Unit 2: UML Modelling to Support Secure System Planning

Required Reading

Dadzie, J. (2005) Understanding Software Patching, IEEE.

Summary
The document "Understanding Software Patching" provides an in-depth exploration of the software patching process, emphasizing its critical role in maintaining software security and functionality. It outlines the steps involved in creating, testing, and distributing patches, highlighting the importance of ensuring that patches do not conflict with existing software versions. Key considerations include the need for robust uninstall or rollback processes, the protection of patches from tampering, and the necessity of clear communication with users regarding vulnerabilities and patch details. The document also discusses the challenges developers face, such as managing multiple versions of software and ensuring that patches are easily deployable and do not disrupt system operations.

Reflection
The insights from this document underscore the complexity and importance of software patching in today's digital environment. As software becomes increasingly integral to both personal and professional activities, the need for timely and effective patches is paramount. The balance between security and functionality is a recurring theme, reminding us that while addressing vulnerabilities is crucial, it is equally important to maintain user experience and system stability. This document serves as a valuable resource for both developers and users, highlighting the collaborative effort required to ensure software remains secure and functional. It prompts reflection on the responsibilities of both parties in the patching process and the potential consequences of neglecting timely updates.

Sharma, A., Bawa, R.K. Identification and integration of security activities for secure agile development. Int. j. inf. tecnol. 14, 1117–1130 (2022). https://doi-org.uniessexlib.idm.oclc.org/10.1007/s41870-020-00446-4

Summary
The document discusses the integration of security activities into agile software development, addressing the challenges posed by the rapid delivery and flexibility of agile methodologies. It outlines a framework that identifies and incorporates essential security practices from established security engineering processes, such as CLASP and Microsoft SDL. The research employs empirical methods, including the Analytic Hierarchy Process (AHP) and PROMETHEE, alongside a systematic literature review and survey study, to gather industry feedback and select beneficial security activities. The framework aims to enhance security without compromising the agility of the development process, ultimately providing a balance between speed and security.

Reflection

The integration of security into agile development is crucial in today's fast-paced software environment, where vulnerabilities can be exploited quickly. This research highlights the importance of proactive security measures and the need for a structured approach to incorporate these measures seamlessly into agile practices. The use of empirical methods to validate the selection of security activities demonstrates a commitment to evidence-based practices, ensuring that the proposed framework is grounded in real-world applicability. As organizations increasingly adopt agile methodologies, the insights from this study can serve as a valuable resource for enhancing security while maintaining the benefits of agility. This balance is essential for fostering trust and reliability in software products, ultimately leading to better outcomes for both developers and users.

Additional Reading

Pillai, A.B. (2017) Software Architecture with Python. Birmingham, UK. Packt Publishing Ltd.

- Chapter 1.
- Chapter 6.

Accenture security (2020) The Cost of Cybercrime, Ninth Annual Cost of Cybercrime Study.

National Institute of Standards and Technology (2020) Cybersecurity Framework.

Object Management Group (2020) Unified Architecture Framework (UAF) Domain Metamodel. Version 1.1.