UNITED STATES COPYRIGHT OFFICE

Long Comment Regarding a Proposed Exemption Under 17 U.S.C. § 1201

[ ] Check here if multimedia evidence is being provided in connection with this comment

ITEM A. COMMENTER INFORMATION

GitHub, Inc. is the world's largest software development platform, enabling more than 56 million individual developers, students, startups, small businesses, large companies, NGOs, and governments to collaborate on building great software.

GitHub makes it easier for developers to be developers: to work together, to build on each other's work, to solve challenging problems, and to create the world's most important technologies. As part of this mission, GitHub provides tools to develop secure code and partners with developers to raise the bar for software security across the entire software ecosystem.

Contact:

Justin C. Colannino
Director, Developer Policy and Counsel
GitHub, Inc.
88 Colin P Kelly Jr Street
San Francisco, CA 94107
royaljust@github.com

ITEM B. PROPOSED CLASS ADDRESSED

Proposed Class 13: Computer Programs – Security Research

ITEM C. OVERVIEW

GitHub submits this comment to enable developers to improve software security and emphasize the importance of non-infringing security research for the software supply chain. Before turning to the specifics of the proposal before the Office, we wish to emphasize four broader points.

Security research makes us all safer. In modern software development, software is almost never written from scratch. Instead, software "depends" on other software, often open source

software, written by third parties.[1] Everything – the cloud, networks, applications – is built on these "dependencies" that constitute the software supply chain.

Security of the software supply chain is an issue critical to the infrastructure on which our society runs. As the recent SolarWinds supply chain compromise highlighted,[2] attacks on the integrity of software components occur even before they are in the customer's system. That a piece of software incorporates cryptography does not make it less important for that component to be examined; instead, it makes that examination all the more important. Indeed, the malicious software responsible for the SolarWinds compromise was unwittingly cryptographically signed by the victim.[3] Developers looking to secure the software supply chain should be encouraged—not discouraged by the threat of 1201 liability.

Important security work is done by all kinds of developers. The Opposition comment of Joint Creators and Copyright Owners focuses on its contested view that the present exemption does not hamper scientific dialogue, academic peer review, or classroom teaching. But other researchers who receive little focus in that comment are critical contributors to security. Not everyone who contributes to the safety and security of software is a professor at a prestigious university. Important contributions to security come not only from academia, but also from those working in enterprises—and from a large community of civic-minded independent researchers who understand the broader impacts of software security, even if not for their day job. While the Opposition comment of SIIA frames business motivations as inconsistent with good-faith security research, nothing could be further from the truth. On the enterprise side, FireEye – a publicly traded company – uncovered the SolarWinds breach while probing its own network and worked with others to diligently disclose the extent of the compromise. On the independent researcher side, those who help make GitHub safer by reporting bugs to GitHub's Bug Bounty program are not always affiliated with big companies or academic institutions.[4] All of these contributions are equally critical to the safety and security of software systems.

FUD chills. In the world of software development, "FUD" refers to "Fear, Uncertainty, and Doubt"—the factors that eat away at a developer's ability to move forward confidently with a project. Section 1201 is a source of FUD as applied to good faith security research. It can be asserted even when a court has decided that there is no copyright infringement of the underlying

[1] These dependencies are often so numerous and layered (software that a developer depends on can depend on other software all the way down) that GitHub provides a "dependency graph" feature so that developers can understand what software they depend on for the software they write. https://docs.github.com/en/github/visualizing-repository-data-with-graphs/about-the-dependency-graph

[2] https://www.zdnet.com/article/microsoft-fireeye-confirm-solarwinds-supply-chain-attack/

[3] https://www.solarwinds.com/sa-overview/new-digital-certificate

[4] https://bounty.github.com/bounty-hunters.html

work, as in the *Corellium* case.[5] It's a reason why a developer can't be confident that there won't be repercussions for engaging in legitimate, non-infringing security research and related development activities. It's a reason why they might decide to do a different project, with less impact, that doesn't help make us all safer to the same extent. That is why this exemption process should be focused on FUD elimination. The Halderman et al. proposal draws clearer lines out of fuzzy lines in the current exemption, giving more certainty to researchers, academics, and enterprises conducting security research. It should be taken seriously.

Security researchers benefit from automation and virtualization services. Just the open source dependencies in the software supply chain number in the millions. GitHub is home to over 100 million projects. And other popular software ecosystems also have projects in the millions. One, NuGet, has over three million different available dependencies.[6] Another, npm, has over 1.3 million.[7] Indeed, the complexity of the software dependency ecosystem is ripe for abuse as demonstrated recently by research into "dependency confusion" attacks that impacted many large technology firms.[8]

With this volume and complexity, analyzing the security of all components requires engaging in a wider umbrella of foundational activity, including automation, vulnerability testing, and virtualization of different environments where the dependency can be run. For example, there are just shy of 300 developer projects hosted on GitHub that advertise themselves as providing security automation.[9] These kinds of security automation activities – like the software virtualization in *Corellium* – should fall into the security research exception to avoid FUD for these important automation services. The DMCA should not be a tripwire for this type of critical automation activity.

ITEM D. TECHNOLOGICAL PROTECTION MEASURE(S) AND METHOD(S) OF CIRCUMVENTION

The technological protection measures and methods of circumvention with respect to the proposals are as set forth in the Long Comment of Halderman et al.

ITEM E. ASSERTED ADVERSE EFFECTS ON NONINFRINGING USES

GitHub appreciates the Office's recommendation that the current security-research exemption be maintained, and its willingness to consider an expansion of that exemption in the current proceeding. More can be done to ensure security researchers and software developers have confidence in their work to make the software supply chain safer.

---

[5] *Apple Inc. v. Corellium, LLC*, No. 19-81160-CIV, 2020 WL 8642269, at *16 (S.D. Fla. Dec. 29, 2020) (holding that security research activity that was adjudicated to be fair use may nonetheless violate 1201).

[6] https://www.nuget.org/

[7] https://github.blog/2020-03-16-npm-is-joining-github/

[8] https://medium.com/@alex.birsan/dependency-confusion-4a5d60fec610

[9] https://github.com/topics/security-automation

This comment focuses on the aspect of the Halderman proposal that GitHub sees as particularly important to its community of developers: removal of the word "**solely**" in reference to the purpose of the activity. This provision appears in bold in the below text.

> (i) Computer programs, where the circumvention is undertaken on a lawfully acquired device or machine on which the computer program operates, or is undertaken on a computer, computer system, or computer network on which the computer program operates with the authorization of the owner or operator of such computer, computer system, or computer network, **solely** for the purpose of good-faith security research and does not violate any applicable law, including without limitation the Computer Fraud and Abuse Act of 1986.

This limitation, without sufficient recognition of its flexibility, may unnecessarily chill activity undertaken for security research purposes that helps improve security and makes us all safer.

Circumvention of software for security research and testing purposes may involve a range of activities undertaken to create secure software ecosystems. A developer setting out to find security flaws may also need to perform other noninfringing activities that relate to creating secure and stable environments, such as reverse engineering for compatibility, fixing a computer program's "bugs", or monitoring applications to see how they use a user's private data, all of which may be done to ensure the security of software, platforms, computing environments, and users.

Developers and researchers do not do only one thing at a time or act out of only a single animating goal. So long as their activity is consistent with undertaking good-faith security research, it should not matter if a specific step be "solely" focused on security – it can and should embrace activities that lead to stable computing environments because stable computing systems are consistent with ensuring secure computing ecosystems. The purpose of the security exemption must be to incentivize, rather than discourage, practices consistent with good faith security research.

Developers and the public have a vested interest in improving the software we all depend upon. There is a tremendous amount of overlap between normal quality assurance work that developers undertake for those who rely on their code and the encapsulated but more narrow heading of security research. When performing this quality assurance work looking for all software issues – including security holes – there is no good reason that just because a developer finds and fixes a bug not impacting security in addition to one impacting security that the work should fall outside of the exemption. Similarly, a step that may be a prerequisite to creating the right type of automation for security research, like virtualizing software in order to look for flaws or vulnerabilities, should not itself be disqualified merely because that specific step is not "solely" related to security research. Eliminating this type of ambiguity is an example of something the Copyright Office can achieve by accommodating the goals of the exemption proponents to achieve clarity for activities consistent with and in furtherance of good faith security research.

Developers conduct valuable and non-infringing security research for a variety of purposes consistent with the purpose of improving security. The Copyright Office should ensure that Section 1201 continues to incentivize, not threaten, these valuable activities. The exemption

should not be limited to only one purpose, but should provide enough flexibility that ancillary and beneficial activities consistent with good faith security research continue to fall within it.

Nobody gains if that developer decides not to engage in circumvention out of concern over 1201 liability. And nobody gains if developers analyzing the supply chain and engaging in activities consistent with good faith security research operate in fear despite 1201 exceptions that were enacted to achieve secure computing ecosystems.

DOCUMENTARY EVIDENCE

GitHub requests that the online sources and information cited and/or linked to herein be considered as documentary evidence in support of GitHub's comment.