Development of a Quantitative Method for Evaluating the Effect of Cyber Security Control Applications on SDLC in NPPs

Chanyoung Lee [a], Poong Hyun Seong [a*]
[a] *Department of Nuclear and Quantum Engineering, Korea Advanced Institute of Science and Technology, 291 Daehak-ro, Yuseong-gu, Daejeon, 34141, Republic of Korea*
[*] *Corresponding author: phseong@kaist.ac.kr*

1. Introduction

Installation of digital I&C systems and continuous cyber attacks against national infrastructures signified that NPPs are susceptible to cyber attacks [1]. As technologies have developed, public perceptions of cyber security for NPPs have been changed. One of the major problems is that the nuclear industry is in the very early stage of cyber security problems [2]. It is caused by late adoption of digital I&C systems, less attention compared to other safety problems, and the limited experience of cyber attacks. Many cyber security regulatory documents, standards have already been published in the nuclear industry. However, there were still difficulties when it comes to defining appropriate requirements for a security control, and deciding which security functions are installed for a security control [3]. Several methods have been proposed to compare the security controls in terms of security effectiveness [4], [5]. Although the verification of security functions used for security controls should be preceded, there has been no research or guide in the nuclear industry. In RS-015, about 20% of the technical security controls include the statement that the adverse effect of configured functions must be verified. Though it is 20%, those controls are required to have functions with high technology implementation level. Having not considered the fact that security functions when the existing digital I&C systems were designed, the effect of verification on safety is very important. Verification of adverse effects of security controls is also the most controversial issue between the safety and the security fields, and it must be solved for the safe application [1].

2. Analysis of the installation of security functions and developer effort

Security control types need to be classified by roles or implementation methods depending on the evaluation purpose. In this study, security controls were classified into two categories according to how configured functions were implemented.

- Category 1: security control configuring functions integrated with the safety system.
- Category 2: security control configuring functions operated in real time.

For the security control classified category 1, it should be carefully applied with consideration of S/W faults that can occur in an integration process. For the security control classified category 2, it should be carefully applied after conducting verification of effect on system operation. The two categories are not exclusive, and security controls belonging both categories should meet the two kinds of considerations. The method for managing faults that can occur during the integration process is focused in this study.

When new functions are installed in the I&C system, the functions are verified through the S/W V&V process in order to assess the safety implications [6]. There has been an assumption that the complexity of the S/W of the NPP I&C system is low in several researches [7]. However, it is necessary to validate the assumption when security functions are installed. If not validated, complexity of the V&V process should be estimated instead to ensure the integrity of functions. Several researches found that incorporation of S/W changes due to the installation of new functions may cause highly scattered code modification [8]. It increases complexity of the system which also negatively affects the S/W quality. It also makes developers difficult to keep tracking S/W changes. These facts are main reasons for the introduction of faults. Based on these researches, it can be concluded that the installation of security functions may lead to new faults. In addition, the process management for finding and fixing new faults is important. Considering that the main cause of software faults is human error [9], it is important to quantify and reduce the efforts of the developers required for the management.

However, many S/W measures appear in various articles in different forms as an indicator of developer's effort. With this regards, the S/W measures should be grouped by similar concepts, and implication of the group should be summarized. A total of 175 S/W measures were analyzed through a literature survey, and the measures were categorized in five groups. The categories of S/W are summarized in Table 1.

Table 1. Categorization of S/W measures

Group of similar concepts	Number
Measures for S/W size	47
Measures for S/W complexity	96
Code distribution degree	8
Module-network structure	24
Programming code pattern	64
Measures for working environment	32
Total	175

The number in this table is just a count of measurement methods, and it does not reflect importance of methods. Factors affecting the developer's effort have a limited relation with S/W itself in this study. The factors are summarized in Fig. 1.

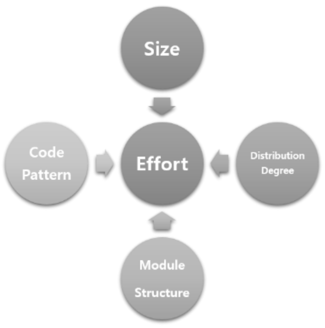


Fig. 1. Factors affecting developer effort

3. Quantification of the Developer Effort

The amount of developer's effort can be quantified as the amount of S/W change information by applying the information theory [10] [11].

$$\text{The amount of } \textit{S/W change information} = -\sum p_i \log_2 p_i$$

In which, p_i is the percentage of the i^{th} unit change among the whole S/W changes.

The concept of entropy can be used for measuring not only quantity of the S/W change process, but also quality [10]. Moreover, it is suitable for quantifying the degree of difficulty that the developer feels [12]. However, the previously developed method only reflects the size and the distribution degree of S/W changes. It cannot reflect complexity of a code pattern and a module structure.

To reflect the factors, common unit tasks between operators in the user interface and developers in the S/W inspection process were investigated [13]. Because the research field of user interface has already tried to quantify the complexity of information type. The methods for quantifying the complexity of the screen structure design and transition patterns were used for developing a weighting variable, which reflects the uncovered complexity types. In addition, the weighting variable was assigned to unit change.

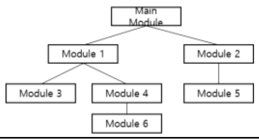
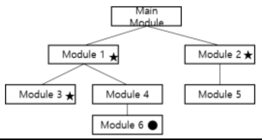
$$\text{Weighting variable of unit change } (c) = \alpha + \beta$$

Where, α is the entropy of the module structure where changes take place, and β is the entropy of diversity of the code pattern required for installation of new functions. The sum of two kinds of entropy is proposed as a weighting variable for the unit change. Variables are summarized with some examples in Table 2. The improved model for quantifying developer's effort is suggested as following equation.

$$\text{The amount of } \textit{S/W change information} = -\sum c_i p_i \log_2 p_i$$

Where, c_i is a weighting variable of the i^{th} unit change.

Table 2. Development of the weighting variables

Example Cases	Variables
A module structure 	Entropy of module α
Code patterns for a new function (★, ★, ★, ●) One figure means an unit change	Entropy of code pattern β
Results of S/W Change 	Weighting variable $c = \alpha + \beta$

4. A Case Study

A case study was performed to prove the validity of the suggested model. It was applied to two assumed scenarios, one was selecting security functions that required the more careful management, and the other was checking the fact that installation of security functions requires more developer effort than existing safety functions. Furthermore, the degree of distinction was compared with the previous method.

Table 3. The results of the case study

Scenario	Features to be applied	Previous Method	Suggested Method	Results (Careful)
1	Security Function 1	3.58	15.59	●
	Security Function 2	2.58	7.75	
2	Security Functions	4.17	16.26	●
	Safety Functions	3.32	9.63	

Security functions that require the careful application can be selected well in advance by using the suggested method. In addition, it can be checked that the introduction of security functions may cause a greater managing effort than that of safety features. Although, the previous method and the proposed method have the similar result, the suggested method shows more noticeable difference, because the suggested methods reflects more factors related to complexity.

5. Summary and Conclusion

Having not considered the fact that security functions when the existing digital I&C systems were designed, verification of adverse effects of security controls is the most controversial issue between the safety and the security fields. Several researches found that the incorporation of S/W changes due to the installation of new functions could cause a highly scattered code modification. Considering that the main cause of software faults is human error, it is important to quantify and reduce effort required for management. Factors affecting the developer's effort have a limited relation with S/W itself in this study, and are summarized as the size, the distribution degree, the module structure, and the code pattern. The amount of developer's effort can be quantified as the amount of S/W change information by applying the information theory. However, the method only reflects the size and the distribution degree of S/W changes. It cannot reflect complexity of the code pattern and the module structure. To overcome this limitation, common unit tasks between operators in the user interface and developers in the S/W inspection process were investigated. In addition, a weighting variable, which reflects the uncovered complexity type was developed and assigned to each unit change. A case study was performed to prove the validity of the suggested model. The suggested method can help estimate how further the system complexity can be and how much more difficulty can be added to developers when cyber security controls are applied. It is also expected that the suggested method can be applied to select security controls which need careful attention in advance. By measuring the complexity and difficulty, the suggested method can help managers establish the specific process for the safe application. However, there are some limitations in this work to estimate developer's effort, because the method for obtaining the weighting variable (c) needs to be elaborated. Also, verification and validation of the suggested method need to be improved.

REFERENCES

[1] J. Song, J. Lee, C. Lee, K. Kwon, and D. Lee, "A Cyber Security Risk Assessment for the Design of I&C Systems in Nuclear Power Plants," Nucl. Eng. …, vol. 44, no. 8, pp. 919–928, 2012.
[2] C. Baylon, R. Brunt, and D. Livingstone, "Cyber Security at Civil Nuclear Facilities Understanding the Risks," Chatham House, p. 53, 2015.
[3] J. G. Song, J. W. Lee, G. Y. Park, K. C. Kwon, D. Y. Lee, and C. K. Lee, "An analysis of technical security control requirements for digital I&C systems in nuclear power plants," Nucl. Eng. Technol., vol. 45, no. 5, pp. 637–652, 2013.
[4] C. Lee, H. Bin Yim, and P. H. Seong, "Development of a quantitative method for evaluating the efficacy of cyber security controls in NPPs based on intrusion tolerant concept," Ann. Nucl. Energy, vol. 112, pp. 646–654, 2018.
[5] J. Shin, H. Son, R. Khalil ur, and G. Heo, "Development of a cyber security risk model using Bayesian networks," Reliab. Eng. Syst. Saf., vol. 134, pp. 208–217, Feb. 2015.
[6] M. Khalaquzzaman, S. J. Lee, M. C. Kim, and W. Jung, "Estimation of reactor protection system software failure probability considering undetected faults," Nucl. Eng. Des., vol. 280, 2014.
[7] S. H. Lee, H. E. Kim, K. S. Son, S. M. Shin, S. J. Lee, and H. G. Kang, "Reliability modeling of safety-critical network communication in a digitalized nuclear power plant," Reliab. Eng. Syst. Saf., vol. 144, pp. 285–295, 2015.
[8] V. B. Singh, K. K. Chaturvedi, S. K. Khatri, and V. Kumar, "Bug prediction modeling using complexity of code changes," Int. J. Syst. Assur. Eng. Manag., vol. 6, no. 1, pp. 44–60, 2015.
[9] E. Erturk and E. Akcapinar, "A comparison of some soft computing methods for software fault prediction," Expert Syst. Appl., vol. 42, no. 4, pp. 1872–1879, 2015.
[10] A. E. Hassan, "Proceedings: Predicting Faults Using the Complexity of Code Changes," pp. 78–88, 2009.
[11] Y. Kamei et al., "A Large-Scale Empirical Study of Just-in-Time Quality Assurance," vol. 39, no. 6, pp. 757–773, 2013.
[12] H. G. Kang and P. H. Seong, "Information theoretic approach to man-machine interface complexity evaluation," IEEE Trans. Syst. Man, Cybern. Part ASystems Humans., vol. 31, no. 3, pp. 163–171, 2001.
[13] H. G. Kang and P. H. Seong, "An information theory-based approach for quantitative evaluation of user interface complexity," IEEE Trans. Nucl. Sci., vol. 45, no. 6 PART 3, pp. 3165–3174, 1998.