A New Consistency Validation Approach to Enhance the Quality of Functional Security Requirements for Secure Software

N. Mustafa[1], M. Kamalrudin[2]
[1]*Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.*
[2]*Innovative Software System and Service Group (IS[3]), Faculty of Information and Communication Technology, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, 76100 Durian Tunggal, Melaka, Malaysia.*
nuridawati @utem.edu.my

***Abstract*— Quality security requirements contribute to the success of secure software development. However, the process of eliciting security requirements is tedious and complex. It also requires requirements engineers to have security experience in the process of eliciting consistent security requirements from the clients-stakeholders. Most of the requirements engineers faced problems in eliciting consistent security compliance requirements from the clients-stakeholders as they misunderstood the real needs and the security term used. Thus, this resulted to inconsistent security requirements being elicited. The inconsistency leads to incorrect and insecure software systems being developed as well as to disruptions of schedule and increase of a project's expenditure. Motivated by these problems, this study is aimed to propose a new approach for consistency validation of functional security requirements. Here, security requirements specifications will be collected from software vendors to analyse the flow of functional security requirements process. Next, visual differencing will be integrated to cross-validate the consistency of the elicited functional security requirements with the best-practise template. Here, security requirements best-practice template pattern library will be designed and a new mathematical formulation that defines the consistency validation rules of security requirements will also be constructed. The formulation will be based on the security-related semi-formalised model, called SecEssential Use Case (SecEUC).This approach will then be realised with a proof of concept prototype tool and will be compared with the existing approaches, focusing on its ability to validate the inconsistency of the functional security requirements. Finally, this study is believed could provide a positive impact to the software industry by reducing the development cost as it allows the requirements engineers to validate the inconsistency that occurs in the elicited security compliance requirements at the early stage of the secure software development.**

***Index Terms*— Requirements Consistency Management; Security Requirements; Security Requirements Validation; Security Requirements Engineering; Secure Software.**

I. INTRODUCTION

Security requirement can be defined as a system specification of its required security, such as the specification towards types and levels of protection that necessary for the data, information, and application of the systems. Example of security requirements are authentication requirements, authorization requirements, intrusion detection requirements, and many others [1]. Security requirements are also divided into two parts: Functional and non-functional. However, capturing accurate functional security requirement is important to the development of secure software. It needs to be accurately defined because poor elicited functional security requirements could cause failure to the development and consume high cost [2]. Further, inaccurate functional security requirements could lead to the incorrect generation of non-functional security requirements. In addition, the process of eliciting security requirements is complex and requires requirements engineers to have security experience in the process of eliciting consistent security requirements from the clients-stakeholders.

Most of the requirements engineers faced problems in eliciting consistent security compliance requirements from the clients-stakeholders as they misunderstood the real needs and the security term used. At present, when capturing security requirements from clients, requirements engineers often use some form of natural language, written either by clients or themselves. These requirements are formed from the discussion and negotiation between both parties; clients and the requirements engineers. However, due to both ambiguities and complexities of natural language and the process of capture, these requirements often have inconsistencies which finally lead to the development of inaccurate secure software. As such, this research aims to examine the following research questions:

1) Why is the consistency of functional security requirements important in the development of secure software?

2) How to validate the consistency of functional security requirements?

This paper is organized as follows. In Section 2, we discussed the background and motivation by outlining the existing consistency management works in handling inconsistencies in requirements. Next, in Section 3, we proposed the new approach to validate the consistency of functional security requirements. Then, in Section 4 we explained the overall methodology of this research. Also, the research activities and progress up till now will we presented in Section 5. Finally, this paper ends with a Section 6 that explain the research implications.

II. Background and Motivations

Many approaches have been proposed to maintain consistency and check the inconsistency. M. Kamalrudin et. al. [3,4] have developed a tool supports, Marama AI that allows for inconsistency checking between textual requirements, abstract interactions that derive from the text and Essential Use Case models. The tool provides consistency checking and notification support allowing requirements engineers to modify any of the three forms of requirements in the tool. Yet, Marama AI does not support managing inconsistencies in security requirements.

C.Thiago et al. [5] presented an automatic analysis of requirements consistent with the method named as B Method. They provide a controlled language for requirements specification which is described by use cases scenarios and safety properties. A CRS allows an automatic generation of B specification, which can be checked for inconsistencies in "just one click" using an appropriate tool. However, this work is still in the initial phase and still some improvement such as describing the grammar rules and supporting more complex scenarios. Further, it also does not support the inconsistencies check of security requirements.

M.Alferez at el. [6] checks the consistency of the semantic relationships among the models between features and use scenarios to realize them. However, they also do not check the consistency of the security requirements.

I.Mirbel et al. [7] enhance the goal-based requirements consistency which implementing an Argumentation-based Approach. This approach is to support consistency checking in goal-based requirements engineering. The approach aims to detect implicit relationships between the requirements and checking the possible inconsistencies among them and also uses argumentation theory to formalize the requirement and their relationships and to detect the inconsistencies. Yet, they did not explore the consistency issue in security requirements.

S. Yahya et al [8] present approach that enhances the process of capturing and analyzing security requirements. They are using a tool called SecMereq. They used the developed essential interactions patterns and essential use case patterns. Their tool allows requirements engineers to automate the elicitation process for capturing security requirements. But still, they do not perform consistency checking on the elicited security requirements.

Security requirements engineering process with a generic system model core has been proposed as in [9]. Decke explains the system model core and demonstrates its extensibility using the example of vehicular systems. They explained two methods for formal inspection of the system model, which are how security engineer can be assisted by consistency checking of the system model, and how to verify the sum of generated security requirements to ascertain the correctness of the security concept. Even though the consistency checking is included in this model, the implementation is still tedious because no automated tool is provided. The implementation suggestion requires the REs to choose the checking on their own, depending on the type of the implementation of the methodology. However, their recommendation to use lambda functions in C++ 11 or Java 8 does not provide a guarantee that the result of consistency checking is achieved.

Houmb et al. [10] proposed a security requirements engineering methodology called SecReq, which is an extension of security requirements engineering by seamlessly integrating elicitation, traceability and analysis activities. This methodology combines three techniques: the Common Criteria (CC), the heuristic requirements editor HeRA, and the UMLsec. The integrated SecReq method supports early detection of security-related issues (HeRA). Their systematic refinement is guided by the CC, and it has the ability to trace security requirements into UML design models. A feedback loop helps to reuse the experience within SecReq and turns the approach into an iterative process for the secure system life-cycle. It is also in the presence of system evolution. However, it has several limitations: The consistency of the elicited security requirements during Step 1 is not being considered, and there is still no guarantee that these requirements will be correct and consistently represented in the solution design and then the implementation.

Similar to the previous work, El-Hadary and El-Kassas [11] also proposed a methodology for security requirement elicitation based on problem frames, which is to assist developers to elicit adequate security requirements during the requirement engineering process with the aid of previous security knowledge. This methodology adopted a security catalog based on the problem frames. It was constructed to help identify security requirements with the aid of previous security knowledge. Abuse frames were used to model threats, while security problem frames were used to model security requirements. They claimed their methodology could extract complete security requirements compared to other relevant methodologies. However, the results are still immature since the comparison was made with two security requirement elicitation methodologies only. Perhaps, the consistency level has not been proven in their paper since more empirical studies on large-scale software systems are needed in order to evaluate the methodology.

In summary, there are number of works done in checking the consistency of requirements. However, almost limited work done found in managing the consistency of security requirements. In addition, the existing consistency management approaches are still immature and have a tedious implementation.

III. Proposed Approach

Motivated by the research background and motivation, we proposed to develop an automated tool to elicit security requirements with a new best-practice template for guidance in writing consistent functional security requirement together with consistency management checking. We strongly believe that this approach will improve the quality of elicited security requirement for secure software development. The overall proposed approach is illustrated in Figure 1.

As overall, there are 10 key steps in our proposed approach that will be shown below:

1) Requirement Engineer (RE) elicits requirement from the client/stakeholder by using conventional method. After using the methods for elicitation, they get the software requirements. Then, the RE will key in the requirements in the text editor provided in the tool.

2) The tool checks and analyze the structures of elicited security requirement from the Security Library (SecLib).

3) SecLib suggested the new corrected security requirement together with the guidance in writing correct security requirement.
4) RE choose an option whether to apply the new changes or revert to the original input.
5) The elicited requirement is represented in SecEUC Model that derived from the Abstract Interaction pattern library and displayed in both user intention and system responsibility.
6) RE needs to choose the functional security template according to the client need/recommendation.
7) The Best practice template generated based on the selected template.
8) The consistency checking by cross-validate between elicited functional security requirements that is represented in SecEUC model and Best Practice Template using visual differencing to highlight potential inconsistencies and incompleteness in the SecEUC Model.
9) RE choose the option to use the suggested best practice template or can choose the functional security template again.
9.1 If choosing the suggested best practice template, the prototype design for visualizations is generated.
9.2 If not, RE can choose another security functional template. Repeat Step (6)
10) Exit the system.

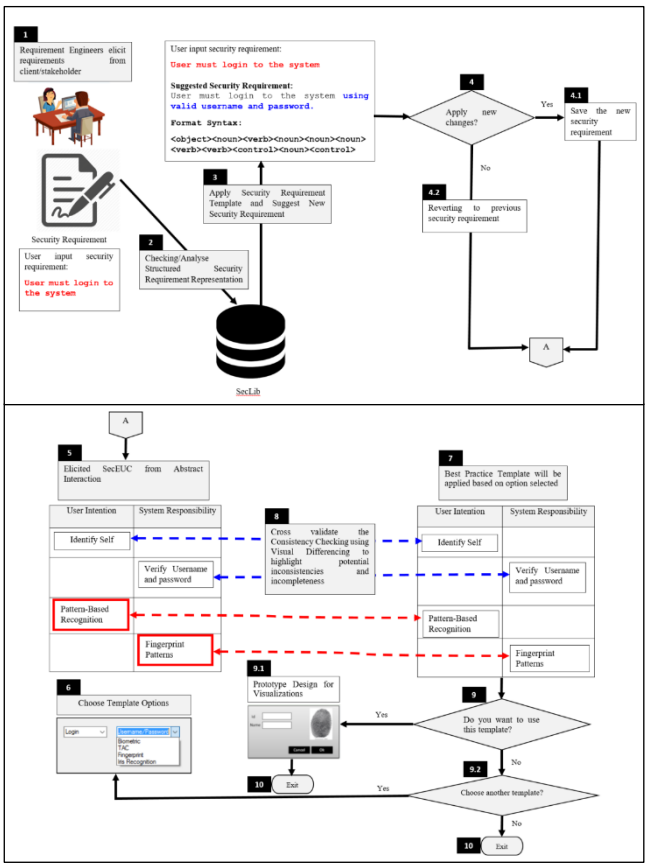

Figure 1: New Proposed Approach

IV. METHODOLOGY

Continuous from the proposed approach, this section outlines the works that we need to accomplish this research. The aim of this research is to design a new consistency validation approach using visual differencing to enhance the quality of functional security requirements for secure software. Figure 2 describes the components and flow of our research to achieve the objectives of the study. As for now, we are at the stage of finalizing the analysis of this research.

Figure 2: Research Flow

A. Analysis

A systematic literature review is conducted to find the gaps in security requirement consistency validation. It will start with reading and gathering relevant information on security requirements consistency management techniques and validation. Security requirements specification will be collected from selected software vendors. At the same time, the survey is disseminated to the requirement engineers to identify the current problems faced by them during the elicitation process, security standards used as the reference, elicitation and validation method, and the important properties considered while developing secure software.

B. Design and Development

At this phase, a new approach to validate the consistency of functional security requirements will be designed. The findings in the earlier phase will be reviewed, and the relevant parameters related to security requirements are collected. Then, a new best practice pattern library will be developed. Then, a mathematical formulation that defines the validation of the consistency checking between the elicited functional security requirements and the best practice template that previously designed. In addition, security requirements are represented using semi-formalised model: SecEUC. Visual differencing will then be integrated to cross-validate the consistency of the elicited functional security requirements. Then, a proof concept prototype tool will also be developed to realize the approach.

C. Testing and Evaluation

The new approach is compared with the existing approaches to validate consistency of functional security requirements. Here, the effectiveness, the usability and the performance of the new approach will be analyzed.

V. RESEARCH PROGRESS AND ACTIVITIES

Based on elaborated research methodology in the previous section, this research is in the stage of finalizing the analysis. We have conducted a systematic literature review to find the related literature that has been summarized in Section 2.0. At the same time, surveys have been conducted to the selected software vendors have to get the feedback on current an existing industry implementation on security requirement.

As overall, Figure 3 shows the entire activities for this research. To highlighted research progress so far that have conducted, the completed research activities are colored in grey. The next stage proceeds to Phase 2 to design and develop the new approach as an automated tool will be carried out.

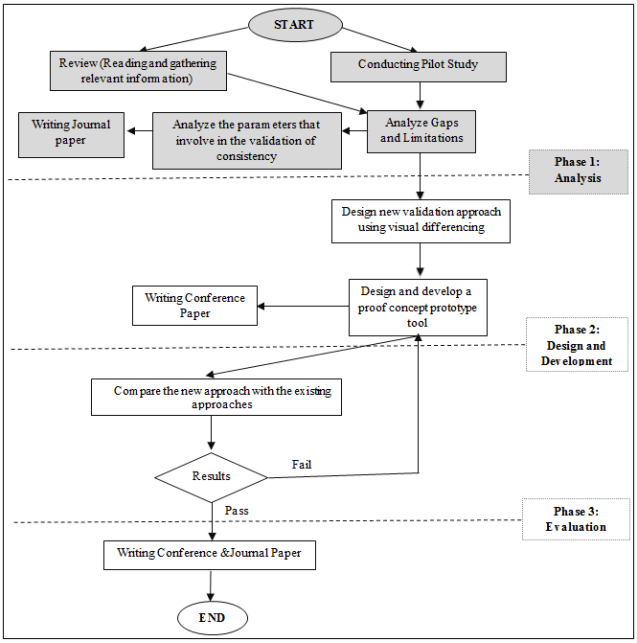

Figure 3: Research Activities

VI. CONCLUSION

In line with the National Key Research Area (NKRA) and National ICT Agenda, it is anticipated that this study could assist requirements engineers and software developers to develop secure software. This approach is believed could help the software community and developers to develop a quality secure software for any application of domains. It also has the potential to minimize time and maintenance cost of developing quality software as it helps to validate the errors at the very early stage of development life cycle. Further, this approach also could increase the confidence and sustainable use of software that being developed.

ACKNOWLEDGMENT

I would like to thank UTeM and MoE for the funding research: FRGS/ 1/ 2015/ ICT01/ FTMK/ 02/ F00291.

REFERENCES

[1] Firesmith, Donald G. “Analyzing and specifying reusable security requirements” in Software Engineering Institute, 2003, pp. 7-11.
[2] Schneider, Kurt, Eric Knauss, Siv Houmb, Shareeful Islam, and Jan Jürjens. “Enhancing Security Requirements Engineering By Organizational Learning” In Requirements Engineering, 2012, 17(1), pp. 35-56.
[3] M.Kamalrudin J. Hosking, John Grundy, "Improving Requirements Quality using Essential Use Case Interaction Patterns," in ICSE’11, Honolulu, Hawaii, USA, 2011, pp. 531-540.
[4] M. Kamalrudin, J.Grundy, J.Hosking,"MaramaAI: Tool Support for Capturing and Managing Consistency of Multilingual Requirements", in Proc. 27th (ASE 2012) Automated Software Engineering Essen, Germany , 2012, pp. 326-329.
[5] Thiago C. de Sousa, Jorge R. Almeida Jr, Sidney Viana, Judith Pavón, "Automatic Analysis of Requirements Consistency with the B Method" in ACM SIGSOFT Software Engineering, 2010. pp.1-4.
[6] Alférez, M., Lopez-Herrejon, R. E., Moreira, A., Amaral, V., & Egyed, A, "Supporting consistency checking between features and software product line use scenarios." in Top Productivity through Software Reuse. Springer Berlin Heidelberg, 2011, Vol. 6727, pp. 20-35.
[7] Isabelle Mirbel, Serena Villata. “Enhancing Goal-based Requirements Consistency: an Argumentation-based Approach”, Michael Fisher and Leon van der Torre and Mehdi Dastani and Guido Governatori. 13th International Workshop on Computational Logic in Multi-Agent Systems” in (CLIMA 2012), Springer, 2012, pp. 110-127.
[8] S. Yahya, M. Kamalrudin, S. Sidek, J. Grundy, “Capturing Security Requirements Using Essential Use Cases (EUCs)” In Requirements Engineering, Springer Berlin Heidelberg, 2014, pp. 16-30.
[9] R. Jindal, R. Malhotra, and A. Jain, “Automated classification of security requirements,” in 2016 International Conference on Advances in Computing, Communications and Informatics (ICACCI), 2016, pp. 2027–2033.
[10] S. H. Houmb, S. Islam, E. Knauss, J. Jürjens, and K. Schneider, “Eliciting security requirements and tracing them to design: an integration of Common Criteria, heuristics, and UMLsec,” Requir. Eng., vol. 15, no. 1, pp. 63–93, Mar. 2010.
[11] H. El-Hadary and S. El-Kassas, “Capturing security requirements for software systems,” J. Adv. Res., vol. 5, no. 4, pp. 463–472, Jul. 2014.