International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

A CRITICAL STUDY ON SOFTWARE SECURITY REQUIREMENT ENGINEERING FOR RISK AND COMPLIANCE MANAGEMENT

CANDIDATE - Biswanath Mishra

DESIGNATION- RESEARCH SCHOLAR SUNRISE UNIVERSITY ALWAR

Guide name - Dr.Prateek Mishra

DESIGNATION- Associate professor

ABSTRACT

Software security requirement engineering (SSRE) is a crucial process in contemporary software development, focusing on identifying, analyzing, and specifying security requirements to address potential risks and ensure compliance with relevant regulations and standards. This research paper investigates the role of SSRE in risk and compliance management within the software development life cycle. The paper explores various methodologies and best practices employed by organizations to integrate security considerations early in the development process.

Keywords: - Software, Security, Management, Application, Cycle.

I. INTRODUCTION

OVERVIEW OF SOFTWARE SECURITY REQUIREMENTS

Using categorical and morphisms theory, this study aims to provide a software-based security requirement engineering paradigm. Multiple perspectives on parallel processing and the creation of rewrite-based, knowledge-centered models are the primary focuses of previous security requirement engineering models, but these earlier models lack the incorporation of multiple functional mappings between the security objects necessary to choose the optimal approach. The security models do not account for the essential security functions that must be implemented over a wide range of settings and execution depths. To better organize the many categories of security requirement functional objects, the suggested requirement engineering model makes use of the formal theory of category of objects and the morphisms between them, as well as n categories and multiple morphisms.

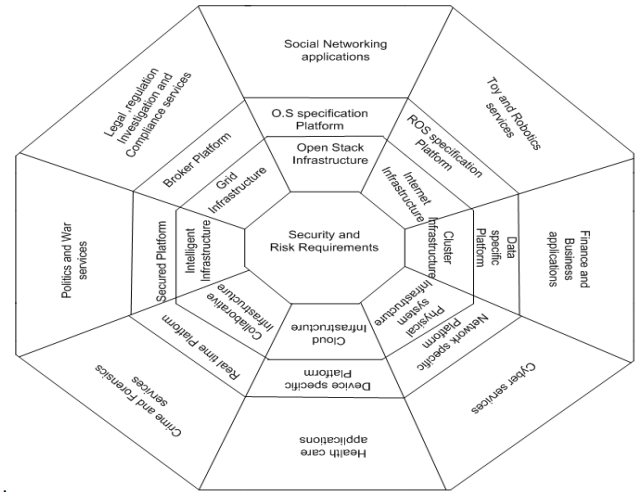
This security need category is handled by treating objects, morphisms, and uncertain events in each given subsystem as algebraic data types on demand. Implicitly employed in the development of category and morphism is the gathering of security requirement items using classification and clustering algorithms. In order to give a security assurance functors with minimal risk on the requirements for the next design state, we first map the risk and compliance in the form of both direct and indirect categories. To ensure minimal security risks via effective compliance management strategies, a 'n' category and 'n' morphic model for software security requirement model are provided.

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

Risks are recognized with many morphisms of non-compliances, and the functions themselves are morphisms between security objects of different sorts in various attack and vulnerability categories. By combining risk analysis with semiformal specification techniques, as in the CORAS approach, a workable framework may be developed for model-based security risk assessment.

II. SOFTWARE SECURITY REQUIREMENT STANDARDS FOR DISTRIBUTED APPLICATIONS

Open stack, Internet, cluster, physical system, cloud, collaborative, intelligent, and in grid infrastructures are all examples of software-as-a-service environments from which information on software security and risk requirements must be gleaned. A device-centric platform, a real-time platform, a secured platform, a broker platform, an operating system-centric platform, a real-time OS-centric platform, a data-centric platform, and a network-centric platform are all part of the platform-as-a-service offering. Health care apps, clinical and forensics services, political and military services, legal regulation investigation and compliance service, social networking apps, toy and robotics services, business applications, and cyber services are all part of the infra structure as a service, as shown in Figure 1.



.

Figure 1 Software Security Requirement in Multi Levels of Execution

III. DISTRIBUTED SOFTWARE SECURITY -CASE STUDY: CONNECTED CARS

The white rear of a tractor trailer stood out starkly against the night sky, but neither the auto pilot nor the driver saw it in time to use the brakes. The accident raises questions about the reliability of autonomous cars' ability to make split-second, potentially life-or-death judgments while driving. When compared to traditional cars and their fallible drivers, the operational hazards associated with automated vehicles and automated roadways are expected to be substantially greater. To rescue the driver or the crowd That is the moral conundrum posed by autonomous automobiles. Accidents of this sort might occur, for example, if sensors are placed so that they cannot detect children who are playing alone. However, the danger of artificial intelligence is too complicated to make sense when evaluating the risk of automation.

Security Requirements Analysis Based on Scenarios Elicitation

International Journal For Advanced Research In Science & Technology
A peer reviewed international journal www.ijarst.in
ISSN: 2457-0362

- Constant and constant notification is required.
- Security decisions and precautions must be taken swiftly.
- Brake system automation must be implemented based on automated signal specifications.
- The sensors need to be set up in a way that allows the items or crowd to be detected.
- Risk and failure are mitigated by the iterative process of dynamic demand elicitation.
- Locating the Security Players
- Who activates, requests, or retrieves the cyber physical system and when.
- Finding the safe ones
- What are the actor's desired system capabilities?
- Does the system have a memory? Who will create this data, who will use it, who will edit it, and who will destroy it?
- Should an actor be informed when there is a change in the system's internal state?
- Is the system aware of any external events? Who communicates such information to the system?
- Localization of the Security User Environment
- The system's limits must be specified precisely.
- Safety Measures Flowchart
- A cyber case diagram is a graphical depiction of the interdependencies between actors and use cases, used to record the desired operation of a system.

IV. CONCLUSION

In conclusion, software security requirement engineering plays a critical role in risk and compliance management within the realm of software development. By incorporating security considerations into the early stages of the software development life cycle, organizations can proactively address potential vulnerabilities and ensure compliance with relevant regulations and standards.

The process of software security requirement engineering involves identifying and analyzing potential security risks, defining security objectives, and establishing security requirements that guide the design and implementation of secure software. It necessitates collaboration between various stakeholders, including software developers, security experts, risk analysts, and compliance officers.

By implementing a structured and well-defined software security requirement engineering process, organizations can achieve several key benefits:

- Risk Mitigation: Early identification and analysis of security risks enable the implementation of appropriate countermeasures, reducing the likelihood of security breaches and data compromises.
- Compliance Adherence: By aligning security requirements with relevant industry standards and regulations, organizations can ensure compliance with legal and regulatory frameworks, avoiding potential penalties and reputational damage.
- Cost Savings: Detecting and addressing security issues during the

requirement engineering phase is more cost-effective than attempting to fix them after deployment or during later stages of development.

- Enhanced Trust and Reputation: Demonstrating a commitment to security and compliance enhances trust among customers, partners, and stakeholders, leading to improved reputation and business opportunities.
- Resilient Software: Security-focused requirement engineering results in more robust and resilient software that can withstand cyber-attacks and evolving threats.
- Long-term Sustainability: Adopting a proactive approach to software security fosters a culture of security awareness and ensures long-term sustainability of software applications.

However, it is crucial to acknowledge that software security is an ongoing process. Threat landscapes constantly evolve, and new vulnerabilities emerge over time. Therefore, it is essential to continuously monitor and update security requirements throughout the software's life cycle.

Moreover, software security requirement engineering should be integrated into an organization's overall risk management strategy. It should complement other security practices, such as vulnerability assessments, penetration testing, and security training for employees.

In conclusion, software security requirement engineering is a vital practice that empowers organizations to build secure, compliant, and reliable software applications while minimizing risks and protecting sensitive data. By fostering a proactive security culture and adopting a holistic approach to software security, organizations can stay ahead of potential threats and maintain a strong position in the face of ever-changing compliance requirements and security challenges.

REFERENCES

1. Canavese, Daniele & , Leonardo & Basile, Cataldo & Coppens, Bart & De Sutter, Bjorn. (2020). Software Protection as a Risk Analysis Process.
2. Islam, Shareeful & Dong, Wei. (2008). Human factors in software security risk management. Proceedings - International Conference on Software Engineering. 10.1145/1373307.1373312.
3. Malik, Vinita & Singh, Sukhdip. (2019). Security risk management in IoT environment. Journal of Discrete Mathematical Sciences and Cryptography. 22. 697-709. 10.1080/09720529.2019.1642628.
4. Asif, Muhammad & Jamil, Ahmad & Hannan, Abdul. (2014). Software Risk Factors: A Survey and Software Risk Mitigation Intelligent Decision Network using Rule Based Technique. 2209.
5. Kure, Halima & Islam, Shareeful & Razzaque, Mohammad Abdur. (2018). An Integrated Cyber Security Risk Management Approach for a Cyber-Physical System. Applied Sciences. 8. 898. 10.3390/app8060898.
6. Jahankhani, Hamid & Nkhoma,

Mathews & Mouratidis, Haris. (2010). Security risk management strategy. 10.1142/9789812837042_0013.

7. Haz, Lidice & Morán, Manuel & Acaro, Ximena & Guzman, Carlos & Espin, Luis. (2019). Implementation of IT Security and Risk Management Process for an Academic Platform. 10.1007/978-3-030-02351-5_43.
8. Magnusson, Christer & Chou, Sung-Chun. (2010). Risk and Compliance Management Framework for Outsourced Global Software Development. 228-233. 10.1109/ICGSE.2010.33.