An improved life cycle for building secure software

Huda Abdulaali abdulbaqi(1), Yasmin Makki Mohialden(2), Sundos Abdulameer Alazawi(3), Adel Muhsin Elewe(4)

1,2,3Department of Computer Sciences, College of Sciences, Mustansiriyah University, Iraq,
4Department of Project & Construction, Mustansiriyah University, Iraq

(1)hudaabdulaali.it@gmail.com, (2)ymmiraq2009@uomustansiriyah.edu.iq,
(3)ss.aa.cs@uomustansiriyah.edu.iq, (4)adelmuhsin@uomustansiriyah.edu.iq

Abstract. Computers advanced very fast technologically in recent times. This progress leads to high demand for more secure systems because sophisticated security problems have arisen. Since modern systems involve several areas of expertise, it is vital for the security engineer to follow an efficient and optimum approach and to take into consideration the mathematical and physical properties of the system. In the present study, the focus was placed on specific aspects of secure software building; the primary objective is to provide efficient and secure software in terms of resources and time. The designer should keep, as well, productivity, security and cost aspects in balance. Safe system requirements are continuously changing. Therefore, the designers should be aware of these changes. At the start of this research, the researchers distributed a survey about the security-engineering principles related to the life cycle of software engineering life and secure web systems. From the data acquired, the proposed method was developed and presented as a secure e-mail system built according to the engineering principles of software. The AHP (Analytic Hierarchy Process) technique correlated with composition operator have been used for finding the optimal concerns of software requirements and keep minimum maintenance and cost effect.

Keywords: secure e-mails, BPR, AHP, security engineering, software, security system

1. Background

Under this section, the definitions and fundamental elements of the two fields required to develop secure software these two fields are software engineering and the principles of security

1.1 Software Engineering

The term refers to the use of sensible engineering principles to obtain reliable and efficient performance [1][2]. The software development process involves three phases, and they are as follows;

a) Definition Phase - In this phase, interviews, questions and observations of users that need software are conducted to gather information regarding the fundamental software analysis elements (problems and needs)[3][4]. This analysis involves system engineering, where the system is divided into fundamental elements, and each element of the job is defined. This procedure also involves a feasibility study where available software choices are made in light of its social, economic and technical aspects. Besides, we conducted a requirement analysis
where the analyst transforms the needs and problems into precise and formal requirements using formal languages and diagrams [5].

b) Development Phase - This phase, an outline is put up describing an operational overview for the analyst and the users to reach an agreement to proceed with the project. In the detail design, the designers examine the inputs and outputs and clarify their processing. Following this, the programmers analyse the software after which, quality assurance is confirmed to ensure that the software meets the quality criteria [6].

c) Maintenance Phase - This phase entails any actions to correct, perceive, adapt and prevent and risks and issues from arising [7]. These activities implementation is mainly dependent on the opinion of the user following the testing of the final product outcome.

1.2 Principles of Security

Security falls under the responsibility of the security engineer, who needs a systematic approach to define security requirements and its characteristics, and in this regard, the approach addresses three characteristics of information security[8][9].

I. Security Attacks – These refer to any act that assaults and compromises the organisation's information security. This actions may take different forms[10]. The first form is an interruption, where a system asset is damaged or made inaccessible or useless. Another form of attack is an interception, where an unlicensed individual gains log on to an asset and primarily attacks confidentiality [11]. Other attacks include modification, where an authorised party tampers with a asset, attacking the integrity of the organisation and fabrication, where an unauthorised party integrates counterfeit objects into the system, attacking its authenticity.

II. Security Mechanisms – This is a detection mechanism, preventing or used to recover from a security attack and the mechanisms that accord the security requirements include cryptography and passwords. Cryptography is the sole defence that ensures private data is encrypted before its transmission so much so that even if the sent bits can be interpreted, only those who can decrypt the message can interpret what it contains. Passwords cover traditional passwords used throughout the years since the advent of computers. Following the request for ID user identification, the password is mandatory.

III. Security Service – This is a security improving service about the processing and transfer of data. The services target security attacks, using a variety of security systems for service provision. These facilities confidentiality, availability, integrity [12], and authentication. Confidentiality ensures that data can only be accessible to those who are authorised and availability safeguards the system resources from attacks that may breach their usability. Integrity, on the other hand, maintains the form of secure data and resources [13]. Lastly, authentication concerns a mechanism that allows the entity to provide and prove the correct identification.

2. Security Engineering

This branch of engineering science refers to the area that deals with real-world system security and integrity. In contrast, the engineering of software uses regular and controlled, measurement method
The aim is to develop, operate and maintain software. Another related component of discipline is Requirement Engineering, which is the process of deciding the type the software to be created and what performance characteristics are needed [16].

Therefore, as a science and discipline, software requirements engineering establishes and documents software requirement, and it comprises different distinct phases, namely, limitation, evaluation, design, certification and management. Notably, significant requirements are becoming increasingly apparent in the customer populations, and these include requirements for software and system security defences as well as countermeasures. The pressure on the importance and demand for security conditions in the engineering arena for software has led to the establishment of new planning for information systems that reflect a new regulation. Also, throughout the process of requirement engineering’s, the engineers should identify how the user define security [17]. The users' definition may vary from protecting every access to extensive protection of data privacy, reliability and accessibility. Moreover, the system engineer should consult a security engineer for the development of a security architecture that can deal with and meet up the needs of clients as well as the needs at the system level.

Customers and end-users often cannot articulate their security needs accurately as they are unfamiliar with the terms [19]. Therefore, developing a secure system is often laden with difficulty owing to the dynamic issues addressed. Added to this, the requirements of providing a user-friendly interface, online services for help or real-time training are requirements that are simple and static [19]. For this type of requirements, the determination of the technical solution is straightforward if the software is created and distributed, and such resolution is feasible throughout the life of the system.

The security solution depends on many factors like the system threats or the likelihood of them, and the technical condition available to protect it. In addition, the technical condition of the system heap, and the recognized value of the enterprise’s information assets are also critical. The solution in the bulk of cases requires development in order to protect against the likelihood of threats. The enterprise threat can vary according to identifiable circumstances, and if the security answer developed by engineers is static, then a protection one should be established to tackle the most significant threats.

3. The Proposed System

Organizations need to develop a secure web system in the probability of security holes and the top crucial choice in building such a system is to build it according to the software engineering principles. This approach is possible through the application of the life sequence of developing and engineering the software for securing the system linked security and process view. Such life cycle is the definition phase followed by three phases as follows.

1) **The Security Engineering Process View Phase (SEPV)** – This phase is initiated with the information collected from customers' collaboration regarding existing processes, limitations, safety policies and desired results, among others. The engineer carries out several analyses including, risk and vulnerability assessments, after which the engineered solution is proposed, the solution test is implemented, the procedure is documented, and the organization is trained on the new procedure refer to Figure 1, which shows the internal life cycle (SEPV) phase, followed by BPR, risk and vulnerability analysis, engineering solution, testing, and documentation.
In the BPR, an engineer and customer reach a common understanding of the business process review (PBR). That agreement includes the engineer, the end-user and other stakeholders. These people collaborate to shed light on the present business process.

Based on the collected information from the customer, the Multi-Criteria Decision-Making (MCDM) techniques is used to specify the priority of information [20]. The suggested method is the Analytic Hierarchy Process AHP which is defined as a method for solving complex decision problems. This technique will rank the collected information to find the satisfactory consideration of the software. The pertinent data will be derived by comparison of collected pairwise of collected data [21]. The outlets will be optimal indicator, which provided a strong basis for taking relevant measures, could be selected by sorting the weights of elements to reduce the number of indicators. Moreover, Analytic Hierarchy Process, will provide combination of qualitative and quantitative assessment outcomes [22]. This method will define a higher-level concerns of software implementation. The founded results will be applied based on composition operators for combining, manipulating and extending the concerns [23]. For instance if C3 proposed as a composed from C1 and C2, the composition operator ⊕ will be:

\[ C_3 = C_1 \oplus C_2 \]

Based on the present consideration, the security of information is achieved when the cost of solutions, remain reasonable in operational terms.

A practical security requirements assessment is one that involves appropriate security measures that a user group used earlier when accessing corporate data sources. Before blindly proceeding with a project to develop a security policy, the scope and limitation of the project have to be decided on for project feasibility. Among the significant issues tackled during such definition are studies of feasibility, determination of security and productivity balance as illustrated in Figure (2) and Figure (3) show optimal balance of security and productivity, which is demonstrated in the lack of security graph.
Assets refer to valuable corporate property that calls for different levels of protection, with the top common asset in the information system environment is the data or information of different confidentiality. The system protects against threats against threats, which processes/people are posing a probable danger to them, and as such, a specific asset can face many threats. Meanwhile, vulnerabilities refer to the manner/path threats attack assets, and they can also be described as the weak connections present in the security architecture that needs identification for each combination of danger/resource. Risk domain comprises of a distinct grouping of an interacted system that shares standard corporation functions and exposure components, with the former being identified during initial risk analysis along with the risk. Engineering solution forms the core of the proposed system, where the engineer determines the defensive methods utilized to develop a suitable policy to secure the system. Many defensive methods that have been created to prevent the vulnerability of assets to threats, and among them are virus protection, authentication, and encryption, and heuristic scanners. In the latter types, the scanner recognizes viruses via their signatures contained within the database. The virus scanner scans each system file for the existence of such signatures, and upon finding one, the virus is eliminated. On the other hand, the heuristic scanner possesses high scanning capabilities that are not limited to a specified number of viruses.

Following all the above phases, testing involves the collection of the entire information linked to the defenses system development and specifies the needs based on such information. The risks and weaknesses surrounding the web are analyzed and based on the risk domain, appropriate protection measure is selected to develop the security of the system. A simplified, secure system is built and tested in an environment that is akin to the actual one. Based on the test outcome, the initial secure system success/failure to achieve user security requirement is confirmed. With the failure of the initiated system, the engineer would go back to
the start and rebuild the system from scratch, but the system works, the engineer can proceed with the following stage. The final stage is documentation, where the security engineer documents the entire stages outcomes in the SEPV and such documentation is defined by the structured scheme, including:

1) DFD (Data Flow Diagram. When the end-users are satisfied with the engineering solution and the system testing results, the first SEPV phase ends.
2) The Development Phase – This phase follows the general development procedure of software, whether security software or any software product. Security software also requires the following development stages; outline and detail design, implementation, testing and quality guarantee. However, a top-most criterion for security software is selecting secure programming languages like object-oriented languages (e.g., Java, JavaScript and Visual Basic script).
3) The Maintenance Phase Based on the user’s views, the configuration of the software would determine how to maintain the operation, even if the software is already secure as illustrated in the following section.

4. EXAMPLE

E-mail is a highly used network-based application. The users, directly/indirectly linked to the Internet, can send mail to others, irrespective of the host operating system or its communication suite. The increasing dependence on e-mail needs authentication and confidentiality services. This study attempts to develop secure software under the principles of engineering the software to protect a confidential e-mail in government web from breaches.

The lifespan follows three phases; SEPV, development phase and maintenance phase. The SEPV comprises of the following sub-phases; First, in the BPR (Business Process Review), information relating to building secure email is collected, and in the present study, this involves dealing with facilitators keeping the design, obtaining and documenting security needs and stakeholders like designers, users and owners. Although it is crucial to have all the user groups represented the number of meeting participants should be kept to 5-6 individuals. Owners and users should be involved in all contexts and made aware of each party’s goals and requirements. In case the facilitators or stakeholders do not possess technical security knowledge, then security experts have to be involved in using expert knowledge in the analysis of risks and the stage of security design.

In the analysis of risks sub-phase, the model for the asset system and the security requirements are explained and depends on the information collected in the first sub-phase, the risk domain is developed, and the asset is confidential as top-secret data. The threats to look out for are interception, interruption, modification and fabrication by hackers or crackers. In this sub-phase, vulnerabilities are described as transmission paths of the e-mail from the transmitter to recipient.

The third sub-phase is an engineering solution, and this involves the identification of the most cost-effective countermeasure. Based on the outcome of the prior two sub-stages, the top optimum solution for the protection of e-mail is selected, which is Pretty Good Privacy (PGP).
The First International Conference of Pure and Engineering Sciences (ICPES2020)
IOP Conf. Series: Materials Science and Engineering 871 (2020) 012009
doi:10.1088/1757-899X/871/1/012009

Figure (4): pretty good privacy [24]

Figure (4) both services may be used for the same message. In summary, when both services are used, the sender first signs the message with its own private key, then encrypts the message with a session key, and then encrypts the session key with the recipient's public key that solution provides service of secrecy and verification that is suitable for e-mail. In mainly, PGG operations, consist of five services as follows.

In authentication (refer to Figure 4 for digital structure of PGP); i) the sender creates an email, ii) SHA-1 is utilized for producing a hash code of 160-bit, iii) this code is then RSA encrypted using the sender's private key. The result is encoded in the email messages, iv) the recipient uses RSA with the public key for the decryption and recovery of the hash code, and v) the recipient produces a new hash code for the message and compares it with the one that was encrypted. Matching hash codes, the message authentication validation is completed. Merging SHA-1 and RSA creates a valid digital signature, and Thanks to RSA power, the processor matching private key can produce the signature. Moreover, the strength of SHA-1 guarantees that no other person can create a new message with identical hash code and the original message signature. Confidentiality is given through coding sent or stored messages as local files. In this case, CAST-128 conventional encryption algorithm may be utilized, with two options IDEA or TDEA using the 64-bit cypher feedback (CFB) mode. In the context of PGP, every conventional key is used singly; a new key is produced randomly using 1280-bit number for every message. Although it is called the session key in documents, it is a one-time-key. The session key is linked to and relayed with the message .it is even encrypted together with the receivers public key for protection (refer to Figure 5b for the sequence).

The sequence, according to Figure 5 is as follows; the sender, using a random 128-bit session key, produces a message. The encrypted message is also using CAST-128/IDEA or TDEA with the session key encrypted with RSA. The receiver, while utilizing the pre-ended public key, makes use of RSA with its private key to decrypt and recover the session key. The session key is finally applied for message decryption. Figure 4 indicates the use of both the confidentiality and authentication services for the same message. Generally,
the sender, using both services, signs the message with a private key, and encrypts it with a session key, after which the recipient decrypts with the session key encrypted with the public key.

Another PGP service is compression. PGP compresses the messages, by default, following signing the message before encryption. This service provides e-mail space-saving for transmission and storage purposes. The compression algorithm placement (indicated in fig 4 by Z for compression and Z1 for decompression) is essential due to two reasons. First, the signature is produced before compression in order to store along with the signature for future verification. This procedure would prove difficult due to PGPs compression algorithm. This non-deterministic algorithm is used due to different tradeoffs in compression ratio versus running speed, as a result of which, different compressed forms are produced. Second, message encryption is applied following compression in order to support cryptographic security. The compressed message is better compared to the original plain text making cryptanalysis more difficult. ZIP is the optimum algorithm used.

E-mail compatibility is another PGP service, where when POP is employed, where an encrypted part of the block is transmitted. If the sender uses only signature service, the message digest is encrypted (with the private key of the sender), and if the confidentiality service is used, the message is encrypted along with the signature applying a one-time symmetric key. In other words, a part or whole resulting block comprises of a stream of arbitrary 8-bit octets. For every group of three octets of binary data, four ASCII characters are mapped in a format that adds a cyclic redundancy check (CRC) for the detection of errors in transmission (refer to Appendix 5B for description.

In Figure 5, the relationships among the above-discussed services are provided. If a signature is required on transmission, a hash code of the compressed plain text is used for encryption purposes. The plain text and signature (if present) are compressed afterwards. In case confidentiality is required, the block consisting of compressed plain text/compressed signature plus plain text, is encrypted and pre-pended with the public-key-encrypted conventional encryption key. Lastly, the whole block is transformed to radix-64 format.

(a) Generic Transmission Diagram (from A)            (b) Generic Reception Diagram (to B)

Figure (5) Transmission and Reception of PGP Messages [25]
On the receiving side, the radix-64 block is first transformed to binary form, and in case of encrypted messages, the session key is recovered by the receiver to decrypt the message, the resulting block is decompressed after. If the message is signed, the receiver recovers the hash code and conducts a comparison of it by calculating the hash code.

The final PGP services are segmentation. A maximum message length is specified for email facilities. For example, some of the Internet-accessible facilities require a maximum length of 50,000 octets. Any message that is longer should be segmented into smaller messages, each to be mailed separately. In order to resolve this limitation, POP sub-divides a too-large message into smaller messages sent through e-mail. Such segmentation is conducted following the entire processes, with the inclusion of the radix-64 transformation. Hence, the session key component and signature component only appear once (at the onset of the first segment). At the recipient’s end, POP strips off the e-mail headers and proceeds to reassemble the whole block before proceeding with the steps as demonstrated in Figure 3.

The fourth SEPV sub-phase is testing – and following the culmination of the above stages, the engineer develops a simplified, secure e-mail and tests it in an environment that is akin to the Internet of the user. The initial secure e-mail will be a success if it meets the user security requirement. If it fails that, the engineer goes back and redevelops the system. An Effective, secure e-mail testing would lead to the documentation sub-phase.

In the documentation phase, the security engineer documents the entire preceding SEPV sub-phases outcomes. After the users receive the documentations, they will want to proceed with developing the security system once satisfied with its evaluation, the proposed protective measures and the testing results. That stage marks the end of the first SEPV phase, and the engineer can proceed to the development phase. In case the users are not satisfied with some or the entire outcome, then the engineer can revert to the beginning and eliminate and rectify errors that have led to user dissatisfaction.

In the development phase, the details are expanded to develop secure e-mail while keeping the selection of secure programming language into consideration. In the maintenance stage, the web administrators can choose the software configuration

5. CONCLUSION

This study proposed a system that is a new approach to information security. That system should be dynamic, as opposed to static, and it has to be updated owing to new attack methods. To meet this condition, we are developing software system that is secure. Security engineering develops effective systems considering both time and resource. The collaboration between the engineer and the users should provide a great deal of help and convenience to both. Any misunderstanding or errors would be determined and identified in the first phases, increasing the performance of the engineer and ensuring that the developed system meets the user security requirements. Through such a system built under the principles of software engineering, while accurately going through each of the SEPV sub-stages. The AHP technique that correlated with composition operator is considered the best method for finding the optimal concerns in case of implementing effective software and keep minimum maintenance and cost effect.

Acknowledgements: Authors thankful Department of Computer Science, Collage of Science, Mustansiriyah University, for supporting this work.
Reference

[1] Pohl, K. (2010). *Requirements engineering: fundamentals, principles, and techniques*. Springer Publishing Company, Incorporated.

[2] Metzger, A., & Pohl, K. (2014, May). Software product line engineering and variability management: achievements and challenges. In *Proceedings of the on Future of Software Engineering* (pp. 70-84). ACM.

[3] Pohl, K., Böckle, G., & van Der Linden, F. J. (2005). *Software product line engineering: foundations, principles and techniques*. Springer Science & Business Media.

[4] Črepinšek, M., Liu, S. H., & Mernik, M. (2013). Exploration and exploitation in evolutionary algorithms: A survey. *ACM Computing Surveys (CSUR)*, 45(3), 35.

[5] Chanda, J., Kanjilal, A., Sengupta, S., & Bhattacharya, S. (2009, December). Traceability of requirements and consistency verification of UML use case, activity and Class diagram: A Formal approach. In *2009 Proceeding of International Conference on Methods and Models in Computer Science (ICM2CS)* (pp. 1-4). IEEE.

[6] Jacobson, I., Booch, G., Rumbaugh, J., Rumbaugh, J., & Booch, G. (1999). *The unified software development process* (Vol. 1). Reading: Addison-wesley.

[7] Robertson, S., & Robertson, J. (2012). *Mastering the requirements process: Getting requirements right*. Addison-wesley.

[8] Mouratidis, H., & Giorgini, P. (2007). Integrating Security and Software Engineering. *Idea Group Inc, Hershey, PA, USA*.

[9] Mellado, D., Blanco, C., Sánchez, L. E., & Fernández-Medina, E. (2010). A systematic review of security requirements engineering. *Computer Standards & Interfaces*, 32(4), 153-165.

[10] Fernández-Alemán, J. L., Sánchez-Henarejos, A., Toval, A., Sánchez-García, A. B., Hernández-Hernández, I., & Fernandez-Luque, L. (2015). Analysis of health professional security behaviors in a real clinical setting: An empirical study. *International journal of medical informatics*, 84(6), 454-467.

[11] Scandariato, R., Wuyts, K., & Joosen, W. (2015). A descriptive study of Microsoft’s threat modeling technique. *Requirements Engineering*, 20(2), 163-180.

[12] He, D., Chen, C., Bu, J., Chan, S., Zhang, Y., & Guizani, M. (2012). Secure service provision in smart grid communications. *IEEE Communications Magazine*, 50(8), 53-61.

[13] Tuballa, M. L., & Abundo, M. L. (2016). A review of the development of Smart Grid technologies. *Renewable and Sustainable Energy Reviews*, 59, 710-725.

[14] Bloch, M., & Barros, J. (2011). *Physical-layer security: from information theory to security engineering*. Cambridge University Press.

[15] Liu, Y., Chen, H. H., & Wang, L. (2016). Physical layer security for next generation wireless networks: Theories, technologies, and challenges. *IEEE Communications Surveys & Tutorials*, 19(1), 347-376.

[16] Li, B., Fei, Z., Chu, Z., Zhou, F., Wong, K. K., & Xiao, P. (2018). Robust chance-constrained secure transmission for cognitive satellite–terrestrial networks. *IEEE Transactions on Vehicular Technology*, 67(5), 4208-4219.
[17] Hamamreh, J. M., Furqan, H. M., & Arslan, H. (2018). Classifications and applications of physical layer security techniques for confidentiality: A comprehensive survey. *IEEE Communications Surveys & Tutorials, 21*(2), 1773-1828.

[18] Wu, Q., Mei, W., & Zhang, R. (2019). Safeguarding wireless network with UAVs: A physical layer security perspective. *arXiv preprint arXiv:1902.02472*.

[19] Joung, J., Choi, J., Jung, B. C., & Yu, S. (2019). Artificial Noise Injection and Its Power Loading Methods for Secure Space-Time Line Coded Systems. *Entropy, 21*(5), 515.

[20] Suneeta Mohanty, Sharmistha Roy, Mrinmoy Ganguly, Prasant Kumar Pattnaik (2019). Risk Assessment for Project Construction Based On User Perspective: An Experimental Analysis Using AHP. *IJITEE, ISSN: 2278-3075, Volume-8, Issue-9*.

[21] Devi Yunita BA, Ridhawati E, Huda M, Hashim A, Mat KS, Teh PT, Shankar K, Maseleno A (2019). Application of Analytical Hierarchy Process Method in Laptop Selection. *International Journal of Recent Technology and Engineering, ISSN: 2277-3878, Volume-8, Issue-2S3*.

[22] Mingxiang He, Xin An (2016). Information Security Risk Assessment Based on Analytic Hierarchy Process. *Indonesian Journal of Electrical Engineering and Computer Science Vol. 1, No. 3, March 2016, pp. 656 ~ 664 DOI: 10.11591/ijeecs.v1.i3.pp656-664*.

[23] Aksit, M., Tekinerdogan, B., & Bergmans, L. (2001). The six concerns for separation of concerns. In *15th European Conference on Object-Oriented Programming, ECOOP 2001*.

[24] Katz, J., Menezes, A. J., Van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC press.

[25] Davin, P. (2009). *U.S. Patent No. 7,600,121*. Washington, DC: U.S. Patent and Trademark Office.