Career Paper: The Secure Software Developer

Cali J. Zuk

Department of Cybersecurity, Old Dominion University

Professor Diwakar Yalpi

CYSE 201S: Cybersecurity and the Social Sciences

November 24th, 2024

Introduction

The career field of software development has been growing rapidly for the past decade. Despite recent economic issues slowing the growth of the field, it is still considered a premier line of work. In this paper, the term software developer and secure software developer are used interchangeable to refer to the career field of secure software developer classified according to the Workforce Framework for Cybersecurity developed by the National Institutes of Standards and Technology (*Workforce Framework for Cybersecurity (NICE Framework) | NICCS*, 2024). Software developers create all of the applications that the American consumer works with on a daily basis. Traditionally, security and accessibility were considered afterthought for developers; they were something to be aspired for, but not necessary to include in their applications if time was short. Developers seldom devoted the time required to ensure secure software practices were followed (Kalhoro et al., 2021) and that the applications were accessible to all end users (Ferati & Vogel, 2020). The fundamental concepts of social science intersect with software developers when it comes to these two aspects, as lack of secure software design results in privacy exposures for end users and a lack of accessibility education results in software that is often unusable to those with disabilities or may include dark features that leave users vulnerable to victimization.

Secure software design and privacy

Secure software development is a critical first step in ensuring the right to privacy of end users. In many cases, intensive development cycles and crunch can result in developers producing applications that have not been thoroughly tested for vulnerabilities. This churn can result in poor refactoring practices and increased code complexity that can better hide bugs (Shin

et al., 2011). Crunch time at large companies feels necessary, but this is generally due to arbitrary deadlines imposed by corporate leadership rather than intentional neglect by developers.

While not a criminal act, software developers often resort to lines of thinking quite similar to those of neutralization, the two most common being that security is not their responsibility (Kalhoro et al., 2021), or that if the app is insecure, it was for the good of the company as ensuring a timely release date was more important than implementing good secure coding practices; after all, they can always be added later (R. L. Jones & Rastogi, 2004). If the rationale behind those justifications seems familiar, it is because they are denial of responsibility and appeal to a common good from neutralization theory in criminology. They are excuses preached by both professionals and the corporations that employ them.

The need for secure coding practices has been known for decades, as even in the early internet, academic researchers were sounding the alarms (R. L. Jones & Rastogi, 2004). The end result of all these practices is often the failure to ensure user data confidentiality, but poor security in coding can also deny availability of resources as seen in the Crowdstrike outage that obliterated corporate networks the world over and was caused by insecure C language practices in a low level kernel driver (Crowdstrike Holdings, Inc., 2024).

Failures to ensure that software is secure at release can leave journalists and activists vulnerable to harassment or even death as in the alleged case of Washington Post journalist Jamal Khashoggi (Priest, 2021). The software potentially used to monitor his movements was the Pegasus Spyware, developed by Israeli firm NSO Group. It relies on a number of vulnerabilities in both Apple's iOS and Google's Android operating system to work (Kareem, 2024). Dissident and journalist rights are threatened by insecure coding practices that allow malware to be

installed on the devices they rely on to maintain their personal safety and security. Low level vulnerabilities in operating systems present an intolerable threat to the privacy of individuals, but as the Pegasus Spyware also showed, it they can also be used by state actors in cyberwarfare operations (S. Jones, 2022).

As the evidence cited in this section has shown, it is critical that software developers view themselves as secure software developers and treat security as a core requirement of their codebases. Not only must they include security as a core requirement, but they must also be aware of the social connections and systems that their applications will interact with and the consequences that poor code can create for the dissidents, activists, oppressed groups, journalists, and governments that rely on their codebases.

Accessibility Concerns

The second important prong for secure software developers is the accessibility and usability of their applications. Accessible systems are ones that are usable to all users; this means that there are system is also designed for the disabled, those on non-laptop or non-desktop devices, and those with limited access to broadband (The Mozilla Foundation, 2024). Ideally, systems should be designed from the ground up to be accessible, but many software developers will simply add accessibility suites in at the end of development via standard libraries, plugins, or extensions. As in the case of secure coding practices, this lack of accessibility by design is often a result of poor levels of education (Ferati & Vogel, 2020), ableism inherent in the industry (Mason, 2024), belief that the cost may not be worth the benefit (Leitner et al., 2016), and poor understanding of accessibility standards (Silva et al., 2019). Secure software developers largely work from high end desktops and develop, test, and deploy apps to those devices. These

developers often do not rigorously test their applications on other devices such as smartphones and other mobile devices and this can affect the usability on those devices. While many government standards require accessible design, developers are often unable to meet them due to the aforementioned causes and if these laws have an enforcement mechanism it is seldom used. The end result of these factors is an internet that is largely inaccessible to many users with disabilities, especially visual ones (Silva et al., 2019). The solution is to increase mandatory education for secure software developers in the realm of accessibility, properly enforce existing accessibility laws, and create more accessible documentation for secure software developers to utilize.

Conclusion

In the Information Age, software is king because it has become the primary communication method of the masses. From social media to enterprise workplaces to streaming, software has captured all of us. This places great responsibility in the hands of the modern day craftspeople who create these tools and platforms for our consumption. Their choices in design can allow state intelligence services to suppress dissidents, facilitate cyberbullying of the young, weaponize insecurity in young people, and deny this modern day Library of Alexandria to those with disabilities and the lower classes. Software is a tool of great societal uplifting, but also one of repression, bigotry, and discrimination. It is critical that secure software developers embrace this great task to ensure that the future of the Internet is secure, fair, and equitable for all.

Works Cited

Crowdstrike Holdings, Inc. (2024). *Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024*. Crowdstrike. https://www.crowdstrike.com/wp-content/uploads/2024/08/Channel-File-291-Incident-Root-Cause-Analysis-08.06.2024.pdf

Ferati, M., & Vogel, B. (2020). Accessibility in Web Development Courses: A Case Study. *Informatics*, *7*(1), 8. https://doi.org/10.3390/informatics7010008

Jones, R. L., & Rastogi, A. (2004). Secure Coding: Building Security into the Software Development Life Cycle. *Information Systems Security*, *13*(5), 29–39. https://doi.org/10.1201/1086/44797.13.5.20041101/84907.5

Jones, S. (2022, May 10). What we know about Spain's cyber-espionage spyware scandals. *The Guardian*. https://www.theguardian.com/news/2022/may/10/what-we-know-about-spains-cyber-espionage-spyware-scandals

Kalhoro, S., Rehman, M., Ponnusamy, V., & Shaikh, F. B. (2021). Extracting Key Factors of Cyber Hygiene Behaviour Among Software Engineers: A Systematic Literature Review. *IEEE Access*, *9*, 99339–99363. IEEE Access. https://doi.org/10.1109/ACCESS.2021.3097144

Kareem, K. M. (2024). A Comprehensive Analysis of Pegasus Spyware and Its Implications for Digital Privacy and Security. *International Journal of Intelligent Systems and Applications in Engineering*, *12*(3), Article 3.

Leitner, M.-L., Strauss, C., & Stummer, C. (2016). Web accessibility implementation in private sector organizations: Motivations and business impact. *Universal Access in the Information Society*, *15*(2), 249–260. https://doi.org/10.1007/s10209-014-0380-1

Mason, E. (2024, May). *Confronting ableism to build a more inclusive web*. https://assistivlabs.com/articles/confronting-ableism-to-build-a-more-inclusive-web

Priest, D. (2021, December). UAE agency put Pegasus spyware on the phone of Hanan Elatr, Jamal Khashoggi's wife. *The Washington Post*. https://www.washingtonpost.com/nation/interactive/2021/hanan-elatr-phone-pegasus/

Shin, Y., Meneely, A., Williams, L., & Osborne, J. A. (2011). Evaluating Complexity, Code Churn, and Developer Activity Metrics as Indicators of Software Vulnerabilities. *IEEE Transactions on Software Engineering*, *37*(6), 772–787. IEEE Transactions on Software Engineering. https://doi.org/10.1109/TSE.2010.81

Silva, J. S. e., Gonçalves, R., Branco, F., Pereira, A., Au-Yong-Oliveira, M., & Martins, J. (2019). Accessible software development: A conceptual model proposal. *Universal Access in the Information Society*, *18*(3), 703–716. https://doi.org/10.1007/s10209-019-00688-5

The Mozilla Foundation. (2024, August 2). *What is accessibility? - Learn web development | MDN* [Documentation]. MDN Web Docs. https://developer.mozilla.org/en-US/docs/Learn/Accessibility/What_is_accessibility

Workforce Framework for Cybersecurity (NICE Framework) | NICCS. (2024, June 17). https://niccs.cisa.gov/workforce-development/nice-framework