INDIAN JOURNAL OF DEVELOPMENT RESEARCH AND SOCIAL ACTION
VOLUME 13 • NUMBER 2 • JULY-DECEMBER 2017 • PP. 131-146

An Empirical Framework to Address Security Requirement

Nikhat Parveen[1], Sandeep Kumar Nayak[1] and M. H. Khan[2]

ABSTRACT

For any software to be secure, security must be analyzed at core level, which is the early stage of requirements. Since, all security analysis has to be carried out through prejudiced approach such as rules, defined-laws, models and policies. However still there is a lack in security requirements that behave as a constraint for proper functionality of the secure system. It is very difficult to capture security requirement in order to fulfill business goals that protect assets from threats and hence produce trustworthy systems. This is the fact that any security breaches are caused directly by vulnerable software. A scrupulous review has carried out on information security that helps to gather the information regarding the fact as there is no such standard methodology available that can secure requirement phase. There is no policy, rules or any guidelines exist hence, it is desirable for developing a prescriptive framework that address security at requirement phase. In this paper security requirement framework is proposed at initial stage of software development. The chronological approach of framework is presented that helps security experts to analyze security and mitigate threat at requirement phase.

Keywords: *Software Security, Security Requirement Engineering, Risk Analysis, Requirements Engineering, Security Requirements, Argumentation.*

Introduction

The most important aspect of life is security and the increasing incidence and potentiality of IT infrastructure makes security a serious concern. In recent years, several reports related to security failures have become usual. The publicity agreed to failures of IT security has made a pressure on software developers to construct and assure software with appropriate level of security. The most identified problem related to security failures in software contains numerous flaws and errors that help the hackers to exploit the security and make the software compromise by any means (CCB, 2009). The author C. Mann in an article 'Why is Software So Bad' reveals the fact of bad habits and the loopholes that are available due to inadequate software life cycle processes

[1] *Department of Computer Application, Integral University, Lucknow, India*
[2] *Department of Computer Engineering, I.E.T, Lucknow, India*

and hence led to the development of insecure application (Man, 2002) . There are number of vulnerability reported in an application where security is not concerned as requirement of the complete system. For example in credit card system the exposed data of millions of card holders detail can easily be hacked through the history of transactions stored by the software (Sullivan,2014). Still there is another source where the security has been easily compromised is dictionary password and attack of PIN that hardly reflect to one's mind. These examples suggest that if the system would not improved with security features it will directly lead to financial impact. While building secure systems, the ground up level of security should be considered first which a fundamental principle for any secure software development is. Requirements engineering is the most critical accomplishment phase of any major advancement of projects. Most development projects create new versions based on earlier requirement releases. The requirements phase being the initial phase of software development offers the best opportunity to build secure software. Due to the criticality of security requirements, the practice should be applied as early as possible in the requirement engineering process.

The Security Requirements Engineering (SRE) technique supports interlink of model building and analysis that allows for reasoning about incremental models. It should also represent and assess alternative options so that an optimum way to security can be selected. Therefore, it is highly desirable to define security requirements during software requirement specification. This paper explores how to determine adequate security requirements for a system. For adequate security requirements, the procedure follows to elicit requirements that lead to satisfy security goals. A system must able to determine whether the security requirements satisfy the security goals and whether the system satisfies the security requirements. Security requirements must take into consideration of developer implicit or explicit assumption that an objective of the system will behave as expected. This paper concise and unify our previous work to form a framework which shows how our earlier contributions can be applied together coherently and effectively.

Software Security: Need and Analysis

Software security is about understanding software that induced with security risks and how to manage the risk when threat occurs (McGraw and Mead, 2003). It is an idea of engineering software that manages itself to function correctly under malicious attack. The process of taking care of security mechanism and requirement of security are the main objective of software security. Many defensive technology such as watermarking steganography, code signing are used by software developers in order to improve the security

of software (McGraw, 2005). Software security process involves acquiring, designing, testing and implementation of secure software.

A literature review and research study reveals the fact that in the field of security, the feature cannot be added at a glance (Taylor and McGraw, 2005). It is the process that should be integrated right from the inner core of the software. The integration of security is necessity feature that should be the part of software development process (Beznasev and Cess, 2008). Due to the lack of security, many software industries have been ruined just because of not implementing security from the beginning. It is not sufficient as to have patch of feature, rather it should lie within the software as the vulnerabilities can compromised the software at any point of time. Security is needed when an industry establish some resources in the form of assets, assets can be tangible such as cash or intangible such as confidential information or industry reputation (Chivers and Fletcher,2005, and ISO/IEC,1999) that one's want to protect from any attackers. The presence of vulnerability can easily harm the assets as a result attacker can affect the violation and breakdown the security through which the integrity of software may loss.

Security analysis has to be obtained through proper understanding of the safety needs of an industry. The needs should not only secure the whole system but it should well define that what needs to be secure such that the customer is satisfied by end- results. These needs can be further alienated into discrete requirements such that the requirements would be complete, clear, concise and correct. Requirement phase is the initial phase towards problem domain that also produces solution to the domain. It plays as foundation stone for the desired product and hence becomes the specification for the software process. Therefore, it is the most appropriate phase that helps to ensure the security requirements which provide accurate blue print of what customer believes for satisfying system.

Security Requirement Framework Approach

Literature survey on security requirement reveals that to maintain security at any stage costs very expensive. Therefore, security requirement process should be integrated well in advance, preferably during the initial stage of SDLC i.e. requirement phase. Following points forms a strong theoretical basis in support for the approaches in order to develop a prescriptive framework for security requirement.

- Find out the deficiency as well as benefits of current security requirements approaches.
- Analyze and integrate security at requirement phase

- Develop the outcomes to decide existing desires of security requirements engineering.
- Verify whether certain approaches are valuable enough for security requirements and if so, assess the past requirements into architecture, design, and implementation.
- If no such approach is found valuable enough, then the foundation of the development of perspective security requirements framework is to be introduced in order to maximize the advantages and minimize the deficiency found in existing approaches.

Based on following points, an effort has been made to develop a framework from security perspectives. Lack of knowledge about which requirement must be considered when it comes to focus security has been considered. Requirement engineering is appropriate discipline for application of technical principles and approaches for developing, communicating, and managing requirements. Usually, security is highlighted at different stages in the software lifecycle, more specifically the requirements stage is highly important.

Developments of an application with prior knowledge of security are much safer than those applications where the security is an afterthought .The techniques that are used in security are basically to develop secure information, design phase, coding phase or any other communication that are affected through threats. Researchers and Developers working in the field of Software Security Engineering have paying attention are so-called best practice in the software lifecycle process. These are number of existing methodologies that are enhanced with security features which provides an integrated framework, or in some cases, phase-by-phase supervision for promoting security-enhanced software throughout the phases of SDLC. The next section follows security enhanced software development approach proposed by various researchers and practitioners.

SQUARE Approach

SQUARE is process that build security at early stages of software development life cycle. It is designed specifically for security a requirement engineering that consists of major nine steps which help to deliver categorized and prioritized security requirements. Square is based on interaction between requirements software engineers and stakeholders of an IT project that facilitate requirements engineering team as a major importance (www.cert.org).

The approach is as follows: a) Agree on definitions. b) Elicit security goals. c) Develop artifacts. d) Perform risk assessment. e) Decide elicitation

technique. f) Identify security requirements. g) Classify requirements. h) Prioritize requirements. i) Inspect requirements. The SQUARE approach is best suited by the requirements engineers of a project's and security experts, with the perspective of supportive executive management and stakeholders.

Agile Development with A Focus On Extreme Programming (XP) Practices.

The methodology used in AGILE is to practice and promotes continuous iteration of development and testing procedure through development lifecycle of the projects. This process includes security requirements. The main ideas of development processes are: (Boström Gustav *et al.,* 2006). AGILE development emphasizes on four core values they are represented as Identify critical assets, Formulate abuser stories, Assess abuser story risk, Negotiate abuser and user stories, Define security-related user stories, Define security-related coding standards and Cross-check abuser stories and countermeasures.

Haley and His Colleagues' Framework

A framework for Security Requirements Engineering is based on following 4 activities which always carry out in iteration (Haley *et al.* 2008). The most important part of the framework is iteration of activities between requirement and design phase. Satisfying a security requirement leads to new assets, resulting in new security requirements. Identify functional requirements, Identify security goals, Identify security requirements and Verify the system are the four activities that define necessary security requirements.

Clasp Approach for Securing Software Development Life Cycles

This approach being Comprehensive Lightweight Application Security Process (CLASP) consists of 30 process pieces that can be incorporated to improve security requirements engineering life-cycle. This approach helps to insert security into every phase of software development life cycle Graham,2006) . The main ideas regarding the approach are: a) To determine risk mitigations and resolve deficiencies and conflicts and b) Identify resources and trust boundaries.

Microsoft Trustworthy Computing Security Development Lifecycle

The framework forecast on security activities and deliverables to every phase of Microsoft's software development process (Lipner and Howard, 2005). The major objective of the framework is to state security feature requirements that are based on consumer demand and consistently maintain standards. The following activities would normally be distributed across all phases of SDLC. They are:

- Identify assets.
- Identify dependencies
- Identify threats.
- Identify use scenarios.

SDL performs an important role in integrating security during early stage and all phases of software development. Best practices, process improvements and metrics are considered the three elements of secure software developments. The aim of using these elements is to minimize vulnerabilities in requirement, design, code and documentation to detect and eliminate vulnerabilities as early as possible.

Axelle Apvrille and Makan Pourzandi Approach For Security Requirements

The approach is based on following steps that incorporate security requirements during analysis phase. The developers predetermined themselves to perform these steps, but could not able to explain in details (Torr, 2005). The steps that follows in this approach are

- Identify the security environment and objectives.
- Determine the threat model.
- Choose security policy, which includes prioritizing based on sensitive information..
- Evaluate risk.

Security Requirements Engineering Process Approach (SREP)

The SREP comprises of nine-step that is based partially on SQUARE but incorporates Common Criteria and notions of reuse (Mellado *et al.,* 2007). SREP is similar to SQUARE. The activities performed are:

- Agree on definitions
- Identify risk or critical assets
- Identify security objectives
- Elicit threats and develop artifacts.
- Risk assessment
- Elicit security requirements
- Label and prioritize requirements
- Requirements inspection
- Repository improvement

Gary McGraw's Approach: Gary McGraw being the author of his book Software Security: Building Security In describes Seven Touch points for Software Security (Gary,2006). He describes seven touch points as lightweight best practice that can be applied to various software development artifacts. The touch points are placed in order of effectiveness as follows: a) Code review b) Architectural threat analysis c) Penetration testing d) Threat-based security tests e) Misuse cases f) Security requirements and g) Security operations.

Applying software security into practice requires changes according to the way organizations build software. These security best practices treated as the basis of high-quality software engineering and provide platform for security situation throughout the software life cycle.

TSP-secure: This framework is provided by SEI's Team Software Process (TSP) that consists of set of operational process and regimented methods that apply on software engineering principles by the software production team (Humphrey, 2002). TSP for Secure Software Development (TSP-Secure) enhances the TSP whose purpose is to target directly on the security of software applications. The principal objective of this framework is to construct TSP- based processes such that it can produce secure software.

Secure Software Development Model (SSDM): For producing secure software the major steps has to be taken by any developer is to integrate Software Engineering (SE) process with Security Engineering (Sodia et. al, 2006). Secure Software Development Model (SSDM), integrates security engineering with software engineering in such a way that effective production of secure software products can be achieved (Man, 2002, Gnewa, *et al.*, 2003). SSDM is based on unified model that combines the laws into software security techniques. The model consists of features that guarantee the successful production of secure software products.

AEGIS: It's a socio- technical software engineering methodology that develops secure software system that seeks to hold security and complex factors, such as functionality, scalability, simplicity, time-to-market, etc. Appropriate and Effective Guidance for Information Security is a software development process to develop secure and usable software system based on security requirements identification, assets, risk analysis and context of use. AEGIS is formulated to be a lightweight process that can wrap into any software development life cycle process and can be seen its application to Spiral model of software engineering (Ivan *et al.*, 2006).

Rational unified process-secure: The Rational Unified Process is the most popular and complete process models based on iterative process. It is not a single concrete perspective process rather it is an adaptive process. The steps involved in RUP's are:-

- Develop software iteratively
- Manage requirements
- Apply component-based architectures
- Visualize model software
- Verify software quality
- Control changes to software

Most of the guiding principles and activities in this process model are based on software engineering standards. RUPSec is an pervasive process used for accomplish secure software systems (Rez *et al.*, 2005). These extensions are integrated with number of Activities, Roles and Artifacts in order to capture document and model threats with security requirements of system (SSA, 2007).

Oracle's Approach (OSSA): Encompassing every phase of secure product development, Oracle Corporation has made an extensive effort for developing security into the design, testing, and maintenance of its products. It's a methodology which includes a comprehensive set of security assurance mechanisms both for customer system and leverage products. The objective of these processes is to improve the effectiveness of security mechanisms and reduce the probability of security vulnerabilities in products. Collectively these assurance mechanisms and processes are known as Oracle Software Security Assurance (OSSA) (www.oracle.com).

An exhaustive review of literature concludes that there is no such framework, tool or model is available that quantify security of software requirements. Hence, it's a high demand to develop such a model or framework that can quantify the security at the early phase of software development. This is the fact that any changes made at final stage of development of software costs very expensive. Therefore, security requirement process should be integrated right from the beginning of the development of software, preferably at the requirement analysis phase.

THEORETICAL BASIS

During the past studies, researchers and developers has revealed the fact related to development of secure software that it is very complex and time consuming process. Keeping this in mind security process should be integrated well in advance preferably at requirement process. The most ignored part during secure software development is security requirement. Unfortunately the security is being considered as technical issue that should be handled at design or implementation phase of the software. For better secure software, it is important to have clear idea regarding secure

requirements. To develop a prescriptive framework for security requirement the following points forms a strong theoretical basis in reinforcement of the methodology followed in the paper.

- Si* is a modeling tool for Secure Tropos.
- Fernandez (Fermendez, 2004) – use cases are the most helpful actions that determines the rights each actor needs and for considering possible attacks.
- Peterson (Peterson, 2004) – suggested that use cases and misuse cases are considered as a basis for security requirements.
- Van Wyk and McGraw clearly suggest of using abuse cases (Gary, 2006).
- Core security requirements artifacts (Moffett *et al.*, 2004) takes an artifact view in order to achieve better security requirements.
- Security patterns behave as bridging a gap between requirements to architectures and then designs (Haley *et al.*, 2007, Weiss *et al.*, 2007).
- Tropos (Giorgini *et al.*, 2007) is a self-contained life cycle approach that specify in terms of how to retrieve requirements specification.
- Software Cost Reduction (SCR) being a formal specification approaches that has also been used to specify security requirements (Heitmeyer *et al.*, 2002).
- The Guidelines of Common Criteria (www.niap-ccevs.org) also provide similar results.

With this study of SRE methods, we are able to analyze which method can be adopted by the software developer in order to achieve Security Requirements Engineering. An effort has been made in this paper to address security properties from requirement perspectives. To evaluate security the main reason behind the failure is lack of knowledge about which properties should be considered. It has been strongly argued by the researchers that, security being a quality attributes, which can be measured at the architectural level (Taylor and Chess, 2005). The facts discussed above forms the strong basis for the proposed framework. This paper describes on security properties at requirement phase, requirement characteristics, requirement time security metrics, and bridging the correlation between requirement attributes and identified security factors. This appears that the work have been highly neglected in past.

The Framework

In consideration to needs and significance of construe software security, a prescriptive framework is proposed. The framework may be used in

requirement phase to predict software security quantitatively. Goal of software Security Requirement Framework [SRF] is to provide high level protection against the vulnerability and threat to the software that helps to contribute the mitigation of security failures. A qualitative conclusion can also be drawn with respect to results obtained using the framework. Security Requirement Framework has been proposed on the basis of congenital and essential components of software security. As shown in figure.1, Security Requirement Framework consists of five phases as follows:

1. Elicit Requirements
2. Elicit Security Goals
3. Analyze & Quantify
4. Verify & Validate
5. Review & Packaging

Security Requirement Framework

Phases of Security Requirement Process are Being Described in the Following Section

Phase I: Elicit Requirements

This phase is the initial phase that deals with various issues related to problem-solving activity which assesses need and importance of requirements in particular software. One of the crucial tasks of this phase is to set application & quality goals that helps to examine the possibility, scope and necessary tools for development. Goal of this phase is to identify functional requirement & non- functional requirement with the specification that fulfill the user needs. The next process is to map functional requirement to non- functional requirement in order to provide primary set of specifications to successive phases of development. These activities motivate developer to work out on a model that serves on large account. In this phase, following activities are proposed to elicit requirements.

- Assess need and importance
- Set Application & quality goals
- Identify functional requirement & non- functional requirement (Parveen *et al.*, 2014).
- Map functional requirement to non- functional requirement

Phase II: Elicit Security Goals

This phase consists of four steps related to security attributes that will elicit the requirement of the software in better way and helps to advancement of better production of software security. Security of software cannot be precise directly, to do this some security objectives have to be set forth that can be treated as a roadmap for developing secure software specifications. An effort will be made to identify related factors regarding requirement parameters and security attributes in order to quantify security requirement. The next step is to identify requirement parameter which is one of the important activities. The ultimate goal is to address security through requirement characteristics. Hence, identifying requirement parameter is necessary. Identify security attributes will encompass commonly accepted set of security factors. It will help to bridge the gap between requirement and security. The identified security attributes will affect the requirement and metric predictions; as a result correlation has to be established between them. In this section, a set of activities has been defined which performed to identify security goals, as follows:

- Identify security issues (Parveen *et al.*, 2014)
- Develop security Objectives (Parveen *et al.*, 2015)
- Identify requirement parameters and security attributes (Parveen *et al.*, 2015)
- Bridge the gap between requirement and security (Parveen *et al.*, 2015).

Phase III: Analyze & Quantify

Before performing security analysis, it is necessary to understand what is to be built. This task involves reviewing of all existing high-level system documentation. It includes activities such as ensuring developers security awareness, global security policy, conducting risk analysis of requirement. The sub activities performed in this section is to establish correlation between requirement parameters and security attributes in accordance to its anticipated influence and importance. A model is developing for quantifying security at requirement phase. Quantification enables the evaluation and assessment of security and help to establish security goals and cost. As explained earlier following activities are performed in this phase.

- Analysis of security requirements
- Establish correlation between requirement parameters and security attributes
- Model development (Parveen *et al.*, 2015)
- Security quantification

Phase IV: Verify and Validate

Common wisdom, intuition, speculation and proof of concepts may not be the reliable sources of reliable knowledge; hence it is essential to place the specified condition under testing. Testing is considered as best empirical strategies performed through quantitative analysis in order to verify and validate. This can be assuring by theoretical basis through literature survey and analysis. The primary goal of verification is to test whether the developed models actually measures what it is supposed to measure. This can be carried out by adequate exposure from experts to get the better solutions. However informal reviews may be carried out at any of the stages performed in the requirement specification process. The validation processes capture all involved activities to test the building process of right product. It is to perform with realistic data in order to prove the developed models through valid measures so that the model can be accepted through quantifiable values. Following activities will be used to verify and validate.

- Assure Theoretical Basis
- Perform Expert Review
- Validation through Tryouts
- Accept Model by Analyze Results

Phase V: Review & Packaging

This phase is informal in nature and has been placed as the fifth phase with full liberty to enter at any of the former phases. The idea of such prescription is to have ample adequate exposure and then twist back for better review, in consideration to all the previous phases. However, informal reviews may be carried out at any of the stages in the requirement specification development process. Packaging is a conclusive phase of the specification development process. It is now ready to developed requirement specification with the needed accessories to prepare ready- to- use product, like any other usable product.

Motivations and future work

Security is an essential characteristic and composite factor in an application. Development of security plan for software is not a onetime built-in process; it is based on existing security specifications that can be reused. It is recommended by many researchers that security should be integrated at the initial stage of the software. To achieve the purpose, secure requirement is must during development of life cycle. The paper presents a software security requirement framework for quantitative evaluation of software security. Each phases of proposed framework are seemingly experimental in nature and comprise of three components, each contiguous study, development and validation of tasks.

Conclusions

Application designed with security is always safer than those where security is reconsidered. The issues regarding security are generally considered during the Design phase of development of software once the requirement specification has been frozen. On theoretical basis a prescriptive software security requirement framework [SRF] has been proposed. The proposed framework comprises of five phases. Study of each phase is discussed in brief. The framework will support the requirement, design and analysis of non-functional properties for systems at the software architecture level effectively. Security requirement framework has been proposed in this paper which will detect and remove defects in advance, which in turn, will reduce development time, effort and budget of the software.

References

Boström Gustav, Wäyrynen Jaana, Bodén Marine, Beznosov Konstantin, Kruchten Philippe (2006), Extending XP practices to support security requirements engineering. In: Proc. 2006 int'l workshop software eng for secure systems (SESS). ACM Press,. p. 11–8.

Common Criteria Board (2009), Common criteria for information technology security evaluation, version 3.1.

D. Taylor and G. McGraw (2005), "Adopting a Software Security Improvement Program", *IEEE Security & Privacy*, pp. 88-91.

Fernandez EB. (2004), A methodology for secure software design. In: Proc of the int'l symp web services and applications (ISWS);. *www.cse.fau. edu/_ed/ EFLVSecSysDes1.pdf*>.

Gary McGraw, (2006), Software Security: Building Security In, Addison Wesley.

Giorgini P, Mouratidis H, Zannone N. Modelling (2007), Security and trust with secure Tropos. In: Mouratidis H, Giorgini P, editors. Integrating security and software engineering, Hershey. p. 160–89.

Graham, Dan. (2006), Introduction to the CLASP Process. Build Security in; <*http:// buildsecurityin.us-cert.gov/daisy/bsi/articles/best-practices/requirements/548.html*>.

Guerra, P.A.D.C., C. Rubira and R. de Lemos, (2003), A Fault-Tolerant Software Architecture for Component-Based Systems. Lecture Notes in Computer Science. 2677: 129-149. Springer.

H. Chivers and M. Fletcher (2005), "Applying Security Design Analysis to a Service-Based System," *Software: Practice and Experience,* vol. 35, no. 9, pp. 873-897.

Haley CB, Laney R, Moffett JD, Nuseibeh B. (2007), Arguing satisfaction of security requirements. integrating security and software engineering. In: Mouratidis Haralambos, Giorgini Paolo, editors. Advances and future visions;. p. 16–43.

Haley CB, Laney R, Moffett JD, Nuseibeh B. (2008), Security requirements engineering: a framework for representation and analysis. *IEEE Trans Softw Eng*.; 34(1): 133–52.

Heitmeyer C. Software Cost Reduction. In: Marciniak John J, (2002), Encyclopedia of software engineering. New York (NY): John Wiley and Sons.

Humphrey, Watts S., (2002), Winning with Software: An Executive Strategy. Boston, MA: Addison Wesley, (ISBN 0201776391).

ISO/IEC (1999), "Information Technology—Security Techniques—Evaluation Criteria for IT Security—Part 1: Introduction and General Model," ISO/IEC, Geneva, Switzerland, Int'l Standard 15408-1, Dec.

Ivan Flechais, Cecilia Mascolo and M. Angela Sasse, (2006), Integrating Security and Usability into the Requirements and Design Process, Proceedings of the Second International Conference on Global E-Security, London, UK, *http:// www.softeng.ox.ac.uk/personal/Ivan.Flechais /downloads/icges.pdf*

K. Beznosov and B. Chess (2008), "Security for the Rest of Us: An Industry Perspective on the secure-Software Challenge", IEEE Software, IEEE, Jan /Feb. pp: 10-12.

Lipner S, Howard M. (2005), The trustworthy computing security development life cycle. Microsoft Corp.;. <*http://msdn.microsoft.com/en-us/library/ms995349.aspx*>.

Mann, C., (2002), Why Software Is so Bad, Technol. Rev. (July/August).

McGraw, G. (2003), Software Security: Thought Leadership in Information Security. Cigital Software Security Workshop.

McGraw. G. and N. Mead, (2005), A portal for software security. IEEE Security. Privacy, 3: 75-79.

Mellado D, Fernandez-Medina E, Piattini M. A (2007), Common criteria based security requirements engineering process for the development of secure information systems. *Comput Stand Interf* 29(2): 244–53.

Moffett JD, Haley CB, Nuseibeh B. (2004), Core security requirements artifacts. Technical report. Open University;. ISSN 1744-1986.

Nikhat Parveen, Mohammad Rizwan Beg, M. H. Khan, (2015), Model to Quantify Availability at Requirement Phase of Secure Software, *American Journal of Software Engineering and Applications*. Vol. 4, No. 5, pp. 86-91. doi: 10.11648/j.ajsea.

Nikhat, Parveen, Md. Rizwan Beg, *et al.* (2014), "Software Security Issues: Requirement Perspectives." *International Journal of Scientific & Engineering Research* ISSN 2229-5518. Volume 5, Issue 7, July, pages: 11-15.

Parveen, Nikhat, Md. Rizwan Beg, and M.H Khan. (2014), "Bridging the Gap between Requirement and Security through Secure Requirement Specification Checklist." Proceedings of 16 th IRF International Conference, 14 th December, Pune, India, ISBN: 978-93-84209-74-2, Pages: 6-10.

Parveen, Nikhat, Md. Rizwan Beg, and M.H Khan (2015), "Model to Quantify Confidentiality at Requirement Phase" Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering & Technology(ACM ICARCSET - 2015) 6th & 7th March ISBN: 978-1-4503-3441-9; doi:10.1145/2743065.2743117.

Parveen, Nikhat, Rizwan Beg, and M. H. Khan (2014), "Integrating Security and Usability at Requirement Specification Process." *International Journal of Computer Trends and Technology (IJCTT)* 10: 236-240.

Peterson G. (2004), Collaboration in a secure development process part 1: information security bull; June p. 165–72.

Reza, M., A. Shirazi, P. Jaferian, G. Elahi, H. Baghi and B. Sadeghian, (2005), RUPSec: An Extension on RUP for Developing Secure Systems- Requirements Discipline, Proceedings of World Academy of Science, Engineering and Technology Vol. 4: ISSN 1307-6884, pp: 208-212.

Rosado David G, Gutiérrez Carlos, Fernández-Medina Eduardo, Piattini Mario (2006), Security patterns and requirements for internet-based applications. Inter Res; 16(5): 519–36.

Sodiya, Adesina Simon, Onashoga, Sadia Adebukola, Ajayi, Olutayo Bamidele (2006), Towards building secure software systems, Proceedings of Issues in Informing Science and Information Technology; June 25-28; Salford, Greater Manchester, England. Vol. 3. *http://informingscience.org/proceedings/InSITE2006/IISITSodi143.pdf*

Software Security Assurance, (2007), State-of-the- Art Report (SOAR) Information Assurance Technology Analysis Center (IATAC) Data and Analysis Center for Software (DACS) Joint endeavor by IATAC with DACS. Oracle Software Security Assurance [web page] (Redwood Shores, CA: Oracle Corporation). *http://www.oracle.com/security/software-securityassurance.html.*

SQUARE (2012), Tool. Software Engineering Institute, Carnegie Mellon University. <*http://www.cert.org/sse/square-tool.html*>;

Sullivan, Richard J. (2014), "Controlling Security Risk and Fraud in Payment Systems," Federal Reserve Bank of Kansas City, *Economic Review,* vol. 99, no. 3, pp. 47–78.

Torr P. (2005), Demystifying the threat modeling process. IEEE Secur Privacy 3(5): 66–70.

Weiss M. (2007), Modelling security patterns using NFR analysis. In: Mouratidis H, Giorgini P, editors. Integrating security and software engineering, Hershey;. p. 127–41.

www.niap-ccevs.org/cc-scheme/; 2007.