Software Security

CSC 424 Software Engineering II
Courtney Hill
4/23/2021

In the world of computing, software development is a set of computer science skills dedicated to the process of creating, designing, and deploying software. Software is a set of instructions that tell a computer what to do. There are three basic types of software: system software, programming software, and application software. System software is used for operating systems, disk management, and hardware management. Programming software provide tools such as text editors, compilers, and debugger. Application software or apps is used to help users perform tasks. Each of these types of software require skilled computer scientist to produce code in a safe and secure manner. How does one produce safe and secure code? This paper will discuss certain guidelines and practices that a coder can take to ensure that what they produce secure efficient code, and ways to test using certain tools [1].

Software security breaches are quickly becoming the norm in this day and age. Until most recently the only response to these types of information breaches would have been to provide patches to fix the vulnerability or blame the users for their lack of caution [3]. Most breaches can be attributed to software design defects. This has caused a rise in internal and external penetration of these software systems. Many organizations rely on software systems for their day to day operations even our government. Due to this increased use of software there has been increased scrutiny of software security [2]. It has come to light that a large percentage of the software deployed in industrial and commercial applications is of poor quality and contains many flaws that are being taken advantage of by cyber-attacks [3]. A security risk is the probability of sustaining a loss of a specific magnitude during a specific time period due to a failure of security system [2]. One can incorporate certain measures to improve the situation like examining the final production code and look for possible problems or plan for security in the beginning [3]. When building secure software, security techniques have to be implemented in all stages of the software engineering [2]. Meaning, that these development techniques be in every phase of the software development phase. From requirements in design, implementation, testing, and deployment [2]. In a recent study, Microsoft found that 50% of software security problems were caused by design flaws [2]. In a study published called *Software assessments, benchmarks, and best practices conducted* by Jones (2000) conducted on hundreds of software projects stated that defects content of released software varies from 1 to 7 defects per thousand lines of new and changed code [2].

Development methodology is important to the production of secure systems. The main idea in proposed methodology is that security principles should be applied at each stage of the design process. In the requirements stage, use cases need to be define the required interactions of the system. This determines the rights needed by each individual; a need-to-know policy. The analysis stage determines the conceptual model where repeated applications of the Authorization pattern realize the rights determined by the use cases. Analysis patterns can be built with predefined authorizations according to their use cases. During the design stage, the interfaces can be secured using the Authorization pattern. In each level we use patterns to represent appropriate security mechanisms. Lastly, the implementation stage requires reflecting in the code the security rules defined for the application. These rules are express as classes, associations, and constraints. In this stage certain security packages, such as firewall products or cryptographic packages, can be selected.

It has been stated that security requirements are normally poorly specified due to three things: inconsistency in the selection of requirements, inconsistency in the level of detail and a having few standard requirements on security solutions [2]. Security is usually seen as having a login, backup, and access control [2]. Software security needs to be planned from the beginning, and security principles need to be applied throughout the whole life cycle [3]. There have been proposed methods for improving software security by researchers that include correctness-by-correction, which operates on the principle that errors should not be introduced in the first place and that errors should be removed as close as possible from the point at which they are introduced. The correctness-by-correction method produced defect densities ranging from 0.04 to 0.75 defects per thousand lines [2]. Another method for secure software is the Cleanroom, which incorporates incremental development, functional-based specification and design, correctness verification, and statistical testing. The overall performance of applications that use the Cleanroom method range from 0.1 errors per kilos of lines of code [2].

Sodiya stated that there are 3 software security goals for a computer system: confidentiality, integrity, and availability. Confidentiality has to do with preventing unauthorized disclosure of software resources (code, data, documents, files, etc.). Integrity, meaning to prevent unauthorized modification of software. Availability concerning unauthorized denial of services of the software resources [2]. Once one has incorporated secure coding practices and developed some form of software, one of the most important stages of development is the testing stage. There are hundreds, if not thousands of software testing tools available on the market. One has to choose a competent software testing tool to run the programs through for accurate analysis. There is different testing software for different areas of the software development lifecycle [6]. The main testing tools include ones for functional testing, non-functional testing, automation testing, and agile testing [6]. Here are some software testing tools and descriptions of what each offer:

1) One of the more well-known and established software testing tools. Selenium is a tool for automating web applications for testing with multiple solutions. This software stands out because of its ability to support multiple frameworks, and also offers the greatest automated browser testing functionality [6]. Selenium helps testers to write test in programming languages like Java, PHP, C#, Python, Groovy, Ruby, and Perl [7].

2) This is a testing tool for web and WPF applications designed for functional, performance, load, and API testing. This software contains an easy interface to navigate and intuitive icons and menu systems [6]. Telerik offers one solution to automate desktop, web, and mobile application testing. The basic languages that are supported by this platform is HTML, AJAX, ASP.NET, JavaScript, Silverlight, WPF, and MVC [7].

3) Ranked as a top automated testing software that lets you generate and execute tests on all Oss, browsers, and devices [6]. Being a free platform, its software is built on open-source automation frameworks. This tool includes a full package of powerful features that help overcome common challenges in web UI test [7].

4) Offers a complete pace of various automated testing solution for web, software, database, API, mobile app, regression test suite maintenance, optimization, and automation testing.

5) This software is used by large corporations such as Visa and Walmart and is a cloud-based platform for we and mobile app testing [6].

Each of these platforms offers unique abilities for different testing necessities. As mentioned before this is just a few of the hundreds of platforms that one can choose from. Depending on finances and software capabilities, one needs to find a testing tool that best suits their needs.

So, the next question to be asked is what would I do to make sure that I create secure software? As suggested above, the best approach would be to make sure to incorporate security measures from the beginning of the development process. When I began a new project, I find it helpful to break everything down into simple steps; try to complete one facet at a time. One a program is broken down into simple steps, I can stay on top of simple practices, so they do not overwhelm me by the end of the project. Naming practices are one way to ensure secure code. Having intuitive names for variables will help oneself, as well as others if they need to step through or update certain files. When one breaks a program down to simple steps, it is also best to keep the program as simple as possible. If there needs to be extra components added to the project; it is good to have a basic skeleton as an original file on a main branch that others can pull to their technologies. The positive thing about most of the modern technologies is that there is automation. Automation is an effective way to incorporate security in your code. It reduces the chance of human error and puts important processes on autopilot. It was taught to us to think of the craziest way that a user could break your software. Building code to combat attacks allows the developer to build appropriate defenses. Walk through a compromise and see how easy a data breach could happen if you write code or design the application wrong. As a new programmer embarking as an entry level programmer. I hope to also, get tutelage from those more experienced and take the practices that I already use and incorporate new techniques to create secure applications.

[1] “What is software development”, IBM Developer Newsletter,

[2] Sodiya, A. S., S. A. Onashoga, O. B. Ajayi; “Towards Building Secure Software Systems”, Issues in

Informing Science and Information Technology, Vol. 3, 2006

[3] Fernandez, Eduardo B.; “A methodology for secure software design”, Conference:

Proceedings of the International Conference on Software Engineering Research and

Practice, SERP '04, June 21-24, 2004, Las Vegas, Nevada, USA, Volume 1

[4] Johnson, Patricia; “Secure coding: a practical guide”, WhiteSource, 2019 June 13.

[5] Howard, Michael; “Building more secure software with improved development processes”

IEEE Computer Society, November/December 2004

[6] Boog, Jason; “10 latest software testing tools QAs are using in 2021”, QA Lead, 2021

January 4.

[7] Satasiya, Pratik; “Top 10 automated software testing tools”, DZone, 2019 November 25.