Consumer Perception of Open Source Software Security and Privacy

Christopher Ferrer

ENC2135: Peyton Wahl

February 2, 2024

Open source software is a type of software that has publicly available code and allows users of that software to freely download, use, modify, and redistribute their personal copy of the software, assuming it is published under the same copyright license. All open source software that has been created since the early 1990s has been published with a GNU General Public License copyright license: a copyright license created by programmer Richard Stallman that has led to the modern definition of open source software, allowing users to freely use, modify, and share the software. One of the most notable examples of open source software is the operating system Linux. Linux is a direct competitor to Windows and MacOS, allowing users to customize their personal experience however the user desires and avoid 'software bloat' - a term in computer science that refers to unnecessary and/or unremovable programs or code that take up space on a device and encumber its performance. For example, iPhones used to be unable to uninstall the iTunes, Apple Store, Podcasts, and Stocks apps that come with every version of iOS. However, with the release of iOS 10, Apple started to allow users to delete these apps that many claimed bloated their iPhones (Apple put out a statement with the release of iOS 10 claiming that the built-in, preloaded apps are "very space efficient", but many users still consider them a nuisance). [1] [2] While customization in open source software is one of the most prevalent reasons for its use, especially among individuals with technological backgrounds, it still has major benefits for those who may know very little about tech, such as transparent business practices and the perception of increased security. While consumers may perceive open source software as more secure, for various valid reasons, peer reviewed research has seemingly come up with no significant difference between the security practices of open source software

[1] Chris Welch, "IOS 10 Will Let You Uninstall the Apple Apps You Never Use," The Verge, last modified June 13, 2016, https://www.theverge.com/2016/6/13/11923112/apple-ios-10-delete-stock-apps.
[2] Apple, "Remove Built-in Apple Apps from the Home Screen on Your IOS 10 Device or Apple Watch," Apple Support, n.d. https://support.apple.com/en-us/100567.

compared to proprietary software; nonetheless, this beneficial perception of open source software does have merit to it, including protecting the user of the software from the software and its developers.

The first main point of contention for open source software users is security. As noted before, peer reviewed research has typically concluded that there is no significant difference in the security between open source and closed source software, although there is a major difference in design philosophy when it comes to security (it is incredibly important to know that recent research on this subject is lacking; most research on the average security of open and closed source software is at least a decade old, which is incredibly outdated for modern technological standards).[3] Open source software relies on the vast population it can reach to check its code and ensure that it is secure and up to standard. The underlying concept behind this is that errors are more likely to get fixed (if they exist) if there are more checks in place for that error to be found. On the other hand, closed source software relies on a design philosophy widely referred to as “security through obscurity”. This design philosophy is focused on the idea that it is hard to find flaws or vulnerabilities in code if a would-be attacker cannot see the code in the first place. In theory and isolated environments, both design philosophies could work incredibly well. However, they both also have critical flaws. Open source software could easily be incredibly secure, however it relies on the contentious idea that humans are naturally good and will report any problems they find instead of abusing them for their own personal gain. Obviously, not all individuals have this good-mannered inclination and may abuse any issues they find instead of promoting the common good. This is the first main argument against open source software, and a valid argument at that. Closed source software, alternatively, takes a near

---

[3] Russell Clarke, David Dorwin, and Rob Nash, "Is Open Source Software More Secure?," Courses.cs.washington.edu, n.d. https://courses.cs.washington.edu/courses/csep590/05au/whitepaper_turnin/oss(10).pdf.

polar opposite contentious idea and assumes that anyone who does want to see the source code will either expose bad practices or abuse vulnerabilities they discover. This can typically be effective, but the main issue with closed source software's design philosophy is the core belief that it is hard, if not impossible, to find vulnerabilities without seeing the source code. Any experienced programmer will be able to explain that while it may be hard, it is most definitely not impossible to find and exploit vulnerabilities without knowing intricate details of any system. A highly simplified, yet apt comparison to closed source software would be a balloon artist. The initial state of the input, a balloon, is known. However, the balloon artist would then take that balloon, close your eyes, do their magic, and tell you to reopen your eyes once they have finished their work. While it is hard for any non-balloon artist to determine how the finished project was even created, if this same experiment were tested on an experienced balloon artist, they would likely be able to explain major steps in the process to get to the finished product. The same core principles apply to experienced programmers with closed source software. Thus, while not being able to see the 'magic', the programmer can still assume certain major processes that are taking place and semi-accurately pinpoint where in the process a vulnerability may occur.

Now that the design philosophies have been identified, the identification and repair of vulnerabilities in software is the next major concern. While there is a lot of complicated mathematics, technology, and research behind identifying vulnerabilities and weak points in open source software, fixing those vulnerabilities typically requires understanding why they are occurring, what is causing them, and understanding the ramifications of any potential fixes. A common rule of thumb in programming is called "Kernighan's Law" - a general rule claiming that debugging and fixing code is about twice as hard as programming it, meaning that you are not qualified to debug code that you wrote if you wrote it as cleverly as possible. This comes into

play heavily in closed-source software, as programmers are often organized into small teams. If the team cannot catch a vulnerability before it is abused, it will likely take a long time for them to fix the vulnerability, or they may have to recruit outside help if necessary. In open source software, this issue is typically avoided by users of the software being able to suggest edits to the official releases of the software; someone more qualified to debug the code written as cleverly as possible by the original developers may see the project and want to help out. This may also lead to a possible downside however, as those *unqualified* to fix a certain error may suggest an edit that is then erroneously accepted. Even so, it is likely that the design philosophy of open source comes back into play in this circumstance, and other users correct the suggested edit, claiming that it is worse or less secure in some way.

While security from attackers or potential hackers is important, an equally important form of security to take into consideration is the security and privacy of the user and the user's data from the developers of any given program. **In later drafts, this paragraph will go on to talk about the transparency benefits of open source software over closed source software and why a potential user may choose open source over closed source software.**

In later drafts, this paragraph will transition from transparency of open source software to the particular user base of open source software in general, as briefly hinted at in the introductory paragraph.

In later drafts, this paragraph will serve as the conclusion, ultimately concluding that open source software is typically better perceived and a net-positive for almost all involved with the project. This conclusion will also explain some beneficial niches of open source software that cannot be fulfilled by closed source software, using this as a final argument as to why open source software is valuable.

Bibliography

Apple. "Remove Built-in Apple Apps from the Home Screen on Your IOS 10 Device or Apple Watch." Apple Support. n.d. https://support.apple.com/en-us/100567.

Clarke, Russell, David Dorwin, and Rob Nash. "Is Open Source Software More Secure?" Courses.cs.washington.edu. n.d. https://courses.cs.washington.edu/courses/csep590/05au/whitepaper_turnin/oss(10).pdf.

IBM. "What is Open Source Software?" IBM.com. n.d. https://www.ibm.com/topics/open-source.

Imtiaz, Nasif, Aniqa Khanom, and Laurie Williams. “Open or Sneaky? Fast or Slow? Light or Heavy?: Investigating Security Releases of Open Source Packages.” IEEE Transactions on Software Engineering 49, no. 4 (2023): 1540–60. https://doi.org/10.1109/TSE.2022.3181010.

Li, Yuancheng, Longqiang Ma, Liang Shen, Junfeng Lv, and Pan Zhang. “Open Source Software Security Vulnerability Detection Based on Dynamic Behavior Features.” PloS One 14, no. 8 (2019): e0221530–e0221530. https://doi.org/10.1371/journal.pone.0221530.

Suman, Rajiv Ranjan, Bhaskar Mondal, and Tarni Mandal. “A Secure Encryption Scheme Using a Composite Logistic Sine Map (CLSM) and SHA-256.” Multimedia Tools and Applications 81, no. 19 (2022): 27089–110. https://doi.org/10.1007/s11042-021-11460-4.

Welch, Chris. "IOS 10 Will Let You Uninstall the Apple Apps You Never Use." The Verge. Last modified June 13, 2016. https://www.theverge.com/2016/6/13/11923112/apple-ios-10-delete-stock-apps.