International Journal of Security and Its Applications
Vol. 10, No. 6 (2016) pp.145-154
http://dx.doi.org/10.14257/ijsia.2016.10.6.15

Adaption of Integrated Secure Guide for Secure Software Development Lifecycle

Ki-Hyun Lee[1] and Young B Park[2]

[1]*Department of Software Security,*
[2]*Department of Computer Engineering*
Dankook University,
[1]*Juk-Jeon,* [2]*Cheon-An*
South Korea
qkqn147@gmail.com, ybpark@dankook.ac.kr

Abstract

As interests in security increases, several software development lifecycle considering security have been proposed. Actual results of applying software development lifecycle considering security reduced threats have been reported. But to apply software development lifecycle considering security, requires expertise and experiences. In this paper a secure software development lifecycle applying integrated secure guide is proposed. Secure Guides such as Misuse Case, Security Patterns and Secure Coding are integrated in the proposed method. And then, by applying integrated Secure Guide in each software development phase, the security becomes available in every phase of software development. Since, proposed secure guide provides more security information than available secure guides, software developer can use more security information while they are developing software. As a result, it is expected that developers could consider security more easily when developing software even though developers lacks expertise in security or experience.

Keywords*: Integrated Secure Guide, Secure Software Development lifecycle, Misuse Case, Security Patterns, Secure Coding*

1 Introduction

Security accidents have been increased every year, and in 2015, accidents were increased by 38% comparing to 2014[1]. 75% of security accidents were caused by software vulnerabilities, and accidents which were caused by hardware vulnerabilities were just 4% [2]. As software threats increase, the importance of software security has also increased. Therefore, to develop a secure software, many researches that adapt integrated secure guide in each software development process have been progressed [3, 4]. These software development processes, considering security actually resulted the reduction of software vulnerabilities [5]. But, because software development process, considering security often needs expertise in security or experience, developers may encounter difficulties in adapting a modified software development process. To solve these difficulties, a case of having a training session about security before software development exists [5], but extra charge is needed in software development process. To solve these various problems, this paper proposes a method which allows considering security easier in software development process, by using integrated secure guide. The reason of using integrated guides instead of just using existing secure guides is because, a problem of secure guides having different information according to the institution that provides secure guides. So by integrating secure guides, this problem can be solved.

ISSN: 1738-9976 IJSIA

This paper consists of 6 parts. Part 2 describes related works. Part 3 describes integration of secure guides. Part 4 describes adapting integrated secure guide in software development process. Part 5 describes about the evaluation of proposed method. And finally, in Part 6 describes the conclusion of this paper.

2. Related Works

2.1. Secure Software Development Lifecycle

Secure Software Development Process is a development process to consider security before software development. Secure Software Development Process considers security in every software development phase by adapting security technologies which are capable of adapting in each software development phase. Because a security technology that is being adapted in each phase of Secure Software Development Process is not standardized, difference in security technologies in each phase exists between development groups. So, various Secure Software Development Process are being proposed, due to the difference in applying security technologies [6, 7, 8]. MS-Security Development Lifecycle [5], CLASP (Comprehensive Lightweight Application Security Process) [9], 7Touchpoints are typical cases[10]. Especially, MS-SDL proposes methods that consider security not only in development process but also before and after the development process.

2.2. Misuse Case

Misuse Case is an expanded model of Use Case, and is proposed to complement the difficulties of applying non-functional requirements [11]. Misuse Case is created from existing Use Case, by adding cases of system-misusing occasion, ≪Threaten≫ and ≪Mitigate≫ relations. So, Misuse Case can reflect both functional and non-functional requirements. By using Misuse Case, it is possible to predict threats of the software to be developed and predicted threats lead to consideration about security in Requirement & Analysis phase. An extended model of Use Case which is similar to Misuse Case are Abuse Case and Security Case [12, 13, 14], and they will be used to reflect non-functional requirements as well.

2.3. Security Patterns

Security Patterns is a security technology that can be considered in Design Phase. Security Patterns provide Secure Architecture and Secure Design. Security Patterns is proposed from existing Design Patterns regarding security. These Security Patterns are proposed in multiple institutions, and provides information as a form of guide [15, 16, 17]. Security Patterns provided as a form of guide can be used to define Secure Architecture and Secure Design in software development group, and as it is provided as a form of guide, information can be easily used. Considering security in Design Phase is available by using Security Patterns Actually, researches for applying Security Patterns in software development to consider security are currently in process [18].

2.4. Secure Coding

Secure Coding is defensive programming technique that prevents threats in source code beforehand.

Secure coding aims for cyber-attack prevention, corresponding to well-known threats, threat-modification cost reduction, and securing software stability and reliability. So it provides information such as threat information, threat-included code, and reinforcement codes as a form of guide. By using information of Secure Coding, writing secure code while developing software is available. Secure Coding is standardized. Such as CWE (Common Weakness Enumeration)[19], OWASP(Open Web Application Security)[20],

CERT(Computer Emergency Response Team) Secure Coding[21] provides information about Secure Coding.

2.5. Penetration Testing

Penetration Testing is a technique that considers security in Testing Phase of software development.

Penetration Testing is a testing technique that virtually attack software using predicted threats in software to verify the existence of software's threats [22]. By Penetration Testing, it is possible to detect threats which were not predicted in software development process, and verify whether alternative plans for predicted threats are established correctly. As there be existent various types of threats, developers suffer difficulties in executing Penetration Testing. Because of this reason, Penetration Testing is executed by using Penetration Testing tool. By using Penetration Testing tools, allows P-Testing effectively by using predefined Penetration model. Metasploit is a well-known tool for penetration testing [23].

3. Secure Guide Integration

3.1. Misuse Case Integration

Several Misuse Cases can exist according to the methods to extend Use Case. These several Misuse Cases have different feature according to the purpose of extension. Table 1 shows the comparison between Misuse Case, Abuse Case, and Security Case. Looking at Table 1, there are differences in information between each case. Because of these differences, additional operations occur to select proper Misuse Case. In this paper, Misuse Case is integrated to minimize additional operation. By integration of Misuse Case, it is possible to supplement the information that was not able to express in each Case. Integration of Misuse Case is done by integrating Misuse Case which is defined with Misuse and Relation and defined Security Case based on security requirements. As Security Case include Misuse Case, integration of Misuse Case and Security Case is done by adding Relation of Misuse Case to Security Patterns. This integrated Misuse Case can express more information comparing to existing Misuse Case.

Table 1. Misuse Case Comparing

Guide / Info	Alexander Misuse Case	McDermott Abuse Case	Firesmith Security Case
Aspect	Attacker	Attacker	Defender
Case Info	Misuse	Abuse	Misuse & Security
Actor Info	X	Resource, Skill Objective	X
Relation	≪Threaten≫ ≪mitigate≫	X	X

3.2. Security Patterns Guide Integration

As Security Patterns Guide is not standardized, Security Patterns Guide provides different information according to institution that provides guide. Table 2 shows comparisons of Security Patterns according to providing institutions. Looking at Table 2, it is possible to find out that some information is not provided according to each Security Patterns Guide providers. Developers might face problems of while applying Security Patterns. To solve this problem, this paper suggests integrating Security Patterns Guide. Integration of Security Patterns Guide uses 'Security Patterns name' which is one of the information that guide provides. The reason of using Security Patterns Guide's name is because, even if Security Patterns Guides according to several institutions are different, the information about name of Security Pattern is commonly provided. Integration of Security Patterns Guide having different information is done by using commonly provided Security Pattern's name. This integrated Security Patterns Guide can provide much more information compared to existing Security Patterns Guide.

Table 2. Security Patterns Guide Comparing

Guide / Info	Carnegie Mellon	East Tennessee State	MunawarHalfiz
Category	X	Security Category	STRIDE Model
Part of System	X	X	Core, Perimeter, Exterior
Intent	O	O	O
Motivation	O	O	O
Applicability	O	O	O
Structure	O	O	X
Participants	O	O	X
Consequence	O	O	X
Implementation	O	X	X
Sample code	O	X	X
Know Uses	O	X	O

3.3 Secure Coding Guide Integration

Unlike Security Patterns Guide, Secure Coding Guide has international standards. But, there are several institutions that provide Secure Coding Guide like Security Patterns Guide. Table 3 shows comparisons between Secure Coding Guide according to providing institutions. Looking at Table 3, it is possible to find out that some information is not provided according to each Secure Coding Guide providers. This can cause problem when developers could not get proper information while applying secure coding. To solve this problem, this paper suggests integrating Secure Coding Guide. Integration of Secure Coding Guide is based on CWE. The reason that is based on CWE is, CWE is not dependent on specific language, and provides more information than other Secure Coding Guide Integration of CWE based Secure Coding Guide uses threat name. In the case of Secure Coding Guide, as it provides information for each threat name, so it is possible to integrate information base on Secure Coding Guide. This integrated Secure Coding Guide can provide much more integrated information than existing Secure Coding Guide.

Table 3. Secure Coding Comparing

Guide / Info	CWE	OWASP	CERT
Category	7 Pernicious Kingdoms	X	Each Language Rules
Risk Assessment Summary	O	O	O
Applicable	Common	Web Application	Each Language
Attack Info	O	O	O
Detect Method	O	O	X
Prevent Method	O	O	X
Code Example	O	X	O

4. Secure Software Development Lifecycle.

Secure Software Development Lifecycle proposed in this paper extends software development methodology using Integrated Secure Guide proposed before [24]. Picture 1 shows the process being proposed in this paper.

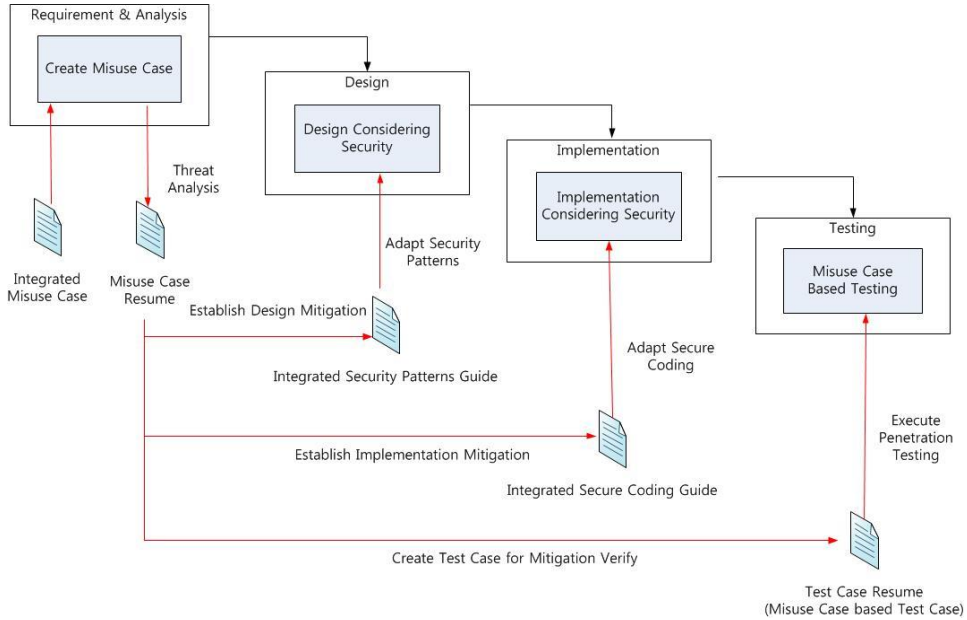

Figure 1. Suggested Secure Software Development Lifecycle

As shown at picture 1, to consider security at each phase, proper security technology is being implied, and this implementation composes Secure Software Development Lifecycle. It will be explained in detail at next section.

4.1. Requirement & Analysis Phase

At Requirement & Analysis phase threat analysis is executed using integrated Misuse Case Guide. At integrated Misuse Case, threats existing in software are predicted and expressed as sentences. As Misuse Case expressed in sentences is insufficient for expressing information about threat, making Misuse Case Resume is required for supplementation. Differ from Misuse Case which is written on simple sentence, Misuse

Case Resume is written with more details. At Misuse Case Resume, information about security properties about threats, object of threat and threats that can exist are written. Written Misuse Case Resume are used to apply Mitigation at Design Phase and Implementation Phase.

4.2. Design Phase

To mitigate threats analyzed in Requirement & Analysis Phase, this paper applies Integrated Security Patterns Guide in Design Phase. To Secure Design by applying Integrated Security Patterns Guide, identifying proper Security Patterns is necessary. To identify proper Security Patterns, Misuse Case Resume is used. A method to identify proper Security Patterns is to use information about Misuse Case Resume and Integrated Security Patterns Guide's security properties. Misuse Case Resume and Integrated Security Patterns Guide identify proper Security Patterns for threat mitigation by matching threat information. Additionally, by using Integrated Security Patterns Guide, it is possible to use more information than existing Security Patterns Guide.

This allows developers to deal with Secure Design more easily.

4.3. Implementation Phase

At Implementation Phase, Integrated Secure Coding Guide is applied to mitigate analyzed threats.

By applying Integrated Secure Coding Guide, eliminating analyzed threat existing in source code of on-developing software beforehand is possible. But, to apply Integrated Secure Coding Guide, process of identifying proper Secure Coding is required, like when applying Integrated Secure Coding Guide. Like Security Patterns, Misuse Case Resume is used to identify proper Secure Coding. Identifying method is to match information about the facts written in Misuse Case Resume about vulnerabilities that analyzed threats have and vulnerabilities existing in Integrated Secure Coding Guide and then, identify the proper Secure Coding. As Integrated Secure Coding Guide provides more information than existing Secure Coding Guide, considering threats not only from Requirement & Analysis Phase but also from Implementation Phase is possible.

4.4. Test Phase

At Test Phase, verification process of Mitigation which is being applied in Design Phase and Implementation Phase is held. Verification of Mitigation uses Misuse Case-based Penetration Testing. The reason of executing Misuse Case-based Penetration Testing is because, both threat analysis and threat mitigation are based on Misuse Case. Simple process of Misuse Case-based Penetration Testing is, first to create Misuse Case-based Test Case and then execute Penetration Testing by using previously created Test Case. According to the result of Penetration Testing, it is possible to recognize that threat mitigation is applied properly if Misuse Case-based Test Case is unable to run in developed software. However, it is possible to recognize that threat mitigation is not properly applied if Misuse Case-based Test Case is able to run in developed software. In this case, re-configuring of Mitigation of Design & Implementation is required.

5. Evaluation

Comparing between existing Secure Guide and integrated Secure Guide was held to evaluate proposed method, Comparing between integrated Secure Guide applied Secure Software Development Lifecycle, existing MS-SDL and security-unconsidered Software Development Lifecycle was also held.

5.1. Integrated Misuse Case

To evaluate proposed Integrated Misuse Case, comparison between Misuse Case, Abuse Case, and Security Case was held. Table 4 compares each Cases and Integrated Misuse Case. Integrated Misuse Case can provide Security Case that Misuse case could not provide and Relation that Security Case could not provide. Additionally, Integrated Misuse Case can analyze threats in attacker's and defender's point of view. As Integrated Misuse Case analyze threat in every point of view, it is possible to express Misuse Case presenting threats and Security Case presenting mitigations. Also, expressing relations between Misuse Case and Security Case is possible by adding <<Threaten>> and <<Mitigate>> relations.

Table 4. Integrated Misuse Case Comparing

Guide / Info	Alexander Misuse Case	Fire smith Security Case	Integrated Misuse Case
Aspect	Attacker	Defender	Attacker& Defender
Case Info	Misuse	Misuse& Security	Misuse& Security
Actor Info	X	X	X
Relation	≪Threaten≫ ≪mitigate≫	X	≪Threaten≫ ≪mitigate≫

5.2. Integrated Security Patterns Guide

To evaluate proposed Integrated Security Patterns Guide, comparison between proposed method and existing Security Patterns Guide was held. Integrated Security Patterns Guide can provide information that existing Security Patterns Guide could not provide. Looking at the comparison between Security Patterns Guide proposed at Carnegie Mellon and Integrated Security Patterns Guide in Table 5, Carnegie Mellon's method does not provide information about Security Patterns Category, however, Integrated Security Patterns Guide can provide information about Security Patterns Category. Also, it is possible to find out that Integrated Security Patterns Guide provides more information than other Security Patterns Guides. By applying Integrated Security Patterns Guide, using more Security Patterns information at Design Phase is possible.

Table 5. Integrated Security Patterns Comparing

Guide / Info	Carnegie Mellon	East Tennessee State	Munawar Halfiz	Integrated Security Patterns
Category	X	Security Category	STRIDE Model	Security Category & STRIDE Model
Part of System	X	X	Core, Perimeter, Exterior	Core, Perimeter, Exterior
Intent	O	O	O	O
Motivation	O	O	O	O
Applicability	O	O	O	O
Structure	O	O	X	O
Participants	O	O	X	O
Consequence	O	O	X	O
Implementation	O	X	X	O
Sample code	O	X	X	O
Know Uses	O	X	O	O

5.3. Integrated Secure Coding

To evaluate Integrated Secure Coding Guide, comparison between proposed method and existing Secure Coding Guide was held. Looking at Integrated Secure Coding Guide in table 6, as Integrated Secure Coding Guide is written based on CWE, Integrated Secure Coding Guide provides similar information to CWE. However, as Integrated Secure Coding Guide integrates not only CWE but also OWASP and CERT, it provides more information than CWE. Comparing with OWASP and CERT, Integrated Secure Coding provides more information than any other Guides. So, by applying Integrated Secure Coding, using more information about Secure Coding at Implementation Phase is possible.

Table 6. Integrated Secure Coding Comparing

Guide / Info	CWE	OWASP	CERT	Integrated Secure Coding
Category	7 Pernicious Kingdoms	X	Each Language Rules	7 Pernicious Kingdoms
Risk Assessment Summary	O	O	O	O (Integrated Info)
Applicable	Common	Web Application	Each Language	O (Integrated Info)
Attack Info	O	O	O	O (Integrated Info)
Detect Method	O	O	X	O (Integrated Info)
Prevent Method	O	O	X	O (Integrated Info)
Code Example	O	X	O	O

D. Secure Software Development Lifecycle

To evaluate Secure Software Development Lifecycle, comparison between proposed method and existing method was held. Table 7 shows comparison between proposed Secure Software Development Lifecycle and existing method. Show the content of table 7, to consider security, existing methods add security activities and phases in Software development lifecycle. To consider security in existing Software Development Lifecycle, MS-SDL adds Training Phase before development to learn expertise in security. Expertise in security before development allowed professional security activities in software development lifecycle. Also, MS-SDL considers security by adding various security activities in every software development lifecycle. In the case of CLASP and 7 Touchpoint, by adding security activities, it considers security at software development lifecycle. But, these security activities require additional costs, so applying existing method is difficult for poor development group. However, proposed Secure Software Development Lifecycle requires lower cost and fewer activities than existing method. Also, even if a developer lacks expertise, applying Secure Software Development Lifecycle is available by using Integrated Secure Guide. But, threat treatment depends on Integrated Secure Guide, threat treatment that is not included in Integrated Secure Guide is limited.

Table 7. Suggested Secure Software Development Lifecycle Comparing

Info \ Lifecycle	MS-Secure Development Lifecycle	CLASP	7Touchpoints	Secure Software Development Lifecycle
Accessibility	Difficult	Difficult	Normal	Normal
Extra Phase	2 (Training, Response)	X	1 (Feedback Phase	X
Extra Activity	17	9	7	4
Extra Cost	High	Normal	Normal	Low
Threat Analysis	O(Detailed)	O(Detailed)	O(Detailed)	O(Brief)
Threat Mitigation	O	O	O	O(Known Threat)
Mitigation Verify	O	O	O	O
Security Operation	O	O	O	X
Expertise	O	High	Normal	X

6. Conclusion

This paper proposes Secure Software Development Lifecycle applying Integrated Secure Guide to consider security more easily in software development phase. As Integrated secure guide is being used in proposed method, Misuse case, security patterns, secure coding which are existing Secure guide are integrated in proposed method. Integrated secure guide can provide more security information than existing secure guides. Integrated Secure Guides allows Developers to consider security based on more security information. Also, to minimize additional cost, proposed method adds minimum security activities. In additional security activities, it was predicted about threat by using the integrated misuse cases guide in Requirements & Analysis phase, and also it was set mitigation up about predicted threat by using integrated the secure coding guide and security patterns guide in Design & implementation phase. And Integrated Secure Guide is verified by Penetration Testing based on Misuse case. Adaption of Secure Software Development Lifecycle is simplified by applying Integrated Secure Guide and adding minimum security activities.

But Secure Guide used in suggestion only provides information about well-known threats. By this reason there are some limits handling unknown threats. This limitation can be handled by immediate updates, but it is not that easy work to update Secure Guide yet. In the case of unknown threats, Identification of unknown threat is necessary a lot of time, because it is difficult to identify unknown threats. For this reason, to reflect the information of unknown threat, it is difficult to update the Secure Guide. As Secure Guide proposed in this paper only uses Misuse Case, Security Patterns, Secure Coding, it has limitation reflecting various features of software being developed. To overcome this limitation, further research finding additional Secure Guide which can be adapted to Secure Software Development Lifecycle is on progress. Also, it will be researched for identification of secure guide that software features reflected. Personal research if these researches are well-closed, it is expected to progress Secure Software Development Lifecycle using proper Secure Guide.

Acknowledgments

This work was supported by the "employment contract based master's degree program for information security" supervised by the KISA (KOREA INTERNET SECURITY AGENCY) (H2101-14-1001).

References

[1] PwC, "Insight from the Global State of Information Security Survey", **(2015)**.
[2] Gartner, "Now is the time for security at Application Level", **(2005)**.
[3] Apvrille, Axelle, and Makan Pourzandi, "Secure software development by example", IEEE Security & Privacy 4, **(2005)**, pp. 10-17.
[4] J.-R. Choi and Chang-hwan Bae, "Case Analysis and Utilization of Methodologies for Security Development Lifecycle", In Proceedings of Korean Society for Internet Information, **(2010)**, pp. 21-222.
[5] Microsoft, "Introduction to the Microsoft Security Development Life cycle", http://www.microsoft.com/security/sdl.
[6] M.-g. Choi and M.-J. Jeon, "Analysis of Methodologies for Security Development Lifecycle for Security Enhancement System", In Proceedings of the Korea Society of Management Information Systems, **(2010)**, pp. 418-425.
[7] S.-H. Lee, J.-Y. Choi, I.-H. Kang, D.-S. Seo, J.-Y. Ahn and K.-H. Han, "Software Development Lifecycle Trends for Secure Software Development", Communications of the Korea Information Science Society, vol. 28, no. 2, **(2010)** February, pp. 41-47.
[8] B. De Win, "On the secure software development Lifecycle: CLASP, SDL and Touchpoints compared", Information and software technology, vol. 51, no. 7, **(2009)**, pp. 1152-1171.
[9] J. Viega, "Security in the Software Development Lifecycle: An introduction to CLASP, the Comprehensive Lightweight Application Security Process", Secure Software, Inc., McLean, Virginia, USA, White Paper, **(2005)**.
[10] G. McGraw, "Software Security: Building Security In", Addison-Wesley Professional, **(2006)**, pp. 99-118.
[11] I. Alexander, "Misuse cases: Use cases with hostile intent", Software, IEEE 20.1, **(2003)**, pp. 58-66.
[12] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases", Requirements engineering, vol. 10, no. 1, **(2005)**, pp. 34-44.
[13] J. McDermott and C. Fox, "Using abuse case models for security requirements analysis", In Computer Security Applications Conference, 1999(ACSAC'99) Proceedings, 15th Annual. IEEE, **(1999)**, pp. 55-64.
[14] P. M. Stern and H. P. Green, "The Oppenheimer Case: Security on Trial", New York, Harper & Row, **(1969)**.
[15] J. Y. Dangler, "Categorization of Security Design Patterns", East Tennessee State University, **(2013)** May.
[16] C. R. Dougherty, K. Sayre and R. Seacord, "David Svoboda, and Kazuya Togashi, Secure design patterns", Carnegie Mellon University, **(2009)** March.
[17] M. Hafiz, "Security Pattern Catalog", http://www.munawarhafiz.com/securitypatterncatalog/index.php, **(2013)**.
[18] D. M. Kienzle and M. C. Elder, "Final technical report: Security patterns for web application development", DARPA, Washington DC, **(2002)**.
[19] B. Martin, "2011 CWE/SANS top 25 most dangerous software errors", Common Weakness Enumeration 7515, **(2011)**.
[20] J. Williams and D. Wichers, "OWASP top 10–2010", OWASP Foundation, https://www.owasp.org, **(2010)** April 1-5, pp. 8.
[21] Seacord, Robert, "The CERT C secure coding standard", Pearson Education, **(2008)**.
[22] G. McGraw, "Software Security: Building Security In", Addison-Wesley Professional, **(2006)**, pp. 103-104.
[23] D. Maynor, "Metasploit toolkit for penetration testing, exploit development, and vulnerability research", Elsevier, **(2011)**.
[24] K.-H. Lee and Y. B. Park, "A Study of Security Software Development Methodology Using Integrated Secure Guide", In Proceeding The 4th International Conference on Smart Media and Applications, **(2016)**.

