Web Application Development Model with Security Concern in the Entire Life-cycle

Musa Bala Shuaibu, PhD.
Department of Computer Science and Engineering
Yanbu University College,
P.O. Box 31387, Yanbu Industrial City, Saudi Arabia.
dandaloto@gmail.com

Ruqayyat Ahmad Ibrahim
Department of Science and Technology
University of Jos
Plateau State, Nigeria
Ruqayyat_musa@yahoo.com

***Abstract* - e-commerce involves a great deal of credit cards, fund transfers, web shopping, and other forms of transmitting private information that needs to remain secure. Although, the knowledge of how to develop a secure e-commerce application is known, the major concern is the insufficiency of the right developmental model and practice. The use of inappropriate methodology in developing an e-commerce application can seriously undermine organizations' confidentiality and integrity. Having said that, many of the web application development models may have some kind of security considerations that may not be applied across the entire Web Application Development Life-cycle. This is unfortunate. This study proposes a web application development model that inculcates security considerations across the Web Application Development Life-cycle which can be adapted by security critical applications like the e-commerce sites. The implementation and evaluation of this proposed model, with previous web application security development models, have shown that a 96 percent security level is achieved, despite some four percent failures. The failures are of information severity status, which are not typically critical to the security of the underlying application.**

Index Terms – Web Application Development Life-cycle; Web Engineering; Application Layer; Security; e-commerce.

I. Introduction

The concept of secure software development emerged when practitioners who relied upon firewalls and intrusion detection systems finally understood that it is better to produce a secure system at the very beginning than to embrace defenses after deployment [1]. This understanding directed the attention of researchers on improving the development process by considering security as a requirement, instead of a corrective measure or after thought issue. Though, software development teams have realized the need for immediate software security, unfortunately, many applications use the traditional 'penetrate and patch' approach for security.

Security is one attribute of both software and web applications that should always be carefully considered. A simple defect in software or web can leave users open to attackers who find such defect for exploitation. Thus, there is a need for an appropriate development methodology to be created. This must consist of security considerations in order to avoid vulnerabilities that can be inherited at any stage of the Web Application Development Life-cycle. In other words, the risk is high for architectural and design security flaws to be present if an adequate job of identifying flaws is not completed.

Open Web Application Security Project [2] contends that four out of the current top ten vulnerabilities in the web application layer are due to improper input validation, which is a serious security breach. This error can result in compromise of data or an escalation of privileges, which may further result in revenue leakage and reputational damage for an organization.
Furthermore, it has been argued by practitioners that security breaches or vulnerabilities in the web application layer has nothing to do with secure socket layer or firewalls. Attackers are savvy enough to penetrate an application as legitimate users, but if safety programming methods are used the security breaches could be avoided or, in the least, reduced[3].
"Reference [4] argues that web application vulnerabilities are most obvious in web application code, not on the underlying technology". However, technologies, such as a fire-wall or intelligent detection and prevention systems, offer comparatively secure protection when achieved at the host and network layers. "Reference [5] added that strategies are crucial to the design of defense mechanism".

Practitioners and researchers focusing on the mechanisms of building security into web applications have propounded numerous studies, but few provided a security emphasis during the Web Application Development Life-cycle. Emphasis is centered on identifying important threats and dealing with it at early stage [6]. Thus, this study investigated the state of the art security considerations in web application development models with the need to focus on security issues throughout the life-cycle.

II. Related Work

Although the concept of a secure software development approach is new in the software industry and among the software development community, there are many approaches of developing secure software. Three of the well-known approaches are the Microsoft's Security Development Life-cycle (SDL) [7], the Comprehensive Lightweight Application Security Process (CLASP) [8], and the Software Security Touch Points [9].
Even though the approaches differ, the key points remain the same:

- Risk assessment and management is essential in all the approaches
- Utilization of best practices is also crucial
- Security education is emphasized in all the approaches mentioned.

A few studies discovered in the course of this research, directly inculcates security at each stage of the Web Application Development Life-cycle. However, the majority of research considers security only at a single stage of the life-cycle, typically at the beginning, end, or as a complete after thought issue.

For example, [10] uses security testing methodology approaches to look for vulnerabilities and to detect XSS in web service. This was accomplished through techniques such as penetration testing and fault injection. In addition, spoofing services and denial of services allowed for vulnerabilities to be located. Though the approaches were successful, the concern of inculcating security focus across the development of the life-cycle was not rebutted. This is important in conjunction with e-commerce which is an opportunity for attackers.

Studies such as [11] and [12], uses the threat modeling approach to tighten security and to ensure that vulnerabilities are not introduced during software development. Other studies, such as [13] use security patterns as a technique for mitigating vulnerabilities. “Reference [14] and [15] uses some formal definitions for security mitigation”. “Reference [16] have demonstrated how threats can be assessed in a web application, which included details of semantics and context information”.

“Reference [17] presented an automatic generation of security tests as an approach by using a formal threat model as predicate/transition nets”. This generates all attack paths in the threat model by converting them to executable test codes according to the Model-Implementation Mapping (MIM) specifications. Though the approach was successful in unveiling various paths and threats, it was somewhat restricted by model implementation.

Although all the related techniques mentioned are effective in dealing with vulnerabilities to some extent, the existence of a technique for curbing vulnerabilities is not sufficient to produce a highly secure application if there does not exist a guiding principle to implement the technique [18].

Different approaches or techniques are used in some studies to mitigate a specific security threat that may not be applicable when it comes to other threats, it may be more effective if different techniques are combined in security critical applications like e-commerce. The stage in the Web Application Development Life-cycle where security approach or technique is emphasized varies with different studies. Some studies have considered security checks during requirement gathering, such as[19], other studies, such as [20], and [21], have considered security in the coding stage. Although, studies that considered security around coding stage of development emphasis the fact that attacks are more likely due to improper coding practices such as SQL injection, it is necessary to build security in the entire stage.

Furthermore, this study found to the best of its knowledge that [22] and [23] has focus on the security consideration in web application development framework particularly e-commerce.

On one hand, Ge et al. model investigated general-purpose information system development methods, particularly the feature driven development and risk analysis which was then integrated to address the development of secure web applications. In this approach, risk assessment is fused into agile processes as an engineering method that satisfies the functional requirement for web applications. The joining of these two mechanisms (feature driven and risk analysis) is aimed at mitigating security in web development processes.

On the other hand, Sengupta et al. model delved further into the e-commerce domain by putting forth a model for security using a general life-cycle approach which then highlighted perceived threats and vulnerabilities with a proposed framework for mitigation of those perceived threats.

The major challenge is how to fit the security enforcement mechanism into existing models without re-inventing a new Web Application Development Life-cycle from scratch that may lead to serious design challenges.

Therefore, the inspiration from these related views drive our expedition to answer whether web applications will have an improved security if security concerns are inculcated at each stage of the Web Application Development Life-cycle. It is possible to improve the security of a web application by improving the engineering approach or through process improvement in the development of that application, which will lead to a higher quality and secure product.

III. Methodology

The methodology therefore, follows two major categories. First category involves the following activities:

- Selection of web development framework from existing web development frameworks
- Building new security framework with checks and balances around the web development framework
- Developing a prototype from the new security framework and
- Testing the prototype based on security concerns.

The Second category involves evaluation phase for the new security framework with the existing security frameworks for web application using an experiment method known as “within subject design”.

The implementation focuses on the features that are systematically obtained from the design and implementation rules associated with security frameworks because it is expected that the properties built in the frameworks which follows the security description will manifest in the application and then provide the necessary defences.

This study therefore, investigates the security of existing security framework by developing a prototype based on the framework in order to determine what it was and the characteristics or factors which are of interest, then comparing the test result with the prototype of the new security framework that contains interventions.

Ideally, a possible causal characteristic have been manipulated, while keeping constant other factors or characteristic. The investigation is hence carried out using penetration and vulnerability assessment of the security breaches in order to measure if the security checks and balances that are incorporated as intervention has any effect on the end result.

Our proposed security model considers security at early and throughout the entire life-cycle of web application development. Although the model is based on Extreme Programming (XP) methodology, it is unique in terms of the inculcating security consideration in all the development phases. The inclusion of threat modeling during the design stages, the conduct of code reviews by pair programmers during coding, the use of code-scanning tools and manual testing during the testing phase have demonstrated significant security focus during the life-cycle of web development which is not implemented in other models as detailed in the phases of our proposed web development model.

The key elements in the phases is the stakeholders collaboration activities at the requirement that ensures a common understanding to be gained among business representatives and development team on security implications, considerations, and requirements which gives an initial thought on key security milestones.

The design specification with security consideration, abstraction of security representation, description of how new vulnerabilities will be introduced in the design are activities proposed as a solution in this study. The result of these activities will highlight the specified security controls provided and areas where further planning or risk mitigation is required.

Secure coding practices and security infrastructural enforcement, security best practice violation policies are activities for quick initial review into highlighting errors and potential challenges. The use of small release in iterative manner with acceptance test for evaluation is one of the activities proposed under this study. The solution is flexible in order to accommodate the needs of different applications and web development organizations because it outlines a basic structure that can be easily implemented.

The contribution extends to demonstrate that the case study produced as a by-product of the model, after undergoing penetration testing for security issues by a team that are independent from its development group, will have significant security assurance and a reduced rate of external discovery of security vulnerabilities. This is when compared to applications that have not been subject to the same intervention. Provided in Fig 1 are the phases in a top down manner with iterations in-between phases as indicated by the arrows:

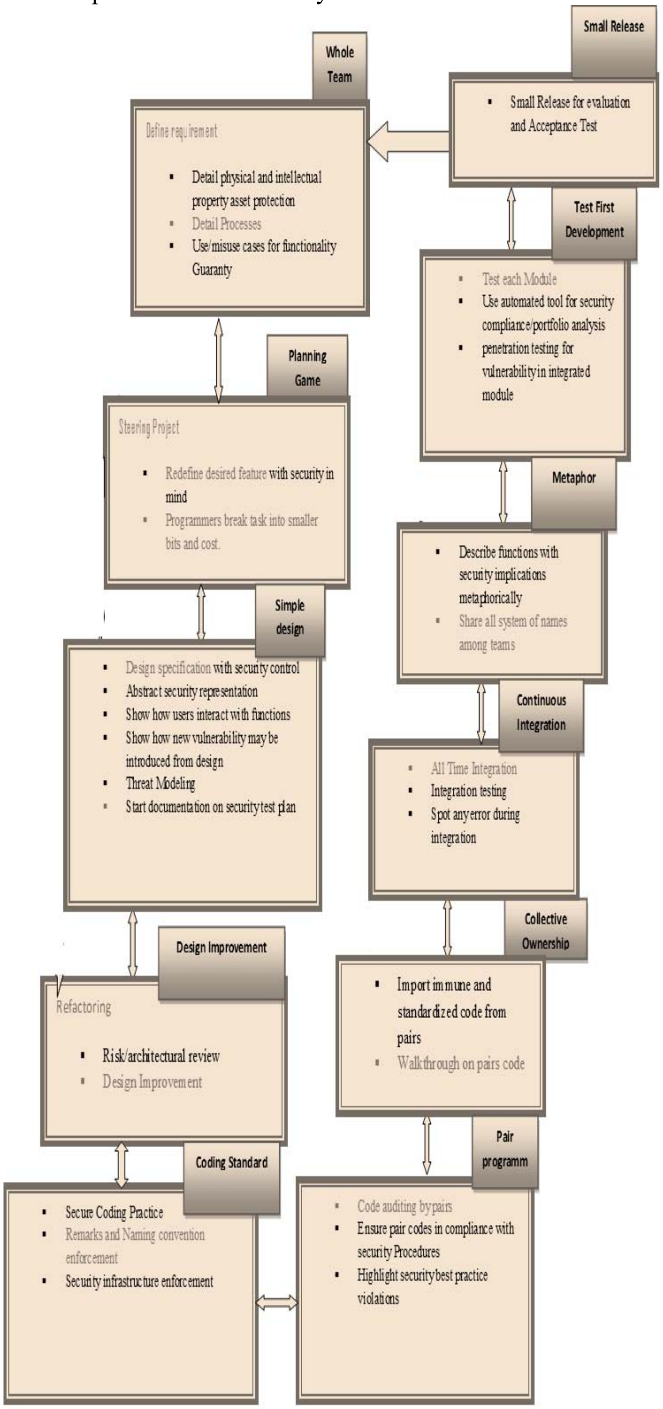
Small Release
Small Release for evaluation and Acceptance Test
Whole Team
Define requirement
Detail physical and intellectual property asset protection
Detail Processes
Use/misuse cases for functionality Guaranty
Test First Development
Test each Module
Use automated tool for security compliance/portfolio analysis
penetration testing for vulnerabilityin integrated module
Planning Game
Steering Project
Redefine desired feature with security in mind
Programmers break task into smaller bits and cost.
Metaphor
Describe functions with security implications metaphorically
Share all system of names among teams
Simple design
Design specification with security control
Abstract security representation
Show how users interact with functions
Show how new vulnerability may be introduced from design
Threat Modeling
Start documentation on securitytest plan
Continuous Integration
All Time Integration
Integration testing
Spot anyerror during integration
Design Improvement
Refactoring
Risk/architectural review
Design Improvement
Collective Ownership
Import immune and standardized code from pairs
Walkthrough on pairs code
Coding Standard
Secure Coding Practice
Remarks and Naming convention enforcement
Security infrastructure enforcement
Pair programm
Code auditing bypairs
Ensure pair codes in compliance with security Procedures
Highlight securitybest practice violations

Fig 1: Web Application Development Model with Security Concern

Step 1: Whole Team
Step 2: Planning Game
Step 3: Simple Design
Step 4: Design Improvement
Step 5: Coding Standard
Step 6: Pair Programming
Step 7: Collective Ownership
Step 8: Continuous Integration
Step 9: Metaphor
Step 10: Test First Development
Step 11: Small Release

IV. Implementation

The common practice of e-commerce transaction is through the use of third party merchant companies. The merchants companies offer coupon codes as an incentive to buyers to purchase from their Web site. The study uses the coupon alternative with the expectation that improved security controls will lead more customers to make purchases online without inordinate fear of losing their money. The online book store application used under this study is a virtual store on the Internet that facilitates searching books from a list of categories of books of interest. Selections made by the customer are grouped together in an online "cart" and later presented as an order at the checkout point where details of the customer will then be required in completing the transaction. The user is required to supply details of billing information such as shipping address or delivery address and sometimes a discount coupon code. Finally, an invoice is generated to complete the order and payment. Considering the payment gateway obtainable in the third party transaction, a validation process occurs between the shopping cart and the financial establishment by authenticating the information supplied by the customer. This information is then encrypted and passed through a secure channel where it is later sent back as decrypted information to the customers' shopping cart entity with an appropriate debit or credit balance.

An experienced web development company having a wide market-based reputation that provides valuable and creative services about web design and web development is given the task of developing the book store prototype based on two security models. Prototype 1 is developed based on our proposed security model under this study, while prototype 2 is developed using Ge et al. security model.

A three man committee of experienced developers was assigned to implement the security models. The mandates for these individuals were to design and build the end product under the supervision of this study. This study provided guidance in the security aspect and proper definition of model controls. A weekly meeting was scheduled to discuss and observe progress and subsequently provide clear guidance in any questionable areas of the model. After the implementation of the first model, a questionnaire was presented to the team about the satisfaction of the website developed. The opinions expressed include: clarity of model as a reflection in quality, availability of information to reflect on quality, service quality, and security assurances in shopping with the website. The entire team agreed that there was clarity in the model with enough information to back-up the development process. Both prototype 1 and 2 case studies were developed using the same website content. However, not all stages in the model were applicable to this particular case study.

The stages in the proposed model such as whole team, planning game, coding standard, pair programming, and test first development are fully implemented during the case study. Other stages such as simple design, collective ownership, continuous integration and small release are partially implemented. The reason for partial implementation is due to the nature of the case study which does not encompass a full blown e-commerce application such as Amazon.com where every aspect of user connection has to be authenticated.

Hence, our security model constitutes in the design and design improvement stage, an abstraction of how users interact with the system and functions, how vulnerabilities could be introduced, and risk reviewed.

A. Dataset

A comparison to measure the security between prototype 1 and prototype 2 was conducted. The same dataset sample is used for testing the two prototypes. Prototype 1 represented the types of security considerations inculcated across the development life-cycle of web application for this study. It was hoped that Prototype 1 would show improvements in security when compared with Prototype 2, for the latter represented the kind of systems in use in e-commerce today.

Penetration testing which is believed to be a general practitioner standard for determining application strength is used on the two implemented websites. This is used to measure the possibility of an attacker circumventing or violating the application defenses. A good result from such penetration testing means gaining more assurances about the security of the web application and further attributing the security consideration within the Web Application Development Life-cycle.

Bias was avoided by using a black box testing method in which the tester knew nothing of the application or defense mechanisms.

The dataset, in this regard, was the aggregation of various types of injection techniques and the attack surfaces used to circumvent the defense of web application.

This study divided the test cases into 10 categories :
1. Information Gathering Test
2. Configuration Management Test
3. Authentication Test
4. Session Management Test
5. Authorization Test
6. Business logic Test
7. Input validation Test
8. Denial of service Test

9. Web service Test
10. Ajax Test (Asynchronous JavaScript and Extended Make-up Language XML)

The dataset seeks to identify trends currently affecting the security of web applications which will help e-commerce businesses to improve security by highlighting problem areas and to identify areas where specific classes of vulnerability are on the increase.

A similar dataset was used in the Open Web Application Security Project OWASP [24] to measure and ascertain the most critical web application vulnerabilities annually.

B. Result of Prototype 1 compared with Prototype 2

As shown in Fig 2, out of the 50 test cases planned and executed on both prototype 1 and 2, about 96 percent success rate is recorded on Prototype 1 as compared to about 29 percent recorded on Prototype 2. And 4 percent failure in prototype 1 as compared to 72 percent failure in prototype 2.

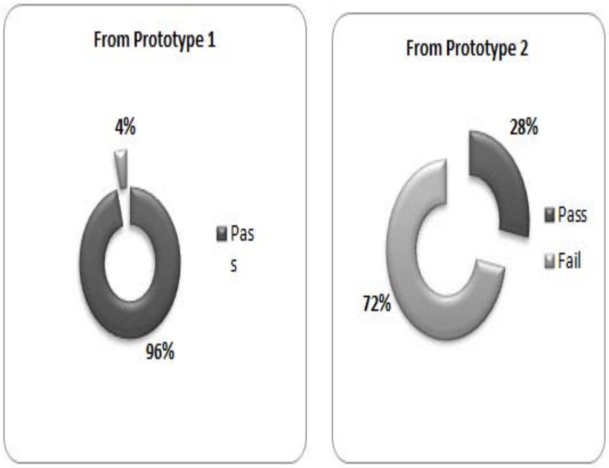

Fig 2: Comparative Success and Failure Summary

On one hand, considering only the areas covered by the 96 percent pass for prototype 1, it is found that the passes are covering 46 percent high severity vulnerability cases, 44 percent medium vulnerability cases, 6 percent low vulnerability cases and 6 percent information vulnerability cases .

This means that the success rate in prototype 1 is more on high and medium severity cases than for low and information severity cases. Whereas in contrast to prototype 2 which only has 28 percent pass rate is found to cover medium severity with 36 percent medium severity case, 29 percent information severity case, 28 percent high severity case and 7 percent low severity. This means that the few passes it has achieved is mostly detecting medium and information severity rather than on high severity cases.

VI. Conclusion

Therefore, it is pertinent to mention that since the test cases are standardized across both prototypes, the testing was conducted externally to prevent any bias of having prior knowledge of the system rudiments for both prototypes. The same e-commerce implementation with the same functional features is administered.

This study ascertains, based on the result that a significant level of security is by far attributed to the effect of inculcating security considerations during the Web Application Development Life-cycle and that websites will have improved security if security concerns are inculcated at each stage of the development life-cycle of the web application as the case in Prototype 1.

By extension, the result provides the security assurances to critical web applications despite not having a perfect 100 percent result, as a result of some limitations of the study.

Furthermore, it is pertinent to state that existing security web applications are still vulnerable to some form of attacks due to lack of security consideration throughout the stages of the development life-cycle. And vulnerability trend could be attributed to enormous funding on technology and functionality rather than security issues in web application development processes.

Our major contribution is the provision of a solution to the security challenges faced by most web application in which security considerations are an afterthought issue.

Acknowledgment

We wish to thank Professor Muhammad Muda (Universiti Sains Islam Malaysia) for his guidance and profound support.

References

[1] B. Mansoor, "Chapter 8 - Intranet Security," in *Network and System Security (Second Edition)*, J. R. Vacca, Ed., ed Boston: Syngress, 2014, pp. 221-258.

[2] D. Wichers, "The 2013 OWASP Top 10," in *AppSec USA 2013*, 2013.

[3] D. Stuttard and M. Pinto, *The web application hacker's handbook : discovering and exploiting security flaws*. Indianapolis, IN: Wiley Pub, 2008.

[4] N. Macia and F. G. Tinetti, "Chapter 29 - Improving Security in Web Sessions: Special Management of Cookies," in *Emerging Trends in ICT Security*, B. Akhgar and H. R. Arabnia, Eds., ed Boston: Morgan Kaufmann, 2014, pp. 481-491.

[5] A. Almaatouq*, et al.*, "If it looks like a spammer and behaves like a spammer, it must be a spammer: analysis and detection of microblogging spam accounts," *International Journal of Information Security,* vol. 15, pp. 475-491, 2016.

[6] I. Uusitalo*, et al.*, "Towards Evaluation of Security Assurance during the Software Development Lifecycle," in *Availability, Reliability and Security,*

2009. ARES '09. International Conference on, 2009, pp. 817-822.

[7] M. Howard and S. Lipner, *The security development lifecycle*: O'Reilly Media, Incorporated, 2009.

[8] W. Enck*, et al.*, "On lightweight mobile phone application certification," in *Proceedings of the 16th ACM conference on Computer and communications security*, 2009, pp. 235-245.

[9] B. De Win*, et al.*, "On the secure software development process: CLASP, SDL and Touchpoints compared," *Information and Software Technology,* vol. 51, pp. 1152-1171, 2009.

[10] M. I. P. Salas and E. Martins, "Security Testing Methodology for Vulnerabilities Detection of XSS in Web Services and WS-Security," *Electronic Notes in Theoretical Computer Science,* vol. 302, pp. 133-154, 2014.

[11] A. Mourad*, et al.*, "An aspect-oriented approach for the systematic security hardening of code," *Computers & Security,* vol. 27, pp. 101-114, 2008.

[12] R. Hassan*, et al.*, "Goal-oriented, b-based formal derivation of security design specifications from security requirements," 2008, pp. 1443-1450.

[13] J. Dong*, et al.*, "Automated verification of security pattern compositions," *Information and Software Technology,* vol. 52, pp. 274-295, 2010.

[14] R. Tissot*, et al.*, "AB Formal Framework for Security Developments in the Domain of Smart Card Applications," *International Federation for Information Processing Digital Library,* vol. 278, 2010.

[15] M. H. Alalfi*, et al.*, "A verification framework for access control in dynamic web applications," in *Proceedings of the 2nd Canadian Conference on Computer Science and Software Engineering*, 2009, pp. 109-113.

[16] Xiaohong Li, "A Unified Threat Model for Assessing Threat in Web Applications," *International Journal of Security and its Applications,* vol. 2, 2008.

[17] D. Xu*, et al.*, "Automated security test generation with formal threat models," *Dependable and Secure Computing, IEEE Transactions on,* vol. 9, pp. 526-540, 2012.

[18] R. Kissel*, et al.*, "SP 800-64 Rev. 2. Security Considerations in the System Development Life Cycle," 2008.

[19] C. Mao, "Experiences in security testing for web-based applications," presented at the Proceedings of the 2nd International Conference on Interaction Sciences: Information Technology, Culture and Human, Seoul, Korea, 2009.

[20] S. Chong*, et al.*, "Building secure web applications with automatic partitioning," *Commun. ACM,* vol. 52, pp. 79-87, 2009.

[21] A. Sengupta*, et al.*, vol. 3803 LNCS, ed, 2005, pp. 328-331.

[22] X. Ge*, et al.*, "Agile development of secure web applications," 2006, pp. 305-312.

[23] A. Hopkins, "Web Application Vulnerability Statistics 2010-2011," 2012.

[24] S. Christey*, et al.*, "CWE/SANS Top 25 most dangerous software errors," ed, 2011.