Software Security Paper
CSC 424 G001 Software Eng II
Charles Young
4/23/21

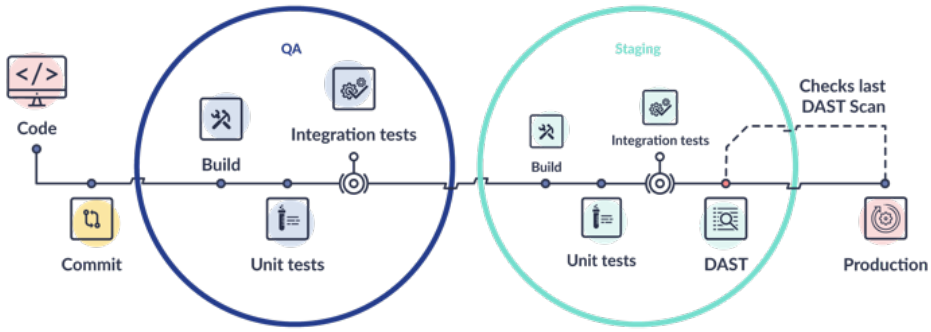
Before we dive deep into what constitutes secure software, I'd like to first cover the base of software in general. Software is a collection of instructions and data that tell a computer how to work and what to do. This can be as simple as a game of pong or as complex as AI drone tracking. Because software is so versatile, people use it for quite literally any and everything. The government uses software, banks use software, everyday moms on the street use software…and this is exactly where software security comes into play. Because everyone uses software, valuable info is often stored with it as well which makes it very lucrative to hackers, people that use technology to access important data. Secure Software thus is software developed or engineered in such a way that its operations and functionalities continue as normal even when subjected to malicious attacks. The systems and resources in its environment remain safe and the attacks detected and removed. This means that the software can detect and fight against any attack while also continuing to function normally.

There are many characteristics you could use to describe secure software, one being a secure database. The database is where most of the sensitive data is going to be stored so it's only natural this must be protected. One of the most common database attacks is an SQL injection, which inserts bad code into the software which allows for back-end executing of malicious queries. This essentially gives the hacker control over the database and all sensitive data within it. As well as security of the database, the data should also be encrypted as well. Another characteristic of secure software is Data Validation, which ensures that input data is safe and accurate. This is very important as malicious input can be carried on through the system to output. Another example deals with access controls, which are rules that define permissions for different users. Default rights should be set to 'no access' to prevent any unauthorized access.

Aside from characteristics of secure software, there are also many tools that one can use to test the strength of said security. Some of the most popular tools are:

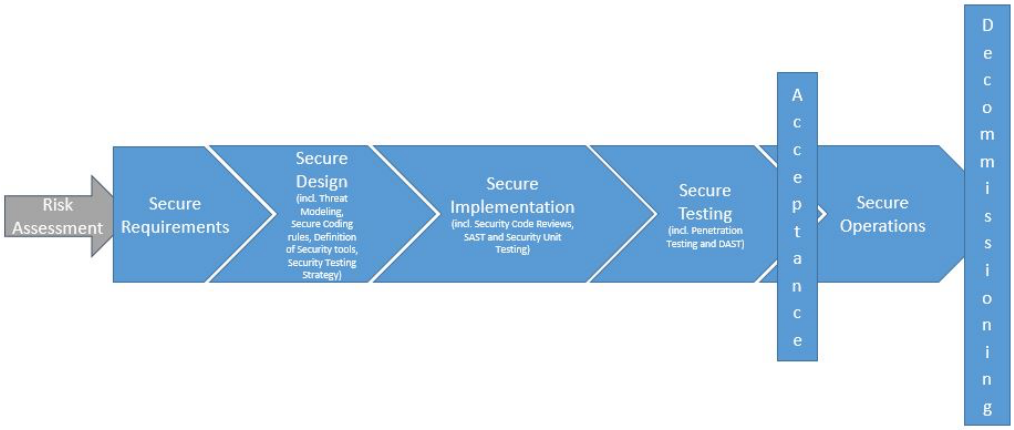
- **Static Application Security Testing (SAST)**
  - SAST tools can be thought of as white-hat or white-box testing, where the tester knows information about the system or software being tested, including an architecture diagram, access to source code, etc. SAST

tools examine source code (at rest) to detect and report weaknesses that can lead to security vulnerabilities.
    - Source-code analyzers can run on non-compiled code to check for defects such as numerical errors, input validation, race conditions, path traversals, pointers and references, and more. Binary and byte-code analyzers do the same on built and compiled code. Some tools run on source code only, some on compiled code only, and some on both (Scanlon).
- **Dynamic Application Security Testing (DAST)**
    - In contrast to SAST tools, DAST tools can be thought of as black-hat or black-box testing, where the tester has no prior knowledge of the system. They detect conditions that indicate a security vulnerability in an application in its running state. DAST tools run on operating code to detect issues with interfaces, requests, responses, scripting (i.e. JavaScript), data injection, sessions, authentication, and more.
    - DAST tools employ fuzzing: throwing known invalid and unexpected test cases at an application, often in large volume (Scanlon).



    -

- **Mobile Application Security Testing (MAST)**
    - MAST Tools are a blend of static, dynamic, and forensics analysis. They perform some of the same functions as traditional static and dynamic analyzers but enable mobile code to be run through many of those analyzers as well. MAST tools have specialized features that focus on issues specific to mobile applications, such as jail-breaking or rooting of the device, spoofed WI-FI connections, handling and validation of certificates, prevention of data leakage, and more(Scanlon).



- **Application Security Testing as a Service (ASTaaS)**
  - As the name suggests, with ASTaaS, you pay someone to perform security testing on your application. The service will usually be a combination of static and dynamic analysis, penetration testing, testing of application programming interfaces (APIs), risk assessments, and more. ASTaaS can be used on traditional applications, especially mobile and web apps (Scanlon).

Though these testing methods are fantastic tools to use after a product is finished, there is still plenty you can do while on the job creating code to maximize security in your software. One good habit is to only ask your consumers for information that you need and will use. For example, if you do not plan on sending physical mail, you shouldn't ask for a physical address as this is just more sensitive information that also takes up unnecessary space and processing. Another method is also creating more barriers that secure the users password/login info. Complex passwords are everywhere now, as almost every site wants you to have a lengthy password with an uppercase, lowercase, and special character. This is a great start, but you can also implement other methods to protect user login info as well like security questions and second-factor authentication like email or text. Another good rule of thumb is to have your code looked at in a couple ways. The first of which involves a real person, or auditor, who will look over your code to identify flaws and give suggestions to better the code. The second is a code analyzer, which is a program that looks through your code for mistakes. Most of these are small and easily corrected but some can really be fatal, which is why it's always good to have eyes, real and virtual, look over your code (Wayner).

Works Cited

Scanlon, Thomas. “10 Types of Application Security Testing Tools: When and How to Use Them.” *SEI Blog*, 9 July 2018, insights.sei.cmu.edu/blog/10-types-of-application-security-testing-tools-when-and-how-to-use-them/.

“Take Online Courses. Earn College Credit. Research Schools, Degrees & Careers.” *Study.com | Take Online Courses. Earn College Credit. Research Schools, Degrees & Careers*, study.com/academy/lesson/secure-software-definition-characteristics.html.

Wayner, Peter. “Safeguard Your Code: 17 Security Tips for Developers.” *InfoWorld*, InfoWorld, 4 Feb. 2013, www.infoworld.com/article/2078701/safeguard-your-code--17-security-tips-for-developers.html.