

Trustworthy Software Development – practical view of security processes through MVP methodology

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

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

Abstract—With the popularization of 5G, one of its characteristics supports a large amount of IoT devices and a lot of open sources will be introduced. Therefore, trustworthiness has become more important for users and developers. According to IBM Systems Sciences Institute, we understand that taking software security into consideration in the early stages of development will save 99% cost. Therefore, security by design can not only save development costs but also improve the overall security of the software.

In this paper, we mainly focus on the trustworthiness aspect in the software development process based on ISO/IEC/IEEE15288, NIST800-160, DoD Enterprise DevSecOps Reference, and A Guide to the Project Management Body of Knowledge (PMBOK Guide) - Seventh Edition and the Standard for Project Management concept. A trustworthiness framework will be proposed to make software more reliable and trustworthy for users and allows enterprises and developers to control and test the trustworthiness of production. After that, the analysis of software security is provided. This platform will provide the guidance for industries easier to use and apply to their product.

Keywords—Security by Design, trustworthiness, vulnerability, security development lifecycle, National Vulnerability Database, National Institute of Standards and Technology (NIST), Open Web Application Security Project (OWASP), DevSecOps, V Model

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. According to the National Vulnerability Database (NVD), 93% of vulnerabilities are application-level vulnerabilities. The software meets the functional requirements, but it cannot be guaranteed to be safe. According to the IBM Systems Sciences Institute [1], the cost of troubleshooting the software in the design phase is one percent of the maintenance phase. Microsoft Security Development Lifecycle (SDL), National Institute of Standards and Technology (NIST) Special Publication (SP) 800-64, Open Web Application Security Project (OWASP) Software Assurance Maturity Model (SAMM), etc. all have definitions of software secure. Since 2004, Microsoft has adopted the self-developed SDL as a company-wide policy. The SP 800 series proposed by the NIST is a series of security standards related to Information Technology (IT). Among them, SP 800-64 Security Considerations in the Information System Development Life Cycle (SDLC) are security considerations

that need to be paid attention to when discussing the system development life cycle. Assist the machine in the establishment of basic safety activities and integrate them into the establishment of the SDLC, making it more cost-effective, risk-based, safety control identification, development, and testing. SAMM is a framework proposed by the OWASP organization. Its purpose is to assist organizations facing risk threats to develop software security strategy that corresponds to their implementation.

II. BACKGROUND KNOWLEDGE

To better understand the so-called trustworthiness software, some standards and reference guide are described below, including ISO/IEC/IEEE 15288: 2015 (E) [2] which describes system life cycle processes, National Institute of Standards and Technology (NIST) Special Publication (SP) 800-160 [3] which defines trustworthiness properties and system life cycle, and then DoD Enterprise DevSecOps Reference Guide [4] introduces the implementation and operational guidance for DevSecOps.

A. ISO/IEC/IEEE 15288: 2015 (E)

As the system becomes more and more complex, the artificial deficiency is generated, so how to create a suitable product will depend on a well-defined framework. This International Standard provides a common process framework for describing the life cycle of systems created by humans, adopting a Systems Engineering approach. It defines some phases: firstly, define stakeholder needs and required functionality early in the development cycle. Secondly, define documenting requirements. And then proceeding with design synthesis and system validation while considering the complete problem. The full life cycle of systems includes acquisition and supply of systems, conception, development, production, utilization, supporting and retirement of systems. This document contains (1) overview (2) conformance (3) normative references (4) terms, definitions, and abbreviated terms (5) key concepts and application of this international standard (6) system life cycle processes. The first 4 items provide information. The 5th provides the important concept and 5 processes catalog which are agreement, organizational project-enabling, technical management, and technical. Each catalog contains a several processes and each process has its title, purpose, outcome, activities, and tasks. See below figure 1 for details.

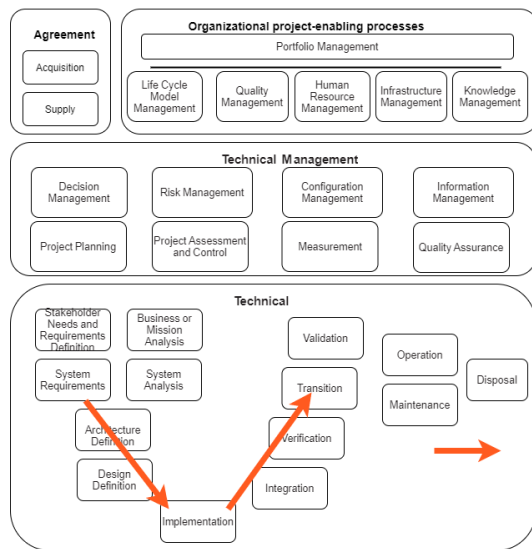


Figure 1: 30 processes for SDLC

From Figure 1, the concept process has two processes, namely Acquisition and Supply. After that, the organization project-enabling process was started, including Life Cycle Model Management Process, Infrastructure Management Process, Portfolio Management Process, and Human Resource Management Process, Quality Management Process, and Knowledge Management Process. After the organization is determined, the plan management process starts, including Project Planning Process, Project Assessment and Control Process, Decision Management Process, Risk Management Process, Configuration Management Process, Information Management Process, Measurement Process, and Quality Assurance Process. The final start of the technical process includes Quality Assurance Process, Stakeholder Needs and Requirements Definition Process, System Requirements Definition Process, Architecture Definition Process, Design Definition Process, Design Definition Process, Implementation Process, Integration Process, Verification Process, Verification Process, Validation Process, Operation Process, Maintenance Process, and Disposal Process. In addition to operation, maintenance, and disposal procedures, technical procedures are a V model of the software development life cycle.

B. National Institute of Standards and Technology (NIST) Special Publication (SP) 800-160

The need of trustworthy secure systems from stakeholder because of the need of the mission, business, and a spectrum of other objectives and concerns. Today's systems have dimensions and complexity that require a structured engineering approach to achieve any expectation that the inherent complexity can be effectively managed within the practical and feasible limits of human capability and certainty. One way to achieve in a secure complex system is to apply activities to each phase during the development life cycle. Trustworthiness means worthy of being trusted to fulfill

whatever critical requirements may be needed for a particular component, subsystem, system, network, application, mission, enterprise, or other entity. Trustworthiness requirements can include attributes of safety, security, reliability, dependability, performance, resilience, and survivability.

This standard focus on two parts: the fundamentals and system life cycle. The object of the fundamentals is security, including system security engineering, system and system elements, system security perspective, and systems security engineering framework. The systems security engineering discipline provides the security perspective to systems engineering processes, activities, tasks, products, and artifacts. System elements include technology/machine elements, human elements, and physical/environmental elements. System security perspective concerned on protection capability and security, system security and failure, strategy for system security, beyond verification and validation-demonstrating system security, system characteristics and system security, and role of systems security engineering. System security engineering framework contains problem context, solution context, and trustworthiness.

The system life cycle describes the security considerations to system life cycle processes, extending from ISO/IEC/IEEE 15288.

C. DoD Enterprise DevSecOps Reference Guide

The DoD Enterprise DevSecOps Reference Design describes the DevSecOps lifecycle, supporting pillars, and DevSecOps ecosystem. The reference also provides implementation and operational guidance. It mentions that the DevSecOps architecture must have the capability to meet the operational software requirements, including business systems, control systems, embedded and weapon systems, intelligence analysis systems, autonomous systems, and assisted human operations. The DevSecOps also mentions that there are some key principles should be considered such as automation as much as it can to avoid human being errors, deploys common tools through the whole software development cycle. The DevSecOps means development, security, and operation. The cross-functional skill sets of Development, Cybersecurity, and Operations throughout the software lifecycle in parallel instead of waiting to apply each skill set sequentially. Security risks of the underlying infrastructure must be considered, so that software risk can be understood. Each organization has its own culture, practice will bridge the gap among developers, security team, and the operation team. Due to different organization's culture, the processes should be unique. The best practice is to shift their culture, evolve existing processes, adopt new technologies, and strengthen governance. In this guidance, it separates the software lifecycle phases to nine phases: plan, develop, build, test, release, deliver, deploy, operate, and monitor. Security is embedded within each phase. The DevSecOps, the software development lifecycle is not a linear process as waterfall. The DevSecOps lifecycle is adaptable and has many feedback loops for continuous improvement. Four pillars for the DevSecOps are organization, process, technology, and governance. To start small with some tasks that are easy to automate, then gradually build up the

DevSecOps capability and adjust the processes to match is a good way to practice the DevSecOps. A software system can start with a Continuous Build pipeline, which only automates the build process after the developer commits code. Over time, it can then progress to Continuous Integration, Continuous Delivery, Continuous Deployment, Continuous Operation, and finally Continuous Monitoring. The instantiation of the DevSecOps environments can be orchestrated from configuration files the DevSecOps tool configuration scripts, configuration code, and the application run-time configuration scripts. Those configurations are treated as software and go through the whole software development processes including design, development, version control, peer review, static analysis, and test. Governance is the function to manage the risks throughout the lifecycle. Through the DevSecOps ecosystem, a set of tools and the process workflows can be created and executed throughout the full DevSecOps lifecycle. The plan phase involves activities on time, cost, quality, risk, and issues. These activities include business-need assessment, project plan creation, feasibility analysis, risk analysis, business requirements gathering, business process creation, system design, DevSecOps design and ecosystem instantiation, etc. It is a best practice to develop a minimum viable product (MVP) for critical business needs as the first thing to develop. In the production environment, the released software is pulled from the released repository and deployed. And then the operations, operation monitoring, and security monitoring are performed. Production operation tools should be selected based on system functional requirements.

III. IMPLEMENTATION

On 5G era, billions of devices are connected to internet. Trustworthiness becomes more and more important for users and developers. A trustworthiness framework with software development process will be proposed. The platform will start with the minimal viable product (MVP) which is a platform that determines if a project is compatible with the required software development, ensuring reliability, trustworthiness, and security fundamentals. The framework will examine the source code and binary code during different development phases. The framework makes software more reliable and trustworthy for users and allows enterprises and developers to control and test the trustworthiness of internal self-production software and integrated external source code. This MVP contains several phases that is requirements, design, coding & testing, build, testing, release, operating, and disposal (see figure 2.) In contrast to PMBOK [5], the requirement is the initial stage, design is the plan stage, coding & testing to testing is in execute & monitor stages, the release is in the close stage. For DevSecOps, operating and disposal stages are added. On this trustworthiness framework, each phase has corresponding security documents and testing to make sure the software developed under this environment is secure.

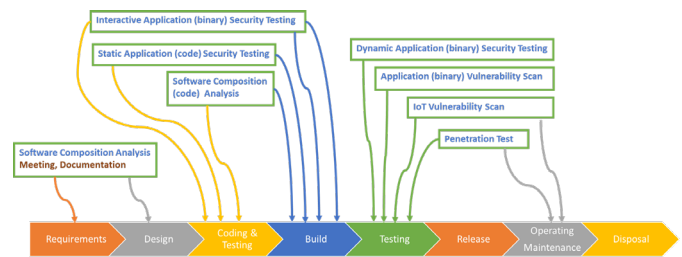


Figure 2 Software Development Life Cycle and corresponding security testing

Firstly, on the requirements and design stages, agreement process, organizational project-enabling process, technical management, and part of technical will be applied. (See figure 2.) Those processes will include several activities which are meeting, documents relative to the meeting, and software composition analysis (SCA). The meeting activities are for acquisition, supply, portfolio management, life cycle model management, infrastructure management, human resource management, Quality Management, knowledge management, decision management, risk management, configuration management, information management, project planning, project assessment and control, measurement, quality assurance, business or mission analysis, stakeholder needs and requirements definition, system requirements definition, architecture definition, design definition, and system analysis. The SCA is performed to evaluate security, license compliance, and code quality, especially on the open sources' licenses.

Secondly, on the coding & testing stage, which is execute phase according to PMBOK, implementation process will be used. The activities have meeting, documentation, software (code) composition analysis (SCA), static application (code) security testing (SAST), and interactive application (binary) security testing (IAST). SAST is a white-box testing tool and is used to scan an application's source, binary, or byte code. It identifies the root cause of vulnerabilities and helps remediate the underlying security flaws and is a frequently used on application security. SAST can analyse an application from the "inside out" and do not need a running system to perform a scan. IAST analyses code for security vulnerabilities while the application is run by testing, or any activity "interacting" with the application functionality and reports vulnerabilities in real-time, which means it does not add any extra time to your CI/CD pipeline.

Thirdly, on the build and testing stage, integration, verification, transition, and validation processes will be applied. The activities have IAST, SAST, SCA, Dynamic Application (binary) Security Testing (DAST), Application (binary) Vulnerability Scan (AVS), IoT Vulnerability Scan, and Penetration Test. DAST is a black box testing without knowledge of the application's internal interactions or designs at the system level, and with no access or visibility into the source program. An application's responses to simulations to determine whether the application is vulnerable and could be susceptible to a real malicious attack. A penetration test is an attempt to evaluate the security of an IT infrastructure by trying to exploit vulnerabilities.

Fourthly, the release stage which is part of close phase. In this stage will build the knowledge by lesson learn and make the root cause analysis report which will be knowledge process and be an assessment of the organization.

Finally, Operation stage contains operation/maintenance and disposal. There are some operation/maintenance and disposal relative security documents should be defined. For maintenance, the reliable of the software source and update is very important. The relative security testing will be executed to make sure the trustworthiness.

To have this platform implemented, there are some modules should be defined which are project management, account management, configuration management, test management, schedule management, tool resource management, resource management, and vulnerability tracking management. Not all processes will need to apply to software development cycle because of the organization culture, size of the product, and so on. On the platform, we should provide the correct configuration to make sure the corresponding testing is done and no critical issue before the next stage being launched.

IV. CASE STUDY

To better understand the platform, we use a software product without any security requirements and design to do testing. Compared with a well-design software, there are not only vulnerability issues but also bugs, code smells, and duplications (see figure 3.) There are a couple of processes to defect those issue before it goes to product stage by using this platform.



Figure 3 SAST testing

To solve the above issue, we back to plan management process to define more secure cipher algorithm and review the implement process about the code review. After that, we could find the most of bugs got fixed. However, this is a small-scale of software and we must do more experiments to understand how these processes improve the security of software.

V. CONCLUSIONS AND FUTURE WORK

A large amount of open sources' software is widely used in software development, it causes more probabilities to have vulnerabilities exist and difficult to detect.

It is not secure enough if only follow the individual industry certification standards. For example: you might follow ETSI EN 303 645 and pass the testing. However, it might not be trustworthy because that the library you employed might not be safe. Nowadays, the software structure become more complex, so the waterfall could not fit the rapid need. The NIST recommends change the development phases to processes. The users can define which processes should be employed to the development life cycle by the users' need.

However, this makes the design and development of the platform more difficulty. In addition to, the devices rapid needed because of the Massive Machine Type Communications (mMTC) characteristics of 5G. The operation system and the language are diverse. It's difficult to defect the vulnerabilities. The future work, we will work on the multiple development environments and work with different industries to find a good guidance.

References

- [1] Dr. P. K. Suri, Pooja Study of Software Quality and Risk Estimation and Quality Cost Analysis using empirical study 2015 IJECS Volume 4 Issue 7
- [2] ISO/IEC/IEEE15288: 2015 <https://www.iso.org/standard/63722.html>
- [3] NIST800-160 2018
- [4] DoD Enterprise DevSecOps Reference 2020 https://dodcio.defense.gov/Portals/0/Documents/DoD%20Enterprise%20DevSecOps%20Reference%20Design%20v1.0_Public%20Release.pdf
- [5] A Guide to the Project Management Body of Knowledge (Pmbok Guide) - Seventh Edition and the Standard for Project Management 2021
- [6] ETSI EN 303 645 V2.1.1 2020



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 IoT, 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.



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.



Chi-Chin Tsai is working in TTC (Telcom Technology Center) Taiwan and received his M.S. degree in Yuan Ze University. His research interests include software engineering, information security and cloud computing.