Security in Software

CSC 424 Software Design

Hunter Johnson

4/23/21

The secure nature of software has been a long-standing concern. Users who trust any software with their information want to be assured that the chance of it being stolen or leaked is virtually none-existent. Businesses with software contracts, or even in-house development, want to prevent the reveal of their clients' information and other forms of sabotage. Developers themselves want to guarantee that their liability when creating any software will not put them in a position of culpability.

This growing need can be realized within the industry media as various papers, op-eds, and even listicles became more common in efforts to address this need. This is not to say that it has only recently become an important topic, as it is addressed in earlier textbooks, only that the importance has seen a perceptual rise in recent years. The growing library of information espousing the importance of secure software will be helpful when available. It will be important to define what makes software secure, as well as how to grade tools on their effectiveness, and especially for the individual developer to perform these practices while on the job.

Secure software is any software to which the application of security techniques have been applied. The result of which should give an innate ability to the software to prevent vulnerabilities or deter outside intrusions. Not all software is built with these techniques in their development life cycle, or by those who concern themselves with the integration of high-level security. Secure software quickly becomes defined as secure personnel and secure hardware.

The most common method of securing personnel comes in the form of training. This training is meant to take place early on, either at orientation to establish expected behavior or during initial training to reinforce positive practices. Beyond basic training changes it is also recommended to move the importance of security to a higher place within the software development life cycle. Changes to development documentation are a recurring item of advice.

Securing hardware is as important digitally as it is physically. Limiting access to qualified personnel only ensures that the human variable is restricted to trained individuals and reduces the chances of malintent. Digital security for hardware exists as automated tools which continuously check for open ports or security misconfigurations. The ability to determine which tools are strictly necessary becomes an important skill. [1]

Two such tools are the Static Application Security Testing tool and the Dynamic Application Testing tool. The former is meant to analyze code and find vulnerabilities during

compile time. The latter is meant to analyze running code and discover security vulnerabilities. Once again, it is recommended to run these tools consistently and continuously to ensure adequate protection of data. [2]

Available tools for Static Application Security Testing are varied and are thus subject to rankings and review sites. Among the top of these tools are GitHub and GitLab, the former of which has benefitted from increased investment from Microsoft. The next best three, according to one review site, are: Appknox, a mobile app security platform; CodeScan, used exclusively for Salesforce and DevOps; and Coverity from the Synopsys group. [3] Each of these applications maintain strengths based on target audience.

The tools for Dynamic Application Security Testing are also subject to rankings and review sites. The top two for this method of security testing are Pentest-tools.com, which is geared toward website vulnerabilities, and GitHub, again benefitting from a large user base and Microsoft investment. The remainder of the top five consist of: GitLab, benefitting from its nature as an open-source DevOps platform; Appknox, taking its place as another multirole testing platform; and Netsparker by Invicti, boasting a unique combination of DAST and IAST. [4] Knowing which tools are most helpful also requires knowing when to apply these tools.

Proper training, mentioned before, is also an effective measure in creating secure software. It is recommended to include it earlier in the software development life cycle at an earlier time. It is also important that this training occur regularly, awareness training for all employees, such as vigilance against phishing attempts, will be one of the most effective training methods available. Training developers to create more secure code can be a cheap, long-term method as well. [1]

The efforts of the individual developer to create secure software, while on the job, could be considered symbiotic. Initial efforts would include taking note of standard coding practices, within the office or similar workspace, and doing ones best to ensure code written is up to those standards. Familiarizing oneself with provided tools will lower the time it takes to correctly apply those tools to one’s code, reducing the time it takes to bring code to review and also reduce

the chance of code causing issues later on. Best practices are not difficult, they simply require an investment of time which will save a greater amount of time in the long run.

Overall, the methods to develop secure software which will protect the data and privacy of, not only your clients, but your employees as well, are bettered by diligence and practice. The tools with which these same goals, and in essence legal protections, are achieved protect numerous people daily. The practices and habits of a given business should be judged by the individual based upon a given standard of security. An ability to identify and create secure software simply takes practice.

References

[1] Synopsys Editorial Team, "Top 10 software security best practices | Synopsys," 29 June 2019. [Online]. Available: https://www.synopsys.com/blogs/software-security/top-10-software-security-best-practices/. [Accessed 20 April 2021].

[2] T. D. Edmiston, "What is Software Security?. Build better, safer software… | by Taylor D. Edmiston | theFramework | Medium," 28 August 2019. [Online]. Available: https://medium.com/the-framework-by-tangram-flex/what-is-software-security-e03a5ee7a6b5. [Accessed 20 April 2021].

[3] G2, "Best Static Application Security Testing (SAST) Software in 2021 | G2," G2, [Online]. Available: https://www.g2.com/categories/static-application-security-testing-sast?__cf_chl_jschl_tk__=eb440fad70290e35dc69cebf50a729e4207ebf84-1619043421-0-AT_-wlZSZ91iBuHTMfPUqUbOjJrOTtBg6yk5KHpe3mfqBv0_C69UB0eQq_feNcA9UJHkBbFqkZfRf5hKRvvm-a09X-_orgmFBM3ABcztkWGNj5. [Accessed 20 April 2021].

[4] G2, "Best Dynamic Application Security Testing (DAST) Software in 2021 | G2," G2, [Online]. Available: https://www.g2.com/categories/dynamic-application-security-testing-dast?utf8=%E2%9C%93&order=top_shelf. [Accessed 20 April 2021].