Secure Software Development

Unit 12 – Reflective Piece

By Trevor Woodman

* Module 3 ePortfolio branch repo available here: <https://github.com/turbits/essex_eportfolio/tree/m3_final>
* ePortfolio Website available here: <https://essex.trevorwoodman.ca>

# Introduction

Developing secure software is an essential practice for any software developer. With the pace at which technology is advancing, and as attack methods become more sophisticated, it is more important than ever to have a security-first approach to developing software. Software intended to be used in a production environment are particularly vulnerable. This is especially true for software used in cloud computing, arguably the largest and most important advancement of enterprise computing in human history. Cloud computing allows unprecedented levels of scalability and flexibility, but it introduces dizzying levels of complexity and risk. The cloud is commonly a shared environment, and the security of the cloud is only as strong as it's weakest link. One needs only to look at any bug bounty program to see the level of complexity in software today. Developing software for scale is an incredibly difficult task to get right and the larger the target, the more enticing it is to malicious actors.

# Developing Secure Software

A crucial step in developing secure software is design; this includes threat modeling, a structured process that is meant to outline and detail security requirements, possible threat vectors, to quantify threat and vulnerability criticality, and to outline a remediation process and the methods that would assist with remediation (Synopsys, no date).

Another critical step in the secure software development pipeline is testing; a robust testing solution is a necessary step. Without thorough testing, software is prone to vulnerabilities or bugs that may lead to vulnerabilities. In a production ready software, several types of testing should be done, including unit testing, interface testing, and penetration testing, to ensure the software is secure and reliable.

Proper coding practices play a part in the security of development. Using best practices and standards ensure that code is secure and does not inherently introduce vulnerabilities. Secure coding techniques, such as input validation and sanitisation, output encoding, and proper error handling can prevent the most common attack vectors such as injection and cross-site scripting attacks.

True software security requires a willingness and a determination to build and feed a culture of security-first. Project stakeholders, developers, QA testers, and project leads should all prioritize security in the development of secure software. It needs to be the first thought at all stages of the software lifecycle.

# Summary

In conclusion, secure software development is a critical practice that ensures security and reliability of software. It involves many stages, some critical ones being threat modeling and robust testing, but to truly be secure requires a group mindset that security is to come first. Coding standards and best practices need to be followed and strict testing requirements will allow developers to create secure software applications that are inherently resistant to malicious actors.

# References

Synopsys (no date) Threat Modeling – Definition. Available at: <https://www.synopsys.com/glossary/what-is-threat-modeling.html> [Accessed 15 April 2023]