoted to cotton, especially given the risks posed by the boll weevil. The presumptive solution to monoculture is diver-

---

[27] See Geer et al, supra note 24.

[28] See National Science Foundation Press Release, "Taking Cues from Mother Nature to Foil Cyber Attacks," Nov. 25, 2003 (available at http://www.nsf.gov/od/lpa/news/03/pr03130.htm).

[29] Geer et al, supra note 24, at 7.

[30] John S. Quarterman, Monoculture Considered Harmful, First Monday, February, 2002 (available at http://www.firstmonday.dk/issues/issue7_2/quarterman/).

sification, presumably meaning here that farmers shifted fields from cotton to other crops.

Now I must confess that my knowledge of cotton is limited to thread counts and the supposed virtues of Pima, Supima and Egyptian cotton for sheets (definitely go with the Egyptian), so I am not particularly well situated to make claims about cotton growing in the U.S. in the 1920s to 1940s. But a brief incursion into the world of cotton suggests that the analysis is tricky. Consider the most direct suggestion about boll weevil devastation and diversification. The boll weevil spread throughout the U.S. cotton belt during thirty years, from roughly 1892 in the southern tip of Texas to 1922 in north-eastern North Carolina.[31] Between 1866 and 1892, harvested cotton acreage rose from 7,666,000 acres to 18,896,000 acres.[32] Between 1892 and 1922, while the boll weevil worked its way across the Cotton belt, harvested cotton acreage rose to 31,361,000 acres. The number of bales produced rose as well, from 6.7 million bales in 1892 to 10.1 million in 1922. As a group, farmers were not exiting cotton growing at all, quite the opposite.

We can also look at the data slightly differently. Between 1909 and 1933 in the United States, cotton's share of planted acres fluctuates, but there is barely any net movement between 1909 (10.57%) and 1933 (10.78%).[33] Cotton does decline relatively during the Depression and World War II, and I don't begin to understand why that is, but it seems hard to trace any of this to the boll weevil.[34] While the boll weevil was spread-

---

[31] See Harry Bates Brown & Jacob Osborn Ware, Cotton 203 (3rd ed. 1958).

[32] Historical Track Records, U.S. Department of Agriculture, National Agricultural Statistics Service, April, 2004, pp. 27-30.

[33] This is calculated using the data series for all planted crops, id. at 5, and the comparable series for cotton, id at. 28.

[34] If the numbers are right—and they are sufficiently dramatic that they are difficult

ing, cotton acreage was increasing in absolute terms and cotton held its own as measured against other crops until 1933. We dealt with the weevil by switching to early blooming cotton varieties[35] and by moving production to less humid locations.[36]

Now one might say that this just makes the monoculture point, that switching varieties or growing on different land is an example of heterogeneity in action.[37] But I think that the point is slightly different. The real point is about the cost of generating variety and how quickly adaptations can be made. Think of the point this way: do we need to have an existing stock of varieties in place to be drawn upon at the point where the dominant variety has been found to be wanting or does it suffice to implement just-in-time variety, variety when and as we need it? This is a point about the speed of adaptation in the face of a threat.

But there is a more basic problem with the monoculture idea, at least in farming. For the individual farmer, growing multiple crops is a way of self-insuring against the failure of any one crop. Self-insurance may be sensible if more direct insurance markets are under-developed or aren't sustainable for any of the standard reasons that insurance markets are difficult to establish (adverse selection and moral hazard, for example).

---

to believe—the change in U.S cotton industry in one year was startling. In 1932, roughly 36.5 million acres of cotton were planted and 35.9 million were harvested. In 1933, the corresponding figures were 40.2 million and *29.3 million*. And in 1934, the figures were 27.8 million and 26.8 million. In one season, the harvested numbers fell through the floor and plantings tracked that going forward. And it is unlikely that the change in harvested cotton in 1933 was due to the boll weevil or disease: productivity per harvested acres actually rose from 173.5 lbs. in 1932 to 212.7 lbs. in 1933 (presumably in part as a result of only harvesting the most productive fields). Id. at 28.

[35] Basil G. Christidis & George J. Harrison, Cotton Growing Problems 506 (1955).

[36] Brown & Ware, supra note 31, at 202.

[37] Neil Katyal made this point in his oral remarks on his paper at the conference.

But fundamentally, the monoculture idea says nothing about how much cotton should be grown. While individual farmers might want to grow a mix of cotton and corn to self-insure against the boll weevil, we shouldn't grow corn if no consumer wants it.

The cotton-corn trade off is a great example of the difference between supply-side and demand-side substitutes. Cotton and corn might be supply-side substitutes for the individual farmer—grow one, grow the other, grow both (but also might not be, as we clearly just shifted cotton production across states). But for the consumer, cotton and corn are poor substitutes: we do not see magazines extolling the virtues of corn-silk sheets and no one suggests that you serve cotton as a side dish at your next July 4th celebration. The monoculture notion completely ignores consumer demand: it is a supply-side production idea tailored to the individual farmer for use when insurance markets are incomplete.

B. Heterogeneity and Autarky

Those concerned about monoculture might respond to this by noting that individually rational decisions can be collectively foolish. In choosing a computer and an operating system, the individual may take into account some of the externalities associated with computers. So computer software is frequently discussed as having network externalities: for example, I benefit when other users have the same software, as it makes it easier for us to swap documents. Sheer number of users can give rise to less direct network externalities, as many users will support a greater variety of software.

Individual users probably pay little attention to whether they should seek to contribute to software diversity by using software that runs against the mainstream. Some individuals may value difference and choose accordingly and that would

have the same consequence for increasing diversity in the installed base of computers. In contrast, large-scale choosers might be more sensitive to diversity benefits. They might seek to minimize the chance of a correlated failure of their computer systems by sprinkling pockets of Linux and Macs in a Windows-dominant population.

This means, in an almost a double-negative fashion, that the disconnect between the monoculture argument and what consumers want shouldn't necessarily be dispositive against the monoculture argument. But there is another set of arguments to consider, in particular those organized around the ideas of interconnection and autarky. Interconnection is the great issue of modern network industries. We impose connection obligations on firms that control key bottleneck facilities and seek ways to simplify how those connections are made. So, to take quick examples:

- *Electricity*. As electricity generation ceased to be subject to substantial economies of scale, we moved to encourage merchant generation by imposing an open access regime on the electricity grid. Vertically-integrated grid owner/generators wouldn't be able to advantage their own generation over competing outside generation.[38]
- *Telecommunications*. To switch from electricity to phones, under the Telecommunications Act of 1996, we imposed on telephone incumbents three key sharing obligations: they must interconnect with entrants, so that new customers of entrants can call the incumbent's customers; an entrant can buy telcom services from in-

[38] Promoting Wholesale Competition Through Open Access Non-Discriminatory Transmission Services by Public Utilities, 61 FR 21540, 21544 (1996), *substantially affirmed sub. nom.* Transmission Access Policy Study Group v. FERC, 225 F.3d 667 (D.C. Cir. 2000), *affirmed sub. nom.* New York v. FERC, 535 U.S. 1 (2002).

cumbents at wholesale prices for resale to the customers of the entrant; and, most onerous, entrants can mix and match pieces of the incumbent's network and other facilities at cost-based prices under the unbundled network elements regime.[39]

- *Windows*. Interconnection is not only the dominant issue in traditional physical network industries. How the Windows operating system interconnected with other software was one of the key issues in the antitrust actions brought by the United States and the European Union against Microsoft. The consensual final judgment between the U.S. and Microsoft requires Microsoft to make available to third parties certain communications protocols to facilitate communications between third-party software and Windows.[40] The European Union case focused on two issues: bundling of the Windows Media Player with Windows and server interoperability, meaning, how well do computer servers communicate with Windows?

But as we should have guessed, there is a real downside to all of this connectivity: problems percolate quickly throughout an interconnected system, and problems that might have been just local disturbances end up everywhere. The August 14, 2003 power blackout in Canada and large chunks of the Eastern United States, which affected nearly 50 million people, emphasized again how a local problem—here overgrown trees

---

[39] 47 USC § 251(d)(2). For discussion, see Douglas Lichtman & Randal C. Picker, Entry Policy in Local Telecommunications: Iowa Utilities and Verizon, 2002 Sup. Ct. Rev. 41 (2003).

[40] Final Judgment, ¶ III.E (available at http://www.usdoj.gov/atr/cases/f200400/200457.htm).

in Northern Ohio—could spillover throughout the electricity grid.[41]

The monoculture is another name for a homogenous, connected system. In the monoculture framework, heterogeneity is used as a barrier to the spread of a virus throughout a connected computer system. The anti-monoculture idea also taps into our sense of necessary biodiversity. It is reasonably straightforward to articulate possible benefits of biodiversity and to simulate those in a simple environment.[42] Systems without sufficient diversity can be very brittle, especially as conditions change. An adaptation poorly matched to one environment may become the dominant adaptation as the environment changes.

But heterogeneity isn't equivalent to redundancy: if the University of Chicago Law School used only Windows computers, while Yale Law School used only Macintoshes, a Windows-only virus would decimate Chicago, and while Yale could continue to produce text, we know that the world would be a very different place without the Chicago papers. As this example suggests, we can achieve redundancy through heterogeneity only if we have done a good job of matching the level of het-

---

[41] U.S.-Canada Power System Outage Task Force, Final Report on the August 14, 2003 Blackout in the United States and Canada: Causes and Recommendations p. 45 (April, 2004) ("After 15:05 EDT, some of [FirstEnergy]'s 345-kV transmission lines began tripping out because the lines were contacting overgrown trees within the lines' right-of-way areas. … The loss of the Sammis-Star line triggered the cascade because it shut down the 345-kV path into northern Ohio from eastern Ohio. Although the area around Akron, Ohio was already blacked out due to earlier events, most of northern Ohio remained interconnected and electricity demand was high. This meant that the loss of the heavily overloaded Sammis-Star line instantly created major and unsustainable burdens on lines in adjacent areas, and the cascade spread rapidly as lines and generating units automatically tripped by protective relay action to avoid physical damage.")

[42] See Randal C. Picker, SimLaw 2011, 2002 U. Ill. L. Rev. 1019 (2002).

erogeneity with the level of unique assets. So if Stanford Law School and Yale Law School are good substitutes, we could afford to have each school specialize in a computer operating system, so long as each specialized on a different operating system. In contrast, to ensure that we don't lose Chicago scholarship—a unique asset in the system in my world!—my Law School needs to create heterogeneity internally (most dramatically, my colleagues Richard Epstein and Cass Sunstein should each have at least one Windows machine, one Linux box and a Macintosh to ensure that not a single moment of writing time is lost).

And the last example suggests some of the complexities of using heterogeneity to achieve redundancy. How would we ensure that substitutes use different operating systems? Across-firm heterogeneity might arise spontaneously, as might be the case if we had two substantial choices, with perhaps a 60/40 market share. But unless the differentiated inputs are an important part of production—can you really write better papers on a Macintosh?—we shouldn't expect substitutes at the firm level to necessarily choose different substitutes in inputs, here different operating systems.

But heterogeneity may be a particularly clumsy approach to redundancy. Take two steps back on our path: we went from monoculture as connected homogenous computers to looking at a system of connected heterogeneous computers. Maybe we just need to sever the connection, to isolate computers and to head towards an autarkic computer network. Think Tom Cruise and the first Mission Impossible movie: Ving Rhames can't just hack into the CIA computer at Langley to get the NOC list, because the computer isn't on the network, so Tom has to dive in and hang from the ceiling.

Embracing isolation—disconnection or autarky—breaks the modern pattern of network industries. But interconnection

is not always good and we need to focus on an optimal amount of interconnection. These are obviously not new points to professionals whose job is to engineer safety. So the Nuclear Regulatory Commission has regulations that address how safety-related computer systems need to be isolated or send-only.[43] At the same time, the regulatory push towards electricity generator neutrality is precisely about making better information available to outsiders about the state of the transmission grid. If done poorly, however, requiring interconnections for competitive purposes may create security problems.[44]

The extent of autarky is a choice, and in some cases, we have reduced the degree of autarky in critical systems by moving more communications onto the public Internet. In many critical infrastructure industries, equipment is operated and assessed through SCADA systems (supervisory control and data acquisition systems). The SCADA system are the eyes-and-ears of these systems, and systems that once ran on closed, isolated networks—autarkic networks—are migrating to the Internet.[45] Moving control systems back to separate communications networks—so-called "out-of-band" management—is one of the approaches being considered by government officials to enhance cyber security.[46]

---

[43] See "NRC Issues Information Notice on Potential of Nuclear Power Plant to Worm Infection," NRC Press Release No. 03-108 (September 2, 2003).

[44] See National Security Telecommunications Advisory Committee, Information Assurance Task Force, Electric Power Risk Assessment ("Although not all utilities have an interface between the control center and the corporate information system, the distinct trend within the industry is to link the systems to access control center data necessary for business purposes. One utility interviewed considered the business value of access to the data within the control center worth the risk of open connections between the control center and the corporate network.") (available at http://www.ncs.gov/n5_hp/Reports/EPRA/electric.html).

[45] National Strategy to Secured Cyberspace, supra note 5, at 32.

[46] Id. at 31. See also Robert Lemos, "Sprint touts off-Net networks," CNET

So here is the question: should we buy redundancy through heterogeneity or through autarky (isolated systems)? Heterogeneity and autarky are substitutes, but quite imperfect substitutes. At an abstract level, we would need to do a social cost-benefit analysis on the costs of redundancy as compared to our tolerance for downtime and then figure out the different ways in which we might implement the same level of redundancy.

Try this: we can have ten connected computers running different operating systems or ten isolated computers running Windows. We know that it is cheap to make the next computer, quite expensive to make the next operating system. Meaningful variety in connected computers is quite expensive, if that means creating different operating systems. This is expensive redundancy. Simply creating the operating systems would be quite expensive; adding the associated software ecosystems—the actual application programs that do something—would make the costs extraordinarily high. In contrast, we can isolate ten computers running the same operating system for next to nothing. And of course this overstates the cost of autarkic redundancy. Software and data have a zero marginal cost and computer infections don't affect monitors and CPUs. We are really talking about redundant hard disks, and even an infected hard disk can be wiped clean and reused.[47]

Autarky works best for critical infrastructure, where we can invest the resources required to have isolation and parallel independent communication paths. Autarky addresses cyber-

News.com, July 22, 2004 (available at http://news.com.com/2102-7355_3-5280148.html).

[47] And, for better or worse, the telecommunications bubble has created much redundant infrastructure, making it much cheaper now to create separate, isolated communications networks, with some estimates suggesting that only 3% of the fiber in the ground is being used. See Yochi J. Dreazen, "Behind the Fiber Glut," The Wall Street Journal, September 26, 2002.

terrorism, but autarky makes little sense to dealing with cyber-crime. We cannot very well tell Amazon.com to take its servers off the network to "solve" its cyber-crime problems. Amazon lives and dies on the state of the public network. But Amazon also is a good example of the distinction between critical and non-critical assets. I would find it disruptive if Amazon were offline for a month, but we have many good substitutes for Amazon (BN.com, physical bookstores, books that I already own, libraries). Take the electricity system offline for a month, and much of our current infrastructure—almost of all which runs off of electricity—starts to break down.

C. The Cost of Engineering Heterogeneity

We can now circle back to the core remedies suggested in the monoculture literature, namely, mandatory porting of Office and Internet Explorer to other platforms and a 50% cap on the market share of Windows:

- "Instead, Microsoft should be required to support a long list of applications (Microsoft Office, Internet Explorer, plus their server applications and development tools) on a long list of platforms. Microsoft should either be forbidden to release Office for any one platform, like Windows, until it releases Linux and Mac OS X versions of the same tools that are widely considered to have feature parity, compatibility, and so forth."[48]
- "A requirement that no operating system be more than 50% of the installed base in a critical industry or in a government would moot monoculture risk."[49]

[48] See Geer et al, supra note 24, at 18.

[49] Id. at 19.

Mandatory porting of Office to "a long list of platforms" is an almost certainly an extraordinarily expensive way to seek heterogeneity, and one with no assurance of achieving the end in mind. Microsoft would be required to invest resources in creating a Linux version of Windows independent of any possible consideration of the economic returns from doing so. This remedy was suggested in the remedial phase of the U.S. antitrust case and was squarely rejected by the government "as far beyond the violations found."[50] So we won't impose a mandatory porting obligation as an antitrust remedy. And, in *Trinko*, the Supreme Court recently narrowed the circumstances under which a dominant firm might have an independent antitrust duty to deal with a rival, so we won't get porting through antitrust.[51] This would require federal legislation and that would raise other issues. The possible Taking Clause claims associated with the duty-to-deal obligations in telecommunications and electricity have never been fully litigated, and a porting obligation might be far more onerous than those obligations.

It also might not work and this takes us back to the cotton-corn discussion. Consumers might continue to purchase Windows even with Office ported to Linux. After all, Office has

---

[50] See Response of the United States to Public Comments on The Revised Proposed Final Judgment ¶¶ 433-34 (February 27, 2002) ("The Court of Appeals did not find that Microsoft's unlawful actions created the barrier to entry. The United States crafted the [Revised Proposed Final Judgment] to restore the competitive conditions in the market that were impeded by Microsoft's actions, allowing consumers, software developers, OEMs, and others to make decisions based on the competitive merit of their options. In this way, the market will determine whether particular products will erode the applications barrier to entry. The commentors' and Litigating States' proposal, however, goes far beyond the violations found by imposing on the market a porting requirement for Office that substitutes for competition on the merits and preordains the market outcome.")

[51] Verizon Communications Inc. v. Law Offices of Curtis V. Trinko, LLP, 124 S.Ct. 872 (2004).

been available on the Mac OS for sometime and we don't see consumers heading in droves to the Mac. Office is just one piece—a key piece to be sure—of the Windows ecosystem.

But the point of the second remedy—limiting the market share of Windows to 50% in critical industries—is to make sure that other OSs thrive when they would not otherwise. Assume that we can surmount the question of which industries are critical—though we have struggled with that question for as long as we have had regulated industries[52]—and turn to the merits of the proposal. Given the advantages of autarky over heterogeneity, we should be focusing on the marginal benefit that we would achieve in a more heterogeneous environment.

How many operating systems would we need to mitigate the monoculture problem? If we were a bi-culture or a tri-culture, would that be sufficient? Unsurprisingly, Microsoft believes otherwise having taken the position that a truly diverse operating system culture would require thousands of operating systems.[53] The 50% cap in the monoculture literature is just asserted without any reason being offered to believe that the limit would actually be effective.

And to take the cyber-security question seriously, we need to switch from sport hackers seeking an idiosyncratic version of fame if not fortune, to cyber-terrorists intent upon taking down

---

[52] New State Ice Co. v. Liebmann, 285 U.S. 262 (1932) (considering whether ice business in Oklahoma was a public business and therefore appropriately subject to state regulation).

[53] Warning, supra note 24 ("[Scott Charney, chief security strategist for Microsoft] says monoculture theory doesn't suggest any reasonable solutions; more use of the Linux-source operating system, a rival to Microsoft Windows, might create a 'duoculture,' but that would hardly deter sophisticated hackers. True diversity, Charney said, would require thousands of different operating systems, which would make integrating computer systems and networks nearly impossible. Without a Microsoft monoculture, he said, most of the recent progress in information technology could not have happened.")

key infrastructure. Sport hackers probably just seek attention and dislike Microsoft, so for them, Windows is the natural target. As other operating systems grew in market share, they might become attractive targets as well.[54]

Dedicated cyber-terrorists would take into account the organization of the network and the number of operating systems at work.[55] A cascading failure—a failure that starts with one node and propagates out throughout the network as loads are redistributed—is most likely to occur if the loads are distributed unevenly across the network and the node that fails first has a relatively high load.[56] An attacker seeking to bring down the entire system—power grid or Internet, for example—might naturally concentrate her attack on key nodes, perhaps the root servers in the case of the Internet.[57] And a cyber-attack that relied on congestion, as occurs in a typical denial-of-service attack,[58] would almost certainly seek to exploit any substantial operating system.

In sum, I see little reason to think that a strategy of forced heterogeneity in operating systems would yield meaningful returns at any acceptable cost. This really would be market engineering of a particular sort, and would seem to have more traditional responses available that will do a better job of creating meaningful redundancy and cyber-security. We have frequently isolated networks from other networks—a strategy of discon-

---

[54] Whittaker, supra note 24.

[55] Hackers already write malware that infects across multiple platforms. See, e.g., the description of the Virus {Win32,Linux}/Simile.D available at http://securityresponse.symantec.com/avcenter/venc/data/linux.simile.html.

[56] Adilson E. Motter & Ying-Cheung Lai, Cascade-based attacks on complex networks, 66 Physical Review E 065102(R) (2002).

[57] See Whittaker, supra note 24, at 2.

[58] See, e.g., CERT Advisory CA-1998-01 Smurf IP Denial-of-Service Attacks.

nection or autarky—and I see little reason to think that we should not continue that policy in preference to a strategy of forced heterogeneity.

II. Understanding Computer Software Product Quality

If we are likely to continue to have a monoculture in operating systems—and not just in operating systems, as, for example, Cisco's market share of high-end routers has been running in the 65 to 70% range[59]—what, if anything, can the legal system do to improve software quality? Microsoft gets a lot of attention, but as the Department of Homeland Security's cyber security bulletins make clear, Microsoft isn't the only company that produces software with vulnerabilities, far from it in fact. For me, at least, the more relevant question is what is the relationship between software quality and the liability rules that relate to it? And what should be the source of the liability rules: voluntary contract, default or mandatory warranties tied to contract, or perhaps tort?

We should start with a typical Microsoft End User License Agreement, the basic contract between Microsoft and those using its software. Microsoft disclaims all warranties to the maximum extent permitted by law, seeks to limit any possible damages and seeks to limit any other possible remedies.[60] For Microsoft to be held liable for software defects, an end-user would have to surmount these contractual barriers. Of course, producers have been disclaiming warranties for some time but only with limited success in the face of judges willing to expand

---

[59] Marquerite Reardon, "Cisco bets on new high-end router," CNET News.com, May 24, 2004 (available at http://news.com.com/2102-1033_3-5218356.html).

[60] See End-User License Agreement for Microsoft Software § 8 (available at http://www.gotdotnet.com/team/clr/samples/eula_clr_cryptosrc.aspx).

tort doctrines of product liability. In *Henningsen v. Bloomfield Motors, Inc.*,[61] the New Jersey Supreme Court ran right over a warranty disclaimer, in, as my colleague Richard Epstein puts it, "inaugerat[ing] the modern age of products liability.[62]

Torts liability here would be especially tricky, as we will frequently have three parties to consider: Microsoft, as producer of the software; the hacker, who has created the virus or worm; and the harmed end-user, who very well may have contributed to the harm by clicking when he shouldn't have done so. We would need to sort through complex tort doctrines relating to causality, the intervention of third parties and basic questions regarding strict liability, negligence and contributory negligence. These are not sure winners for Microsoft, as key infrastructure providers have been held liable even in the face of malicious acts by third parties that might naturally be understood to be the actual source of the harm. So the railroad was held liable in *Brauer* when it struck a horse-drawn wagon and thieves made off with empty barrels and a keg of cider.[63] And Consolidated Edison was held liable for damages resulting from looting and vandalism when its gross negligence allowed the lights to go out in New York City in 1977.[64] But these issues, while critical for anyone seeking to impose liability on

---

[61] 161 A.2d 69 (1960).

[62] Richard A. Epstein, Modern Products Liability Law 50 (1980). To put it mildly, Richard is skeptical of the path launched by *Henningsen*: "However popular the *Henningsen* approach might be today, it remains clearly flawed in all its essentials." Id. at 53 (footnote omitted). For additional discussion in the context of software, see Peter A. Alces, W(h)ither Warranty: The B(l)oom of Products Liability Theory in Cases of Deficient Software Design, 87 Cal. L. Rev. 269, 287-88 (1999).

[63] Brauer v. New York Cent. & H.R.R. Co., 103 A. 166 (N.J. Ct. Errors and Appeals, 1918).

[64] Food Pagent, Inc. v. Consolidated Edison Co., 429 N.E.2d 738 (N.Y. 1981); Koch v. Consolidated Edison Co. of New York, Inc., 468 N.E.2d 1 (N.Y. 1984).

software providers, are not my target here and I will instead consider how liability rules influence software quality and software adoption decisions.

A. Consumer Software Adoption with Full Liability

Start with a system of full liability: Microsoft would be liable for the harms that result from its products. With full liability, how would consumer behavior change? A key issue for consumers of software is what version of a product to purchase. Street "wisdom" has it that Microsoft doesn't get its products right until at least the third version, so a prudent user may wait before jumping in. When Microsoft issues a new version of Windows, you are frequently advised to wait until the first service pack is issued before embracing the new operating system. In each case, the consumer trades off the benefits of the new product against the costs of that product. Those costs include the actual purchase price, the hassle of making changes but also should include the expected costs associated with buggy software. Buggy software costs include downtime from software that works poorly, the cost of installing frequent patches and the possible costs of a hacker harming your computer system.

A consumer compares the benefits of the new software against these costs in making a purchase/installation decision. These benefits and costs are private information, known only to the individual consumer. Hacker insurance—actual insurance or de facto insurance imposed under the guise of product liability—could change that decision, as the consumer would no longer face the hacking costs. Such is the nature of insurance: the insured rationally ignores real social costs. As a society, for given software, we want consumers to wait for revised software or not install the software at all if the full social costs of the software exceed the benefits.

Mandatory full insurance—put differently, broad products liability—would result in over-consumption of the software. In a competitive market, mandatory insurance would result in a cross-subsidy from one set of consumers to another. Consider a simple example to make this point.

We have two types of consumers. C1 values the software in question at a benefit of $200 and has hacking costs of $0. C2 values the software at $20 and has hacking costs of $50. We have 9 C1's and 1 C2. It costs $50 to make the software and nothing to make each copy. Use a zero profit condition to define a competitive outcome. Full costs if all ten consumers buy the software are $100. The social benefit from the software is 9*200 + 20, or 1820 against costs of $100, so we should build the software. If we sell ten copies, then a price of $10 per copy would cover the costs of the software. At that price, all ten consumers would buy and the net gain from the software would be $1720.

But without bundled mandatory insurance we would do better. C2 wouldn't buy and we would have costs of $50, benefits of $1800 and net benefits of $1750. The bundled "insurance" is worthless to the C1 type consumers and when required, we have over-consumption by C2s—a social loss—plus a cross-subsidy running from the C1s to C2 to boot.

B. Quality Investment and Full Liability

Mandatory insurance affects investment in product quality in a number of interesting ways. So, to continue the example, suppose that Microsoft could spend $10 to re-design Windows to eliminate all of the hacking costs. From a standpoint of overall social welfare, we would like this to happen: C2 values the software at $20 but faces hacking costs of $50. We could eliminate those costs by spending $10. Will Microsoft do so?

No, at least not if we assume that Microsoft is just selling one product and therefore must sell the same quality product to each consumer. Our C1 consumers don't value the higher quality product: they don't face any hacking costs and would not want to pay a penny more for a better product that lowers hacking costs. If Microsoft spent $60 to make the software—the original $50 cost, plus the additional $10 to eliminate hacking costs—then Microsoft would need to charge $6 a copy to cover its costs, assuming that all ten consumer bought a copy of Windows. C1s would end up with a net benefit of $194 ($200 - $6). An operating system entrant facing the same costs could build the $50 version of the product and cover its costs selling only to the 9 C1s at a price of $5.55, making the C1s better off. (Obviously, if Microsoft could sell two versions of the product it could separate the market and it would then make the $10 investment. So low-quality Windows would sell for $5, high quality for $15, C1s would buy low and C2 would buy high.)

How would mandatory insurance change these decisions? Again, if Microsoft sells only one version of Windows, mandatory insurance "solves" the quality underinvestment problem. Recall how the prior example worked. With bundled insurance, Microsoft's total costs were $100, the $50 product cost and the $50 insurance payment for C2's hacking harms. Microsoft could lower those costs by spending the $10 to eliminate C2's hacking costs, so that total costs dropped to $60. With mandatory insurance, Microsoft would do this.

So should we think that we have two countervailing effects of mandatory insurance, over-consumption by some consumers but better product design decisions by Microsoft? Not really. To bring the two examples together, from a social standpoint, we would like Microsoft to spend up to $20 to eliminate C2's hacking costs, and no more. Remember that C2 puts a value of

$20 on the product and faces hacking costs of $50. Microsoft is going to build the software for the C1s of the world and there is no marginal cost to put a copy in C2's hands. We gain C2's value if we can eliminate the hacking costs so we should spend up to $20 to do that.

Mandatory insurance would get Microsoft to do that. Unfortunately, as set out in the first example, C2 will take the software period, and won't internalize the costs of that decision. So with mandatory insurance, Microsoft will have an incentive to overspend on quality, to invest up to $50 to eliminate C2's hacking costs. What we would really like, socially, is for Microsoft to spend up to $20 to eliminate the hacking costs and for C2 to stop buying the software if it costs more than $20 to eliminate those hacking costs. That is the first-best outcome but mandatory insurance doesn't get us there.

At least in this framework, unbundling the insurance and allowing Microsoft to offer insurance on a voluntary basis doesn't accomplish anything. No C1 would buy insurance and Microsoft would not sell C2 insurance for less than $50. You can't pool risk without more types than we have in this example.

C. Timing of Software Release and Adoption

When should Microsoft release software? When should a consumer adopt new software? How do the liability and warranty rules matter for these decisions? To build off of the prior example for just a bit, instead of imagining that Microsoft faces an initial design choice, think of Microsoft as learning about its software over time through use by consumers. So a sufficient amount of use during period 1 means that Microsoft can redesign the software for use in period 2. We know of course that this tracks actual practice: Microsoft releases service packs for Office and Windows. We face a real design question here, a

trade off between internal and external testing costs and about the nature of learning, whether simulated use is good substitute for actual use.

In a basic sense, this is a question of the optimal time to release a product, where we might think that software has at least two distinctive features (at least as compared to an exploding Coke bottle). First, we think that we will learn about the product through consumer use. We may learn a bit about Coke bottles in consumer hands, but we should think that we will learn vastly more about a given piece of software as end-users explore the full set of possibilities inherent in software. So we should think that the scope of learning is much greater for software. Second, software can be modified after the fact at very little cost. Said again, software is continuous while physical products are lumpy and discrete. Once the Coke bottle is in my hands, Coca-Cola can alter it after-the-fact only at high-cost. Think of the burdens associated with recalls of physical products for retrofitting. In contrast, software could be adjusted across the network while in place. The learning associated with software and its installed malleability should push towards earlier product release compared to physical goods.[65]

To just focus on learning, suppose that Microsoft will learn from period 1 use how to eliminate C2's hacking costs and that from that learning, it can eliminate those costs at a cost of $1. Period 1 use saves us $9 in reduced design costs compared to the $10 that could have been spent in period 1 to eliminate C2's hacking costs.

---

[65] A more formal analysis of these issues would look to the burgeoning literature on real options, which makes timing and the costs and benefits associated with delay central to its analysis. See, e.g., Avinash K. Dixit and Robert S. Pindyck, Investment Under Uncertainty (Princeton Univ. Press, 1994). Products liability policy almost certainly needs to take into account these critical questions regarding the optimal time at which to release products.

Now we can look at insurance a little differently. A no-insurance approach helps to segregate and sequence adoption by end-users. Consumers who anticipate deriving substantial net benefits from the product adopt early, conferring a use externality on those who wait. These guinea pigs pay a higher price for the software—higher in the form of bearing first-period hacking costs that will be eliminated for second-period users. Absent the insurance, we end up with a form of price discrimination.

Indeed, the story gets even better. We don't see Microsoft selling two versions of Windows—one with bugs and one without—simultaneously. But this is close to what we see occurring over time (OK, perhaps with many bugs and fewer bugs). Selling different quality software at different time separates the market and makes it possible to make the socially-sensible investment in reducing hacking costs that couldn't occur in the one-period version of the model above.

To see this, imagine this path for use and sales. Microsoft anticipates selling one version of the software today, and a second improved version tomorrow. Microsoft spends $50, builds Windows and announces a $5 per copy price and sells without insurance. C1s buy today—an assumption for now and an important issue below—and get a benefit of $195. No entrant can offer them a better deal. C2 doesn't buy today, as the buggy software is a bad deal for it. The next period, Microsoft invests $1 in a service pack for the software. Microsoft raises the purchase price for Windows to $6 and C2 buys, with a net benefit to C2 of $14.

Why didn't Microsoft just spend the $10 in the first period to eliminate the bugs? Two reasons. First, we saved $9 in quality improvement costs in using the software in the first period. Whether that makes sense depends on C2's discount rate, as C2 only gets the software in the second period. But, second,

and more importantly, we avoid the defecting-coalition problem that we saw above. Were Microsoft to spend the extra $10 on Windows in the first period and sell the bug-free version to all, it would charge $6 to everybody. An entrant could produce the buggy version for $50 and sell nine copies to the C1s for $5.55 a piece, and the C1s would defect.

By selling different quality software at different times, it is possible to support the incremental expenditure on quality. Once the C1s have purchased, they can't defect to a competitor. The purchases by the C1s create the learning that reduces the costs of improving the software. With mandatory insurance, C2 would have no incentive to internalize the costs of early adoption of the software.

We have relied on consumer heterogeneity—C2 differs from C1—to get the results we have seen so far. If consumers are identical, then we may confront a waiting problem, where each consumer waits for another consumer to adopt first. Put differently, users may believe that there is a second-mover advantage to software adoption. Or, more jargon, we may have a prisoner's dilemma in software adoption, where each person wants the other guy to go first and no one adopts the software. Insurance, voluntary or mandatory, would help to solve the second-mover problem.

How would this work? Suppose that consumers are identical. In choosing whether to adopt new software today or tomorrow, consumers should compare the net benefits today against the discounted benefits of adoption tomorrow. The key point here is that costs tomorrow depend on the number of adoptions today, where the greater the number of adopters, the lower the costs.

Insurance reduces the costs of adoption today. We could imagine a partial insurance scheme, as the learning that emerges from use may not require all consumers. We should

want an insurance scheme that incurs just enough costs today to learn enough to lower the costs of adoption tomorrow. Note that that won't happen with mandatory insurance. Partial insurance might be implemented by insuring only a fraction of the customers or by insuring only a fraction of the harms.

III. Conclusion

The wonder of the Internet is incredibly capable computers connected with each other under the control of individuals. For all of the reasons that we think that decentralization is a powerful force we have applauded the ability of individual users to set up websites and make their ideas available to others. But there is a dark side as well. Always-on connections, extra computing cycles and gigabytes of storage to burn mean that individual decisions can propagate throughout the network quickly. The small-worlds phenomenon that is the Internet means that my computer is only a handful of clicks away from a malicious computer programmer.

My decisions matter for your computing life. A malicious hacker can turn my computer into a zombie and use my broadband connection and my computer to shut down websites, to send millions of spam emails, or worse. The network is a sea of computing externalities, many extraordinarily positive but others that can range from everyday bothersome to enormously disruptive. And, in the hands of a cyber-terrorist, the more we embed critical infrastructure into the public network, the more we make it possible for a cyber-terrorist to turns our computing resources against us and thereby harm critical infrastructure, such as the electricity grid or our communications networks.

Addressing cyber security is a mixed question of engineering—computing architecture—and legal rules. The zombie PC problem emerges with the rise of the Internet and decentral-

ized control over PCs. The pricing structure of the Internet world—one-price, all-you-can-eat broadband and lumpy computing power in the form of powerful CPUs—kills off many of the natural incentives for an individual to ensure that her computing resources are not being used by others. This can be good, as it creates many opportunities for sharing, but the downside is that there is little reason for the individual computer user to police against zombification.

We need to look for mechanisms, architectural or legal, to boost cyber security. Obviously, we will always pursue cyber-terrorists, but we want to take steps before cyber-terror takes place. We could consider actions targeted at individuals, perhaps legal actions for negligent computer set-up or computer operation, or more centralized approaches to kicking poorly-configured machines off of the network. We might enlist Internet service providers, original equipment manufacturers or software producers in those efforts.

But I don't consider those issues here. Instead, in this article, I have considered two issues in detail. The monoculture argument is one approach to architecting the network. That argument suggests that we should focus on forcing heterogeneity in operating systems to enhance our cyber security. I think that is the wrong emphasis. On its own terms, the argument tells us little about the extent of diversity that would be required to achieve meaningful protection, especially if our concern is the cyber-terrorist. The argument also ignores the more important question of adaptability, meaning how quickly can the current system adapt to new conditions. Instead, I argue in favor of the traditional approach of isolation—autarky—in separating critical infrastructure from the public network.

Second, I consider the way in which liability rules for software might influence the quality of software and software use decisions. Hackers can exploit defects in software to seize con-

trol of machines. Fewer defects to exploit and we might reduce the harms of hackers. This turns out to be tricky. Broad liability rules that would protect consumers from the harms of hacking will lead to the standard moral hazard problem that we see in insurance. Consumers who shouldn't be using computers or on the network will jump on once they are protected from hacking losses.

These are standard products liability issues, but software has two particular features that suggest that we should not just apply our standard approaches to products liability. First, we learn about software through use. One piece of software is combined with other software in a way that a Coke bottle is rarely combined with anything else. Second, software can adapt and can be fixed in place after-the-fact. Both of these features should push towards earlier release of software, for buggy software to be fixed later.