Component Analysis of DevOps and DevSecOps

Jin–Keun Hong
Professor, Division of Information Communication Technology, Baekseok University

Abstract  This paper is analyzed of the characteristics of development operations and development security operations of the software and product, and the use analysis tools from a software code perspective. Also, it is emphasized the importance of human factors and the need to strengthen them, when considering security design rules. In this paper, we consider a secure process for managing change, focusing on fast and accurate decision–making in terms of procedural factors, when considering development security operations. In addition, the paper discussed the need for maturity model analysis in relation to the development security operating characteristics, and analyzed the meaning of the analysis elements through detailed procedures for the strength and integration elements of the dynamic and static elements accordingly. The paper also analyzed factors such as scanning activity and code analysis for threat modeling and compliance and control.

Key Words : Software, Analysis, Safe, Threat, Code

1. Introduction

Demands for development security operations have recently been highlighted throughout the product development cycle, including software. There is a need to apply security technologies to DevOps processes from the beginning of development[1].

In the DevOps environment, each stage is automated and consistent. In this respect,
DevOps is based on cloud application. Then, there is a question about how security elements will be included as a core service of software in this environment. Creating software that meets cloud and security requirements is a basic task for software developers.

Researches related to DevSecOps include as follows. Vaishnavi Mohan et al. presents the best practices related to the automation of the software application process[2]. This paper is emphasized on documentation and logging, collaboration and communication, automation of processes and role separation in the transition of DevSecOps. But what is important among automation is that role-separation elements should be considered. Laurie Williams argue for continued integrated security during the software attack process[3].

In this paper, changes in software are automatically tested and distributed, with measurements taken in hours or days. Studies such as Oswaldo Dian and others focus on DevOps approaches that take security and risk management into account[4]. This paper is a study of DevOps approach considered in the Mexican government's data center construction. This paper presents a success story of information security and risk management. Risk management factors, including task flow or version management, and software management, are considered as a whole.

The configuration of the security utilization cycle consists of that as follows: the planning phase verification phase application phase–product release phase configuration phase and monitoring phase. Looking at each step, the planning stage plans and implements the code based on source code management. During the verification phase, codes are tested and verified according to business objectives. The package application phase identifies, tests and packages the code in the application process. In addition, during the release phase, defects are released along with the product after approval of the release. In the configuration management, i.e. the configuration management phase, the settings for the application are made. And for performance, security, and compliance, there is a monitor for the application.

However, this DevOps concept aims to ensure high quality and reduce the change time between system changes and normal operations. Thus, DevOps will consider quality assurance in terms of software engineering, operations and security on the development side. However, it is natural to apply DevSecOps together in DevOps (Development Security Operations)[5–8].

DevSecOps is understood to provide advantages in terms of compatibility, reliability, speed, resiliency, and automation. In terms of compatibility, it is flexible and can quickly deliver value from customers. On the reliability side, the customer requires a more reliable and available system. In terms of resiliency, systems with recovery capabilities are designed and implemented. On the speed side, it allows them to enter the market more quickly. In terms of automation, it reduces the complexity of the system and increases its scalability.

In this paper, we will examine its characteristics from DevOps and DevSecOps perspectives and focus on the testing tools applied to DevSecOps[9–13].

Research on these topics raises many possibilities and needs for research. The composition of this paper is as follows. Chapter 2 analyzed the characteristics of development operation and development security operation together. The concept of development operation seeks to explore its features and link them to development security operations.

It also focuses on the security analysis tools used in the DevSecOps process, as well as the discussion of secure SDLCs.

2. Characteristics of DevOps
2.1 Development and DevOps

DevOps is considered in terms of automation, sharing, and measurement. However, this development operation is approached from the perspective of continuous integration and continuous application, and is consists of processes such as planning (design and requirements making), coding, build-up (compiled and tested), testing (integrated performance testing), release (release-by-stage testing), application (infrastructure configuration) and operation (monitoring and warning). Finally, the development operation and development security operation processes cover all phases, from planning to operational monitoring[14].

2.2 Software and DevSecOps

Developers must reflect security from the beginning of the development process. In the case of an enterprise, security considerations are not reflected to meet business requirements. Thus, the application program can quickly enter the market without considering the security, when the application program is released on the market. Therefore in order to maintain operations safely and continuously, security must be considered during the development process. Security services are denied being reflected in the development stage due to actual development and release schedules and complexity. Developers must reflect security considerations in the development process. These requests involve applying security controls at the pre-cycle of SDLC. Core security processes must be automated and security vulnerability must be monitored periodically. Security issues must also be discovered and calibrated to improve overall quality of software security. In security activity, these activities must be automated and monitored in conjunction with security audit, monitoring, and alerting systems. In development security operations (DevSecOps), it is important to automatically use code that is considered of security, perform security tests on application programs, and ensure that secure design rules are available[15–17].

In DevSecOps, security responsibility is given to everyone. This is a human element, a link that is the most vulnerable to corporate security. For this reason, security education and awareness training programs need to be conducted regularly on staff to enhance corporate security. For this reason, it must take steps to ensure that security designers work together with developers in the development process. Thus, the quality of the product source can be improved and the cost required for code correction can be reduced. In the development process, human security should fully cooperate with developers. It is required active participation and reviewed the entire software development process (e.g. unit testing, validation and maintenance support for the integrated test process). In doing so, security delays should be reduced.

It is necessary to control the process of safely developing DevSecOps throughout the entire process. For products, versioning and meta-data management are essential components. This element includes change management. Procedures considered for security should be provided a security process. To this end, management must make decisions quickly and accurately. This procedure requires documentation. The entire procedure consists of planning, coding, build-up, testing, release, deployment and operation. In the planning phase, functional and non-functional items are designed to define requirements. The code is stored in the coding phase. it is compiled at the build up phase, and unit testing is done. In addition, integrated security is achieved. In order to ensure the security of performance, an integrated test is conducted during the test phase. During the release phase, a step-by-step test is carried out according to the release schedule. In the management phase, monitoring and alerting are performed.
2.3 Secure SDLC

Thus, the pre-operation cycle of software development consists of each phase such as the training phase, the requirements setup phase, the design phase, the implementation phase, the verification phase and the release phase, and finally the incident response phase.

In configuration requirement phase, it is considered bug quality and conducted a risk assessment of security and privacy. This includes establishing the requirements for design. The implementation phase uses verification tools and removes unsafe vulnerabilities. Static code analysis is performed. During the verification phase, fuzzy tests or dynamic analyses are performed. Then perform an attack interface analysis. In the release phase, an accident response plan is established and the security is finally reviewed.

The final incident response phase is where the accident response plan is implemented. The safe SDLC process includes the following issues: three are included defining security roles and responsibilities, determining human factors for security missions, setting sensitivity levels for the system, classifying information, setting goals for the system’s security profile, creating profiles, decomposing systems, assessing vulnerabilities and threats, selecting and documenting security controls, creating test data, managing and controlling changes, measuring security governance, and withdrawing systems.

In addition, the test items include security tests, integration tests, load and performance tests, system and function tests, regression tests and system acceptance tests.

The concepts of the Security SDLC basic procedures include security design (reduction of the attack surface, DiD, principle of minimum authority, safe preferences), threat modeling (an overview of threat modeling, design of threat models, coding of threat models, testing of threat models), security coding (buffer overruns, integer calculation errors, cross-site scripting, SQL injection, weak ciphering, net, or self-environmental testing).

For NIST, SDLC is divided into initialization phase, development and acquisition phase, evaluation phase, operation and maintenance phase, and disposal phase.

2.4 Maturity model in phase of DevSecOps

To ensure security, the implementation phase should support continued integration and the use of application procedures. It also essentially requires security to be applied to the framework and services. A matured security model should be applied to enhance functionality and a series of security tasks should be strengthened. In addition, developers and operators should be provided with visibility into security activities.

Here, the analysis of the maturity model should be done with balance from a dynamic perspective and a static perspective and from a strength perspective and an integration perspective.

From a static point of view, there is an analysis of how detailed a static code analysis is. Dynamic analysis refers to the depth of dynamic scan implementations. From a strength perspective, the analysis is to determine whether or not robust the system is from the number of attacks carried out. And from an integrated perspective, it is an analysis that identifies how complete a series of analysis processes are.

Of course, security activities that apply at the planning phase include threat modeling. During the coding phase, there have scanning. During the build phase, security activities include testing of security devices, scanning components, and analyzing code. During the test phase, security activities include integrated scanning and testing. During the release phase, there are scanned for the relevant areas.

During the application phase, security activities
include hardening of systems and applications. At
the operational phase, it verify compliance and
control.

Given the policy of security development, the
implementation preparation phase of the
requirements establishes a portfolio and carries
out a risk assessment. It also determines service
levels. The design phase models threats around
the asset. At this stage, threat models and design
reviews are conducted.

In the execution phase, an integrated security
checklist is created and standards. In addition,
this step reviews the code internally and analyzes
the security code. During the verification phase,
a security assessment is conducted and bugs are
tracked. In the release phase, searches and
evaluations are performed at the host level.

When assessing a threat level, the risk level
consists of three steps: high, medium, and low. At
a high level, risks are subject to threat models
and design reviews (code reviews, penetration
tests, privacy reviews, application reviews). In the
interim phase, the risk is subject to a model of
threat, a review of code (white box), a review of
privacy, and an application review. Finally, at a
lower level, the risk is subject to a review of the
model and application of the threat.

2.5 Metrics of DevSecOps

DevSecOps’ common components include code-based
infrastructure, automation and underlying
communications. Platforms to be applied are
from IaaS to PaaS. DevSecOps’ platform consists
of a description, a maturity model, a metric, and
a linearity. In a standard platform, the model of
maturity must meet a certain level of maturity.
Metrics are the main indicators for implementing
the DevSecOps framework.

Management elements include image, logging/
monitoring, patches, platform governance,
change, development and testing, distribution,
authority and credentials, availability, network,
operations and maintenance, backup, contracts,
and so on.

Image management is related to image creation,
maintenance, and transmission. It is provided
logging/warning/monitoring management information
and security robustness. Patch Management
maintains the security intensity. Governance
includes processes for security management and
availability. Changes to the application include
changes to management control. The test is
integrated with the unit test.

Deployment should be supported to ensure
that systems and applications are not down. The
management of authority and credentials is
concerned with the permission or sharing of
secrets. Availability should be provided for
systems and applications. Network management
involves the maintenance of network services.
Operation and maintenance is required after the
implementation process. Backing up requires
restoring transactional data.

The metrics associated with data recovery
include the following factors: Number of
deployments, code commit and deployment time,
volume, failure rate, average recovery time, up
and down time for availability, customer
troubleshooting time, time to resolution of
customer issues, time to fulfillment, and time.
Metric elements related to security development
security are included as a follows: Number of
automated tests based on AI, security patches,
event information of availability, time to engage
in commit code development, implementation or
rejection of platform governance, time from code
commit to change and patch management,
number of deployed platforms, and number of
failed deployed platforms. Failed platform,
number of published images, logging data,
warning number, test (function, integration)
number, test (function, approval) number,
average recovery point, retention control,
instantiation time from request to response,
security variation from application to image
instantiation, security control number, patch
number, patch number, patch time, security compliance number from commencement to completion, operational and maintenance number, test and test and security management number, test and test and test and test and test number and security number.

3. Conclusion

In this paper, it is analyzed its characteristics with a connection between DevOps and DevSecOps of software and products. This paper considers the need for a maturity model analysis in relation to DevSecOps characteristics, and detailed analysis factors can be described procedure for strength of dynamic and static elements, and integration elements in detail considerations. Of course, there are emphasized scanning activities or code analysis for threat modeling, and compliance and control.

REFERENCES


홍 진 근(Jin-Keun Hong)  

학부 교수

- 관심분야 : 융합보안
- E-Mail : jkhong@bu.ac.kr