

The Impact of the Practical Security Test during the Software Development Lifecycle

Chen, Shiang-Jiun, Pan, Yu-Chun, Ma, Yi-Wei, Chiang, Cheng-Mou

Annette.Chen@ttc.org.tw, YC.Pan@ttc.org.tw, yiweimaa@gmail.com, James.Chiang@ttc.org.tw

Abstract— With the advent of the 5G era, due to the innovation of 5G architecture, open source is widely used under the software-defined everything architecture, software security is even more important. According to National Institute of Standards and Technology (NIST) Special Publication (SP) 800-64 Vol2 (Security Considerations in the System Development Life Cycle; SSDLC), the key security roles and responsibilities that are needed in development of most information systems. Sufficient information about the SDLC will improve the development on the secure software.

On this study, how to ensure software security from the initial requirement to the final release, and even the issues of operation and disposal will be explored. We deploy different test methods in different phases of SDLC, including Software Composition Analysis (SCA), Interactive Application Security Testing (IAST), Static Application Security Testing (SAST), and Dynamic Application Security Testing (DAST), etc. From the experiment, the initial security requirements are improved by detecting the problems, and then the security of the software is improved.

Keywords— Security by Design, trustworthiness, vulnerability, security development lifecycle, National Vulnerability Database, National Institute of Standards and Technology (NIST), SCA, IAST, SAST, DAST

I. INTRODUCTION

The architecture of 5G network adopts an open structure with the advantage of rapid and flexible expansion of network functions. The open architecture adopted by 5G also brings new software security issues and the software security issues is not only the software issue and has also become a national security issue [1]. Users of the exploding array of software systems and software-enabled devices across the ‘connected’ ecosystems have lots of the inherent trade-offs embedded in the many benefits of these globally wired systems. Unfortunately, one significant trade-off involves operating risks arising from the fact of connectivity. These risks are particularly hazardous when visited on insecure critical infrastructure systems. The risks of interrupted or improper operation may not only incur costs or delays, but also pose material threats to public health and safety or to national security. In terms of software security requirements, in addition to the guidelines for software security development, security considerations in organizations and business are also important security requirements. In the requirements phase, software security requirements and organizational security strategies are the security soul of the entire SDLC [2-3]. In design phase, some coding guidelines needed to be specified on the design document to make sure the coding will be on the right stream to avoid security issues [4]. Therefore, software

security is not only related to coding and testing, but also closely related to requirements and design. In addition, software updates, user privacy and disposal during or after operation are also closely related to software security [2]

II. BACKGROUND KNOWLEDGE

There are various types of security testing although as per Open-source Security Testing Methodology Manual (OSSTMM) [5], including vulnerability scanning, security scanning, penetration testing, risk assessment, security auditing, ethical hacking, posture assessment, and so on. To better understand the Basics of Security Testing with SDLC Integration [6], the describe as below table 1:

Table 1 Integration of security processes with the SDLC

SDLC Phases	Security Main Activities
Requirements	Software Composition Analysis
Design	Software Composition Analysis
Coding, Unit Testing, Implement (Build)	Software Composition Analysis; Software Composition (code) Analysis; Static Application (code) Security Testing
Testing	Interactive Application (binary) Security Testing; Dynamic Application (binary) Security Testing; Application (binary) Vulnerability Scan; IoT Vulnerability Scan; Penetration Test;
Operation	IoT Vulnerability Scan; Penetration Test

The activities of each phase are different depending on its goal [7-8] and will be discussed below separately.

A. Requirements Phase

Software requirements are to make your software appeal to the customers and stakeholders. Software security should be considered right from the very beginning to make sure the functions work properly and mitigate the cost during the SDLC. Security can be considered during the requirements phase is so-called secure software requirement. The resiliency, the reliability, and the recoverability will be 3 keys of security factors. Doing the security from the initiate requirement phase will be embedded into the whole SDLC and protect the software on every phase to be attacked. Also avoid the customer information being stolen. IBM Systems Sciences

Institute shows security issue fixed on the development phase can save 99% cost than on the maintenance phase [9]. There are a lot of activities needed to be concerned including the choose of tools for the whole SDLC, properly security training for stakeholders, and Software Composition Analysis (SCA). Open-source information is very important for software development nowadays. In the requirement phase, Open-source license limitations and obligations should be aware. SCA is an automated process that identifies the Open-source software in a codebase. This analysis is performed to evaluate security, license compliance, and code quality.

B. Design Phase

The Design phase is commonly defined as the set of actions that will translate the business and customer requirements from their formal and verbose structure into architectural vectors that can be coded (or implemented) into executable pieces of software [10]. The design phase is also where the programming language, model, and framework will be selected by the development team. When discussing software architecture, it is also important to understand the three-tier and four-tier application architectural standards, which represent the separations of processing between different organizational levels [11]. The most of vulnerabilities discovered in software are caused by bad design, incorrect decisions or little attention has been given towards security during the Design phase. However, security is often seen as an “afterthought”. Therefore, the cost of fixing security issues can be higher than the cost of implementing secure design techniques before the software began to be coded. Availability, authentication, authorization, accounting, and auditing are all important aspects of software. However, it provides a good opportunity for attackers when insecurely designed. Security is a key factor, and in the context of complex software architectures, architects should focus to design process for secure software, threat modeling. This process can be used to precisely map security vulnerabilities and apply security countermeasures to avoid the evolution of vulnerabilities into threats to assets. During the design phase of the SSDLC, establishing and incorporating threat models, risk analysis, and security features before actual development (coding) allows everyone on the development team to fully understand the significance of security and the importance of ensuring the integration of security features in the software project. On this phase, the Architecture Risk Analysis can identify flaws within system designs to improve the security posture.

PTA (Practical Treat Analysis) technical is very important and SCA tool can be deployed to analyse Open-source libraries and components, and also to evaluate the quality of the code.

C. Coding and Unit Testing Phases

In this phase, the development team gets the design documents, the software design is further translated into source code. development, etc. One of the first thing to do is to choose a secure programming language and framework and implement policies and guidelines of how to handle untrusted

data safely. All the components of the software are implemented in this phase. To ensure security, a code review and security design review is done by the development team before the code commits. Software Composition Analysis (SCA); Software Composition (code) Analysis; Static Application (code) Security Testing (SAST) [12] also need to be performed to ensure that development is being done securely. SAST is employed to continuously scan the code during development and get rid of security vulnerabilities as much as possible. SCA tools automatically identify signs of various vulnerabilities in the code.

D. Testing Phase

Testing phase after the build modules are released. The developed software is tested thoroughly with any defect found sent back to the development team to get it fixed. Retesting is done until the software meets the customer requirements. The testing phase under a secured SDLC will involve a series of scan including SCA, Interactive Application (binary) Security Testing (IAST), Dynamic Application (binary) Security Testing (DAST), and Penetration Test to make sure no severe bugs make it to production. SCA tools can keep track of an application’s dependencies and send alert if publicly disclosed vulnerabilities are found in the application. IAST analyses code for security vulnerabilities while the app is run by an automated test, human tester, or any activity “interacting” with the application functionality. This technology reports vulnerabilities in real-time. DAST is a black-box testing method. The principle revolves around introducing faults to test code paths on an application. DAST doesn't require source code or binaries since it analyses by executing the application. Penetration Test an authorized simulated cyberattack on a computer system, performed to evaluate the security of the system.

E. Operation Phase

After successful testing, the software is released for customers. Beta testing is performed as soon as the software is employed. If any bugs are found, it will be given to the development team to fix it. Once all beta testing is done, the software is released for the final deployment. Final gap analysis, final security test review, final privacy review, and Open-source licensing review are major activities. SCA and Penetration Test will be performed as well.

III.IMPLEMENTATION

In the software development lifecycle, the strategy of security requirements plays an important role in the completion of an organization and even a single project. Figure 1 indicates the Software Development Life Cycle and corresponding security testing.

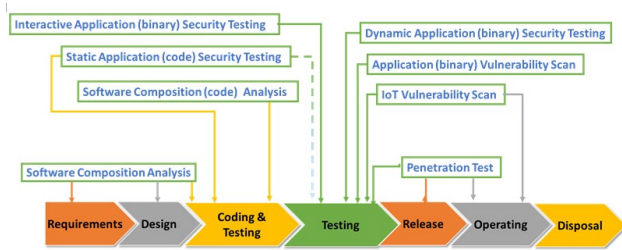


Figure 1 Software Development Life Cycle and corresponding security testing

In this experiment, different security tests were used. In the first experiment, we used black duck SCA tool, and the results are shown in figure 2.

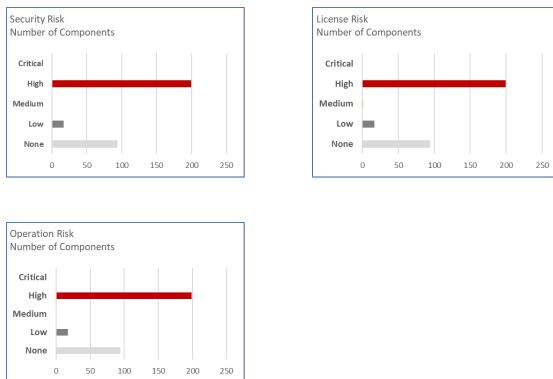


Figure 2 : Result of SCA scan

From figure 2, we can see if the license risk doesn't fix, it will be the operation risk. In the operation risk, in addition to the risk found in the license, the version age is compared, and the activity of the software commit is also compared. For security risk, most of them are exploitability risk. In the second experiment, we used WhiteSource [13] Open-source security & license management to determine the security risks of libraires of open source, version compatibilities, software dependencies, and so on. (See figure 3)

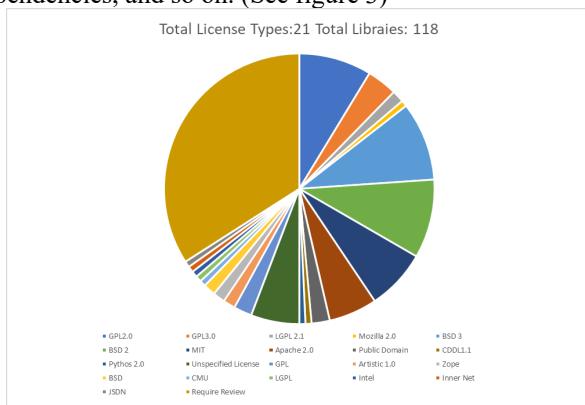


Figure 3 Result of SAST scan

In addition to relying on a good software security architecture, hardware security is also very important. In the third experiment, we used Digit Twin [14] as our security tool

for hardware, configuration, and policies scanning. In addition, in the SDLC process, it is also important to ensure the safety of the process. (See Figure 4)

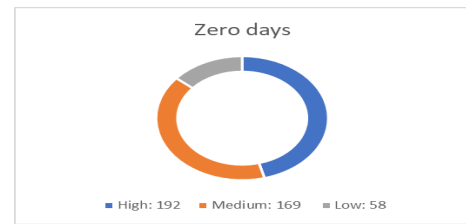


Figure 4: Result of Zero days

IV. ANALYSIS AND DISCUSSION

According to the above experimental results, in the security requirements phase, SCA is used to assist in determining whether the open source used has authorization issues, which can avoid future commercial risks. In the coding phase, the SCA tool can judge whether there is an exploitability risk based on the NDV database, the software dependency. When a product is in operation, there may be risks caused by the open source not being updated to latest version. When the users can perform the architecture risk analysis by using SCA tool in the requirement phase, the risk of using open source can be greatly reduced. Secondly, if the architect can provide a secure architecture for the development to follow, the risk can also be greatly reduced. Therefore, requirement review, code review at each stage and the use of tools to assist in detection can assist to have a secure software product.

The number of the vulnerabilities found might depends on the collection of the database. Once you encounter a new type of state (a vulnerability that may be synthesized by a combination of multiple intrusion methods), it is easy to cause software insecurity. To disassemble to assemble language might be the way to find the vulnerability or defects if there is no an existing database.

According to the benchmark developed by the Build Security in Maturity Model (BSIMM) [15], SSDL Touchpoints can evaluate its software security maturity based on architecture analysis, code review, and security test. It can also be based on its development of penetration testing, software environment, and configuration management & vulnerability management to judge its safety maturity.

In addition to using management tools to ensure that all reviews can be executed correctly and properly, it is also extremely important to use different tools to do cross function scan in different phases. In addition, according to BSIMM, it can help users understand the safety maturity of their products and can improve the safety of the entire product.

V. CONCLUSIONS AND FUTURE WORK

In a software development lifecycle process, in addition to have rigorous security requirements in the requirements phase, various security tests must be carried out at various stages. If

these security tests are performed in the testing phase, and then updated to the latest software, consideration must be given to them. To the compatibility problem, it will lead to the complexity of the repair. In the future, more security testing will be employed to different product to figure out the impact of security testing in each phase. And to discuss whether there are different degrees of impact on the security requirements for different fields.



Cheng-Mou Chiang received the M.S. degree in Electrical Engineering of National Taiwan University of Science and Technology, Taipei, Taiwan. His research interests include network function virtualization, and software-defined network.

REFERENCES

- [1] Jorge Ruiz-Vanoye, Ocotlán Díaz-Parra, Ana Canepa Sáenz, Ricardo A. Barrera-Cámara, and María Beatriz Bernábe Loranca, Alejandro Fuentes-Penna: Strategic Planning for the Computer Security: A Practice Case of an Electrical Research Institute, May 2014
- [2] Jorge Ruiz-Vanoye, Ocotlán Díaz-Parra, I.R. Ponce-Medellín, and Juan Carlos Oliva: Strategic Planning for the Computer Security: May 2008
- [3] code.documentation.Excerpts.pdf (buffalo.edu), Feb 2003
- [4] OSSTMM 3 (isecom.org), 2010
- [5] Basics of Security Testing with SDLC Integration - Advaiya, 2017
- [6] 2013_019_001_297287.pdf (cmu.edu), 2013
- [7] NASA Software Assurance Technology Center. Software Assurance Guidebook, NASA-GB-A201, 2004
- [8] Kelly Blincoe, Daniela Damian, and Giuseppe Valetto: IBM System Science Institute Relative Cost of Fixing Defects | Download Scientific Diagram (researchgate.net), 2017
- [9] Mano Paul, Official (ISC)2 Guide to the CSSLP, 2016
- [10] What Is the Software Development Life Cycle (SDLC) and How Does It Work? | Synopsys
- [11] Mohammad Ubaidullah Bokhari, Shams Tabrez Siddiqui, Hatem S. A. Hamatta: Object Oriented Software Security in Design Phase, 2012
- [12] Static Application Security Testing, What Is SAST and How Does Static Code Analysis Work? | Synopsys
- [13] Whitesource:WhiteSource-Vulnerability-Database.pdf (whitesourcesoftware.com)
- [14] Cybellum: Cyber Digital Twins | Cybellum
- [15] BSIMM: Building Security In Maturity Model | BSIMM



Shiang-Jiun Chen is working in TTC (Telcom Technology Center) Taiwan and received her PhD degree in the The University of Texas at Arlington. Her research interests include numerical analysis, software defined networking, software testing.



Yu-Chuan Pan is working in TTC (Telcom Technology Center) Taiwan and received the PhD degree in the Department of Information Management of National Taiwan University. His research interests include AIoT, Cyber security, Internet service, 5G and cloud computing.



Yi-Wei Ma is an assistant professor in National Taiwan University of Science and Technology. He received the PhD degree in the Department of Engineering Science at National Cheng Kung University, Tainan, Taiwan. His research interests include internet of things, cloud computing, future network and ubiquitous computing.