

An Exploratory Study on Secure Software Practices Among Software Practitioners in Malaysia

Shafinah Farvin Packeer Mohamed¹, Fauziah Baharom¹, Aziz Deraman², Jamaiah Yahya³, Haslina Mohd¹

¹*UUM CAS School of Computing, Universiti Utara Malaysia, 06010 Sintok, Kedah, Malaysia.*

²*Faculty of Science and Technology, Universiti Malaysia Terengganu, 21030 KualaTerengganu, Terengganu, Malaysia.*

³*Faculty of Information Science & Technology, The National University of Malaysia, Bangi, 43600 Selangor, Malaysia.*
shafinah@uum.ed.my

Abstract—Rapid growths of computers, mobile phones and Internet technology have created ways for irresponsible people to undertake computer crimes. Millions of users across the globe have fallen as victims to computer crimes, including Malaysia. It is due to current software environment which is more complex, distributed, keeps confidential data and easily exposed to malicious attacks. Consequently, secure software process is increasingly gaining much importance among software practitioners and researchers. However, even though its importance has been revealed, only few studies were conducted regarding its current practice in the software industry, especially in Malaysia. Thus, an exploratory study is conducted among software practitioners in Malaysia to study their experiences and practices on the secure software process in the real-world projects. This paper discusses the findings from the study, which involved 93 software practitioners. Structured questionnaire is utilized for data collection purpose whilst statistical methods such as frequency, mean, and cross tabulation are used for data analysis. Outcomes from this study reveal that software practitioners are becoming increasingly aware on the importance of secure software process, however, they lack of appropriate implementation of the practices.

Index Terms—Secure Software Practices; Exploratory Study; Software Practitioners; Malaysia.

I. INTRODUCTION

Computer crime is one of the problems that gains our concern in today's cyber world. It is growing very fast compared to other crimes and causes serious damage to the political, economic and social sectors [1]. With the advancement of computer, mobile phone and Internet usage, the software application environment becomes more complex and distributed. Majority of companies which ranges from small companies to large companies are relying on the Internet to run their businesses. Furthermore, with the existence of social media and online transactions, the life of humankind is highly connected to the Internet. Thus all confidential information and personal details are available online at any time. These information are exposed to malicious attack since they can be manipulated by irresponsible people if not protected in a proper manner. There are many kinds of attacks, such as Website crashes, password cracking and security breaches. Viruses could be laying inactive in smartphones or computers waiting to copy banking passwords and social media accounts when connected to public Wifi, or masquerade as a trustworthy entity

in an electronic communication. These circumstances have influenced the level of system performance, quality and integrity of a software application. Consequently, in recent years, there are many serious computer crimes have been reported.

According to CBS Corporation [2], 1.5 million cyber attacks are found annually, which means there are over 4,000 cyber attacks every day, 170 attacks every hour, or nearly three attacks in every minute. They added that in 2014, hackers had stolen personal information from 47% of American adults through data breaches at large companies. The same situation happens in Malaysia. 6167 cases were reported in 2010 which caused RM 63 million loss. Among the frequent crime reported are in Internet banking, VoIP (Voice over Internet Protocol) and e-Commerce. Mostly it involves identity theft and fraud [3]. On top of that, Malaysia has been listed in the Sophos Security Threat Report 2013 as the sixth most vulnerable country in the world for cyber crime, which involves the malware attacks in computer or smart phone [4]. More recently, based on the research performed by CyberSecurity Malaysia, there are more than 30 victims of cyber criminals daily in Malaysia. To make it worst, this number does not include unreported cases and unnoticed victims [5].

Consequently, the customers are becoming more concerned about the security of software produced to them. Since it is estimated that 80% of all breaches are application-related, the traditional perimeter defenses like firewalls, intrusion detection and anti-virus systems are unable to protect software. Thus, most researchers believe that security activities should be considered from the beginning of the software development lifecycle and continuous in all phases [6,7]. McGraw defines secure software practices as “about building secure software: designing software to be secure, making sure that software is secure, and educating software developers, architects, and users about how to build secure things [8]

Despite the importance of incorporating secure software practices during software development, only few studies related to the current industrial practice of secure software practices have been conducted and just focus on the requirement engineering phase. Furthermore, several studies focus on showing the criticality of considering security measures, rather than investigating the current practice in industry. On top of that, to our best knowledge, the current practices of software practitioners in Malaysia regarding secure

software practices is still scarce. Even though there are many studies conducted on the current practices of software development practices in Malaysia, the focus is more on the conventional software practices [9,10]. However it is essential to investigate the current practices of secure software since it has become as a determinant factor for producing high quality software. Based on the abovementioned limitations, an exploratory study is conducted to explore the experiences and practices of software practitioners on the secure software practices.

This paper discusses findings from the study. First, the related studies are described, followed by research implementation, continued with results, discussion and ended with the conclusions.

II. STATE OF THE ART

There are several studies which focuses on secure software practices, nevertheless, their focus is more on showing the criticality of considering security measures in developing software, for instance Whitehat Security investigated the number of vulnerabilities in small, medium and large organizations [11], while National Cyber Security Alliance [12] surveyed the security trainings provided in software companies, the awareness of security initiatives and the security problems they are facing. In addition, Errata Security [13] found out that 57% of the respondents used secure development methods, while 43% do not consider secure development methods at all.

Nonetheless, these studies do not reveal the practices adopted by software practitioners in eliciting, documenting and analyzing security requirements in the real environment [14]. Therefore, Elahi et al. [14] and Wilander and Gustavsson [15] investigated the software practitioners' practices in requirement engineering which focus on security. Elahi et al. [14] conducted a survey among software practitioners and found that the security requirements are not explicitly elicited and documented in the early stages of the development, but they are considered during the implementation phase. On the other hand, Wilander and Gustavsson [15] analyzed the requirement documents of 11 Swedish software projects. They concluded that the security requirements were inconsistent and inadequately identified among the projects.

In Malaysia, there are many studies have been conducted in the software development area which are intended for investigating the current practices of software development in the Malaysian software industry, for instance [9,10,16]. However, these existing studies focused on the conventional software development practices, rather than secure software practices.

Based on the existing studies discussed, empirical studies on the secure software practices is lacking in Malaysia, since their focus is more on the conventional software practices. Furthermore, even though there exist studies which focus on the secure software practices in Western countries, nevertheless, they concern more on showing the criticality of considering security measures in developing software, without practices adopted by software practitioners during software development. Also the studies focus on the requirement engineering only. Consequently, in this study, the secure

software practices being implemented by the Malaysian software practitioners have been investigated, which considers the requirement engineering, software design, coding, testing, security requirement and risk management. Section 3 explains about the implementation of the study.

III. IMPLEMENTATION OF THE STUDY

The study was conducted through four (4) phases, as described briefly in Table 1.

Table 1
Activities and Descriptions

Activities	Descriptions
Instrument design	<ul style="list-style-type: none"> instrument was designed by referring previous works such as [9] and [14]. consists of single and multiple responses, yes/no questions.
Pilot study	<ul style="list-style-type: none"> involved 32 respondents (system analysts and programmers with at least 5 years' experience). they agreed that the questions covers the domain of the secure software practices, however, there are some suggestions: simplify the questions to be more readable and understandable, reduce and reorganize the questions.
Data collection	<ul style="list-style-type: none"> data was collected from samples which were identified from Kuala Lumpur, Selangor, Penang and Kedah, since most software development companies are located there in Malaysia [17]. 93 respondents took part in this study. the questionnaire was distributed through online survey, email or mail.
Data analysis	<ul style="list-style-type: none"> the collected data was analyzed using descriptive statistical analysis: the frequencies, mean and cross tabulation by using the SPSS software.

IV. RESULTS

This section discusses the results on the demographic data and the software practitioners' experience and practices regarding secure software.

A. Demographic Data

The respondents are asked about their position and experience. Cross tabulation analysis is used to classify them, as depicted in Table 2. Most of the respondents are programmers (41.9%). Out of the 93 respondents, only 16.1% have experience more than 10 years, while majority have 1 to 5 years of experience (50.5%).

Table 2
Respondents' Experience

Positions	<1 year	1-5 years	6-10 years	11-20 years	Total
Project Managers	1 (1.1%)	3 (3.2%)	2 (2.2%)	4 (4.3%)	10 (10.8%)
Programmers	7 (7.5%)	26 (28%)	3 (3.2%)	3 (3.2%)	39 (41.9%)
Quality Assurance /Testers	0 (0%)	5 (5.4%)	0 (0%)	1 (1.1%)	6 (6.5%)
System Analysts	2 (2.2%)	10 (10.8 %)	11 (11.8 %)	3 (3.2%)	26 (28%)
Security Advisors	1 (1.1%)	0 (0 %)	0 (0 %)	0 (0 %)	1 (1.1%)
Team Leaders	1 (1.1%)	3 (3.2%)	3 (3.2%)	4 (4.3%)	11 (11.8%)
Total	12 (13%)	47 (50.5%)	19 (20.4%)	15 (16.1%)	93 (100%)

The respondents work in many sectors, such as software development, education/training and manufacturing, as presented in Table 3. Most of the respondents are from private sectors (76%), and 47% from software development organizations.

Table 3
Classification of Organization Sector

Sectors	Organization Types		Total
	Private	Government	
Software Development	44 (47%)	0 (0%)	44 (47%)
Education/Training	10 (11%)	11 (12%)	21 (23%)
Service and Public Administration	5 (5.4%)	4 (4.4%)	9 (9.7%)
Manufacturing	4 (4.3%)	0 (0%)	4 (4.3%)
Consultation	3 (3.2%)	1 (1%)	4 (4.3%)
Telecommunication	4 (4.3%)	0 (0%)	4 (4.3%)
Health & Social Work	0 (0%)	5 (5.4%)	5 (5.4%)
Banking/Financial/Insurance	1 (1%)	1 (1%)	2 (2.2%)
Total	71 (76%)	22 (24%)	93 (100%)

B. Software Practitioners' Experience & Practices in Secure Software

Firstly, the respondents were asked whether they agree that secure software practices can influence the quality of produced software. 96% agreed, while only 4% disagreed. Secondly, the respondents were asked about the security incidents that they faced. It is found that respondents faced many security incidents, as depicted in Figure 1. The most common security incidents faced by them are password cracking (45%), followed by malicious code (39%) and SQL injection (35%). Only small percentage (9%) of them never face any security incidents.

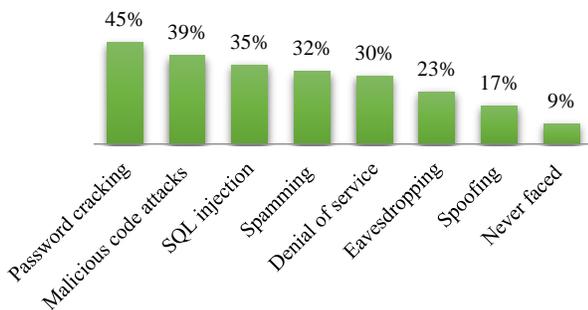


Figure 1: The security incidents faced

Thirdly, the respondents were asked whether they elicit and document security requirements explicitly from early stage. 21.5% of the respondents discuss about the security requirement from early stage. Unfortunately, the requirements are not documented. However, 24% of them are aware of this, whereby they gather and document the security requirements explicitly during requirement gathering. Meanwhile, 32% of the respondents only deal with security issues during the implementation phase or after the system being developed. On top of that, 22.5% do not even deal with the security requirements, as presented in Table 4.

Table 4
Eliciting Security Requirements Explicitly

Answers	Frequency/Percentage
Security issues are only dealt during the implementation phase or after the system being developed	30 (32%)
Security requirements are gathered and documented in the early stages of the projects before the development starts	22 (24%)
Do not deal with security requirements	21 (22.5%)
Security requirements are discussed from early stages butnot documented	20 (21.5%)
Total	93 (100%)

Table 5 depicts the analysis result regarding the notations used to represent security requirements. Unfortunately, the analysis result found that majority of them (76%) do not document the security requirements, while 4% do not use any specific notation to represent the security requirements.

Table 5
Notations used

Notations	Frequency	Percentages
Do not document	71	76%
Abuse case	10	11%
Misuse case	9	10%
Attack tree	7	8%
No specific notation	4	4%
Misuser stories	2	2%

Additionally, the respondents were asked about how they prevent from introducing common attacks that occurred previously. Surprisingly, majority of them did not consider the attacks that have happened in the past (41%). However, fortunately the remaining respondents referred to the document which records the security attacks that have occurred previously (37%), while 35% of them consulted with the security experts. Table 6 demonstrates the analysis result.

Table 6
Prevention techniques from common attacks

Prevention techniques	Frequency
Do not consider attacks that have happened in the past	38 (41%)
Refer to document which records the security attacks that have occurred	34 (37%)
Consult with security experts to prevent common attacks	33 (35%)
Look for well-known common security attacks in attack and vulnerability databases	32 (34%)

Moreover, the respondents were asked about the percentage of security trainings provided for the staff. Cross tabulation analysis was used in order to classify the respondents based on their position and amount of security training provided for them. Most of the respondents (38.7%) are provided with 25% or less security trainings in a year. Quite a big percentage is not provided with any security trainings (19.4%). Only 24.7% are provided with security trainings within 25 to 50 percent in a year. The result of analysis is depicted in Table 7. Meanwhile, the trainings are provided mostly for the programmers and system analysts, 41.9% and 28% respectively.

Table 7
Percentages of security training provided

Positions	Percentages of Trainings per year					Total
	None	<=25%	25% - 50%	50% - 75%	> 75%	
Project Manager	1.1%	3.2%	3.2%	2.2%	1.1%	10.8%
Programmer	7.5%	15.1%	11.8%	2.2%	5.4%	41.9%
Quality Assurance/Tester	2.2%	3.2%	1.1%	0%	0%	6.5%
System Analyst	7.5%	10.8%	5.4%	2.2%	2.2%	28%
Security Advisor	0%	0%	1.1%	0%	0%	1.1%
Team Leader	1.1%	6.5%	2.2%	0%	2.2%	11.8%
Total	19.4%	38.7%	24.7%	6.5%	10.8%	100%

Additionally, the respondents were asked about the software practices that need to be implemented in order to produce high quality software, concerning on secure software practices. The practices are categorized into requirement engineering, software design, coding, testing, security management and risk management. Mean value for each practice is used in the analysis, as it represents the most selected answers by

respondents. The 7-point semantic differential scale is used for these questions, which ranged from Extremely not Important to Extremely Important. The scale was then mapped to equal intervals, as depicted in Table 8.

Table 8
Interval values

Degree of importance (DI)	Interval value
Extremely Not Important (ENI)	1 – 1.86
Not important (NI)	1.87 – 2.73
Less Important (LI)	2.74 – 3.60
Moderately Important (MI)	3.61 – 4.47
Important (I)	4.48 - 5.34
Very Important (VI)	5.35 - 6.21
Extremely Important (EI)	6.22 - 7

The mean values obtained for the important secure software practices subsequently, as in Table 9.

Table 9
Secure software practices

Phases	Practices	Mean	DDI
Req. Engineering	Updating security requirements iteratively, taking place as changes occur	5.51	
	Documenting and maintaining a set of well-defined security requirements to prevent from introducing common attacks that occurred previously	55.47	
	Obtaining security requirements explicitly	5.41	(VI)
	Considering attackers' perspective while eliciting security requirements	5.39	
	Documenting security requirements in a particular notation (e.g.: misuse case)	5.33	
Software Design	Referring the latest lists of common attack patterns, vulnerabilities and threats in order to keep up-to-date with current trends	55.52	(VI)
	Documenting security requirements in a particular notation (e.g.: misuse case, attack tree)	5.33	
	Modeling the possible threats	5.23	(I)
	Performing an external (by someone outside the design team)	5.13	
Coding	Referring to the secure coding guidelines	5.40	
	Coding countermeasures for the identified threats	5.35	(VI)
	Preparing documentation for installing and operating the application securely	5.32	
	Implementing pair programming to reduce vulnerability (with continuous review)	4.81	(I)
	Comparing outcome from automated and manual code review	4.72	
Testing	Performing integration tests focusing on the threats and vulnerabilities	5.31	(VI)
	Creating unit tests by focusing on the identified threats and vulnerabilities	5.26	
	Performing risk analysis again at the end of the phase to ensure all risks are mitigated and to consider remaining risks	5.23	(I)
	Performing penetration test (simulate attack from malicious outsiders)	5.23	
	Performing fuzz testing (use random data as input for tests)	5.17	
Security Management	Sharing the produced artifacts among team members	5.63	
	Producing and revising security policy regularly	5.59	
	Ensuring that all members of the project team are aware of and involved with security engineering activities	5.43	(VI)
	Planning and documenting security plan	5.42	
	Defining the security roles and responsibilities up-front	5.34	
Risk Management	Performing risk analysis iteratively throughout the software development to identify the possible threats, vulnerabilities and impacts of the application	55.35	(VI)
	Planning mitigation strategy to countermeasure the identified threats, vulnerabilities and impacts	55.32	
	Ensuring the newly identified risks are reported and mitigated as soon as possible	55.24	(I)
	Monitoring the identified threats, vulnerabilities and impacts throughout the development	55.18	

V. DISCUSSIONS

The software practitioners are aware with the importance of secure software practices. However, their experience in implementing the proper practices still can be considered as low. Although the respondents faced many security incidents such as password cracking and SQL injection (Refer Figure 1), most of them did not consider security requirements from the early stage of software development, but only dealt with security requirements during the implementation phase or after the system being developed (Refer Table 4). This result is aligned with the outcomes of [14] whereby most of their

respondents left the security requirements undocumented and only consider them implicitly. However, incorporating security in later stages of software development will increase the risks of introducing security vulnerabilities into software. On the other hand, the outcome of Errata Security survey [13] found that half of the respondents gave high concern on security during software development. There exist among the respondents who discuss the security requirement from early stages, yet, they do not document them. Fortunately, some of the respondents gather and document the security requirements from early stage. This explains that there are among the

respondents who are aware about the importance of security activities during software development.

In addition, representing the security requirements in particular notation is vital in order to get good understanding about the requirement of proposed system. Yet, majority of the respondents do not even document the security requirements (Refer Table 5). In contrast, Elahi et al. [14] indicated that their respondents used modelling notations widely. By neglecting this important software practice, the software practitioners might ignore relevant threats that might surface in the proposed system. Fortunately, there exist among them who use abuse case, misuse case, attack tree and misuser stories.

Moreover, to efficiently elicit security requirements, software practitioners should refer to references which provide guidelines on handling security issues. Majority of the respondents referred to the documents which record the previous attacks occurred, which is aligned with the findings from the study of Elahi et al. [14]. They also consulted security experts and looked for the common attacks from the attack and vulnerability database. However, almost half of the respondents did not make any security references while eliciting security requirements (Refer Table 6). This might cause the software practitioners to be outdated from the current threats, attacks and countermeasure available in the industry, as well as repeating the same threats which occurred in previous projects.

Besides, trainings have been accepted as one of the major ways to create awareness on the security issues among the software practitioners. However, less security trainings are provided for the respondents, whereby majority of them attended security trainings only for 25% or less (Refer Table 6). On top of that, there exist among them who did not receive any security trainings. This result is contradicted with the findings in the study of Elahi et al. [14]. Without attending proper trainings may lead to improper implementation of secure software practices, since proper guideline on its actual implementation is not received.

As discussed earlier, many researchers have come out with software lifecycle models which support security activities throughout the lifecycle. Among the most prominent and used in industry to date are CLASP, Microsoft SDL and McGraw [18,19]. These models have been referred in order to establish the secure software practices, as well as security standards which are ISO/IEC 27001[20] and ISO/IEC 21827[21]. Outcomes from the study show that mostly these practices obtained high consideration among the respondents, whereby the mean values are in the range of *Important to Very Important*. This shows that they are important practices in producing high quality software. In addition, it indicates that the practices and perceptions of the respondents are aligned with the literature. They are discussed further subsequently.

A. Requirement engineering for secure software process

Eliciting security requirements explicitly, accurately and consistently has been one of the most fundamental activities for engineering secured software [8,15,22]. However, security requirements are mostly dealt when the system has been designed or put in operation. Only low percentage of respondents (9%) admitted that they document security requirements explicitly in study by Elahi et al. [14], while majority of them (59%) considered it implicitly. On top of that,

31% do not elicit security requirements at all. Furthermore, most of the researchers [6,7,23,24] stress that security requirements should be established from an attacker's perspective and updated iteratively as soon as changes occur. In addition, the security requirements must be documented and maintained for reuse purpose [7]. By doing so, it will help developers to improve the software security as well as learn from past mistakes [25]. Majority of the respondents in this study expressed that all of these practices as important towards producing secured software.

B. Software design for secure software process

Designing security is similarly important as eliciting security requirements explicitly [22]. CLASP [24] emphasizes on auditing the security requirements during design phase. This is to ensure its completeness. Furthermore, during this phase, the possible impacts, vulnerable and threats must be identified, classified, rated and documented [7,21,23,24]. This activity will be more efficient by performing external review [7,23] and referring to the latest list of common attack from online database [26].

C. Coding for secure software process

During this phase, the secure coding guideline should be referred [7,23,24]. There are websites which gives this guideline such as Software Engineering Institute (SEI). The most important part in this phase is coding the countermeasure for the identified risks- threats, vulnerabilities and impacts [7,23,24,27,28]. Besides, these codes must be reviewed with automated tools as well as manual review and both results should be compared [7,29]. In addition, pair programming is useful to reduce vulnerability- by having continuous review [28]. Besides producing security emphasized coding, Microsoft [23] prepares documentation for user and help manuals, administrators manual and developer documentation, as well as user configuration tools. Additionally, CLASP [24] insists of producing document for installing and operating the application securely.

D. Testing for secure software process

Testing for secure software process is slightly different from traditional testing as it emphasizes what an application should not do rather than what it should do [26]. Thus, testing for producing secured software must include testing the security functionality besides the standard functional testing. They are the fuzz test and penetration test [7,23]. Traditional tests such as unit tests and integration tests are performed as well, but focused more on the threats and vulnerabilities [26]. Consequently, the test cases are created by focusing on the identified mitigation strategies [7,23,24]. Additionally, to ensure all risks are mitigated and to consider other residual risks, McGraw [7] includes analyzing the risks again at the end of testing phase.

E. Security management

In managing the security, the usage of security policy is very important to ensure that appropriate controls are put in place [30,31]. Besides, it should be reviewed and revised regularly as well as ensuring that it is being properly followed by the workers [31]. Additionally, security plan is another means of

ensuring good security management [23,30]. On top of everything, as human is important in conducting the security activities, thus their awareness on security [30,32] and involvement in security engineering is vital[21]. CLASP [24] defines security roles upfront for the team members. Furthermore, it also emphasizes on providing separate team for security engineering, similar to [23].

F. Risk management for secure software process

Risk management is the main activity in secure software practices [33]. Basically all of the traditional risk management activities exist in this approach: risk identification, risk analysis, risk planning and risk monitoring [34]. However, their concern is more on the threat, vulnerabilities and impacts [7,21,24,27]. These activities are implemented iteratively throughout the software development and ensuring the newly identified risks are reported and mitigated as soon as possible [7,23,24,27].

VI. CONCLUSIONS

This study has discussed the software practitioners' experiences and practices with the secure software process. It is found that software practitioners in Malaysia are increasingly becoming aware on the importance of the secure software practices. However, there still exist among them who did not consider security practices during software development, whereby they only consider security requirements implicitly. On top of that, majority of them left the security requirements undocumented, without proper notations. Also, they did not refer to references which provide guideline on handling security issues. This might lead them to be unaware with the current threats available in the industry. Besides, they might repeat the same threats which they faced in previous projects. These scenario highlight that the software practitioners are lack of proper implementation of secure software practices. This might possibly because less security trainings are provided to them. 19.4% of them did not even attend any security trainings, which can cause them to implement security activities improperly due to inadequate knowledge and awareness. This paper has given some insights on the implementation of secure software practices among Malaysian software practitioners. For our next step, the important secure software practices that influence the quality of software will be included in the proposed software process certification model. Interested readers may refer to [35, 36] which discuss the background of the research.

ACKNOWLEDGEMENT

The authors would like to sincerely thank the participants of the exploratory study and special thanks go to the Ministry of Higher Education for supporting this research through the FRGS grant (S/O Code: 11889).

REFERENCES

[1] Hong, L., H., Bin, L., and Taylor, M. "A Comparative Analysis of Cybercrimes and Governmental Law Enforcement in China and the United States. *Asian journal of criminology*. Vol. 5(2), pp. 123-135, 2010.

[2] CBS Corporation. 2015. These Cybercrime Statistics Will Make You Think Twice About Your Password: Where's the CSI Cyber team when you need them?. Retrieved from <http://www.cbs.com/shows/csi-cyber/news/1003888/these-cybercrime-statistics-will-make-you-think-twice-about-your-password-where-s-the-csi-cyber-team-when-you-need-them/>

[3] Lee, H. B. 2011, July 26. RM 63 juta rugi angkara jenayah siber. Utusan Malaysia. Retrieved from http://www.utusan.com.my/utusan/info.asp?y=2011&dt=0726&pub=Utusan_Malaysia&sec=Jenayah&pg=je_01.htm

[4] Bernama 2013, May 6. Malaysia sixth most vulnerable to cybercrime. The Star. Retrieved from <http://www.thestar.com.my/News/Nation/2013/05/16/Malaysia-sixth-most-vulnerable-to-cyber-crime/>

[5] Cheng, N. 2015, October 26. More than 30 Malaysians fall prey to cybercrime daily. The Star Online. Retrieved from <http://www.thestar.com.my/news/nation/2015/10/26/cybercrime-30-msians-daily/>

[6] Mead, N. R. 2010. Security requirement engineering. BSI Articles, SEI Institute.

[7] McGraw, G. 2006. Building security in. Boston: Pearson Education

[8] McGraw, G. 2004. Software security. *Security & Privacy, IEEE*, 2(2), 80-83. doi: 10.1109/MSECP.2004.1281254

[9] Fauziah Baharom, Aziz Deraman and Abdul Razak Hamdan 2005. A survey on the current practices of software development process in Malaysia. *Journal of ICT*. pp. 57-76.

[10] Yazrina Yahya, Maryati Mohd Yusof, Mohammed Yusof and Nazlia Omar. The use of Information System development methodology.

[11] Whitehat Security 2013. Website security statistics report, WhiteHat Security, California.

[12] National Cyber Security Alliance 2010. National small business study.

[13] Geer, D. "Are companies actually using secure development life cycles?". *Comp.* vol. 43(6), pp.12-16, 2010.

[14] Elahi, G., Yu, E. and Tong, L. "Security requirements engineering in the wild: a survey of common practices. *IEEE Ann. Comp.Soft. and App. Conf.* pp. 314-319, 2011.

[15] Wilander, J. and Gustavsson, J. 2005. Security requirements—A field study of current practice. *Symp. on Req. Eng. for IS*.

[16] Amjed Tahir, Rodina Ahmad and Zarinah Mohd Kasirun. 2010. An empirical study on the use of standards and procedures in software development projects. *Int. Conf.on Soft.Tec. & Eng.*

[17] Ani Liza Asnawi, Gravell, A. M. and Wills, G. B. 2012. Factor analysis: Investigating important aspects for agile adoption in Malaysia. *AGILE India*. pp. 60-63.

[18] De Win, B., Scandariato, R., Buyens, K., Gregoire, J., and Joosen, W. 2009. On the secure software development process: CLASP, SDL and Touchpoints compared. *Information and Software Technology*. Vol. 51(7): pp. 1152-1171, 2009.

[19] McGraw, G. 2011. Technology transfer: A software security marketplace case study. *Software, IEEE*. Vol. 28(5), pp. 9-11, 2011.

[20] ISO 2015. ISO Standards. Retrieved from <https://www.iso.org>

[21] Davis, N. 2013. Secure software development lifecycle process. Retrieved from <https://buildsecurityin.us-cert.gov/articles/knowledge/sdlc-process/secure-software-development-life-cycle-processes>

[22] Karpati, P., Sindre, G., and Opdahl, A. L. 2011. Characterising and analysing security requirements modelling initiatives. *Sixth International Conference on Availability, Reliability and Security*. 710-715.

[23] Microsoft. 2012. Microsoft Security Development Lifecycle SDL Process Guidance Version 5.2. Retrieved from <http://www.microsoft.com/en-my/download/confirmation.aspx?id=29884>

[24] OWASP. 2006. CLASP best practices. Retrieved from https://www.owasp.org/index.php/Category:CLASP_Best_Practice

[25] Rios, E. et al. 2009. A qualitative evaluation of model-based security activities for software development. Proceedings of European Workshop on Security in Model Driven Architecture, 14-21. Retrieved from <http://www.utwente.nl/ctit/publications/workshopproceedings/2009/wp0906.pdf#page=14>

[26] Julia, H. A., Barnum, S., Ellison, R. J., McGraw, G., and Mead, N. R. 2008. Software security engineering. Boston: Addison-Wesley.

[27] Evans, R., Tsohou, A., Tryfonas, T., and Morgan, T. 2010. Engineering secure systems with ISO 26702 and 27001. *5th International Conference on System of Systems Engineering (SoSE)*. 1-6.

- [28] Ashbaugh, D. A. 2009. *Security software development assessing and managing security risks*. Boca Raton: CRC Press.
- [29] Merkow, S. M. and Raghavan, L. 2010. *Secure and resilient software development*. Boca Raton: Auerbach Publications.
- [30] Ai, C. Y., Md Mahbubur Rahim, and Leon, M. 2007. Understanding factors affecting success of information security risk assessment: the case of an Australian higher educational institution. *Proceedings of PACIS. Paper 74*. Retrieved from <http://aisel.aisnet.org/pacis2007/74>
- [31] Syed Irfan Nabi, Abdulrahman A. Mirza, and Khaled Alghathbar 2010. Information assurance in Saudi organizations- an empirical study. In Tai-Hoon, K., Wai-Chi, F., Muhammad Khurram Khan, Arnett, K. P., Heau-jo, K., & Slezak, D., *Security technology, disaster recovery and business continuity*. Berlin Heidelberg: Springer Berlin Heidelberg
- [32] Siponen, M., Pahlila, S., and Mahmood, M. "Compliance with information security policies: an empirical investigation". *Computer*. Vol. 43(2): pp. 64–71, 2010.
- [33] Olsson, R. 2006. *Managing project uncertainty by using an enhanced risk management process*. Sweden: Malardalen University Press.
- [34] Sommerville, I. 2007. *Software Engineering 8th Ed*. Harlow: Pearson Education Limited.
- [35] Fauziah Baharom, Jamaiah Yahya, Aziz Deraman, and Abdul Razak Hamdan 2011. SPQF: software process quality factor for software process assessment and certification, *International Conference on Electrical Engineering and Informatics*.
- [36] Shafinah Farvin Packeer Mohamed, Fauziah Baharom and Aziz Deraman. "ESPAC Model: Extended Software Process Assessment and Certification Model". *ARPN Journal of Engineering and Applied Sciences*. Vol. 10(3), pp. 1364-1373, 2015.