Security Requirement for Software Quality - A Survey of Engineering Discipline

Mohammad Ubaidullah Bokhari
Associate Professor,
Dept. of Computer Science
Aligarh Muslim University, India

Mahtab Alam
Research Scholar,
Dept. of Computer Science
Mewar University, Chitorgarh, India
alam_mahtab@rediffmail.com

ABSTRACT: ***The requirements are the foundation stones upon which the entire software depends. The quality of software entirely depends upon the software requirements attributes. The software attributes like reliability, availability, dependability etc. are well considered in the beginning phases of the software life cycle. Problems in building and refining a system can be mostly traced back to errors in requirements. The requirement elicitation, specification, and validation are the important criterion to assure the quality of the software. The requirement engineering discipline is getting more and more popular in the last few years due to its dependability of the software quality. There is no 'golden metrics' available for assessing the relative security of the system. But we have found in practice that the use of a standard set of security analysis guideline is very useful. In this paper we present a survey report of the research work done on the requirement engineering discipline.***

KEYWORDS: ***Software Requirement, Security Attributes Information Flow, Secrecy and Integrity, Checklist, Risk Assessment, Confidentiality.***

I. INTRODUCTION

Software requirement is one of the major stages of software development life cycle and the success of entire project is depending upon the quality requirement. A number of authoritative studies have shown that due to requirement engineering defects the cost varies from 10 to 200 times as much to correct once the software has been deployed as if it were corrected during requirement phase [5]. Major problems due to poor quality requirements are over budgeting and scheduling, inadequate scope of application, and improper functioning. Requirement engineer usually faces a number of problems such as there are not any tools available to analyze and model the quality of requirements, requirement identification does not consider all the stakeholders and ignorance of quality and nonfunctional requirement.

Security means an unintended user are prevented to operate the system under any circumstances. Due to increase rate of software system usability and its easy operation all the valuable assets of business and critical Increase rate of software system usability and its easy operation all the valuable assets of business and critical mission are stored in computer-bases system. The assets are increasingly misused by the abuse-actor due to worldwide accessibility of the internet and the automation of systems. Security is a measure's of the system ability to restrict malicious user and provide services to legitimate use. Any attempt to breach security is called an attack; it can be in a large number of forms [25].

The rest of the paper is organized as follows. Security requirement engineering is explained in section II. Security attributes are explained in section-III, Security, Privacy, Policies and Common Criteria are explained in Section-IV and V respectively. Information Flow, Trust Management and Risk Assessment are explained in section VI, VII and VIII respectively. Concluding remarks are given in section IX.

II. SECURITY REQUIREMENT ENGINEERING

Researchers and engineers are continuously concerning about secure software, whose primary ambitions is to implement a security protocol or

mechanism, that functions correctly under malicious use and that does not contain loopholes. **Baskerville** [5] pointed out some discrepancies in the security requirement engineering discipline in the context in which the software operates, he presents the three generation of security design methods as:

- **Checklists**: A checklist is lists of questions to be checked, on the assumption that previous experience of applications that can be applied to the current one.
- **Mechanistic engineering methods**: A method is used, which focuses on security in isolation from other aspects of system design.
- **Integrated design**: A development process includes security as a facet of the whole development.

Consumer heavily relies on the security requirement specification for the products. If the requirements are poorly specified and elicited, there is no need to say about strive for security. To build accurate, reliable and secure software, consistent security requirement must be specified. **John** and **Jens** [23] have investigated current practice by doing a filed study of eleven requirement specification, which is being built from 2003 to 2005. They present the complete list of security requirements found in the specification and divided them into security areas and every requirement is categorized as functional, non-functional, or security assurance. An ISO/IEC standard for security management has been used as an example of how a standard could help to specify better security requirements. Finally they note that some of the requirement specification found in their studied was hard to categorize in a clear way, just because of the diversity in definition of non-functional requirements.

Requirement engineering is an art to investigate systematic approach to analyze security threats and security requirements. Security requirements are non-functional requirement mainly concerned with how to protect the assets from misuse or harm. Requirement engineers have to identify the scope of protection, cause of security threats, and evaluate the trade-off among different design decisions. **Lin** [28] proposed how abuse frames can provide a means for bounding the scope of security problems in order to analyze security threats and derive requirements using Jackson's Problem **Frames** [42]. They introduce two conceptual tools - anti-requirements and abuse frames - and deploy these tools systematically to explore security problems arising from requirement level. An anti-requirement is the requirement of a malicious user that exploit the existing system. Thus, anti- requirement defines a set of undesirable phenomenon imposed by the malicious user that will ultimately exploit the software. An anti-requirement uses security threats represented by an abuse frame which share the same notation as the normal use frames, but each notation is treated as opposite meaning. The functional requirement varies between applications to applications across different problem domain. However, same cannot be said about their security requirements which are no-functional. Most application needs to specify levels of identification, authentication, authorization, integrity, and privacy. At the high abstraction level, the application tends to have common vulnerable assets subjects to the same type of attack.

The similarity of threats and attacks pose to considerable uniformity when it comes to the architectural security mechanism. **Firesmith** [39] presents the reusable parameterized templates for specifying security requirements with an example of such template and its associated usage. He described about the security quality factors (characteristics, attributes, aspects) and decomposed them in sub factors[19][35] such as: Identification, Authentication, Authorization, Immunity, Integrity, Intrusion Detection, Non-repudiation, Privacy, Security Auditing, Survivability, Physical Protection. The reuse of security requirement is then found in the sense of the security sub factors as a basis of requirement. At the highest level of abstraction, the security requirement teams perform the iterative procedure to analyze the security requirements as incremental, parallel, and time-boxed manner which are as follows:

- Identify the Valuable Assets, threats, attackers, relevant situation, and relevant template.
- Estimate vulnerability.

- Determine negative outcomes, security criterion, measure, and required level.
- Prioritize vulnerabilities.
- Consider security sub factor.
- Specify requirement.

All the above process could be used to create security requirement based on the reuse of parameterized templates for security use cases. The discipline of engineering of the requirements for a business, system center involves merely its functional requirements. One must engineer about its quality, data, and interfaces with other as well as its architecture, implementation, testing, maintenance, and security constraints. Most engineers are poorly trained to elicit, analyze, and specify the security requirements, and few of them which are trained have only given an overview of security mechanism such as password and data encryption. The different types of security requirements and provides associated examples and guidelines with the intent of enabling requirements engineers to adequately specify security requirements. He provides guidelines for security requirements engineers and categorizes the security requirements according to their objectives[46]. **Alam** [30] presented a software secure requirement metrics using a checklist in which all the security parameters proposed by Jan Jurjen are considered. With the help of these available checklists Degree of Secure Requirement (DSR) metrics can calculate the security concern of any proposed requirement.

III. SECURITY ATTRIBUTES

Organizations and companies are realizing the importance of security in the life cycle from a network security, to system security and application security. Software is an integrated end-to-end process [16] [6] [24]. Security is a process not a product; it is a continuing process to meet the changing required by the stakeholders. There would be highly negative impacts of risk if security is not integrated into the development life cycle. **Pankaj** [41] presents waterfall model as the reference model to illustrate the correspondence between software engineering and security engineering, and briefly discuss each stage. **Kazman** and **Clements** [25] proposed some common security requirements attribute which are stated as under:

- Non-repudiation is the property that a transaction (access to or modification of data or services) cannot be denied by any of the parties to it.
- Confidentiality means the data or services are protected from unauthorized access.
- Integrity is the property that data or services are being delivered as intact.
- Assurance is the property that the parties to a transaction are who they purport to be.
- Availability is the property that the system will be available for legitimate use.
- Auditing is the property that the system tracks activities within it at levels sufficient to reconstruct them.

Security's defining features are historical and continued standoff between attackers and stakeholders. No system is completely secure; attackers invariably increase their caliber as security engineers improve protection measure. To remain secure, the system must change according to the environment changes, preferably by predictive changes. These observations suggest that the need of security engineering quite relative to agile mindset. **Johan** [21] presents an agile security requirement engineering practice by using user stories and enhance the requirement engineering by abused stories. User stories express the capabilities of the requirements that deliver business goal. The requirements are ranked according to their perceive value and a score has been assigned according to ranking. The score may be varying throughout the development life cycles. Further, he pointed out that agile development is an iterative process. The goal of iteration it to realize the greatest possible value within the limited time frames. The security related details in user stories is not ambiguous, therefore, it can be enhanced by using another abused stories. Similar to ranking and scoring according to business value in user stories, abused stories may be also ranked and scored according to prone threat they poses to assets. Implementing the user story may widen the attack surface of a system, while abused stories allows business value to be traced more

accurately and facilitates rational planning of the effort required for security-related development.

The path of the software development process starts from requirement. Requirement has primarily focused on elicitation and representation of concrete business requirements. Security is generally forefront of the stakeholders concerns, except comply with the basic standards. **John Viega** [22] described a resource-centric approach which is a subset of the CLASP (Comprehensive, Lightweight Application Security Process). This approach covers the security requirements better than technology-driven methods. The CLASP approach to formulate the security requirements consists of the steps as: Identify system roles and resources: Roles generally goes to identify the owners and users of resources. Therefore, it is better to identify which roles are parameterized with respect to permission. Resources are any piece of data that can be used by a program. Categorizing resources: Category includes an indication of which role can be own a particular resource, as well as potential value of the resources. The main advantages of categorization are that requirements can be utilized as organization standards and applied across projects. Identification of resources interface: Security requirements on data change through the lifetime. The confidential data may be secure when a user is working on a own machine, means the client side application does nothing special to protect it, but when clients send the data over network to middleware, protection against attack is desirable. Thus, requirement on the user data depends on the data interacts with other resources in the system. Requirement specification: The resources when interacting with other resources is specified by CLASP on the basis of core security service are:

- Authorization: What privileges on data should be granted to the various roles sat various times.
- Authentication and Integrity: How to identify the access of users to the resources.
- Integrity is the data origin authentication.
- Confidentiality: Specifies what confidentiality mechanism is required, and what should identify establishment.
- Availability: How resources are available on request.
- Accountability: What logging is necessary? It's also including non-repudiation. For the above security aspects the **SMART** (Specific, Measurable, Acceptable, Realistic and Traceable) requirement [26] is extended to SMART+. CLASP provides the first structured methodology to deliver secure requirements of a software system which is far more effective than an ad hoc methodology of security requirement.

IV. SECURITY, PRIVACY AND POLICY

Security and privacy have some common characteristics, and both have been active research areas in computing for a long time. **Liu and Yu** [27] proposed a methodological framework for security and privacy analysis based on the concept of strategic social actors {i*}. They emphasizes that security and privacy goals must be identified and dealt with starting from the earliest stages of software engineering process [37][47]. Security and privacy issues originate from human concerns and intents, and thus should be modeled through social concepts [27] [13]. Social concepts mean the relationship of software with other components. An actor is one who involved in the system, whereas, the goal captures is the high-level objectives of the system. One actor can depend on another actor to achieve the goal. An actor can be a malicious user or an intended user. The factors governing the success of attackers are their motivation, vulnerabilities of the system, and capabilities to carry out the attacks. When the attacker identified, a set of system boundaries is formed, and then exhaustively searches for possible attackers. With the help of this methodology a system designer make a decision how to protect the system for security and privacy aspects from potential attackers and vulnerabilities. Security Policy is a set of requirements document that have to be enforced by the system. Policy means protection of information or system from any unintended users. Security requirement is set of tuples in the form of <mi, pk, tj> where m is a message, p is required security property and t is the transaction mode [49].

V. COMMON CRITERIA

Common Criteria is an international standard used as a basis for assessing the security properties of software products or systems. All the earlier work suggests that use cases have increasingly common during requirement engineering but offer limited support for eliciting security threats and requirements [32]. As a result misuse case, abuse cases [47], and security use cases [42] all have been proposed as methods for specifying security requirements. An approach to elicit security requirements based on use case 'actor profile' [45]. Such an approach is not an easy since CC standards by themselves are often too confusing and technical for non-security specialist to understand and utilize [14]. In practice CC present the security requirement for the product under the distinct category of functional and assurance requirement [43]. This presents regular use cases both in UML diagram and textual templates form with misuse cases [14]. They propose a labeling misuse cases along with normal actor and use cases to represent threat and mitigation in a diagram. Misuse cases are generally driven by threats. However, they provide guidelines for helping developers to describe misuse cases textually and map them to security objectives and requirements based on the CC.

VI. INFORMATION FLOW

With increased reliance on software system, protection of confidential data has become an increasingly important research problem. To be sure that a system is secure with respect to confidentiality, it should be regularly analyzed to check if it enforces good confidentiality practices. The analysis demonstrates that the information controlled by a confidentiality policy cannot leak the important data. These policies which monitor the movement of data through the system is called information flow policies. It is traditionally checked at the run time monitoring system [1], or by static analysis of the source code [4]. But these are post-implemented approaches-finding spurious information flow, cause to send back for designing of the software.

The stakeholders provide requirements in natural languages which is not understandable to designers. In modem era requirements are expressed in formal languages like Message Sequence Chart (MSC) [11]. TMSC (Triggered Message Sequence Chart) recently proposed enhancement to classical MSC for checking the information flow properties namely non-interferences [8]. They concerned with non-interferences information flow by using two security levels high and low. Actions having a low security level is observed by anyone whereas not the case with high level. They show how individual TMSCs may be equipped with a ready-set (set of instances in TMSC) semantics, which make data-flow analysis on requirements expressed feasible in practice.

VII. TRUST MANAGEMENT

Requirement engineering is concerned with the characteristics of the system-to-be, which comprises not only software, but also the diverse components needed for it to achieve its purpose. An important element of a system's requirements is its security requirements, which demands a system-level analysis [3]. Using problem frames, one must analyze the behavior of domains within the context of the security requirements. Threat description [45] is very handy for security requirements in a problem frames environment. Threat in the context of system is to exploit vulnerabilities and damage assets. An attacker always tends to exploit the assets in some way. Attackers are a type of stakeholders. Recent research work has taken this approach, looking the requirements and goals of the attacker [20] [9] [40]. An attacker wants a system to have characteristics that create vulnerabilities, whereas requirement engineer try to ensure that requirements of an attacker cannot meet their intentions. Haley and team [34] shows how a trust assumption used by a requirement engineer to define and limit the scope of analysis and decision made during the process. Every system has a problem domains and every domain has interfaces, which describes how the system accomplishes the goal.

This interplay is a specification, describing how the requirements are satisfied [17]. The main difference between specification and requirement is that specification is the behavior of phenomena visible at the boundary whereas requirement is the solution of the problem. When analyzing the problem, how requirement is satisfied depend on the characteristics of the problem domain. Security Requirement and trust assumption has an analogous relation. Trust is defined to be the firm belief in the competence of an entity to act dependably, reliably and securely within specific context [18]. How security requirements are

satisfied depends on the trust management. One of the most discussed issues about security is related to trust assumption [12] [2]. Trust is necessary for system to function in today's diverse environment of software systems.

VIII. VIII. RISK ASSESSMENT

Risk is the probability of exploitation of software vulnerabilities [44]. The exploitation may be tangible or intangible to the owner, and considered in the business world as the source of profit, not just a source of loss [7]. Risk is generally placed into three categories: Qualitative, Quantitative, and **Hybrid** [34] [36]. Qualitative approaches use subjective terminology for risk measurement, quantitative approaches are used for concrete values and considered more accurate than anyone else, but defining the underlying parameters is often difficult [31]. There is an inherent relationship between risk and trust. **Zaid** and **Francesco** [36] present a model to show that small individual risks can be transformed into major risks when combined together in a complex attack.

They evaluate the level of compliance a certain software system's design exhibit the given security policy, especially the probability that transaction fails to meet its security requirement. They first define how to calculate the transaction's risk using information from the security policy and presented by components and channels. As the system grows, the functionality or some components of the system may be added or discarded, which cause the change in architecture. Any mistakes during modification leads to provide attack surface, which focuses to new security requirements or go through the existing one carefully. The risk assessment is performed by the following steps:

The owner of the organization determines the goals or mission to be achieved by identifying critical valuables (Assets) which must need security requirements.

- Find the weakest surface of the system (Vulnerabilities) through which threats can exploit and leads to risk.
- Identify the Threats and categorize them using STRIDE model (Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service, and Elevation of Privilege) which provides risk.
- Rank the threats using the DREAD categories (Damage potential, Reproducibility, Exploitability, Affected Users, and Discoverability) which can be mitigated by using security controls (Countermeasures and safeguards). The controls can be Technical, Operational or Management.

The risk is clearer once threat and vulnerability are defined. An adapted definition of risk, from NIST SP 800-30, is "The net mission impact considering the probability that a particular [threat] will exercise (accidentally trigger or intentionally exploit) a particular [vulnerability] and the resulting impact if this should occur.

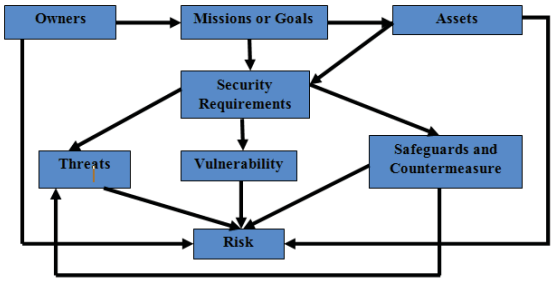

Figure-1, Risks Assessment

IX. CONCLUSION AND FUTURE WORK

In this particular survey paper we discussed about, Security Requirements and some of its factors in which it relies. All the major approaches and models of Security Requirements Engineering discussed by various researchers have been clearly stated and we have developed a security requirement assessment tool for measuring degree of security requirements of an application. We also presented an approach to adopt security policies for secure operation of an organization. In future with these approaches we have planned to design and develop the security requirements models or tool to quantify the security concern of an application.

X. REFERENCES

[1] A. Reniers, "Message Sequence Chart: Syntax and Semantics", PhD Thesis, Eindhoven University of Technology, 1998.

[2] Alberts,C.J.,Dorofree A.J., Managing Information Security Risks: the ACTA VE Approach, Addison Wesley, July 2002.

[3] Alexander, "Modeling the Interplay of Conflicting Goals with Use and Misuse cases, "Proceedings of 8th International Workshop on Requirements Engineering: Foundation for Software Quality (REFSQ'02), Essen. Gern1at1Y, 9-10 Sep 2002. pp. 145-152.

[4] Arnab Ray, Bikram Sengupta, and Rance Cleaveland, "Secure Requirements Elicitation Through Triggered Message Sequence Chat1S", First International Conference (ICDCIT 2004), volume 3347 of Lecture Notes in Computer Science,India, December 2004. Springer Verlag. PP. 273-282

[5] Baskerville, R., Information Systems Security Design Methods: Implications for Information Systems Development. ACM Computing Surveys, 1993.25(4): p. 375-414.

[6] Bishop, Computer Security: Art and Science, Addison-Wesley-Longman, Nov. 2002.

[7] Blakley, B., McDermott, E. and Geer, D., Information Security is Information Risk Management, Proceedings of the 2001 workshop on New security paradigms Sept.-01

[8] C.B. Haley, R. C. Laney, at1d B. Nuseibeh, "Deriving Security Requirements from Crosscutting Threat Description," Proceedings of the 3rd International Conference on Aspect-Oriented Software Development (AOSD '04) K. Lieberherr, Ed. Lancaster UK, ACM Press, 22-26 March 2004, pp. 112-121.

[9] Charles B. Haley, Robin C. Laney, Jonathan D. Moffett, Bashar Nuseibeh, "The Effect Trust Assumptions on the Elaboration of Security Requirements ".mcs.open.ac. uk/ban25/papers/re04 .haley .charles. pdf, 2004.

[10] Common criteria Toolbox Version 6, SPARTA, Inc. February 2003. Retrieved June 15, 2005. http://www.cctoolbox.sparta.com

[11] D. Moffett and B. Nuseibeh, "A Framework for Security Requirements Engineering". Technical Report YCS368. Depat1ment of Computer Science, University of York, York U.K. Aug 2003.

[12] Dimmock, N., Belokosztolszki, A., Eyers, D., Bacon, J. and Moody, K., Using Trust and Risk in Role-Based Access Control Policies, Proceedings of the ninth ACM symposium on Access control modelsand technologies, June 2004.

[13] Donald G. Firesmith. "Specifying Reusable Security Requirements", Journal of Object Technology. Vol 3, No.-l, Jan-Feb. 2004.

[14] Firesmith, "Engineering Security' Requirements", Journal of Object Technology, Vol 2, No.- 1, Jan-Feb 2003.

[15] Firesmith, "Security Use Cases". Journal of Object Technology, Vol.-2, No.-3, May-June 2003, pp. 53-64. [24].G. Sindre and A. I., Opdahl, "Eliciting Security Requirements with Misuse Case", Requirements Engineering 10, Springer-Verlag London Ltd., January 2005, pp. 34-44.

[16] G. McGraw, "Building Secure Software: A Difficult but Critical Step in Protecting Your Business," Cigital, White Paper, available at: http://www.cigital.com/whitepapers/.

[17] Grandison, T. and Sloman, M., A Survey of trust in internet applications, IEEE Communications Surveys and Tutorials 32000, pp. 2-16, 2004.

[18] Internet Engineering Task Force, RFC 2828, www.ietf.org

[19] ISO/lEC 9126-2, Software Engineering - Product Quality - Part-2 : External Metrics, 2000.

[20] J. McDermott, "Abuse-Case-Based Assurance Arguments", Proceedings of the 17th Computer Security Applications Conference (ACSAC'OI), New Orleans LA USA, IEEE Computer Society Press, 10-14 ,Dec 2001, pp. 366-374.

[21] Johan Peeters, Agile Security Requirement Engineering,johanpeeters.com/papers/abuser %20stories. pdf

[22] John Viega, "Building Security Requirements with CLASP", International Conference on Software Engineering, Proceedings of the 2005 workshop on Software engineering for secure systems, 2005, pp. 1-7 [12].Mannion, M. and Keepens, B. SMART Requirements. ACM SIGSOFT, SE Notes 20, 2 (Apr. 1995).

[23] John Wilander, Jens Gustavsson, "Security Requirements-A Field Study of Current

Practice", Symposium on Requirements Engineering for Information Security, August 29, 2005

[24] K. Hoo, A. Saudbury and A. Jaquith, "Tangible ROI through Secure Software Engineering, "Secure Business Quarterly, Q4, 2001

[25] L Bass, P Clements, R Kazman, "Software Architecture in Practice", Second edition, Sei Series in Software Engineering.

[26] Lin Liu. Eric Yu, and John Mylopoulos. "Security and Privacy Analysis within a Social Setting", Proceedings of the11th IEEE International Requirement Engineering Conference, 2003.

[27] Liu, L., Yu, E. Mylopoulos, J. "Analyzing Security Requirements as Relationship among Strategic Actors", 2nd Symposium on Requirements Engineering for Information Security (SREIS'02) Raleigh, North Carolina. October 16,2002.

[28] Luncheng Lin. Bashar Nuseibeh, Dan-el Ince, Michael Jackson, Jonathan Moffett, "Introducing Abuse Frames for Analyzing Security Requirements", Proceedings of the IEEE Intl. Requirements Engineering Conference 2003.

[29] M.U. Bokhari, Mahtab Alam, "Threat Model: A Path of Secure Design", Intl Journal of Technology and Applied Science, Vol-1, pp-4-10, 2010.

[30] Mahtab Alam, "Software Security Requirements Checklist", International Journal of Software Engineering, Vol-2, 2010, pp-58-68

[31] Mahtab Alam, M.U. Bokhari, "Information Security Policy Architecture", IEEE Computer Society 2007, pp-120-122.

[32] McDermott, "Abuse-case-based assurance arguments". Proceedings of the 17th Annual Computer Security Application Conference (ACSAC'O), New Orleans, Los Angeles, 2001.

[33] McDermott, 1.. Fox, C. "Using Abuse Case Models for Security Requirements Analysis", Proceedings 15th IEEE Security Applications Conference. 1999.

[34] McGraw, G., Managing Software Security Risks, IEEE Computer ,Vol. 35, Issue: 4, March 2002, pp.: 99 - 101. [47].Charles B. Haley, Robin C. Laney, Bashar Nuseibeh, "Deriving Security Requirements from Crosscutting Threat Description", ACM Press, March 2004

[35] Michael S. Ware, John B. Bowles, Carolines M. Eastman, "Using the Common Criteria to Elicit Security Requirements with Use Cases". Proceedings of the IEEE Southeast Conference, March 31- April 2. 2006.

[36] MiroslavKis, "Information Security Anti patterns in Software Requirements Engineering", PhD, CISSP, IEEE Transaction, 2002.

[37] Mylopoulos, J., Borgida. A., Jarke, M. and Koubaraka, M. Telos. "Representing Knowledge about Information System", ACM Transaction on Information Systems 8(4), October 1990.

[38] Nancy R.Meed, "How to Compare the Security Quality Requirements Engineering (SQUARE) Method with Other Methods", Technical Note, August 2007.

[39] OPEN Process Framework", Web Site: www.donald-firesmith.com

[40] P. Zave and M. Jackson, "Four Dark Corners of Requirements Engineering", Transaction on Software Engineering and Methodology (ACM). Vol. 6, Bo. 1, Jan 1997, pp. 1-35.

[41] Pankaj Goyal, "CS497- Security Engineering and Software Engineering" Website: www.cse.iitk.ac.in/report-repository/ - 2005/cs497_Y1240.pdf

[42] R. Crook, D. Ince, Luncheng Lin, Bashar Nuseibeh, "Security Requirements Engineering: When Anti-Requirements Hit the Fan", Proceedings of the IEEE Joint International Conference on Requirements Engineering (RE'02) Germany, 2002, IEEE CS Press.

[43] S. Fenton, Information Protection Systems". Ph.D, Thesis, University of Cambridge, England, 1973.

[44] Stoneburner, G., Grogen, A., Dering, A., Risk Management Guide for Information Technology Systems, National Institute for Standards and Technology, SP 800-30.

[45] Van Lamsweerde, "Elaborating Security Requirements by Construction of Intentional Anti-Models. "Proceedings of the 26th International Conference on Software

Engineering (ICSE'04),Edinburgh Scotland, 26-28 May 2004. pp. 148-157.

[46] Wagner, "Static Analysis and Computer Security: New Techniques for Software Assurance, PhD Thesis, and University of California. Berkeley, 2000.

[47] Yu, E. and Liu. L.. "Modeling Trust for System Design Using the i* Strategic Actors Framework", Springer-Verlag, 2000, pp. 175-194

[48] Zaid Dwaikat Francesco Parisi-Presicce, "Risky Trust: Risk-Based analysis of Software Systems", Proceedings of Software Engineering for Secure System-Building Trustworthy Applications (SESS'05), May 15-16.2005 [39].Viega, J., Kohno, T. and Potter, B., Trust (and Mistrust) in Secure Applications, Communications of the ACM, Feb 2001, Vol. 44, No.2.