Danielle Monroe

8-2 Journal: Portfolio Reflection

**Adoption of a secure coding standard, and not leaving security to the end**

Implementing security from the beginning, at all levels, is imperative to building a secure system. If implementing security is left for the end, a plethora of vulnerabilities would arise, hopefully before deployment. Then, the developers with vulnerabilities within their code; which is likely to be many, if not all, throughout every level of the system; will be obligated to return to their code, digging for where the vulnerabilities are. Needless to say, that that would be considerably time consuming. The purpose of adopting a security Coding Standard is to substantially, if not completely, diminish such vulnerabilities and strengthen a systems code. By inspecting all incoming data, it can be determined if it’s safe, unsafe, or undetermined, and only safe data is allowed into an application (Murray, 2022). Coding Standard *Validate Input* could mitigate four different vulnerabilities that are possible with incoming, unchecked data: Cross-Site Scripting, Injection, Insecure Deserialization, and XML External Entities (Murray, 2022). Combined with the *Principle of Least Privilege*, access to sensitive data and the threat of attack are curtailed.

**Evaluation and assessment of risk and cost benefit of mitigation**

An assessment of risk catches and evaluates potential vulnerabilities from data of miscellaneous resources. These vulnerabilities are prioritized by resource and varying risks that feasibly would affect those vulnerabilities (Allen, 2022). The assessment permits an organization the allowance of addressing logged vulnerabilities before an application version is deployed and the progress of any existing security endeavors. Evaluation of the assessment grants decision makers a comprehensive conveyance of the status of the application’s security, make informed decisions as well as evaluate employees’ performances (Allen, 2022). The methodology for the cost/benefit figure, for mitigation, is base off the offset between the calculation of the architectural and policy recommendations, that’ll be necessary to mitigate the risk, and the *Category of Threats*, that has a ranged set of financial loses for average probability occurrences (Chen et al.). The *Category of Threats* is a set of attacks and misuses that can be harmful to an organization and are categorized by threat level. The assessment of risk allows for the *Margin of Safety* to be calculated; the aggregation of probabilities of a *Category of Threats*; and is based on if the *Category of Threats* has been mitigated or not. An unmitigated *Category of Threats* will have what is called a *Baseline Risk*, while a mitigated gets a *Residual Risk* (Chen et al.). The former is the calculated event of risk that could happen, if there are no security solutions in place, at the national average. The latter is the calculated event of risk to a system with security solutions in place (Chen et al.). The *Bypass Rate* is the rate of an occurrence of risk within the *Residual Risk*. A *Bypass Rate* of 0% shows all incidents were solved within the security solutions while 100% reveals that nothing was solved (Chen et al.). Therefore, the evaluation of the risk assessment leads to a categorized *Category of Threats* that allows decision makers to make informed, purposeful determinations on how to address security.

**Zero trust**

*Zero Trust* is a self-explanatory concept that should be implemented into all levels of an organization, security, and the SDLC. In a “zero-trust security architecture,” this involves not trusting devices, users, data, application, and transport/sessions (Kueh). For devices, a unified endpoint management (UEM) set in place equips programmers with the ability to manage the device; monitor and control it. This is done to a list of specified devices that’re owned by an organization and offers a solution to the monitorization of these devices (Kueh). For users, any input coming from a user should not be trusted; it should be validated and then allowed or denied access. Further, users themselves need to be authenticated. A strong conditional access engine (conditional-access policies and password-less and multi-factor authentication to name a few) should be used to enforce the concept (Kueh). Within an *Application*, employees need logical access. User authentication, with single sign-ons, provides a way to retain security and for the user to encounter an enhanced encounter with the application (Kueh). Any importing *Data* should be held to the Coding Standards *Input Validation* and thoroughly sanitized. Inter-system data transporting should enable encryption/decryption practices. These methods adhere do the *Zero Trust* concept by not allowing any incoming or passing data to move through an application without being scrutinized first. *Transportation/Sessions* applies the *Principle of* *Least Privileges*. Here a user or system are only given the minimum required access level to execute the primed operation (Kueh). Implementation of encryption, session protection, and micro-segmentation gives further security by promoting the enforcement of the *Principle of* *Least Privilege* (Kueh).

**Implementation and recommendations of security policies**

Security policies are instrumental to security itself. Security policies briefly express precisely what necessities are entailed in the expectations of employees, instructs which control framework for the best security of the system, methodically holds each person accountable to comply with the behavior that is expected of them, and supports legal and ethical responsibilities (CISA, 2021). Implementation should be done at all levels of security and all stages of the SDLC. It should be checked and reimplemented, if necessary, recurrently. Recommendations include the use of automation throughout some of the processes within the *Pre/Post-Production stages* (like *Threat landscaping* in the *Assess and Plan* stage*,* or the *Log Collection* and *Analytics* processes in the *Monitor and Detect* stage to name a few).

**Works Cited:**

Allen, J. (2022, December 15). *How to conduct a security risk assessment*. PurpleSec. Retrieved December 28, 2022, from https://purplesec.us/learn/security-risk-assessment/

Chen, P., Dean, M., Lopez, L., Ojoko-Adams, D., & Osman, H. (n.d.). *Square project: Cost/benefit analysis framework for information ...* Retrieved December 28, 2022, from https://resources.sei.cmu.edu/asset\_files/TechnicalNote/2004\_004\_001\_14360.pdf

CISA, R. D. (P. A. R. T. N. E. R. |. (2021, October 26). *The importance of a company information security policy*. Linford & Company LLP. Retrieved December 28, 2022, from https://linfordco.com/blog/information-security-policies/

Kueh, T. (n.d.). *A practical guide to zero-trust security*. Threatpost English Global threatpostcom. Retrieved December 28, 2022, from https://threatpost.com/practical-guide-zero-trust-security/151912/

Murray, A. (2022, August 17). *Secure coding: A practical guide*. Mend. Retrieved December 28, 2022, from https://www.mend.io/resources/blog/secure-coding/