A Descriptive Review On Software Security Requirements Using Open Source Software Projects

Subhash Chandra Jat, Dr. C. S. Lamba Dr. Vijay Singh Rathore

Abstract: Security Requirements Engineering (SRE) is a novel area of research in software engineering that requires early analysis of safety in the requirement phase. Software security is primarily part of the software development process view. Security integration in the first steps of software development is the newest environment & there is still a lack of real procedures to address security concerns in early phases of development. Several researchers work in this area, but there is a lack of consideration of security necessities. One of the non-functional necessities, which perform as restrictions on system functions, is security requirements. In order to understand existing security practices, as well as security vulnerability between them, OSS software security studies, have to be repeated. In this paper, we introduce a view of Software Security, Security Necessities, types and SRE, Open Source Software for developing propose.

Index Terms: Security, Software Security, Software Engineering, Open Source Software (OSS), Security Requirements Engineering.

1 INTRODUCTION

The information system requires several elements, such as people, hardware, software, data & networks. Every part has to be secure enough to ensure that the information system works smoothly & reliably. In relation to all other information systems components, software systems are one of the most important elements. The integration of security into software is a long-standing challenge to software development companies. Developing secure software is not an advantage but a requirement for software organization software is increasingly vulnerable to deliberate or accidental malicious in today's networked environment. The primary reason behind software security holes is the failure of the software development process to address security problems. Security is always treated asan afterthought in the process of software development as well as handled through necessary preventive measures after system development phases. By the very start of SDLC(software development life cycle) [1], security needs to be considered[2]. This phase of secure software development will start with security necessities called NFR (Non-Functional Requirements followed by a secure design & implementation, along with procedural code requirements [2]. Software security is a concept of software engineering hence this could work accurately and also work continuously with malicious attack, has recently concerned significant consideration since reactive network-based protection strategies, such as signature-based & firewalls anti-spyware, have shown to be ineffective in achieving secure software. Besides, it is necessary to fix code after releases are much expensive. To longer the security has solved into the expansion cycle, the more costly this turns out to be: after shipping the code, 1$ needed for solving a problem for the period of designing time rises to 60-100 $ for solving the same problem [3].

- *1Research Scholar, RCEW, Jaipur, RTU, Kota, India, PH-09414206706. E-mail:-subhashcjat@yahoo.com*
- *2Professor, RTU, Kota, India, PH-09928000040. E-mail:-professorlamba@gmail.com India*
- *3Professor, RTU, Kota, India, PH- 09414265839.E-mail:-vijaydiamond@gmail.com India*

Therefore, in recent time's software security raised as the main challenge [4] & more no. of additional expertise of security have required according to the requirements for non-functional security. Security, though, considers as seldom at the main concerned of investors, excluding possibly to meet fundamental standards/ authorized requirements. A good specification report for specifications should include all operational requirements (Related to the services to be provided by s/w or by the system) & NFRs (Non-functional requirements) (relevant to so-called quality, performance, portability, security, etc.).Open Source Software (OSS) is a special distribution and licensing software category. OSS systems have to be utilized in modern societies to increment the significant applications such as economics, healthcare, govt., fortifications, & also in other sectors that are susceptible to security. Software certification is a growing interest as a means of ensuring the quality and reliability of such systems. Nevertheless, OSS projects ' design processes and organizational structures may differ significantly from conventional closed-source projects.

2 REQUIREMENT AND POLICIES

Security is in the eye of the spectator, like beauty. A public library will certainly differ from a central clearing for interbank transactions with regard to computer security. Only after careful attention of business context, users ' preferences and/or defense posture can precise security needs of a specific installation be established. TCSEC [5] Glossary describes security policy as"... a collection of laws, rules & practices governing control, defense & distributing of sensitive information by organization." A security necessity is a representation of high-level corporate policy with respect to a comprehensive system in order to reflect current usage of security and software engineering research community, we will loosely misuse following word "security policy." Security policies are similar to standard and functional system requirements, such as customer's features they are a type of non-functional necessity, as are performance & reliability. Such an integral part of requirements engineering, preferred approaches for requirements engg., like cases of use generally do not incorporate security issues. Albeit during requirements engineering, some security issues are

addressed, the majority of security requirements only emerge after completion of functional requirements. As a follow-up to standard (functional) requirements security policies are added. In modern society, software is overwhelming, and we often do not know about it until it becomes difficult. Software is one of the most significant but one of the most cost-effective methods of this era. It is one of the most laborious, complicated & misunderstood inventions in human history as a purely intellectual material. Computer users are often subject to bloat code, which is ugly, ineffective and poorly designed, leading to software failure. Good software that is totally dependable is quite difficult to find. Good software is usable, confident, defective, economic & manageable software. Truth is that most software sucks. Dijkstra [6] notes that “the average computer user has been served so poorly that he expects his system to crash all the time, and we witness a massive worldwide distribution of bug-ridden software for which we should be deeply ashamed.”

3 SOFTWARE SECURITY

The software security is software engineering so that the ideal device is up to control security threats through malicious attacks in an uninterrupted manner. Security provides a desired software functions and protects against security threats. The software with world-class security is not everything that makes its software secure, it is equipped with technology-based defenses and it does not have a well-trained security team. In fact, it is essential to secure software and that is called' conscience' among people who are involved with the collection, design, coding, testing, implementation, and use of software [7].

Attributes of a Good Software

- **Functionality:**

Good software must provide users with essential functionality as well as performance also be secure, acceptable& reliable.

- **Maintainability:**

Good software must be easy to manage & improve so that evolving requirements can be fulfilled.

- **Dependability:**

Software should be reliable. It must be reliable, safe, &secure.

- **Efficiency:**

It must not create inefficient usage of system resources

- **Acceptability:**

Users for which it is intended must accept software. It should, therefore, be usable, understandable, & other systems compatible.

- **Survivability:**

Effective software must be capable of supporting & adapting to any environment it can be used.

4 OPEN-SOURCE SOFTWARE SYSTEM

Many people said that Open-Source software is inherently secure than Closed Source, others said it is not.2 both of them are absolutely true: they are simply flip sides of the same coin. Open source provides more computational resources to attackers & supporters to address software vulnerabilities. Nevertheless, open-source only gives attacker advantage if a defender does nothing about security but, open source also gives defender great advantages, providing access to security technology that is usually impossible with closed source code. Closed source allows users to embrace the vendor's level of security vigilance while open source lets users or other collectives raise security bar to standard they want. This article examines improvements in the security of open-source software. Software security not network security, (which is managed by line protocols and is therefore not influenced by whether other network components are open-source), is what we'll concentrate on. All solutions we consider are applicable to open source systems, but they cannot be fully open source. There are several meanings of the term OSS, as"software which can freely distribute with available source code through the internet and which can freely change code using unpaid persons, is open source software"3.

Following features are described by certain definitions 4:

- Free software
- Distributed software
- Developers communicate through the internet
- Available source code
- Unpaid and large amount volunteers
- Developers are users

There are 4 sets of OSS developers:

- Co-developers
- Core developers
- Usersf
- Bug reporters

Core developers are a small group of experienced developers liable to key system features by writing high-quality codes; manage & control system and draw up development plans, priorities & roadmaps [8] The developers involved are a large group of developers who direct affect development of software you can add new features and other activities such as fixing bugs or evaluating software by peers (based on application modularity). The system is tested by bug reporters. contributing developers and core developers, and system users that do some of this task One of the important tasks of this group is towards making sure that more people test systems on various platforms (if this is enabled by the system). The system is used by users; some are developers. A simple onion model as given in figure 1 can describe the sustainability of software development community groups.

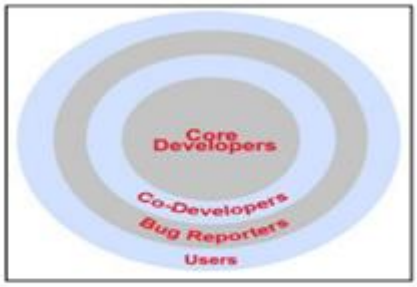

Fig. 1. The onion model of a sustainable software development community [9].

5 SECURITY REQUIREMENTS ENGINEERING

In all areas of human society such as banking, transport, telecommunications, entertainment, military services, education, and healthcare, software systems become more and most difficult; The list goes on & on. These systems are used by individual users and organizational networks as well as corporations & governments. This extensive use of these systems led to the presence of a large no. of critical as well as secure information & processes. It is also important for software systems to be developed as user needs and for systems to be secure. But a common method of integrating security into a software system is, after system specifications, to identify security requirements. That generally means that security enforcement mechanisms in a previous design should be implemented. This leads to serious design challenges that normally lead to the emergence of highly vulnerable computer systems [10]. From the viewpoint of the conventional security model, recent researchers argued that convergence of protection & requirements technology could reduce these problems. From the beginning, security requirements (SR)must be considered & specified in accordance with system requirements specifications. Together with functional requirements, taking security requirements into account helps to reduce the causes of conflict among security / functional requirements by preventing or separating them early on in the computer system procedure. SR Engineering process is to generate, specify as well as analyze SR to fundamental system thoughts such as "what" are SR, with a view to preventing harm in the real-world &consider them as restrictions on functional needs. Several techniques that promote such evaluation of requirements and design of security requirements have been developed. Major goal of this paper is to give a summary & comparison of different Security Requirements Engineering (SRE) approaches. In this section, important activities to security engineering also the use of SR in later phases of SDLC have been analyzed & discussed.

Main activities for SRE

Activities to every early development phase of requirement engineering should be specific to each SRE method. SR must be considered functional as well as following key activities included in the Security Requirements Engineering phase:

a) Identify assets, threats as well as vulnerabilities.
b) Threat modeling to show potential software threats so as to specify suitable security features & mechanisms.
c) Risk analysis in order to prioritize accepted SR & security evaluation in order to assess the security status of specification requirements [10].
d) Design of SR by the terminology of design or simulation for the elimination of defined errors.
e) Requirements Analysis &Check Requirements to define security errors by use of a checklist of possible SR.[11]

6 SECURITY REQUIREMENTS TYPES

In order to build information security, developers should have knowledge about various types of security requirements. It can be reduced to rely on security experts. This section addresses the form of security requirements found in [12] and how they are important to creating a secure & stable software application.

Analysis & visualization of data

- Giving feedback to supportive instructors
- Recommendations to students
- Forecasting student performance

A. Identification requirements

AnSRrequires the extent to which an organization, application component/center must recognize these external elements (for example, human actors & external applications) prior to interacting. [9]

B. Authentication requirements

It's a security requirement specifies to what extent application, business, component/center must validate the identity of these external parties before interacting with them (for example, human actors, as well as external applications). Traditional purposes of a prerequisite for authentication are to ensure that externals are who or what they claim to be as well as to prevent cooperating protection of an applicant.

C. Authorization requirements

It is a security prerequisite that sets out authenticated users and server applications ' access and use privileges.

D. Immunity requirements

It is a security requirement to specify to what degree an application/component should prevent unauthorized undesirable programs (for example computer worms, viruses, and Trojan horses) from being infected. The traditional purpose of an immunity requirement is to stop the destruction or degradation of information as well as applications by any undesirable program

E. Integrity requirements

It is SR to ensure that application or component does not deliberately compromise information or interaction through unauthorized development, alteration or removal of application/component. The typical aim of an integrity requirement is to ensure trust in communication& information.

F. Intrusion detection requirements

It is a security requirement to detect & record tentative access or modification through unauthorized people in respect of an application or component.

G. Non-repudiation requirements

It is anSR that sets out to what extent an organization, application/component prevents a party by refusing any participation in entire or part of the interaction (e.g. transaction, message).

H. Privacy requirements

It specifies the extent to which sensitive data & communication from unauthorized individuals or programs are kept by an organization, application, part or center in the private sector.

I. Security auditing requirements

It is anSR that designates how far security personnel may audit status and utilization of their security mechanisms through a firm, application, component or center.

J. Survivability requirements

It is a security requirement that stipulates the extent of intentional loss or destruction of a component by an application or facility. Typical objective is to ensure that an application or center each gracefully fails or remains functional (probably in degraded mode) even if some modules are deliberately damaged or destroyed.

K. System maintenance

It is a security requirement that determines how much-approved modifications (for example defect fixes, updates, &enhancements) of an application, part or center unintentionally fail to comply with its security mechanisms.

L. Physical protection requirements

It requires the extent to which an application or a center protects against physical attacks. Application or center is secured from physical damage, degradation, theft/replacing equipment, software, or staff components from vandalism, sabotage or terrorism as a standard goal of physical safety requirements.

7 LITERATURE SURVEY

Manal Binkhonain and Liping Zhao [2019] report of 24 methods ML-based to define and classify NFRs. The article is motivated by three research questions and aims at explaining what have ML algorithms used in these types of methods, how they have the operation performed on these algorithms and also describe that how are they performing. In these methods, sixteen diverse ML algo's have set up; most common are supervised learning algorithms. All 24 approaches followed a generic NFR identification and classification method. Mostly common usable metrics for measuring the performance of these all methods are precision and recall. The analysis notes that while ML-based strategies can define and recognize NFRs, they are faced with some of the common issues that would impact its quality & practice. Review calls for close cooperation among RE and ML researchers to resolve open challenges facing real-world ML systems development. ML used inside RE offers stimulating chances for developing new intelligent & smart systems to help RE activities & processes. It means that RE has turned in a traditional expert systems application [13]. Nan Niu and Daniel Mendez Fern´ andez [2019]activities of agile & open source software projects aim towards being just in time, as opposed to traditional initial RE, categorizedthrough lightweight representation & continual improvement of demand. Exceptional problem includes 6 papers, in the range from theoretical views to experimental learning via logical reasoning. The set provides significant & appropriate assistance to advance RE perceptive in the application of manufacturing knowledge and Industry 4.0 [14]. Celia Chen et. al. [2018]We are investigating how maintainability is demonstrated with reparability and modifiability subsets by testing over 6000 actual bug findings from several Mozilla products. We also study manually how qualities change as products mature, what are root causes of bugs toevery quality and how each quality affects & depends on it. Our results indicate areas to be addressed in order to guarantee maintenance at various stages of development as well as maintenance process [15] S. Pandey et. al.[2018] The main objective of the project is to give scientists & developers, also studying science, technology, engineering, mathematics, as well as a variety of other orders access to spaces for small payloads which can be planned and executed for a huge number of planned missions. Every system is designed and developed by specification technology as it defines the system's functional and non-functional specifications. The research discussed in this paper focuses on the design of a requirement engineering tool that can help to elicit specifications and to determine system-wide values based on scenarios of performance. The requirement technology method described in the paper encourages non-functional requirements & functional needs to be related. It uses examples of performance attributes that Carnegie Mellon Software Engineering Institute (SEI) is proposing [16] Tanmay Bhowmik and Anh Quoc Do [2018]describe a novel system of modification & declaration which met whole criteria reviewed from its inception to completion and successive releases. Noting some of the motivating phenomena that are extra relevant to NFRs, they are conducting a 2nd study that explores 50 NFRs out of given 3 subjective system described above. They reveal a little diverse method into refining & addressing NFRs. They give additional confirmation that in the systems analyzed, NFR modification & declaration actions are added implementation based modification in compare to specification creation. Our work gives novel nearby into how is he OSS project teams build performance quality as of basic early JIT specifications explanations, expose much detailed JIT Requirement Engineering (RE) activities for an NFRs, and also starts some latest opportunity for more research in the field of evolving JIT RE [17].

8 CONCLUSION

In this paper, we examined different aspects of software security that are critical to normal everyday life regarding the security of software-controlled systems. Software engineers face a relentless & challenging task to develop these systems timely, efficiently & correctly. Its requirements & policies and its kinds of security requirements. This paper describes some of the important open problems that software engineering researchers face

9 REFERENCES

[1] Wang, Huaiqing, And Chen Wang. Taxonomy Of Security Considerations And Software Quality. Communications Of The Acm 46.6 (2003): 75-78

[2] Devanbu, Premkumar T., And Stuart Stubblebine. Software Engineering For Security: A Roadmap. Proceedings Of The Conference On The Future Of Software Engineering. Acm, 2000.

[3] .S. Hoo, A.W. Sudbury, and A.R. Jaquith, "Tangible ROI Through Secure Software Engineering", Secure Business Quarterly, vol.1(2), 2001.

[4] M. Zulkernine, S.I. Ahamed, Software security engineering: toward unifying software engineering and security engineering, in M. Warkentin, R.B. Vaughn (Eds.), Enterprise Information Systems Assurance and System Security, Idea Group Publishing, 2006.

[5] I. Jacobson, M. Griss, and P. Jonsson. Software Reuse: Architecture Process and Organization for Business Success. Addison Wesley, 1997.

[6] Rothenberger, M.A., Dorley, K.J., Kulkarin, U.R. and Nada, N. (2003) Strategies for Software Reuse: A Practical Component Analysis of Reuse Practices. IEEE

Transactions on Software Engineering, 29, 825-837.
[7] Mellado, D., Blanco, C., Sánchez, L. E., & Fernández-Medina, E. (2010). A systematic review of security requirements engineering. Computer Standards & Interfaces, 32(4), 153–165.
[8] J. J. Heiss. The meanings and motivations of open-source communities. Aug 2007, from Oracle, http://www.oracle.com/technetwork/articles/java/opensource-phipps-137190.html
[9] Mark A. Achieving Quality in Open Source Software. IEEE.Jan-Feb 2007. p. 58-64
[10] Viega J, McGraw G. Building secure software. Addison-Wesley; 2001.
[11] Apvrille A, Pourzandi M. Secure software development by example. IEEE Secur Privacy 2005;3(4):10–7.
[12] Salini, P., & Kanmani, S. (2012). Survey and analysis of Security Requirements Engineering. Computers & Electrical Engineering, 38(6), 1785–1797.
[13] Binkhonain, M., & Zhao, L. (2019). A review of machine learning algorithms for identification and classification of non-functional requirements. Expert Systems with Applications: X, 100001, pp. 1-13.
[14] Nan Niu, Daniel Mendez Fern ´ andez, Special Issue on Just-in-Time Requirements Engineering for Software Integration, Journal of Industrial Information Integration, Volume 14, June 2019, Pages 1-2.
[15] C. Chen, S. Lin, M. Shoga, Q. Wang and B. Boehm, "How Do Defects Hurt Qualities? An Empirical Study on Characterizing a Software Maintainability Ontology in Open Source Software," 2018 IEEE International Conference on Software Quality, Reliability and Security (QRS), Lisbon, 2018, pp. 226-237.
[16] S. Pandey, S. Pokharel and H. Reza, "Towards Cyber-Physical Requirement Engineering Elicitation Tool Support," 2018 World Automation Congress (WAC), Stevenson, WA, 2018, pp. 1-5.
[17] Bhowmik, T., & Do, A. Q. (2018). Refinement and resolution of just-in-time requirements in open source software and a closer look into non-functional requirements. Journal of Industrial Information Integration, Volume 14, pp. 24-33.