Lee Kitchen

CS-405

21OCT2024

8-2 Journal: Portfolio Reflection

Adopting a secure coding standard and integrating security early in the development process is crucial for building secure and reliable systems. Waiting until the end of development to address security flaws is often ineffective and costly. By implementing secure coding practices, such as those recommended by OWASP and CERT, during the coding phase, developers can proactively mitigate risks such as buffer overflows, SQL injection, and improper memory management. Integrating these practices throughout the development lifecycle ensures that vulnerabilities are identified and remediated early, reducing the overall cost and effort required to maintain system integrity.

Effective risk evaluation involves assessing both the severity and likelihood of vulnerabilities while weighing the cost of mitigation. A structured risk management approach helps prioritize vulnerabilities based on their potential impact and the feasibility of mitigating them. Focusing resources on high-severity and high-likelihood threats first, such as SQL injection or memory corruption, allows organizations to maximize their return on investment in security measures. Additionally, conducting a cost-benefit analysis of mitigation strategies ensures that resources are allocated efficiently, aligning security efforts with business objectives.

The concept of Zero Trust is increasingly vital in modern cybersecurity strategies. This approach operates under the assumption that no entity, whether inside or outside the network perimeter, should be trusted by default. Every access request must be verified continuously through methods like multi-factor authentication (MFA) and least privilege access controls. This aligns with the principle that threats can emerge from both external and internal sources, making it essential to validate every access attempt and monitor all user activities in real time. Implementing Zero Trust not only reduces the risk of unauthorized access but also supports compliance with regulatory requirements by ensuring comprehensive logging and accountability. Security policies should be clear, enforceable, and integrated into an organization’s workflow. A comprehensive policy includes guidelines on secure coding practices, data encryption, access control, and incident response protocols. For example, dynamic application security testing (DAST) tools can be incorporated into the CI/CD pipeline to identify runtime vulnerabilities. Training for development teams is also critical; inadequate knowledge of patch management and secure coding practices can leave systems vulnerable to exploitation. Organizations should prioritize ongoing education and policy updates to keep pace with evolving threats.

In summary, a proactive approach to adopting secure coding standards, evaluating risks, and implementing Zero Trust and security policies ensures robust defense mechanisms are embedded throughout the development lifecycle. By integrating these practices early, organizations can reduce costs and enhance system security, leading to a more resilient infrastructure.

References

Sharp, H., Preece, J., & Rogers, Y. (2019). Interaction Design (5th ed.). Wiley Professional Development (P&T). https://mbsdirect.vitalsource.com/books/9781119547303

Howard, M., LeBlanc, D., & Viega, J. (2010). 24 Deadly Sins of Software Security: Programming Flaws and How to Fix Them. McGraw-Hill Education.

Shostack, A. (2014). Threat Modeling: Designing for Security. Wiley.

CISA. (2021). Remediation guidance for the SolarWinds Orion supply chain compromise. U.S. Cybersecurity and Infrastructure Security Agency (CISA) https://www.cisa.gov/news-events/news/remediating-networks affected-solarwinds-and-active-directorym365-compromise

Equifax. (2017). The 2017 Equifax data breach. Federal Trade Commission (FTC). https://www.ftc.gov/news-events/news/press-releases/2019/07/equifax- pay-575-million-part-settlement-ftc-cfpb-states-related-2017-data-breach

OWASP Foundation. (2021). OWASP dynamic application security testing (DAST). OWASP. https://owasp.org/www-project-devsecops- guideline/latest/02b-Dynamic-Application-Security-Testing

Cimpanu, C. (2017). WannaCry ransomware attack: What happened and how to prevent it. ZDNet. https://www.zdnet.com/article/wannacry- ransomware-crisis-one-year-on-are-we-ready-for-the-next-global-cyber-attack/