Comprehensive Integration Methods for Automatic Security in the Software Development Lifecycle

Jingyu Yang, Enzhe Li, Yu Wang, Dongmei Zhai

China TravelSky Holding Company Limited, Beijing, 101318, China

1614124889@qq.com

Keywords: Software development lifecycle, Automatic security, Comprehensive integration methods

Abstract: In recent years, China's science and technology sector has made significant achievements driven by the rapid progress of the social economy. Software development technology is advancing rapidly, which has to some extent increased the complexity of the software supply chain and the characteristics of global development. Based on this, how to improve the security of the software development lifecycle has become an urgent task. In the traditional sense, software development security management is mainly aimed at standardizing the later stages of software development work. However, with the diversification of information technology security risks and the continuous deepening of harm levels, it is clearly necessary to do a comprehensive integration of automatic security in the software development cycle. This article analyzes the comprehensive integration method of automatic security in the software development lifecycle, providing reference and guidance for relevant personnel to timely discover and address security vulnerabilities or defects in software.

1. Introduction

Faced with various security risks that frequently occur in the software development lifecycle, major software enterprises urgently need to solve the comprehensive integration of automatic security in the software development lifecycle. This can not only help software companies maintain a good market reputation and avoid a decrease in customer base, but also prevent data information leakage and various network attacks [1]. Based on this, relevant software development enterprises should fully recognize the necessity of integrating network security into the software development cycle, and then design comprehensive integration methods according to the principle of automated testing to maximize the quality and efficiency of software development and improve the stability of software applications.

2. Necessity of Integrating Network Security into the Software Development Cycle

The reasonable design and implementation of a comprehensive integration method for automatic security in the software development lifecycle can improve the security and stability of software development. Network security integration technology is a testing method that utilizes automated testing tools and scripts to simulate user behavior and check various programs or software functions. Compared to manual detection mode, network security integration testing technology has the following advantages.

Firstly, improve testing efficiency. Network security integration testing technology can quickly and accurately execute testing instructions, greatly improve testing efficiency, run a large number of test cases in a short period of time and promptly identify potential security risks. Secondly, reduce testing costs [2]. Network security integration testing technology can reasonably reduce testing costs, shorten testing cycles, and accelerate software development efficiency. Thirdly, improve software quality. Network security integration testing technology can comprehensively detect software performance, improve software quality, reduce manual testing errors, and further improve software

  DOI: 10.25236/icmmct.2024.001

stability and reliability.

Applying network security integration testing technology well in the software development lifecycle can improve testing efficiency and quality, and maximize the maintenance of software development quality and operational stability.

3. Principles for Implementing Automated Testing

3.1 Determine Testing Objectives

The software development team first needs to clarify the testing focus, such as key business processes, core functional points, etc., based on the type of software development, complexity, and level of existing technology research and development.

3.2 Determine Testing Scenarios

Further expand the coverage of testing scenarios and test cases based on the established testing objectives. The testing scenarios can be specifically divided into basic testing scenarios and complex testing scenarios. For example, conducting security testing on basic application scenarios of e-commerce websites can be manifested as login and registration, while security testing in complex scenarios can be manifested as shopping cart, order management, etc. [3].

3.3 Evaluate the Feasibility of Automated Testing

The software development team needs to further test which test cases can be automated and which test cases need to be manually executed.

3.4 Develop Automated Testing Plans

Specifically manifested in the selection of testing tools and the development of testing frameworks. The software development team also needs to test the feasibility of script writing and execution, as well as detect data information and environmental conditions.

3.5 Evaluate the Effectiveness of Automated Testing

After the implementation of automated testing, the software development team should scientifically evaluate the testing effectiveness, coverage, etc., and then make reasonable adjustments and optimizations to the automatic security comprehensive integration method of the software development lifecycle based on testing [4].

It is worth noting in this process that the software development team should establish automation testing goals and scope based on the specific situation and limitations of the software development project, and track and adjust the progress of automation testing in real time to ensure the synergy between the effectiveness of automation testing and the collaborative progress of software development.

4. Comprehensive Integration Methods for Automatic Security in the Software Development Lifecycle

At present, due to the complexity of the software development lifecycle, major software development companies are developing various software development lifecycle methods, such as waterfall model, V-shaped model, big bang model, iterative model, incremental model, etc. [5]. However, currently, most software development enterprises still choose agile development integration methods as the first choice in the comprehensive integration process of automatic secure software development lifecycle. By dividing software development projects into small modules and delivering them in continuous cycles, we can flexibly control changes in requirements, thereby improving resource utilization, reducing security risks, and improving software development quality and operational stability.

4.1 Requirements Analysis and Design

The primary task of fully integrating automatic security into the software development lifecycle is to further clarify security requirements and design them reasonably based on actual situations. Before the formal start of a software development project, developers need to discuss security objectives and requirements with stakeholders such as product development managers, software security experts, etc. This should include security requirements such as authentication, access control, and data retrieval encryption for software applications. Then, software developers design security management frameworks, security risk response methods, etc. based on specific security requirements, to ensure that the application design phase in the software development lifecycle has a strong security foundation to the greatest extent [6].

4.2 Security Coding and Review

Doing a good job in secure coding is also one of the comprehensive integration methods for automatic security in the software development lifecycle, which can ensure the scientific and secure nature of software development. Based on this, software development teams should strictly follow the principles of best practices in security management and write security standard code based on actual situations. For example, codes that circumvents common security vulnerabilities are specifically manifested as buffer overflow management code, SQL injection security code, cross-site school-based attack code, etc. In addition, to maximize the quality of secure code, software development teams should also make good use of static analysis tools to carry out automated code detection and work, and test code security. During this process, software developers should strengthen communication and cooperation with security experts to ensure that there are no potential vulnerabilities or risks in the security code.

4.3 Automated Security Testing

The current software development delivery time and R&D complexity continue to increase, and the manual security testing mode used by software enterprises in the past is no longer in use. Based on this, software development technology teams should fully utilize automated security testing technology and continuously improve the integrated security of the software development lifecycle. Automated security testing tools can simulate various network attack scenarios, such as technical vulnerabilities, security vulnerabilities in web applications, identity authentication login, and can also conduct security testing, ambiguity detection, and so on. The above tools can promptly identify various security issues that exist in the software development lifecycle, and provide security repair suggestions for software developers to ensure software development security.

4.4 Continuous Integration and Delivery

Continuous integration and delivery are key aspects of the agile development process in the software development lifecycle. In this regard, the software development team should integrate security into the continuous integration or continuous delivery process, and conduct security checks and testing at every stage of the software development lifecycle. For example, software development enterprises can apply static code analysis and security testing tools in the continuous integration process, to timely discover and fix security vulnerabilities. In addition, automated security testing technology can be applied in the continuous delivery process, and a refined security assessment can be conducted before the software is officially launched.

4.5 Deployment and Configuration Management

The software security development process involves automated deployment, secure communication, and rollback planning, which can help software development teams restore security risks to known states before they escalate. In the process of security configuration management, the software development team should design standardized configurations, conduct regular configuration audits, and timely update software control versions to improve the sensitivity of security storage and management. In addition, technicians can also achieve comprehensive

automation and security integration by dynamically monitoring security vulnerabilities and repairing security patches in a timely manner. They can execute security patch management in temporary environments to improve software development security.

4.6 Operations and Maintenance

The final stage of the software development lifecycle is software maintenance, which involves fixing known security vulnerabilities or defects, adding new features appropriately, and managing software upgrades. In this regard, software development enterprises should establish a major security incident response plan, clarify the work responsibilities of team members, dynamically monitor the software development lifecycle and related basic equipment, and timely discover and repair security risks. In addition, the software development team should also backup and promptly recover data information during ransomware attacks, and regularly conduct security training for the development team, such as safety awareness promotion activities, safety skills training activities, safety seminars, etc., to comprehensively enhance the safety awareness and risk response ability of employees.

5. Conclusion

In summary, accurately designing comprehensive integration methods for automatic security in the software development lifecycle based on actual situations has strong practical significance. Not only can it improve testing efficiency and reduce testing costs in the software development lifecycle, but it can also accelerate software update speed, ensuring software development quality and operational stability to the maximum extent. Based on this, relevant software development enterprises should take measures such as requirement analysis and design, security coding and review, automated security testing, continuous integration and delivery, deployment and configuration management, operation and maintenance to maximize software development efficiency, continuously enhance software stability, and promote the healthy progress of China's software development industry.

References

[1] Gu Kangkang, Zhang Li, Lv Youdong, et al. Exploration of the Current Situation and Application Practice of Computer Software Development Technology [J]. Software, vol.44, no.9, pp.104-106, 2023.

[2] Jie Chunle. Exploration of the Current Situation and Application Practice of Computer Software Development Technology [J]. Network Security Technology and Application, no.7, pp.55-56, 2023.

[3] Gan Jianwen. Research and Application of Cluster Ensemble Algorithm Based on Complex Relationship Mining [D]. Shanxi University, 2023.

[4] Peng Xinyue. Research on Office Chair Design Evaluation Based on LDA and Optimized FAHP Integration Method [D]. Central South University of Forestry and Technology, 2023.

[5] Li Guofan. Research and Inspiration on the Selection Method of Cloud Software Lifecycle Model [J]. Science and Technology Innovation and Productivity, no.3, pp.18-23, 2023.

[6] Liu Mingwei, Xu Liguan, Zhang Lingzhi. Research on Quality Measurement Method for Integrated Software Development Based on Indicator Fusion [J]. Automation Technology and Application, vol.42, no.1, pp.97-99+103, 2023.