Joshua Williamson

CS-305

September 18th, 2025

Module 3-2 Journal Reflection.

**What is your role in solving security concerns as a developer? What might solving security concerns as a developer involve?**

As a developer, my responsibility is to ensure that security measures are implemented as early as possible in the Software Development Life Cycle (SDLC). This means that when gathering requirements, the focus is not only on system functionality but also on identifying and incorporating the necessary security requirements. While writing code, making it a priority to “apply secure coding practices and security principles such as confidentiality, integrity, availability, and accountability while coding software modules” (Jeganathan, 2019).

**Where does security fall within the software stack and development life cycle?**

In the DevSecOps methodology, security is integrated into every stage of the Software Development Life Cycle (SDLC). What sets this approach apart from the traditional SDLC is that it emphasizes implementing security measures early in the development process, rather than waiting until the testing phase as is often done with traditional methods.

**How might you add security measures to transform a DevOps pipeline into a DevSecOps pipeline?**

In a DevOps pipeline, the primary focus is on Continuous Integration and Continuous Delivery. To transform this into a DevSecOps pipeline, however, Continuous Security must also be incorporated. This involves implementing controls to address vulnerabilities such as those identified in the SANS 25 and OWASP Top 10. Key measures include input validation, session management, user credential and access validation, data protection and privacy safeguards, logging, and both static application security testing (SAST) and source code analysis. Additionally, it includes live and offline log analysis, as well as threat modeling techniques like STRIDE and DREAD (Jeganathan, 2019).

**The article suggests creating and following a plan to secure the entire DevOps life cycle. What is included in the suggested plan? Would you recommend following the plan?**

Jeganathn (2019) outlines a plan in the journal that includes the following steps:

* Develop a rapid, high-level threat model.
* Quantify identified risks.
* Secure the DevOps lifecycle tools, examples such as Azure or GitLab, by:
  + Protecting access points.
  + Implementing SSO or federation for logins.
  + Enforcing two-factor or multi-factor authentication (2FA/MFA).
* Safeguard keys, service accounts, and API credentials.
* Establish infrastructure protection controls, such as determining whether developers truly need access to production environments.

This plan provides a comprehensive checklist of critical considerations. One important addition would be to schedule and plan for testing. Some testing data can be sensitive or confidential, which means its access must also be secured and carefully managed. This requires a clear plan, along with documentation specifying who accessed the data and when.

**Sources:**

Jeganathan, S. (2019). DevSecOps: A systemic approach for secure software development. ISSA Journal, 20–27. Information Systems Security Association.

Oracle. (2024). Secure coding guidelines for Java SE. Oracle. https://www.oracle.com/java/technologies/javase/seccodeguide.html

OWASP Foundation. (2010/2021). OWASP secure coding practices – Quick reference guide (v2.1 draft). OWASP. https://owasp.org/www-project-secure-coding-practices-quick-reference-guide/

OWASP Foundation. (2025). Dependency-Check: The dependency-check-maven plugin. OWASP. https://jeremylong.github.io/DependencyCheck/dependency-check-maven/index.htm