Atlantic Council

Testimony of

Dr. Trey Herr
Director, Cyber Statecraft Initiative
Atlantic Council

Before the
United States Senate
Committee on Homeland Security and Governmental Affairs

"Responding to and Learning from the Log4Shell Vulnerability"

February 8th, 2022

Chairman Peters and Ranking Member Portman, members and staff of the Committee, thank you for the invitation to speak today. While the bulk of our discussion may involve the intricacies of technology and cybersecurity, I want to underline the ultimate importance of this topic to innovation. Software is the logic by which we mold base metal and electromagnetic impulses into computers central to economic and political life. Open source is the wellspring of that transformation, and it underpins a period of remarkable technical progress stretching back more than two decades.

Our task is to ensure the long-term viability and security of open source as it enables these important and widely used technologies. In working to improve the security of open source we should not seek to "fix" these communities, but to become a better partner to them to enable open source developers, maintainers, and consumers to better secure each other.

Log4Shell and the Vulnerabilities of Software Supply Chains

For the past two and a half years, my team and I have studied the security of software supply chains, cataloguing more than 140 attacks and vulnerability disclosures going back to 2010. Software supply chain attacks remain popular, impactful, and are being used to great effect by states.[1] The sustained growth of software supply chain attacks is caused at a technical level by continued failure to secure code integrity. Attackers continue to find ways to access accounts and bypass code signing, app stores struggle to verify the innocuity of all application software, developers embed insecure design choices at the lowest level of computing, and vendors have difficulty fully grasping the scope of their software dependencies and reliance on supply chain service providers. These are complex technical challenges with neither easy nor immediate solutions, and they further complicate the lapse in policy progress to secure a supply chain that has grown critical to both industry and national security.

The most disconcerting trend in this data is the consistency with which these attacks occur against sensitive portions of our supply chains—this is not a new problem. A 2010 report from Carnegie Mellon University's Software Engineering Institute profiled the DoD's concern about vulnerabilities buried deep in software and exploited by malicious parties; and indeed we live that reality today.[2]

Log4j is a logging program, software that collects information about the behavior of other software. Log4shell is a means to take advantage of a flaw in that logging program to spread malicious software. The flaw itself is rooted in a feature of the logging program allowing it to

---

[1] This and several other portions of the following testimony are drawn from "Breaking Trust: Shades of Crisis across an Insecure Software Supply Chain", Trey Herr, June Lee, Will Loomis, and Stewart Scott – https://www.atlanticcouncil.org/in-depth-research-reports/report/breaking-trust-shades-of-crisis-across-an-insecure-software-supply-chain/ and "Broken Trust: Lessons from Sunburst", Trey Herr, Will Loomis, Emma Schroeder, Stewart Scott, Simon Handler, and Tianjiu Zuo" - https://www.atlanticcouncil.org/programs/scowcroft-center-for-strategy-and-security/cyber-statecraft-initiative/breaking-trust/

[2] Robert J. Ellison, John B. Goodenough, Charles B. Weinstock, and Carol Woody, "Evaluating and Mitigating Software Supply Chain Security Risks," Carnegie Mellon University's Software Engineering Institute, May 2010, https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=9337.

look up information, including web addresses, but without good means of checking the inputs it receives. The actual flaw is tragic but not unique or altogether unpredictable. Log4j's notoriety comes from the fact that the program is widely used in other software, often several layers deep, which has made identifying and patching the affected program difficult for many.

The Log4j vulnerability and resulting Log4shell exploit are significant challenges but they are not exceptional. Software, both open source and proprietary, has been victim to and remains vulnerable to widely exploited flaws. The structure of open source maintenance means that version control, ownership, repository management, dependency tracking, and even naming conventions all impact the ecosystem's security. The Cyber Statecraft Initiative's Breaking Trust dataset[3] found that 29 percent of the most popular open source projects contain at least one known security vulnerability .[4] That is not to say that open source software is inherently more or less secure than proprietary software, and in fact the great bulk of code sold as proprietary depends to great extent on open source. In widely used proprietary operating systems, open source is rampant. Early versions of MacOS famously shared a great deal of code with the FreeBSD (Berkeley Software Distribution) version of Unix,[5] and Windows 10 now includes a Microsoft-developed version of the full Linux kernel.[6] The same transparency and mutability that make open source so useful to the entire software ecosystem also present security concerns.

Open Source

As software continues to spread at an unprecedented pace, developers are under pressure to create new products and services ever faster and at lower cost. Open-source software is a crucial layer in the software ecosystem whose security is under-addressed, especially as many vulnerabilities are hidden under layer after layer of dependencies. Across the software supply chain, there are a variety of types of codebase interleaving in a complex web of software inter-reliance. Not all open source software is developed or maintained by hobbyists and volunteers, there is a profusion of codebases with paid or even full time maintainers. Log4j had multiple full time funded developers associated with it and indeed this may have helped contribute to the risk as features were being added in response to request without a sufficient understanding of how this new code could be abused.

Open source software operates under licenses that allow others to use and modify it under limited conditions—the "source (code)" is *open* to public viewing and modification. For open

---

[3] Trey Herr, Nancy Messieh, June Lee, Will Loomis, and Stewart Scott, "Breaking Trust: The Dataset," The Atlantic Council, July 26, 2020, https://www.atlanticcouncil.org/programs/scowcroft-center-for-strategy-and-security/cyber-statecraft-initiative/breaking-trust/.

[4] Mayank Sharma, "Supply Chain Attacks On Open Source Repositories Are Reaching New Highs," TechRadar, September 15, 2020, https://www.techradar.com/news/supply-chain-attacks-on-open-source-repositories-grew-by-over-600.

[5] Klint Finley, "Apple's Operating System Guru Goes Back to His Roots," Wired, August 8, 2013, https://www.wired.com/2013/08/jordan-hubbard/.

[6] Mary Jo Foley, "Windows 10 is Getting a Microsoft-Built Linux Kernel," ZDNet, May 7, 2019, https://www.zdnet.com/article/windows-10-is-getting-a-microsoft-built-linux-kernel/.

source without paid support schemes, this code is typically offered "as is" without warranty. Some of the most common software systems in use today are open source: the Linux operating system, Apache Web Server, and more. The open source ecosystem benefits software development itself, providing packages, libraries, and tools for widespread use while allowing interaction with a greater number of developers than any single entity could provide. Any final product might contain dozens or hundreds of open source packages—one 2018 report found that 96 percent of applications in sampled commercial codebases had open source components.[7]

Open source projects are not simply text files of freely editable source code floating around the Internet. Rather, a specific developer or entity maintains them. Others can come along, copy that code, and make changes. However, to incorporate those changes into the repository, contributors need to submit their version to the project maintainer for review (often called a pull request). The maintainer decides whether to incorporate the changes. Alternatively, the editor can host their own branch of the original code independently, preserving the identities of the original maintainers while still allowing the code itself to be altered and redistributed. Version control, changelogs, and ownership practices are crucial to an effective ecosystem, and there are many hosting services where developers can upload projects for others to use, edit, make pull requests, perform independent reviews, and more. These repositories of open source projects, libraries, and packages can even be built on open source components themselves. For example, the most widely used version control system, Git, is itself an open source program, while GitHub is a web-hosted interface allowing users to interact with repositories and other management tools that relies on an underlying Git infrastructure. Other popular open source repository systems include GitLab, BitBucket, and PyPI.

Where proprietary software is code owned by a single individual or organization, much of open source is developed along the lines of something closer to "an interacting, self-governing group involved in creating innovation with members contributing toward a shared goal of developing free/libre innovation."[8] Some open source projects start as proprietary code while other open source is sold together with software support programs. Many developers voluntarily collaborate to develop software that is valuable to them or their organization and many open source communities across the world find graduate students, long time software engineers, and technically minded volunteers outside the technology industry working side by side. Open source has become the bedrock of technological innovations like cloud computing, software-as-a-service, next generation databases, mobile devices, and a consumer-focused Internet.[9]

Project, or community-based, OSS is the most common example: a distributed community of developers who continuously update and improve a codebase. Ruby is a classic case of this

---

[7] Mary K. Pratt, "5 open source software problems – and how to manage them," TechTarget, September 13, 2018, https://www.techtarget.com/searchcio/tip/5-open-source-software-problems-and-how-to-manage-them.
[8] Michael Ayukawa, Mohammed Al-Sanabani, and Adefemi Debo-Omidokun, "How Firms Relate to Open Source Communities," *Technology Innovation Management Review*, January 2011, https://timreview.ca/article/410.
[9] Peter Levine, "Why There Will Never Be Another RedHat: The Economics Of Open Source," Tech Crunch, February 13, 2014, https://techcrunch.com/2014/02/13/please-dont-tell-me-you-want-to-be-the-next-red-hat/.

model of open-source development. It is a community-based open-source codebase created in 1995 by Yukihiro "Matz" Matsumoto with a focus on "simplicity and productivity."[10] Twitter, Hulu, Shopify, and Groupon are just a few well-known sites built with Ruby. Individuals can manage their own software packages and dependencies on a day-to-day basis to ensure their quality. Attacks on Ruby feature in four different incidents in the Breaking Trust dataset including a March 2019 attack on the "strong_password" gem which inserted a backdoor into code used to evaluate the strength of passwords on websites. The gem was downloaded more than 500 times before a single developer auditing the code noticed the change.

Another, less intuitive, model of open-source project is commercial open-source software (COSS). Up until its purchase by IBM in 2019, Red Hat was the largest COSS entity in the world. The company runs and operates an eponymous distribution (version) of the Linux operating system. Linux has existed since 1991 and is found in everything from cars and home appliances to supercomputers.[11] Red Hat maintains profitability by giving away its OSS, but charging customers for support, maintenance, and installation.[12]

Open source is not unique to information technology and can be found in operational technology environments as well. Here though the longer time horizon for new equipment and tighter limits on downtime make patching or major version updates more challenging.

Many open source projects are maintained by developers in their free time, and they do not always update their own dependencies, resulting in "trickle-down" vulnerabilities.[13] According to a report by Veracode, open source developers do not update project packages 79 percent of the time.[14] The installation of just one package can result in an average of eighty indirect dependencies, as packages themselves frequently rely upon other open source components.[15] While these dependencies drastically reduce the need for developers to "reinvent the wheel" for every product they create, they also exponentially increase the attack surface of any given piece of code in an already resource-constrained environment.

---

[10] Kimberley Cook, "Python vs. Ruby: Which Is Better for Every Programmer and Why?" House of Bots, November 6, 2018, https://www.houseofbots.com/news-detail/3957-1-python-vs-ruby-which-is-better-for-every-programmer-and-why.

[11] Jenni McKinnon, "A Citizen's Guide to Open Source Communities," Pagely, April 18, 2019, https://pagely.com/blog/citizen-guide-open-source-community/.

[12] Levine, "Why."

[13] Suzanne Ciccone, "Announcing State of Software Security v11: Open Source Edition," Security Boulevard, June 22, 2021, https://securityboulevard.com/2021/06/announcing-state-of-software-security-v11-open-source-edition/.

[14] Suzanne Ciccone, "Announcing Our State of Software Security Open Source Edition Report," Veracode, May 19, 2020, https://www.veracode.com/blog/research/announcing-our-state-software-security-open-source-edition-report.

[15] Mark Russinovich, "Collaborating to Improve Open Source Security: How the Ecosystem is Stepping Up," RSA Conference, February 20, 2020, https://published-prd.lanyonevents.com/published/rsaus20/sessionsFiles/18542/2020_USA20_KEY-F02S_01_Collaborating-to-Improve-Open-Source-Security-How-the-Ecosystem-Is-Stepping-Up.pdf.

Failures to mitigate open source vulnerabilities can have catastrophic consequences due to their widespread use and ambiguous ownership. For example, OpenSSL, responsible for managing website security certificates and implementing the TLS secure protocol, contained a serious vulnerability called Heartbleed, which allowed attackers to steal and eavesdrop on encrypted internet traffic and left massive numbers of users vulnerable.[16] Another popular package, event-stream, was compromised in 2018 when a malicious actor inserted code into the program that stole Bitcoin wallets to compromise copay-dash, a Bitcoin platform dependent on event-stream.[17] It remained undetected for months. A new report from the cybersecurity firm Sonatype found a 650 percent increase in open source repository attacks in 2021.[18]

Addressing the Security of Open Source Through Federal Policy

Though not unique, the Log4j vulnerability and resulting exploitation impacted, according to one firm, thousands of private sector networks around the world.[19] Last year, the Atlantic Council issued a report on the SolarWinds/Sunburst campaign and its aftermath, diagnosing systemic shortfalls in the US government cybersecurity strategy and supply chain security policies as well as concerns with the security governance of widely deployed cloud computing services. Amid this discussion of proprietary code however, the report warned 'Don't forget about open source.'

One of the popular distribution vectors for software supply chain attacks in this report's dataset was open-source packages and libraries. These are often not the most consequential attacks, but they are exploited through largely trivial effort on the part of the attacker, pointing to a concerning trend given the wide dependence on open-source code in commercial and national security applications. Continuing efforts by the White House to incorporate open-source software as a means of sharing code across different agencies and with the technical public further raise the stakes of securing open-source software development.[20]

The following recommendations aim to support more effective and consistent security practices across open-source projects and in the governance of the open source ecosystem. Policymakers should *not* endeavor to "fix" the open-source community. There is no one open-source community, and effective change comes from resources, tools, education, and time—not trying to upend cultures. Rather, the public sector can provide long-term infrastructure

---

[16] James Turner, "Open Sources Has a Funding Problem," Stack Overflow, January 7, 2021, https://stackoverflow.blog/2021/01/07/open-source-has-a-funding-problem/.
[17] Zach Schneider, "Event-stream Vulnerability Explained," personal blog, November 27, 2018, https://schneider.dev/blog/event-stream-vulnerability-explained/.
[18] "Open Source Continues to Fuel Digital Transformation, Sonatype's 2021 Software Supply Chain Report Reveals Important Trends," Sonatype, September 15, 2021, https://www.sonatype.com/press-releases/sonatypes-2021-software-supply-chain-report.
[19] "A Deep Dive into a Real-Life Log4j Exploitation," Check Point Software Technologies, accessed February 3, 2022, https://blog.checkpoint.com/2021/12/14/a-deep-dive-into-a-real-life-log4j-exploitation/.
[20] US Department of Commerce, Open Source Code, accessed July 15, 2020, https://www.commerce.gov/about/policies/source-code.

support and a vision for security across the technology ecosystem alongside additional resources for security like grant funding, policy evangelism for best practices, and incentives toward better security outcomes for industry.

These recommendations focus on the role of DHS and its components given the jurisdiction of this Committee but it is important to note that open source supply chain security is a global challenge. This code is developed across national boundaries and consumed in the same way. While these actions focus on near term investments and activity from the Cybersecurity and Infrastructure Security Agency (CISA), they should be considered as downpayments to collective action across the United States and its allies and partners. The resilience and long term health of the open source ecosystem is in the collective interest of countries around the world., especially as circumstances like the current tensions around Ukraine highlight the potential consequences of vulnerabilities in our digital and physical infrastructure.

Executed together, these policies wold help improve the stability of open-source security efforts and strengthen channels between the public and private sectors. They should also help mitigate the challenges arising from the rapidly growing use of containers in cloud service deployments, including registries and hubs for container and other cloud images. The goal of these changes is to improve the health of the open source software ecosystem, create a more robust public-private partnership on software security issues, and lay the groundwork for long term investments in this critical digital infrastructure.

1. Build a Good Government Partner
2. Get Serious about Global Risk Assessment for Software Supply Chains
3. Investing in Critical Digital Infrastructure

1. Build a Good Government Partner

At present, there is no clear single point of partnership for open-source security in the US government, leaving a burgeoning array of private sector efforts to coordinate piecemeal. This absence risks costly duplication of effort and a missed opportunity to align open-source security with the long range security priorities, and ecosystem wide perspective, of US government stakeholders.

Congress should authorize the creation of a small (six- to eight-person) open-source security outreach and partnership program office inside DHS CISA's Cyber Security Division. Open-source security should be part of mainstream supply chain security policymaking, and this office would be charged with supporting those efforts while acting as the single point of contact for external stakeholders. The office would have an important and complementary role to efforts like the Linux Foundation's Open Source Security Foundation and other industry initiatives, providing long-term perspective, resources, and insights from federal cybersecurity priorities. The CISA Open Source Office should be a source of support and advocacy for open source security as a component of supply-chain security policy work, coordinating across government agencies and offices to ensure that open source security is no longer tied to a crisis cycle.

This office should further encourage collaboration among the United States and allies in supporting the security of open source projects identified as critical by the office and act as a community liaison/security evangelist for the open-source community across the federal government. This office would require new funding in the long term but could be spun up out of an existing program and initially staffed using similar authorities as those used to bring outside cybersecurity experts in to support Operation Warp Speed and CISA's work with the health sector.[21]

2. Get Serious about Global Risk Assessment for Software Supply Chains

One of the pressing challenges for open source maintainers and consumers alike is the lack of a clear hierarchy of risk. Log4j's pernicious nature was not due to a dastardly means of exploitation or a high consequence target so much as to the fact that the logging tool was used on such a wide variety of applications and was embedded deeply in codebases. Thus, consumers may not have even been aware of its inclusion in their codebase or dependences, making it all the more difficult to find and quickly update. Efforts by CISA to track major known vulnerabilities that need to be patched across the federal enterprise as well as industry efforts to index the most widely used projects and prioritize funding for their security are certainly welcome. But what log4j demonstrated, in eerie similarity to the Sunburst/Solar Winds campaign of the previous December, is that assessing risk in software supply chains requires looking beyond what products and services are widely in use—it requires looking deeper, to the codebases with which what these products and services are composed. The most challenging vulnerabilities will be those found in ubiquitous developer tools and codebases and libraries that form the foundation of other widely used code.

Suffice it to say that no single vendor or consumer will have the sufficiently global perspective on the open-source ecosystem necessary to make determinations about the true risk of an open-source library or package on their own. DHS CISA should lead assessments of open-source risk across the technology ecosystem (not limited to widely used tools in the .gov) and look to understand common dependencies. The National Risk Management Center may be better positioned to undertake short-term studies and conduct analysis to support this work in the coming year as the organization continues to realign. CISA should leverage an expanding community of SBoM producers and their data on the provenance of software to support this risk assessment work as well.

Merely tracing the most widely used packages is not enough, and while it may be the focus of near-term industry efforts, the public sector's responsibility, and indeed vested interest, is to consider risk on a longer and wider horizon. The product of these assessments is urgently needed to guide investment decisions by both the public and private sectors intended to better secure open-source software supply chains and to prioritize federal actions to proactively manage risk rather than wait for future crises.

---

[21] "CISA Adds Top Cybersecurity Experts to Join COVID-19 Response Efforts," Cybersecurity and Infrastructure Security Agency, July 22, 2020, https://www.cisa.gov/news/2020/07/22/cisa-adds-top-cybersecurity-experts-join-covid-19-response-efforts.

3. Investing in Critical Digital Infrastructure

Open-source software constitutes core infrastructure for major technology systems and critical software pipelines. The absence of US public support to secure these products, at a cost point far below what is spent annually on other domains of infrastructure security, is a striking lapse. Luckily there is a clear pathway to begin to address this issue. Before the House of Representatives, there is a proposed amendment[22] to HR 4521,[23] the America COMPETES Act of 2022, that would authorize the creation of a set of Critical Technology Security Centers inside of DHS, including one focused specifically on open-source security. The important role this amendment grants to the CISA Director in shaping how the open-source center would award funds and to what ends would in turn support our broader recommendation to create an open-source evangelist/program office in CISA. Adequately resourced, this CTSC for open source would provide a starting point for federal efforts to improve the health and long-term security of the open-source ecosystem. This committee should support the CTSC provisions of the bill when it reaches the Senate, with the understanding that a substantial portion of moneys appropriated for these centers, on the order of $20 to $30 million a year, is dedicated to this open-source mission. These funds can be used in such a way as to avoid duplicating private sector investments, focusing on secure developer tools and "foundational" infrastructure for the open-source ecosystem, to incentivize rebuilding codebases in memory-safe languages, to support audits and volunteer labor to identify and patch vulnerabilities, and to support efforts to drive security talent into this space and towards the most impactful libraries and packages in open source.

A portion of the funds dedicated to this open-source CTSC should be reserved for direct grantmaking at the discretion of the CISA Director in addition to the DHS-determined funding. These grants should involve an open application, widely circulated, to allow code maintainers and open-source projects to demonstrate their suitability and need for small (less than $500,000) support grants. This would support a degree of 'market-driven' risk assessment to complement the other, more top-down, approaches endorsed here. External observations are that CISA's ability to award grants is markedly slower and more complicated than comparable industry practice, presenting a material risk to any grant making activity. In line with this recommendation, this Committee would benefit from asking CISA directly how to radically streamline their grant-making authorities and processes.

The funding in line with this CTSC for open source doesn't need to occasion a net increase in CISA's overall budget. One of the most striking findings from last year's Sunburst campaign, and a continuing concern in the larger analysis of the cybersecurity capabilities of the .gov, is the EINSTEIN program. Hard questions should be asked about the program including:

---

[22] U.S. Congress, House Rules Committee, *Rules Committee Print 117-31 Text of HR 4521, the America COMPETES Act of 2022*, HR 4521, 117th Cong., 2nd sess., introduced in House Rules Committee February 1, 2022, https://amendments-rules.house.gov/amendments/LANGEV_068_xml220128115401119.pdf.

[23] U.S. Congress, House, *Bioeconomy Research and Development Act of 2021 [America COMPETES Act of 2022],* HR 4521, 117th Cong., 1st sess., introduced in House July 9, 2021, https://rules.house.gov/sites/democrats.rules.house.gov/files/BILLS-117HR4521RH-RCP117-31.pdf.

- Is EINSTEIN currently delivering capabilities adequate to address adversary activities and known risks in the .gov technology landscape?
- Does the program's technical roadmap deliver capabilities against likely near-term changes in these adversary activities or technical risks?
- Does the EINSTEIN program have a performance history to suggest that it would be able to deliver such capabilities in the near future if not currently planned for?
- Does the budgetary commitment CISA makes to the EINSTEIN program adequately reflect either a) the program's capabilities or b) its value against CISA's priorities?
- Does EINSTEIN, as currently constituted, map well to CISA's current authorities and relationship with the Office of the National Cyber Director especially as laid out in HR 6497, the Federal Information Security Modernization Act of 2022[24]

It is possible that in such an analysis, the EINSTEIN program may reveal significant opportunities for cost savings and more efficient programmatic approaches to obtain the technical and security capabilities currently desired by CISA.

The open-source CTSC should look to support the long-term health of the software supply chain ecosystem by curating, maintaining, and integrating security and developer software tools released across the open-source ecosystem. Working with these tools, the Center could invest federal resources to ensure these tools are accessible and easily integrated with major package managers to ensure developers can find consistent support for versioning, security checks, integrity verification, dependency mapping, and update notification across different repositories and platforms. At the core of open source is the ability to solve a problem once and enable others to solve it markedly faster and easier the next thousand times it is encountered. Good software tools are important for lowering the barrier of accessibility for good security behaviors in low-resource projects. They also make it easier for a wider array of talent to identify vulnerabilities, mitigate them, and notify consumers. Yet, maintaining these tools, including ensuring their own security against compromise and abuse, can be a costly and time-consuming task unlikely to feature in new investment schemes and funding limited to the highest priority projects. Tooling is a direct pathway to improve the security performance of developers across the open source ecosystem and the CTSC for open source can play a critical role in curating and supporting these tools for all.

In the next three to five years, US government financial support for open source and software supply chain security should be expected to rise above $100 million per year on the simple basis of the overwhelming value that open-source software provides to society and in particular to the technology industry. Efforts are underway to improve our collective ability to fight fires that have long been ignored and risk growing worse. Medium- and long-term efforts should now seek to move the open-source community and industry toward effective fire prevention. The investments contemplated here are to address urgent near-term needs and bring some measure of balance in open source investment between industry and the US government. Long-

[24] U.S. Congress, House, *Federal Information Security Modernization Act of 2022,* HR 6497, 117th Cong., 1st sess., introduced in House January 25, 2022, https://www.congress.gov/bill/117th-congress/house-bill/6497?s=1&r=1.

term changes from industry-led programs and federal policy alike will require resources and expanded funding to address this as the infrastructure health and security issue it is.

A Note on SBoMs

The expansion, and formalization, of the Software Bill of Materials (SBoM) as a leading mechanism for software supply chain transparency are nothing but encouraging. SBoMs, if widely used and rigorously implemented, will provide policymakers, vendors, and most importantly software consumers a necessary wealth of information about the products and services they depend on. These bills of materials can tell organizations a great deal about the composition of the software we use and provide information for broader risk assessment and management efforts.

But insight does not yield change, and the SBoM is not a silver bullet to substitute for a robust public-private partnership to better govern the security of open-source software, global assessments of software supply chain risk, and investments in the security of the open-source ecosystem. SBoMs provide a clear and accessible pathway to enable consumers of software to know much more about what they consume. SBoMs provide a basis on which we might build a better means of assessing or managing risk in software supply chains. Their adoption is critical, and our task now is to use this data to shift from reactive to proactive management of this cybersecurity risk.

Open source is not the problem. Software supply chain security issues have bedeviled the cyber policy community for years. Log4j is an exceptionally widely used logging program and addressing its flaws has required significant effort and public attention but it will not be the last time this kind of incident occurs. The policy community can be better prepared and help limit the consequences of these incidents but improvement will require sustained attention and partnership with industry and developers. The key for this body, and a watchword for Federal efforts to improve the security of open source, is to fund the mundane. Providing resources where industry might not, or where public attention fades, to drive structural improvements in the security of software supply chains across all developers and maintainers. Better securing software supply chains and open source code is an infrastructure problem and the same long term investment model applies. At risk is a wellspring of digital innovation but this moment of crisis also presents opportunity.

Thank you again for the opportunity to speak with you today. I look forward to your questions.