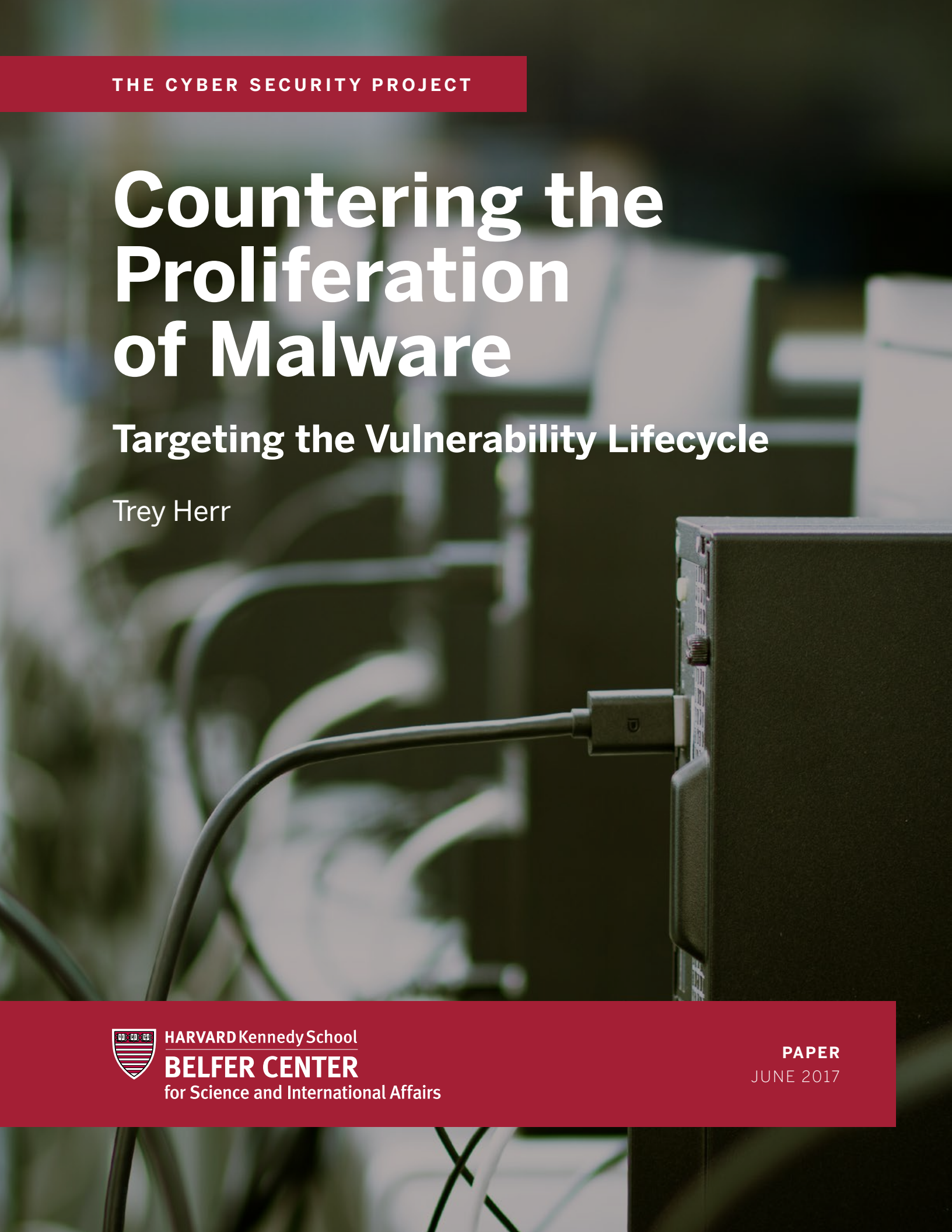


THE CYBER SECURITY PROJECT

Countering the Proliferation of Malware

Targeting the Vulnerability Lifecycle

Trey Herr

HARVARD Kennedy School
BELFER CENTER
for Science and International Affairs

PAPER
JUNE 2017

The Cyber Security Project
Belfer Center for Science and International Affairs
Harvard Kennedy School
79 JFK Street
Cambridge, MA 02138

www.belfercenter.org/Cyber

Design & Layout by Andrew Facini

THE CYBER SECURITY PROJECT

Countering the Proliferation of Malware

Targeting the Vulnerability Lifecycle

Trey Herr

HARVARD Kennedy School
BELFER CENTER
for Science and International Affairs

PAPER
JUNE 2017

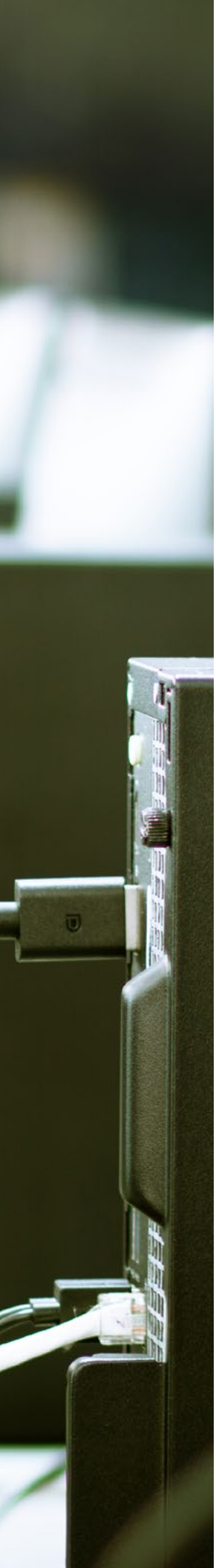
About the Author

Trey Herr, Ph.D, is a postdoctoral fellow with the Belfer Center's Cyber Security Project at the Harvard Kennedy School. His work focuses on trends in state developed malicious software, the structure of criminal markets for malware components, and the proliferation of malware. Trey is co-editor of *Cyber Insecurity—Navigating the Perils of the Next Information Age*, an edited volume on cybersecurity policy, and is a non-resident fellow with New America's Cybersecurity Initiative. He previously worked with the Department of Defense to develop a risk assessment methodology for information security threats. He holds a Ph.D. and M.A. in Political Science from George Washington University and a B.S. in Theatre and Political Science from Northwestern University.

Acknowledgements

This work gratefully acknowledges support from the Belfer Family and the Flora and William Hewlett Foundation. Credit to **Michael Sulmeyer** for his substantial contributions in style, prose, and structure to this paper. Thank you to **Annie Boustead** and the Belfer Cybersecurity Project workshop as well as **Joseph Nye**, **Rob Morgus**, **Casey Ellis**, **Scott Shackleford**, **Ben Buchanan**, **Ryan Ellis**, **Jessica Malekos Smith**, **Jim Waldo**, **Charley Snyder**, **Katie Moussouris**, **Kate Bjelde**, and **Stuart Russell** for discussion and feedback.

Table of Contents



Executive Summary

States have turned to export controls to block the international transfer of malicious software and limit its harmful effects. Based on the nature of the software and the identity of the end user these controls should, in theory, keep malware out of the hands of the worst actors including those with sinister human rights aims. In practice, export controls fail to check the transfer of malware because they ignore the incentives of those who develop and use this software. Even worse, the controls chill the work of legitimate security researchers, undermining efforts to protect states and users from cyber threats and potentially offering the basis for broader information controls.[1] Recognizing these shortcomings, a mix of academics, companies, and civil society groups has attempted to reform the current export control regime. However, even these modest reform efforts have produced only token changes.

A more effective proposal would limit the supply of vulnerabilities available to attackers by reducing the amount of time any given vulnerability is available for an attacker to use in malware. Doing so will raise of the cost to build and acquire malicious software that depend on vulnerabilities. Using the United States as a model for implementation, this paper outlines ten recommendations to shorten the life cycle of vulnerabilities clustered around four key activities:

1. *Increase the number of software vulnerabilities discovered* by expanding the accessibility of bug bounty programs to new companies, but narrowing their scope to the most important bugs.

2. *Increase the number of vulnerabilities disclosed by researchers to software developers* by reforming two important pieces of federal law that currently chill security research.

1 Sergey Bratus et al., "Why Wassenaar Arrangement's Definitions of Intrusion Software and Controlled Items Put Security Research and Defense At Risk—And How To Fix It" (Public Comment, October 9, 2014), http://www.cs.dartmouth.edu/~sergey/drafts/wassenaar-public-comment.pdf.

3. *Increase the speed of patch issuance once developers learn of vulnerabilities in their products* by improving transparency around how long it takes software developers to issue security patches.

4. *Increase the number of customers that apply patches to security flaws once issued by software developers* by improving transparency around which companies apply patches – and which ones do not.

Introduction

The number of actors that employ malicious software to target, infect, and manipulate computer systems is growing, as is the potential for these actors to use malware to destroy and disrupt critical systems. States have taken an indirect approach to combat this threat and limit the use of malware by attempting to restrict the transfer of malware across borders through export controls. The multilateral Wassenaar Agreement—originally designed to restrict the transfer of physical goods like explosives and high strength metals—has been one of the primary vehicles through which these export controls have been negotiated. In addition, the European Union imposed its own regulations to control the transfer of malware, emphasizing that "…the proposal will enhance the EU's efforts to prevent non-state actors from gaining access to sensitive items and will thus contribute to the fight against terrorism…".[2]

This is not the first-time that states have used export controls to limit the transfer of software. In the 1980s though the mid 1990s several countries, led by the United States, tried to use export controls to prevent terrorist groups and adversary states from gaining access to cryptographic technologies. Countries worked through several international agreements, including a predecessor institution to the Wassenaar Agreement. However, these efforts yielded little success as export controls failed to block an explosion in the type and variety of cryptographic software developed

2 European Commission, "Proposal for a Regulation of the European Parliament and of the Council Setting up a Union Regime for the Control of Exports, Transfer, Brokering, Technical Assistance and Transit of Dual-Use Items," COM(2016) 616 final § (2016), 2, http://trade.ec.europa.eu/doclib/docs/2016/september/tradoc_154976.pdf.

around the world.[3] In fact, one of the only lasting impacts to come from these controls were legal requirements for companies to intentionally use weaker security technologies, allowing attackers access to some protected communications.[4]

Attempts to block the transfer of malware with export controls are likely to see similar results as those targeting cryptography: little success and harmful unintended consequences. There are at least three reasons for this. First, it is difficult to define the intent of a software program outside of how it is used thus it is hard to define good vs. bad software suitable for an export control. Second these controls largely ignore criminal groups and individuals beyond the control's jurisdiction thus missing significant sources of malware. Third, even if export controls could be designed and implemented, they pose the risk of harming defensive companies and blocking researchers from collaborating and sharing information.

This paper proposes a better way to limit the use of malware - making it more expensive to build, maintain, and acquire by shrinking the supply of software vulnerabilities available to attackers. This can be accomplished by increasing the pace of vulnerability discovery, disclosure, patch development, and patch application, to shorten the vulnerability life cycle. With a shorter life cycle, useful vulnerabilities will become scarcer and thus costlier. These recommendations cluster around four key activities:

Improving the quality of vulnerability discovery:

- The Department of Homeland Security (DHS) should provide matching funds to private bounties in the event of disclosure of an entire class or method to defeat an exploitation technique.

3 Bruce Schneier, Kathleen Seidel, and Saranya Vijayakumar, "A Worldwide Survey of Encryption Products," *Berkman Center Research Publication*, no. 2016–2 (2016), http://papers.ssrn.com/sol3/papers.cfm?abstract_id=2731160.

4 Part of the US sponsored restrictions were to limit cryptographic keys to a size which could be confidently broken by the National Security Agency. These limits would later enable attacks on encrypted communications between websites and users. For an example, see Bodo Möller, Thai Duong, and Krzysztof Kotowicz, "This POODLE Bites: Exploiting The SSL 3.0 Fallback" (Google, September 2014), https://www.openssl.org/~bodo/ssl-poodle.pdf.

- DHS should work with the Core Infrastructure Initiative to fund a public bug bounty for orphaned software and codebases critical to the internet like OpenSSL.

- The Congress should make bug bounty programs and all their associated payouts tax deductible to encourage wider bounty adoption and higher payouts for vulnerabilities.

Increasing the likelihood of vulnerability disclosure:

- The Congress should reform the Computer Fraud and Abuse Act (CFAA) to reduce the chance of criminal and civil penalties for good-faith security research.

- The Librarian of Congress should permanently exempt good-faith security research from action under the Digital Millennium Copyright Act (DMCA), removing software vendor's ability to sue researchers who discover and disclose flaws in their products.

Encouraging more rapid and effective patch development,

- The National Institute of Standards and Technology (NIST) should develop standards to evaluate patch development availability performance from vendors and work with the Office of Management and Budget (OMB) to integrate these standards into Federal software procurement requirements to give preference to high performing firms.

- DHS and the National Science Foundation (NSF) should collaborate to rapidly award small grants, up to $1 million, to researchers with proofs of concept for new defensive techniques in software and help transition these concepts to practice.

- The US Computer Emergency Response Team (US-CERT) and Federal Trade Commission (FTC) should track and publish patch quality and development time for vendors as well as validation time for secondary vendors.

Improving market transparency and speed in applying patches,

- NIST should develop standards for patch application including categories specific to industry type and organization size, then work with OMB to integrate these into federal contracting standards.

- The SEC should require that companies doing business in the United State annually disclose the number of vulnerabilities reported in their software and the delay associated with patching each of these flaws as well as the use of any software no longer supported by the vendor.

The remainder of this paper summarizes the challenges of using export controls against malicious software, explores the basis for the proposed recommendations, and then develops each in more detail. After highlighting several counter-arguments, the paper concludes with an explanation of how these recommendations fit into the broader cybersecurity policy debate.

Background on Export Controls Targeting Malicious Software

Most formal efforts to limit the use of malware have taken place through export controls. Countries appear unwilling or unable to create new institutions and so are adapting old ones to suit their purpose. In 2013, the UK and French governments proposed adding new controls to the Wassenaar Arrangement to target malicious software.[5][6] These changes were

5 James Gannon, "Wassenaar: Turning Arms Control into Software Control," *Internet Governance Project*, May 25, 2015, http://www.internetgovernance.org/2015/05/25/wassenaar-turning-arms-control-into-software-control/.

6 Wassenaar covers 'intrusion software' which is defined as "'Software' specially designed or modified to avoid detection by 'monitoring tools', or to defeat 'protective countermeasures', of a computer or network-capable device, and performing any of the following: [a] The extraction of data or information, from a computer or network capable device, or the modification of system or user data or [b] The modification of the standard execution path of a program or process in order to allow the execution of externally provided instructions". Wassenaar doesn't restrict this intrusion software but controls all products, technology, and software used to develop, support, maintain, or communicate with it. Much of this software could be employed as a component for malware, underlining the difficulty in crafting careful legal definitions of malicious software.

quickly, and unanimously, adopted by the Wassenaar member-states. The Arrangement is a voluntary international legal framework; it has no direct enforcement authority but serves as a regular forum for countries to coordinate the products and services restricted in their domestic export laws.[7] Two years after the change to Wassenaar, the European Union proposed to list malware as a controlled item under its own export regulations.[8] The proposed EU rules mention human rights but repeatedly return to the goal of national and international security, emphasizing that "the need to protect national security and public morals, in consideration of the *proliferation of cyber-surveillance technologies whose misuse poses a risk to international security* as well as the security of the EU…".[9] [10]

In each of these cases, the export controls either explicitly mention proliferation or are part of an institution designed to block proliferation. The Wassenaar Arrangement is one of a constellation of non-proliferation institutions and requires that its members submit to the Treaty on the Non-Proliferation of Nuclear Weapons among others. The EU's dual use controls, as well as the discussions of the United Nation's Group of Government Experts (GGE) in their 2016-17 session, explicitly employ the phrase proliferation when discussing how to limit the use of malware.[11] This choice of language may be an attempt to build an analogy to nuclear weapons or, more likely, is a holdover from dealing with cybersecurity issues in forums developed to prevent the spread of sensitive weapons technology. Non-proliferation describes attempts to prevent the acquisition of capabilities by actors, e.g. blocking the acquisition of a weapon or fissile material. Counter-proliferation, by contrast, attempts to reduce the utility of capabilities already in these actor's possession. The distinction looks something

7 These changes were implemented in all Wassenaar member states but the United States. In the U.S., public feedback during notice and comment periods for the new regulations spurred an American attempt to modify the text of the controls in Wassenaar itself.

8 Meredith Rathbone et al., "Potential Changes to the EU Dual-Use Export Control Regime, Including Cybertechnology" (Steptoe & Johnson LLP, September 13, 2016), http://www.steptoe.com/publications-11489.html.

9 Author's italics. European Commission, Proposal for a Regulation of the European Parliament and of the Council setting up a Union regime for the control of exports, transfer, brokering, technical assistance and transit of dual-use items, 9.

10 Malicious software, what the EU document terms "cyber-surveillance tools", is covered by a catch-all control that applies restrictions to any items where there is potential for misuse. This massively expands the potential scope of the EU controls.

11 Jim Lewis and Kerstin Vignard, "Report of the International Security Cyber Issues Workshop Series" (UNIDIR, August 2016), http://www.unidir.org/files/publications/pdfs/report-of-the-international-security-cyber-issues-workshop-series-en-656.pdf.

like attempting to block a state from building a nuclear device vs. crippling the delivery and guidance systems for that weapon.

Efforts to craft export controls for malicious software that resemble a non-proliferation regime are likely to fail. This is because, unlike nuclear weapons which very few states had (or have), nearly everyone has the capability to build or buy a piece of malware. There are three problems with the export controls approach to limiting malware use. First, it is difficult to identify software as malicious prior to use so controls are difficult to design. Second, where controls could be successfully implemented, they are aimed at legal business entities and researchers, leaving out criminal groups and other significant sources of malware. Third, even if the controls could be effectively designed and implemented, they pose daunting legal hurdles to the defensive cybersecurity community. These three problems cascade, export controls are hard to make work, are generally pointed the wrong way, and even when working and properly oriented they make life harder (rather than easier) on defenders.

Designing Controls is Difficult

Identifying intent in software is difficult before it is used, creating challenges for the design of legal restrictions like export controls. Using export laws designed for weapons is thorny territory because categorizing malware solely as a weapon is inaccurate. Unlike a shoulder mounted missile or rifle, whose purpose to violence is clear, much of what makes up malware is ambiguous with respect its intended use. One prominent example of this is the Metasploit framework, a popular collection of open source penetration testing tools developed by H.D. Moore and now maintained by the Boston-based information security company Rapid7.[12] Metasploit includes the same software tools necessary to break into and manipulate a computer system as could be used by defenders to protect that very same machine.

12 Jim O'Gorman, Devon Kearns, and Mati Aharoni, *Metasploit: The Penetration Tester's Guide* (No Starch Press, 2011).

Export Controls Ignore the Malware Markets

Export controls are generally focused on legal businesses, and so miss large sources of malware such as criminal groups. Even if information transfer across borders could be precisely limited, export regulations would only restrict malware transferred by companies and individuals under the jurisdiction of these regulations. For example, in the case of the Wassenaar Arrangement, this fails to address all criminal groups and the more than a hundred countries not among the Arrangement's members. It is impractical to guarantee only legal actors have access to the controlled information; many software tools widely available for free on the internet can encrypt or delete data beyond recovery.[13] There is a healthy malware economy where a globally diverse population of groups, including companies, criminal groups, and individuals, build and sell malicious software.[14] It does not take rare technical skills even to create malware that can deliver destructive physical effects.[15] Software vulnerabilities and information on potential targets can be found throughout the malware markets. [16]

Making Life Harder on Defenders

Export controls also impose a potentially debilitating burden on independent researchers and defenders in cybersecurity. Knowledge about malware's components – particularly software vulnerabilities – can be of great use in securing computer systems. Researchers, within both companies and the broader community, will often exchange new malware samples

13 Raiu, "Destructive Malware - Five Wipers in the Spotlight - Securelist," December 18, 2013, https://securelist.com/blog/incidents/58194/destructive-malware-five-wipers-in-the-spotlight/.

14 Trey Herr, "Malware Counter-Proliferation and the Wassenaar Arrangement," in *2016 8th International Conference on Cyber Conflict: Cyber Power* (CyCon, Tallinn, Estonia: IEEE, 2016), 175–90, https://ccdcoe.org/cycon/2016/proceedings/12_herr.pdf.

15 Though doing so repeatedly, autonomously, or with a high degree of confidence about the effects does require substantial investment in intelligence collection, development, and testing infrastructure.

16 Lillian Ablon, Martin C. Libicki, and Andrea A. Golay, *Markets for Cybercrime Tools and Stolen Data: Hackers' Bazaar* (Rand Corporation, 2014), http://www.rand.org/content/dam/rand/pubs/research_reports/RR600/RR610/RAND_RR610.pdf; Kurt Thomas, et al., "Framing dependencies introduced by underground commoditization." Proceedings (online) of the *Workshop on Economics of Information Security* (WEIS), 2015.; Trey Herr and Ryan Ellis, "Disrupting Malware Markets" 105-122 in Richard Harrison and Trey Herr, eds., *Cyber Insecurity: Navigating the Perils of the Next Information Age* (Lanham, MD: Rowman & Littlefield, 2016), https://books.google.com/books?id=NAp-7DQAAQBAJ&source.

across time zones as they are working.[17] This allows for continuous effort, twenty-four hours a day, to understand these samples and craft defenses against them. Export controls restrict this information from moving across borders. Export controls can easily trample on important defensive activities and unnecessarily limit harmless software through catch-all provisions like the one suggested by the EU. [18]

Proposal: Shorten the Vulnerability Life Cycle

Instead of using export controls to block malware transfer, states should work to change attacker's incentives to build and acquire malicious software to begin with. Malware does not exist in a vacuum; it is created by individuals and organizations with recognizable incentives and resource limitations. Software vulnerabilities are a building block for many kinds of malware. From destructive attacks like Stuxnet to espionage operations like Red October and even common surveillance tools, malware often use vulnerabilities to gather information about targets, gain access to computer systems, and maintain access on networks.[19] Reducing the supply of these vulnerabilities would limit those available to attackers and increase the cost necessary to acquire them.

Shortening the vulnerability life cycle to shrink this vulnerability supply requires improving how rapidly vulnerabilities are discovered, disclosed, and patched. This entails *discovering* vulnerabilities in software so that they can be *disclosed* to vendors, rather than sold to malicious actors. Once

17 Katie Moussouris, conversations with the author – 19OCT16 and 3MAY17

18 Bratus et al., "Why Wassenaar Arrangement's Definitions of Intrusion Software and Controlled Items Put Security Research and Defense At Risk—And How To Fix It."

19 Nicolas Falliere, Liam O. Murchu, and Eric Chien, "W32. Stuxnet Dossier" (Symantec, 2011), http://www.h4ckr.us/library/Documents/ICS_Events/Stuxnet%20Dossier%20(Symantec)%20v1.4.pdf; Ralph Langner, "Langner - To Kill a Centrifuge.pdf" (The Langner Group, November 2013), http://www.langner.com/en/wp-content/uploads/2013/11/To-kill-a-centrifuge.pdf; Kaspersky, "'Red October'. Detailed Malware Description," *Securelist.com*, January 17, 2013, http://www.securelist.com/en/analysis/204792265/Red_October_Detailed_Malware_Description_1_First_Stage_of_Attack#1; Bill Marczak and John Scott-Railton, "The Million Dollar Dissident: NSO Group's iPhone Zero-Days Used against a UAE Human Rights Defender," *The Citizen Lab*, August 24, 2016, https://citizenlab.org/2016/08/million-dollar-dissident-iphone-zero-day-nso-group-uae/; Stefan Esser, "PEGASUS iOS Kernel Vulnerability Explained | SektionEins GmbH," *SektionEins*, September 2, 2016, http://sektioneins.de/en/blog/16-09-02-pegasus-ios-kernel-vulnerability-explained.html.

known, software owners can work through the *patch development* process to produce a fix and push to have that *patch applied* by users and organizations to protect themselves. Improving any one step in this cycle does not do enough to constrain vulnerability supply; just finding and even disclosing vulnerabilities does little good without systematic improvement in the patching process.[20]

These efforts would shrink the supply of vulnerabilities available to attackers, raising the cost to build, maintain, and acquire malware which relies on these vulnerabilities. By raising the cost to build and acquire malware, groups with few resources might avoid well-defended targets or be put out of business altogether. Less cost-sensitive organizations, like major intelligence agencies, might be forced to change tactics or accept higher risk of discovery and compromise.

This proposal has several advantages for limiting malware use over the export controls approach. First, shortening the vulnerability life cycle bypasses the need to determine software's intent because it accelerates the flow of all information to defenders, rather than trying to identify and block only malicious software. Second, the policy recommendations outlined below would impact the entire malware economy, not just companies and individuals under a state's jurisdiction. This would help move state's focus away from malware sales and towards the more relevant and pressing issue of malware's use. Third, this proposal's emphasis on vulnerability discovery, disclosure, and patching amplifies existing defensive activities. This puts policymakers in the position of reinforcing and expanding the work already being done to secure against malicious software instead of trying to carefully avoid curtailing it. In addition, nothing in this proposal requires limiting information flows across borders and so presents little threat to be used as a basis for further information controls or outright censorship.[21]

20 Adam Segal, "Using Incentives to Shape the Zero-Day Market," Cyber Brief (New York City: Council on Foreign Relations, September 2016), http://www.cfr.org/cybersecurity/using-incentives-shape-zero-day-market/p38294.

21 These recommendations align with similar made by Sandro Gaycken and Felix Lindner and those of a CSIS Cyber Task Force in 2017 including Federal matching funds for private bounty programs and improving laws that might govern disclosure - Sandro Gaycken and Felix Lindner, "Zero-Day Governance: An (Inexpensive) Solution to the Cyber-Security Problem," in *Cyber Dialogue 2012* (University of Toronto, 2012), http://www.cyberdialogue.citizenlab.org/wp-content/uploads/2012/2012papers/CyberDialogue2012_gaycken-lindner.pdf; Jim Lewis et al., "From Awareness to Action - A Cybersecurity Agenda for the 45th President" (Washington, D.C.: CSIS, January 5, 2017), https://www.csis.org/programs/technology-policy-program/cybersecurity/csis-cyber-policy-task-force.

The remainder of this section outlines ten specific policy recommendations that are modeled in the United States, but could be well replicated in other countries.

Discovery

The key policy challenge in vulnerability discovery is to drive attention to the highest impact software flaws. The recommendations in this section are designed to increase the overall number of vulnerabilities discovered while emphasizing major commercial applications and software critical to the internet, like open-source cryptographic libraries. One insight from increasingly public efforts to find vulnerabilities and disclose them to vendors is that a very small minority of these bug-hunters account for a significant percentage of flaws discovered in software.[22] Increasing the incentives this upper tier of talented researchers have to focus on important targets will help increase the rate at which the most impactful bugs are found.[23] Bug bounties, widely accepted means of providing financial incentives to researchers who discover vulnerabilities, can be used as a way to drive attention to the most important software flaws.[24]

1. *Create New Incentives to Discover Bug Classes*

   Policies should encourage discovery of vulnerability classes, flaws that could be replicated across many different pieces of software. The Department of Homeland Security (DHS) should provide funding to match private sector bounties paid out in the instances of disclosure for an entire class or method to end an exploitation technique. This would complement payouts from programs like Google's Vulnerability Rewards or Apple's Bug Bounty. There is a

22 BugCrowd, "The State of Bug Bounty" (BugCrowd, June 2016), https://pages.bugcrowd.com/hubfs/PDFs/state-of-bug-bounty-2016.pdf; HackerOne, "The 2016 Bug Bounty Hacker Report" (HackerOne, September 13, 2016), https://hackerone.com/blog/bug-bounty-hacker-report-2016; Katie Moussouris and Michael Siegel, "The Wolves of Vuln Street: The 1st Dynamic Systems Model of the 0day Market" (RSA, 2015), https://www.rsaconference.com/events/us15/agenda/sessions/1749/the-wolves-of-vuln-street-the-1st-dynamic-systems.

23 Impactful in this sense meaning some combination of the bug's ease of discoverability, its exploitability, and the resulting severity of its use.

24 Andi Wilson et al., "Bugs in the System" (New America Open Technology Institute, 2016), https://na-production.s3.amazonaws.com/documents/Bugs-in-the-System-Final.pdf.

risk that this program could pay individuals willing to disclose a bug today but who might sell another flaw to a malicious actor tomorrow. However, this is the risk borne by any bug bounty and not a novel problem for government programs funding non-state groups.

2. *Implement Public Bug Bounties for Core Software*

DHS should develop a public bounty program to incentivize vulnerability discovery in orphaned codebases, those no longer supported by a vendor, and software critical to the internet, like OpenSSL, Nginx, and OpenSSH. Run in conjunction with the Core Infrastructure Initiative, this bounty would help defend the central resources of the internet's architecture and care for abandoned code.[25] Focusing bounty programs on these central resources may not add tremendously to the defensive efforts of individual targets but would help reduce the use of the internet's trusted infrastructure as a pathway to attack over time. A comparable effort is being piloted in the European Union, where funding for a program to audit open source software in use by the EU has been expanded to support a short-term bug bounty program.[26]

3. *Encourage Bounty Programs Through Tax Policy*

Congress should make bug bounty payouts tax-deductible. This would encourage the expansion of these programs within existing firms and drive their adoption by a wider range of companies. This tax-deductibility would also leave room for bounty payouts to improve, at least marginally, relative to prices paid by criminal and state buyers elsewhere.[27] Expanding the number of bounty programs is a way to leverage a wider network of vulnerability researchers against malicious actors. The result is a more flexible and responsive security marketplace.

25 Similar in scope to the Internet Bug Bounty program - https://internetbugbounty.org/. Core Infrastructure Initiative - https://www.coreinfrastructure.org/.

26 Lucian Armasu, "European Parliament Doubles Budget For 'Free' Software Audit and Bug Bounty Projects," *Tom's Hardware*, December 1, 2016, http://www.tomshardware.com/news/eu-software-audit-bug-bounty,33123.html.

27 Rainer Böhme, "Vulnerability Markets," in *Proceedings of 22C3*, vol. 27, 2005, 30, https://www.wi1.uni-muenster.de/security/publications/Boehme2005_22C3_VulnerabilityMarkets.pdf.

Disclosure

Public policy and law should encourage, not discourage, individuals and groups who discover vulnerabilities to disclose them to software developers to be fixed. Anything else reinforces a cycle of insecurity where flaws are more often used for malicious ends. Small security firms, academic groups, and independent security researchers regularly bring new vulnerabilities to light through independent audits, hacking competitions, and bug bounty programs.[28] Some companies – including Mozilla, who initiated one of the first bug bounties – provide opportunities for researchers to disclose their findings without fear of legal retribution. Others companies – like Oracle, which often protests the notion of bounties – are infamous for their lackluster or outright retaliatory response to vulnerabilities disclosed to them.[29] The consequences of a bad response to disclosure will grow as companies responsible for software ensuring human safety, like automotive and medical device manufacturers, become more prominent.[30] Reforming laws that impact disclosure and encouraging greater adoption of bounty programs will increase the number of vulnerabilities made known to vendors instead of sold to malicious actors.[31]

---

28 Gregg Keizer, "Single Code Typo Triggers Massive Internet Explorer Hack Attacks," *IT Business*, August 4, 2009, http://www.itbusiness.ca/news/single-code-typo-triggers-massive-internet-explorer-hack-attacks/13806; Dan Goodin, "All Four Major Browsers Take a Stomping at Pwn2Own Hacking Competition," *Ars Technica*, March 20, 2015, http://arstechnica.com/security/2015/03/all-four-major-browsers-take-a-stomping-at-pwn2own-hacking-competition/; Matthew Finifter, Devdatta Akhawe, and David Wagner, "An Empirical Study of Vulnerability Rewards Programs," in *USENIX Security*, 2013, https://www.usenix.org/system/files/conference/usenixsecurity13/sec13-paper_finifter.pdf.

29 Steven Lynch, "Full Disclosure: Infosec Industry Still Fighting Over Vulnerability Reporting," *OpenDNS Umbrella Blog*, October 16, 2015, https://blog.opendns.com/2015/10/16/full-disclosure-infosec-industry-still-fighting/; "That Java Vulnerability and the Full Disclosure Debate," *Infosecurity Magazine*, August 20, 2012, http://www.infosecurity-magazine.com/news/that-java-vulnerability-and-the-full-disclosure/.

30 Kim Zetter, "Researchers Hacked a Model S, But Tesla's Already On It," *WIRED*, August 6, 2015, https://www.wired.com/2015/08/researchers-hacked-model-s-teslas-already/; Risk Based Security, "Uncoordinated Vulnerability Disclosure Causing Heart Palpitations For St. Jude Medical Shareholders," *Risk Based Security*, August 26, 2016, https://www.riskbasedsecurity.com/2016/08/uncoordinated-vulnerability-disclosure-causing-heart-palpitations-for-st-jude-medical-shareholders/.

31 Bounties and related payout programs are unlikely to work alone in resolving the underlying insecurity of most software but they remain a valuable way to attract vulnerabilities to vendors and help researchers learn how to better secure software - Mingyi Zhao, Jens Grossklags, and Peng Liu, "An Empirical Study of Web Vulnerability Discovery Ecosystems," in *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security* (ACM, 2015), 1105–1117, http://dl.acm.org/citation.cfm?id=2813704.

4. *Protect Good-Faith Research and Disclosure from Copyright Protections*

   The Library of Congress should make permanent a recent three-year exemption for security researchers under the Digital Millennium Copyright Act (DMCA) - 17 USC 1201.[32] The DMCA is a 1998 U.S. law criminalizing the act of circumventing protections on copyrighted works, like proprietary software. The new exemption shields those who circumvent manufacturer protections during good-faith security research, for example, after disclosing a vulnerability in a braking system.[33] Making the current temporary exemption permanent would remove one avenue for companies to unfairly punish researchers looking for vulnerabilities in their products. The Electronic Frontier Foundation (EFF) is currently suing the U.S. government to modify the DMCA for exactly this purpose, representing a computer scientist at Johns Hopkins University who audits software and discovers vulnerabilities.[34]

5. *Reform the Computer Fraud and Abuse Act (CFAA)*

   Congress should reform the Computer Fraud and Abuse Act (CFAA) - 18 USC 1030. Specifically, such reforms should 1) introduce an intent-based exemption for good-faith security research leading to disclosure and 2) remove the provision for civil liability from 1030 (g).[35] The intent based exemption would eliminate the potential for criminal penalties in the event of good-faith research

32 Direct change to the statute is welcome as well but working through the Librarian engenders less potential for disruption as compared to working through the legislative process. The Librarian of Congress is empowered to review specific exemptions to the law every three years.

33 Jen Ellis, "New DMCA Exemption Is a Positive Step for Security Researchers," *Rapid7 - Information Security*, October 28, 2015, https://community.rapid7.com/community/infosec/blog/2015/10/28/new-dmca-exemption-is-a-positive-step-for-security-researchers; Charlie Miller and Chris Valasek, "Remote Exploitation of an Unaltered Passenger Vehicle" (Illmatics, 2015), http://illmatics.com/Remote%20Car%20Hacking.pdf.

34 Karen Gullo, "EFF Lawsuit Takes on DMCA Section 1201: Research and Technology Restrictions Violate the First Amendment," *Electronic Frontier Foundation*, July 21, 2016, https://www.eff.org/press/releases/eff-lawsuit-takes-dmca-section-1201-research-and-technology-restrictions-violate.

35 Paul Ohm, "The Computer Fraud and Abuse Act: Structure, Controversies, and Proposals for Reform" in *Cyber Insecurity* eds. Rich Harrison and Trey Herr, Rowman & Littlefield, 2016; Jen Ellis, "How Do We De-Criminalize Security Research?," *Rapid7 - Information Security*, January 26, 2015, https://community.rapid7.com/community/infosec/blog/2015/01/26/how-do-we-de-criminalize-security-research-aka-what-s-next-for-the-cfaa.

efforts.[36] As for civil liability, 1030 (g) provides for relief to parties who can show damage or loss under the criminal provisions of the statute. This gives companies a second means to punish researchers who have discovered vulnerabilities in their products. Removing the provision for civil liability would help limit the range of legal measures companies can use to suppress the discovery of vulnerability information. These changes will help encourage those who discover vulnerabilities to disclose them to developers instead of keeping this information secret or selling it to malicious actors.

Patch Development

The discovery and proper disclosure of vulnerabilities does little to limit the use of malware unless these vulnerabilities are patched. Patching is the process by which a software developer creates a fix for a vulnerability and distributes it to users to update their software.[37] Occasionally, a patch might disrupt software's original functionality so developers must take time to design fixes that work and create no new flaws.[38] The time it takes to develop a patch can be critical; if a vulnerability is being actively exploited, and there is no patch available, organizations have little recourse to protect themselves other than to turn off devices. In the case of critical safety systems or medical equipment, even this may not be an option. The time between a vulnerability's discovery and the availability of a patch is one of the highest risk periods in the vulnerability life cycle as attackers are generally faster at writing exploits for these software flaws than companies are at developing patches.[39]

---

36 Edward J McAndrew et al., "Ninth Circuit Vastly Expands Scope of Criminal, Civil Liability for Computer Fraud" (Ballard Spahr LLP, July 15, 2016), http://www.ballardspahr.com/alertspublications/legalalerts/2016-07-15-ninth-circuit-vastly-expands-scope-of-criminal-civil-liability-for-computer-fraud.aspx.

37 Peter Mell, Tiffany Bergeron, and David Henning, "Creating a Patch and Vulnerability Management Program," *NIST Special Publication* 800 (2005): 40.

38 Jared Newman, "Windows 7 Users Urged to Uninstall Broken Update That Wreaks Havoc on Software," *PCWorld*, December 12, 2014, http://www.pcworld.com/article/2859120/windows-7-users-urged-to-uninstall-broken-update-that-wreaks-havoc-on-software.html.

39 Stefan Frei et al., "Large-Scale Vulnerability Analysis," in *Proceedings of the 2006 SIGCOMM Workshop on Large-Scale Attack Defense* (ACM, 2006), 131–138, http://dl.acm.org/citation.cfm?id=1162671; Muhammad Shahzad, Muhammad Zubair Shafiq, and Alex X. Liu, "A Large Scale Exploratory Analysis of Software Vulnerability Life Cycles," in *Proceedings of the 34th International Conference on Software Engineering* (IEEE Press, 2012), 771–781, http://dl.acm.org/citation.cfm?id=2337314.

Patching demands a balance between speed and precision but not all patches are created equal. Some 'shallow vulnerabilities' are easy to find and simple to fix. One patch may make small changes to break a known exploit without fixing the underlying vulnerability while another requires complex engineering efforts to change an entire feature in the software. The Heartbleed vulnerability in OpenSSL required a relatively straightforward fix though it was widely used so applying that patch took time.[40] By contrast, a 2013 vulnerability in Java was deeply embedded in the language's design.[41] Oracle, the company trying to sell Java, took more than two years to design and release a fix. Even then, their patch was incomplete and turned out to be only a temporary solution.[42]

Patch development involves more than just the original developer as companies whose software depends on the patched code must often validate that patch. For example, if Microsoft issues a patch for Windows 10, then the vendor who builds database software relying on the operating system needs to verify that the patch does not break their product. The database software company tests the patch to see that it does not introduce any new security flaws or break functionality but this process takes time. During the delay waiting for validation, customers often can't apply the patch as many software vendors will void the warranty on a product if the user applies a patch that has not been validated. There are no consensus public standards by which to judge this patch development performance and, as a result, no quick or transparent means to identify those who patch well.[43] But this can be remedied.

40 Zakir Durumeric et al., "The Matter of Heartbleed," in *Proceedings of the 2014 Conference on Internet Measurement Conference*, IMC '14 (New York, NY, USA: ACM, 2014), 475–488, http://dl.acm.org/citation.cfm?id=2663755."plainCitation":"Zakir Durumeric et al., "The Matter of Heartbleed," in Proceedings of the 2014 Conference on Internet Measurement Conference, IMC '14 (New York, NY, USA: ACM, 2014

41 Art Manion, "Anatomy of Java Exploits," *CERT/CC Blog - Vulnerability Insights*, January 15, 2013, https://insights.sei.cmu.edu/cert/2013/01/anatomy-of-java-exploits.html.

42 Abraham Marín Pérez, "Vulnerability in Java Reflection Library Fixed after 30 Months," *InfoQ*, April 28, 2016, https://www.infoq.com/news/2016/04/java-reflection-vulnerability.

43 Stefan Frei, Bernhard Tellenbach, and Bernhard Plattner, "0-Day Patch-Exposing Vendors (in) Security Performance," *BlackHat Europe*, 2008, http://www.techzoom.net/publications/0-day-patch/.

6. *Measure and Reward Good Patch Development Performance*

The National Institute of Standards and Technology (NIST) should develop and publish standards to judge patch development performance. These would measure the time between disclosure and the public availability of a patch as well as how often those patches had to be rolled back and modified then released again. The Office of Management and Budget (OMB), through its participation in the Enterprise Software Category Team (ESCT) alongside the DoD, General Services Administration (GSA), and NIST, should then integrate those standards as a set of performance tiers in new software contract vehicles.[44] The aim of these tiers is to award graduated preference in federal software acquisition for companies with more rapid and effective patch development practices.[45]

7. *Fund New Mitigations and Defensive Techniques*

Government should incentivize the development and implementation of new defensive techniques. In developing a patch, companies and open source projects sometimes have a choice between a rapid but incomplete fix, to break a known exploit, or a longer-term mitigation that resolves the underlying software bug and others like it. Identifying new defensive techniques to make these longer-term fixes can be difficult but often has benefit across multiple software products. Address Space Layout Randomization (ASLR), first developed for use in the Linux kernel in 2001, works to randomize the location of software instructions in memory so they cannot be readily predicted and manipulated by an attacker. ASLR spread to many operating systems after its creation and has had a substantial impact on software security.[46]

---

44 Anne Rung and Tony Scott, "Category Management Policy 16-1: Improving the Acquisition and Management of Common Information Technology: Software Licensing," OMB Memorandum (Washington, D.C.: Office of Management and Budget, June 2, 2016), https://obamawhitehouse.archives.gov/sites/default/files/omb/memoranda/2016/m-16-12_1.pdf.

45 While financial penalties are not suggested here, sanctions between private sector parties could be possible as patch application standards are integrated into private sector best practices and security frameworks.

46 Hector Marco-Gisbert and Ismael Ripoll, "On the Effectiveness of Full-ASLR on 64-Bit Linux," in *DeepSec*, 2014, http://cybersecurity.upv.es/attacks/offset2lib/offset2lib-paper.pdf.

DHS, together with the National Science Foundation, should award small grants (under $1 million) to researchers and organizations who have developed or have working proof of concept for new defensive techniques. This award process should also facilitate a connection between researchers and developers to transition the concept to practice in production software.

8. *Rate Patch Validation Performance*

The US Computer Emergency Response Team (US-CERT), in conjunction with the Federal Trade Commission (FTC), should record and publish the time between initial patches and their validation by secondary vendors and open source projects. The Department of Commerce should convene a multi-stakeholder process, like the Vulnerability Disclosure working group, to develop transparent standards to evaluate patch validation performance using this data. Where feasible, all parties should encourage the adoption of these standards and performance data by private sector actors.[47]

47 Such a working group like approach could also be the basis for developing patch development and application standards.

Patch Application

Discovery, disclosure, and patch development help address the early and middle stages of the vulnerability life cycle but organizations often fail to apply patches. A 2015 report found that 99.9% of vulnerabilities observed in use had been known for more than a year, most with a patch available.[48] Shrinking the supply of vulnerabilities useful for attackers also requires rapidly and comprehensively applying available patches. Public policy should also incentivize vendors to certify each other's patches, where appropriate, and induce organizations to apply them rapidly.[49]

Understanding how well organizations apply patches is an important metric of security behavior. But there is little consistent and reliable public information about most organizations' rates of patch adoption. This prevents customers and business partners from using this information to sort the marketplace into high and low security performers. The result is that organizations have difficulty selecting companies for good security behavior. Government procurement rules offer an opportunity to reward companies with good patch application performance and help encourage adoption of these standards elsewhere in the private sector.

9. *Track Patch Application Performance*

   NIST should work with the OMB to develop and publish standards for patch application performance. Where recommendation 6 deals with vendors and their patch *development* performance, this recommendation addresses all other organizations and their patch *application* performance. These standards would measure the time between, when a patch was made available and when it was applied, including specific categories for industry type and organization size. These patch application standards could then be used to preference high performing companies bidding on federal contracts.

48 Verizon, "2015 Data Breach Investigations Report (DBIR)" (Verizon Enterprise Solutions, April 15, 2015), 15, http://www.verizonenterprise.com/DBIR/2015/.

49 Ioannidis, Christos, David Pym, and Julian Williams. "Information Security Trade-Offs and Optimal Patching Policies." European Journal of Operational Research 216, no. 2 (January 16, 2012): 434–44

10. *Inform Market Response to Patch Application Performance*

Customers should be able to incentivize organizations to more rapidly and effectively adopt patches by rewarding good behavior with their business. This requires that information about organization's patch application performance be available in the marketplace. To make this possible, the Securities and Exchange Commission (SEC) should encourage quarterly disclosure of patch application performance by firm. These records should be broken down by category of software in use and include vulnerability reports, the number of patches applied, their corresponding vulnerabilities, and the time between when each patch was made available and then applied.[50] This disclosure should also include notice for any software in use but no longer supported by the original vendor, indicating that no security updates are being made available. Software vendors should report aggregate vulnerability disclosures and patches issued. These changes will give new information to customers and businesses to evaluate the security behavior of those they transact with, encouraging better patching performance.

50 These changes could follow on the 2011 guidance, and be included under "Description of Business" on firms' annual 10-K reporting.

Counter-Arguments

This paper assumes that vulnerabilities are sufficiently sparse in software such that a substantial reduction in supply is possible, but that may not be true. Most bug bounty programs assume vulnerabilities are sparse to some degree; why pay for bugs without the expectation that a patch will remove that vulnerability and improve the security of a piece of software? But the question of vulnerability's density in code remains a point of contention in scholarship.[51] The topic deserves further study as it plays a key role in how the malware economy will respond to constraints on vulnerability supply.

Few piece of malware leverage new vulnerabilities unknown to defenders, so called zero-days, so it could be argued that improving the incentives to find these vulnerabilities will yield little benefit. It is true that most malware doesn't leverage zero-days, which can be costly and time consuming to employ against a target.[52] For most malware authors, it is better to use known vulnerabilities that offer reliability and consistency.[53] There is an inordinate focus on zero-day vulnerabilities in cybersecurity policy debates but this paper addresses the entire vulnerability life cycle rather than being limited to just known or unknown vulnerabilities. Thus, the recommendations above work in concert to produce positive security outcomes regardless of whether a vulnerability is old or new.

Another argument is that these recommendations do little to specifically address the sale and use of surveillance malware. While this proposal does nothing to impede transfer of such software, the recommendations would generally shorten the vulnerability life cycle, impacting surveillance malware as much as any other kind. The same mobile phone whose security

51 Eric Rescorla, "Is Finding Security Holes a Good Idea?," *IEEE Security and Privacy* 3, no. 1 (January 2005): 14–19; Andy Ozment and Stuart E. Schechter, "Milk or Wine: Does Software Security Improve with Age?," in *Usenix Security*, 2006, https://www.usenix.org/event/sec06/tech/full_papers/ozment/ozment_html/; Dan Geer, "For Good Measure: The Undiscovered," *;login:* 40, no. 2 (2015): 50–52.

52 Robert M. Lee, Michael J. Assante, and Tim Conway, "Analysis of the Cyber Attack on the Ukrainian Power Grid," *SANS Industrial Control Systems*, 2016, 9; Lillian Ablon and Andy Bogart, "Zero Days, Thousands of Nights" (Santa Monica, CA: The RAND Corporation, 2017), https://www.rand.org/content/dam/rand/pubs/research_reports/RR1700/RR1751/RAND_RR1751.pdf.

53 Indeed, issuing a patch is often cause for exploit authors in the malware ecosystem to develop new products, reverse-engineering vulnerabilities from the patches announced by vendors and writing an exploit to match. For more on this topic see, Recorded Future, "Hacker Forum Traffic Analysis: 'Patch Tuesday … Exploit Wednesday' and Other Patterns," *Recorded Future*, September 24, 2015, https://www.recordedfuture.com/hacker-forum-traffic/.

could be improved by reducing the supply of these vulnerabilities, and more rapidly patching software flaws, is the same system being targeted for surveillance.[54] This paper is not a comprehensive approach to restraining the marketplace for surveillance tools but, rather, proposes a more practicable starting point to limiting the use of malware than a complicated and deleterious international export control regime.

54 Adam Senft, Jeffrey Knockel, and Ron Diebert, "A Tough Nut to Crack: A Further Look at Privacy and Security Issues in UC Browser" (The Citizen Lab, August 7, 2016), https://citizenlab.org/2016/08/a-tough-nut-to-crack-look-privacy-and-security-issues-with-uc-browser/; Jakub Dalek, John Scott-Railton, and Masashi Crete-Nishihata, "Shifting Tactics: Tracking Changes in Years-Long Espionage Campaign against Tibetans" (The Citizen Lab, March 2016), https://citizenlab.org/2016/03/shifting-tactics/.

Conclusion

This proposal argues that states working to limit the use of malware should shift their focus from export controls to policies that change attacker's incentives in building and acquiring malicious software. Policymakers can accomplish this by incentivizing more complete dissemination of vulnerabilities to defenders as well as facilitating the rapid update of software. These ten recommendations do not present a complete solution to the use of malware, but are a strategy to build on existing defensive efforts and improve the status-quo through the full chain of mitigation from discovery to patching. By implementing these changes, the U.S. and others would incentivize more complete discovery of unknown vulnerabilities and more rapid and effective patching of known bugs.

It is important to recognize that these recommendations are an improvement to the status-quo but are premised on the state of software today, which remains fundamentally insecure. Improving the pace and quality of patch adoption for example should not substitute for a long-term strategy to improve security across the computing ecosystem. In shaping such a strategy, policymakers should encourage use of less error prone programming languages, for example Rust, and pursue a shift towards easier to secure and more rapidly patched architectures that expand the use of cloud computing.[55] With the rapidly growing importance of embedded systems and the "internet of things", policymakers may also want to consider some combination of narrow liability and voluntary standards, like a software development oriented NIST Framework, in critical industries like automotive and medical device manufacturing.[56]

This proposal also helps to avoid a potential collision with states like Russia and China over censorship and content restrictions. Export controls are used to specify and block the sale of certain software products, that is, they are an attempt to control the movement of certain kinds of information across borders. One such set of information controls could easily to lead to debate over others, perhaps focused on software used to

55 Peter Bright, "Mozilla, Microsoft Rebuilding Their Browsers' Foundations without Anyone Noticing," *Ars Technica*, April 20, 2017, https://arstechnica.com/information-technology/2017/04/mozilla-microsoft-rebuilding-their-browsers-foundations-without-anyone-noticing/.

56 For more on this, see Chapters 4 and 5 in Harrison and Herr, *Cyber Insecurity*.

circumvent censorship or specific kinds of internet content. By moving away from export controls to emphasize policies which shrink the supply of vulnerabilities, states avoid the debate over restricting information flows completely.

There is little about malicious software that grants the state a monopoly on capability in cyberspace. Policymakers must recognize the distributed nature of innovation in the malware markets and relatively low barriers to information exchange between groups building malicious software. These make it impractical to restrict the transfer of malware or try to contain the knowledge of how to build it. Shortening the vulnerability life cycle is one way to achieve broad security gains without imposing unnecessary burdens on the defensive community. This proposal offers a way forward to limit the use of malware without relying on ineffective export controls or other ill-suited non-proliferation regimes. The endstate is a greater level of security for average users and organizations, with a substantial reduction in the operational efficacy of hostile groups.

The Cyber Security Project
Belfer Center for Science and International Affairs
Harvard Kennedy School
79 John F. Kennedy Street
Cambridge, MA 02138

www.belfercenter.org/Cyber

Printed in the United States of America