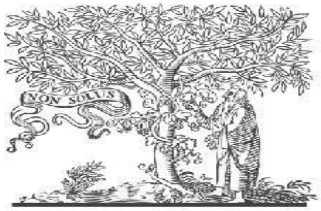
International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

COPY RIGHT



ELSEVIER SSRN

IJIEMR Transactions, online available on 26th Dec 2022. Link :http://www.ijiemr.org/downloads.php?vol=Volume-11&issue=Issue 12

10.48047/IJIEMR/V11/ISSUE 12/84

Title ANALYSING THE SOFTWARE SECURITY, ISSUES & CHALLENGES

Volume 11, ISSUE 12, Pages: 664-668

Paper Authors **Biswanath Mishra, Dr.Prateek Mishra**

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

ANALYSING THE SOFTWARE SECURITY, ISSUES & CHALLENGES

CANDIDATE - Biswanath Mishra
DESIGNATION- RESEARCH SCHOLAR SUNRISE UNIVERSITY ALWAR
Guide name - Dr.Prateek Mishra
DESIGNATION- Associate professor

ABSTRACT
Security risks and regulatory regulations have become key considerations for businesses in the continuously changing world of software development. An all-encompassing strategy for managing software security and maintaining compliance is necessary in light of the rising number of cyber threats and the increased focus on data protection and privacy requirements. This abstract describes the idea of a Software Security Risk and Compliance Management Network, which combines theory and practice to meet these urgent needs. The software security risk assessment and compliance management framework is developed at the abstraction phase. To locate security flaws and evaluate whether or not a business is in compliance with relevant regulations, this framework combines best practices, standards, and procedures. As part of the abstraction, a community of software engineers, security professionals, compliance officials, and auditors is formed to improve communication and coordination throughout each stage of the SDLC

Keywords: - Security, Software, Risk, Network, Management.

I. INTRODUCTION

Software has become the backbone of almost every sector in today's linked digital world, allowing for more streamlined processes, data management, and communication. However, as dependence on software applications grows, so does the potential for security breaches and regulatory noncompliance. Businesses, governments, and people all face serious risks from cybercrime, data breaches, and legal repercussions. As a consequence, infrastructures for managing software security risks and regulations are in dire need of improvement.

The purpose of this introductory piece is to provide a conceptual framework for learning about software security risk and compliance management and to emphasize its practical use in modern corporate settings. The suggested method combines preventative methods to recognize and lessen potential dangers with a methodical approach to compliance to guarantee that all norms, laws, and recommendations are met.

II. ABSTRACTED APPROACH

The first step in the simplified framework for managing software security risks and regulations is conducting a thorough risk assessment. Part of this process involves cataloging the software ecosystem in search of vulnerabilities, dangers, and useful tools. Spoofing, tampering, repudiation, information disclosure, denial of service, and elevation of privilege are all examples of threats that may be modelled to assist identify potential trouble spots. Simultaneously, vulnerability evaluations help in

identifying flaws in the software's framework, source code, and settings.

When threats are discovered, the next step is to rank them in order of importance. Organizations may make more educated decisions by using risk analysis approaches like FAIR (Factor Analysis of Information Risk) to help them put a monetary value on potential outcomes. After identifying high-risk areas, preventative measures may be taken, such as putting in place security controls, doing penetration tests, and developing incident response plans.

On the other hand, the abstracted method relies heavily on compliance management. To guarantee that software systems are compliant with all applicable laws and regulations, as well as any industry-specific standards (such as ISO 27001 or the NIST Cybersecurity Framework), it is important to be familiar with these guidelines. Policies, training and awareness initiatives, frequent audits, and documented security procedures are all part of the compliance process.

III. REAL-WORLD APPLICATION

To successfully use the abstracted approach, businesses must include security risk and compliance management within their SDLC. The DevSecOps (Development, Security, and Operations) paradigm provides a workable approach to embedding security checks at every stage of the software creation process. This allows for an iterative and preventative security strategy, making it simpler to spot problems early and fix them.

Furthermore, software security risk and compliance management networks have applications outside of the typical business setting. The energy, transportation, and healthcare industries are examples of critical infrastructure and are subject to stringent regulatory standards designed to protect the public's safety and privacy. Similarly, in order to safeguard national interests, government entities that deal with sensitive information must place a premium on security and compliance.

Management of software security is increasingly being sought after as a social and cultural good by both public and commercial institutions. The safety of the whole system is based on the code running on several embedded processors and other computer components spread out over the network. Noncompliance in the operating modes of software applications and executable management service functions is the root cause of most software system security vulnerabilities. The security issues may be exploited in a variety of ways, both naturally and as a result of system weaknesses and premeditated assaults on data. Depending on the preexisting vulnerabilities and noncompliances, the thin layer of isolation between hardware and software converts external assaults into either executable deterministic threats or probabilistic risks that may be exploited.

IV. SOFTWARE SECURITY STANDARDS

Standards for code urge developers to adhere to a consistent set of rules and norms established by the needs of the project and the company, rather than by individual experience or taste. These coding guidelines may be used by developers and designers throughout the software development process to produce more robust and secure products. Features of a piece of software that demonstrate

conformance to a number of legislation and guidelines. ISO 19011:2011, the international standard for software security, serves as a primer for verifying conformity with other ISO management system standards. The fundamentals of system security engineering are laid forth in ISO 21827:2011. The full system development life cycle is represented by this model. The criteria for security management systems are laid forth in the ISO/PAS 28000 standard. The ISO 31000:2009 worldwide standard for risk management is based on the understanding that risks may vary in terms of their type, severity, and complexity.

The need of protecting software systems has risen rapidly as companies throughout the globe have grown more and more dependent on software and as the frequency of cyber-attacks has increased. The importance of software security standards in directing businesses toward strong security practices, risk management, and regulatory compliance is essential. This study dives at the relevance of software security standards, how they have changed over time, and how they have affected the creation, distribution, and upkeep of safe programs. This study seeks to illuminate the acceptance, implementation obstacles, and advantages of widely known software security standards by businesses aiming to promote confidence in the digital world via their application.

V. SOFTWARE SECURITY ISSUES AND CHALLENGES

The safety of computer programs is of paramount importance in today's interconnected world. Cyberattacks and data breaches are becoming more likely as technology becomes more widespread in our daily lives. Malicious actors often target software systems including apps, operating systems, and online services in order to exploit vulnerabilities and obtain unauthorized access to data. The most critical problems and obstacles related to software security are discussed here.

1. **Insecure Software Development Practices:**

Insecure software development processes are a root cause of the software security problem. The introduction of security vulnerabilities is common due to rushed development schedules, a lack of security training among developers, and the pressure to deploy new features. If these flaws go undetected, they might be used by hackers to get access to sensitive information.

2. **Lack of Secure Coding:**

The use of secure coding methods is essential in reducing the likelihood of security flaws in software. As a result, many software products have typical coding flaws and vulnerabilities since developers aren't properly taught in safe coding concepts. Attacks such as cross-site scripting (XSS), SQL injection, and buffer overflows are only a few examples.

3. **Insufficient Input Validation:**

In order to keep dangerous data out of a system, input validation is a must. Security problems like command injection and unauthorized code execution may arise from insufficient or incorrect input validation. Data corruption or tampering may also occur if input is not properly validated.

4. **Weak Authentication and Authorization:**

The foundations of access control in

software systems are authentication and authorisation procedures. Use of weak authentication methods, such as readily guessed or factory-set passwords, may result in a breach of security. Similarly, if permission settings are incorrect, attackers may get access to sensitive data and even compromise the system.

5. Inadequate Encryption:

Encryption of data at rest and in transit is essential for the security of confidential data. However, data leakage and compromised secrecy might occur from insufficient encryption techniques. An poor encryption algorithm or careless use of encryption keys might undermine security.

VI. CONCLUSION

The software's safety is contingent on how it was programmed in light of the rules and regulations now in place. Noncompliance in any element of the computational work area of the executable program, including embedded software in the unique hardware, protected and public devices, and hidden third party firmware, might reduce the risk and hazard. In the software development life cycle, security engineering is performed. Internet of Medical Things application needs are determined, appropriate models are built, and security is tested with all conceivable threats and assaults, including formal vulnerability detection, risk assessment using fuzzy rough sets, and security on demand by the devices. In order to describe not only the structure and functions but also the behavioral morphism between security objects, modern methodologies have been developed on solid mathematical foundations such as categorical morphism theory, fuzzy rough set theory, and obfuscation techniques. Fuzzy rough set explorations and threat analysis are used to quantitatively check the findings, and the newest software tools are deployed to ensure that the security flow inside the domain-specific software applications is error-free.

Software security needs must be collected, categorized, structured, and managed to fit the computational profile of development before any security engineering can begin. All conceivable deployments are taken into account when classifying the software security needs into infrastructure security, platform security, and application level security. Software requirement engineering is being done to allow for compliance with the most up-to-date security standards and satisfaction of the domain security needs across all operational modes with an acceptable degree of risk. Prioritization and organization of the gathered distributed software security needs followed a process of genesis, elicitation, and elaboration.

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. Mugarza, Imanol & Parra, Jorge & Jacob, Eduardo. (2017). Software Updates in Safety and Security Co-

engineering. 199-210. 10.1007/978-3-319-66284-8_17.

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. Stewart, Harrison. (2022). Security versus Compliance: An Empirical Study of the Impact of Industry Standards Compliance on Application Security. International Journal of Software Engineering and Knowledge Engineering. 32. 1-31. 10.1142/S0218194022500152.
6. Racz, Nicolas & Weippl, Edgar & Seufert, Andreas. (2011). Governance, Risk & Compliance (GRC) Software - An Exploratory Study of Software Vendor and Market Research Perspectives. Proceedings of the Annual Hawaii International Conference on System Sciences. 1-10. 10.1109/HICSS.2011.215.
7. Alsmadi, Izzat & Xu, Dianxiang. (2015). Security of Software Defined Networks: A Survey. Computers & Security.
8. Ansar, Syed & Yadav, Jaya & Jaiswal, Kriti & Yadav, Amitabha & Khan, Prof. Raees. (2020). Some Common Software Security Risks Factors at Design Phase.