Journal Homepage: - www.journalijar.com
INTERNATIONAL JOURNAL OF ADVANCED RESEARCH (IJAR)
Article DOI: 10.21474/IJAR01/5904
DOI URL: http://dx.doi.org/10.21474/IJAR01/5904

RESEARCH ARTICLE

ON THE SECURITY OF OPEN SOURCE SOFTWARE.

Prattay Sanyal[1], Shubham Sharma[1], DeepaBura[2] and Prasenjit Banerjee[2].
1. Department of Computer Science and Engineering, ManavRachna International University, Faridabad, India.
2. Department of Computer Science, Accendere Knowledge Management System Pvt. Ltd., New Delhi, India.

Manuscript Info

Manuscript History

Received: 18 September 2017
Final Accepted: 20 October 2017
Published: November 2017

Key words:-
open source software, code review, computer security, code review, security vulnerabilities.

Abstract

Nowadays, the usage of Open source software (OSS) is becoming more popular along with its flaws and benefits. By using OSS constantly, it provides several aspects of the internet's infrastructure. There are several commerce-based research questions which may improve the quality of the OSS related to the future of the internet. This software matches with the advantages of OSS in comparison with the key attributes in tomorrow's network that will need mainly in terms of security. OSS presents few arguments which are beneficial for the open source security. This represents qualitative evidence by which the security issues are getting concerned. It surrounds the development and requirement of OSS. The OSS is particularly related to the software that is proprietary. It allows several rights to the user for further redistribution and modification of the source code. In this paper, we have highlighted various benefits of the usage of open source software. We have also mentioned some security concerning issues that allows an easy prey for an attacker to modify and perform some malicious activities by using the software and the risks associated with it.

Introduction:-

Nowadays, we observe that individuals and businesses are strongly dependent on software system and networks. Now with the rising dependency on the open source software, there is a great way of maintaining control over any infrastructure. Generally, any computer security software is designed to enhance the information security of the system. Instead of this security, businesses are getting struck that are not able to rely on the information systems which make them dependent for their survivals. The customers are lacking their belief regarding the security of transactions that was one of the major factors in the place of growth of the internet. The source code which we have manually developed can be secured easily, but the source code collecting from the internet cannot be secured. Though the internet collected programs are freely available, that can build the infrastructure of a secured software. OSS is a free software which contains programs where the source code is easily available.

This software is based on the concept of allowing certain regulations, before providing freedoms to the user. This software will allow the right to every user for accessing the source code in editing mode and promoting redistribution of the software. Then sell the software either as a part of another product or on its own. Generally, OSS does not allow the rights of modification for which the software may be used. OSS has shown its users to be very much secured as compared to the proprietary software from large vendors. While downloading the software,

Corresponding Author:- Prattay Sanyal.
Address:- Department of Computer Science and Engineering, ManavRachna International University, Faridabad, India.

we have to take some precautions to prevent it from copying. We can easily do it by checking the digital signature for some distributors to add on their sites. OSS provides many tools that are required for running a system. A lot of success in terms of money is achieved by utilizing computers that cannot be gained by selling the software. This achievement comes by utilizing software in order to sell or produce goods and services. Now OSS has proved to be much secure as compared to the commercial software. We should be keen in order to verify the signature and the source code for the open source before we download it.

So, we can say that Open Source Software is much better than closed software in terms of vulnerability concerning security. No OSS have caliber for becoming more secure as compared to the closed source. But the open source may face some security issues. Another problem is that, though the source codes are freely available through the open source software, we do not have any right to modify or redistribute the source code. Open source is not considered as a 'free source' instead of being the source code as 'free'. In OSS, there are some different licenses that actually means as a free software but although they have different terms. The two common licenses come under the Berkley Software Distribution (BSD) and General Public License (GPL) [16]. There is no any universal standard that is determining any particular license which is referred as open software. The Debian Free Software Guidelines (DFSG) and Open Source Definition (OSD) [52] represent the sets of criteria that are commonly accepted. The contrast of open source software is very simple. This is mean by licensing traditionally the software that is referred to as 'binary' or 'proprietary-only', it is usually lacking the source code. But a closed source is opposite in meaning as compared to open source. Trojan Horses and other malicious code can severely affect and can cause damage to our systems. Trojan Horses may be included into proprietary software. Trojan horses can breach the system and have the capability to corrupt all the files present in a computer system. An Adware is a software that contains advertisements inserted in the application. Adware is another way for offering the consumers who don't want to give money to the software. There are few ad-supported programs, utilities or games are distributed for adware (or freeware). They are used for creating unnecessary advertisements.

Spyware is software which has the ability to take the information about an organization or a person instead of knowing their permission. Spyware always sends that information to some another person without the persons knowledge. This attains control on any device instead of knowing the user knowledge [1].

We have to differentiate between the exposure of the system, the risk associated while using the system and security of the system. We can say that risk is a combination of any successful attack on a system that is harmful to the assets in it.

Any exposure to the system completely denies its damage that is occurred by any successful attack. This can also be defined as any characteristic of a successful attack. We have to check whether the attackers know the vulnerabilities. Whether the system is a high-profile target or not or will it exploits any vulnerability.

Open source software can easily provide the source code for use and its inspection by any user. Open source software is something in which we will need to construct systems more strong from the attackers. When we open the source code, it allows us for assessment of the disclosure of any system and the hazards involved while using the system. The bugs are patched easily and are developing the quality of code. We need to keep the source code protected by preventing easier access to the data that can easily launch an attack successfully. When the source code is opened, it gives an attacker the information for searching the bugs and vulnerabilities. This raises the exposure of any system. There lies a large difference between the openness of the source and openness of the design. The design is opened so it reveals some logical errors in terms of security which is the worst case. Bugs will be present in the source code, it will not matter how we will verify or test it. Attackers, on the other side, will scrutinize a source. The OSS development can be referred as a core of the system that is developed by any team of workers or a single worker. When any system on the internet is released due to many programmers can redistribute, read and modify the source code of any system. Open source has produced some great products like the Apache Web Server, the Linux Operating System, and the Perl Language. Recently an open source project for Mozilla was launched by Netscape. This is the latest web browser and it provides that there is a serious development of large-scale commercial software by the Open source. The open source presents an important challenge for the traditional software industry. The persons of the community of open source say that it is a wonderful process that can be referred as the joined expertise of a large number of coders who can produce good software as compared with the model which is closed. Here only a limited number of programmers of a single development team have rights and access to the code that we get from the source. The persons who are engaged in the projects of open source are motivated very much as the

programmers make software for their satisfaction and these programmers expect it to be very much successful. As there are some concerns that are related to the development philosophy for the open source can definitely produce better quality software systems. There is a problem where the development procedure is not explained for the open source. Some critical activities for development, such as system testing and documentation are ignored. These needs were explained themselves by the workers. Here debugging activities and coding are indulged with more efforts. The principles which were released earlier are those they gave many eyeballs and all bugs are shallow. The Logiscope is providing their standard of programming after the result of conclusions which occur in later time by investigating billions of lines of programming code. An application was used by some big departments for controlling the procedure of programming. The National Security Agency (NSA) in the US has told that after 3 years of measurement and quality assurance activities after that NSA had analyzed and reported the results from 25 billion lines of source code. Then the software is made which is larger than one million lines of source code. Then NSA had reported some procedures that can be applied in our study. Now NSA is using for enhancing the processes for good quality software where the only level of coding is considered. The program quality was explained to accept these values. For measuring the component quality, the report from these programs was taken. By characterizing the software quality ISO standard is used for determining the software. The OSS must be tested for allowing its fast evolution. It should be easier for allowing subsequent extension and changes to the software. The open source code should describe itself and also be readable in order for performing the actions. Modularity is considered as an essentially open source code feature. Extensive study has involved a large number of products for the open source in various programming languages, the degree of success, growth in success among the programmers and application domains. Some programmers had also found a risk that OSS is having a greater possibility of delivering a code which is difficult to read. The source code is difficult to maintain and can be of bad quality. Programmers have seen these OSS projects are managing for their survival due to many users, for their own interest and are able to correct bugs and provide add-ons. The community of OSS must take it seriously in order for developing better quality code. If every user is developing their style of coding, then they are not allowed for changing the standard of coding. Examples of Open Source Software are:- Apache HTTP Server, Moodle, Firefox, MySQL, OpenOffice.org, PHP, FreeBSD etc.

Related Work:-

Garfinkel [5] in 1999 used the software called as 'TCP Wrapper' that was hacked and the hackers had changed the software code in order to include a back door. Then the code was malicious and modified in one night. Gross [7] worked in 2000. Levy [10] in 2000 was searching the code which can be easily obtained by searching the bugs for security. McAllister and Lettice [11] in 2001 had preferred Linux as a platform for computing due to some consequences. Moody [12] in 1997 fixed patches available for Linux and BSD based operating system. NSA Security-Enhanced Linux [16] in 2000. Raymond [20] in the year 2000 found that all bugs are shallow. Schneier [22] in 2000 found that when any bug is left unfixed, it is most likely to damage the system. For verifying the source code Simpson [24] in the year 1999 using the cryptographic software i.e, PGP. Viega [28] worked in 2000. Lerner and Tirole [30] in 2002 had studied that the utility from OSS is independent and non-exclusive. Lerner and Tirole [30] in 2002 had joined to give their first contribution to the software. Lerner and Tirole [30] in 2000 had proposed the links between the incentives of OSS. Moody [32] in 2001, Raymond [33] in 1999, and Wayner [67] in 2000 had contributed to open source software to the public by introducing the code for a software project. Raymond [33] had given a marvelous meaning and theory of processes used for working in open source. Von Hippel [34] had collectively contributed to the improvement of the software development. Boehm [35] in 1988 and Bollinger [79] in 1999 had found that open source code should be modular and be able to explain itself, for allowing growth at backward sites. Fenton and Pfleeger [36] in 1997 had studied the structural metrics to measure the component quality. Hatton [81] in 1997 said that for a system software, smaller components are less reliable than larger components and it was also observed by Fenton and Neil [37] in 1999. Godfrey and Tu [38] in 2000 had examined the evolution rate of Linux kernel. McConnell [42] in 1999 had found that the OSS process of development is not explained properly. Mockus [43] in 2000 had studied productivity, defeat density, developer participation and core team size. Mockus [43] in 2002 had contributed a code which can be identified uniquely which is based on the procedure of tallying e-mail addresses. O'Reilly [44] and Wilson [48] in 1999 had discussed the benefits and flaws of OSS. Pighin and Zamolo [45] in 1997 had studied statistically 350000 lines of source code written in C. Alhazmi, Malaiya and Ray [49] in 2005 had assumed that after the process of finding vulnerability it should be divided into 3 phases. Anderson [50] in 2001 had said that there are some bugs that are very critical in terms of security which further results to cause higher vulnerabilities in Windows 2000. The OSI [52] in 2006 had provided some specific criteria so that the software can attain it for giving the permission to redistribute and change the code. Ozment [54] in 2005 had found that the vulnerabilities are being correlated. Rescorla [56] had argued not in favor of openness

until the damages are related to each other. Schwarz and Takhteyev [57] in 2008 had provided the contents into the development and history of OSS. Von Hippel [58] in 1998 had studied that the area for innovation has turned towards programmers. Lawrence, Edwards [59] in 1998 and Fielding, Torvalds in 1999 worked for the production of high-quality code. Auguste Kerckhoffs [60] in 1883 had argued, for achieving strong military system …they should not need any of software. Pliskin [61] in 1991, in 1997 Waterson [62], in 1997 Kemerer [62] and Fichman [62] had seen that in complex technologies can stand well for software development and related hurdles for contributing and understanding for both the developers and users of the software. Fichman and Kemerer [62] in 1997 proposed there is a contribution barrier which was erected by OSS technologies. Kohanski [63] in 1998 had found that it was expensive for the new contributors for joining the project. Kohanski [64] in 2000 had reported that OSS had bugs. Clarke's master thesis [65] in 1999 contained a decentralized distributed storage information and retrieval system. Baldwin and Clarke [66] in 2000 had studied that modularization increases the transparency of a project for any software code and efficient use of knowledge. In 1986 Callhoun [68], Taylor and Singleton [69] in 1993 had shared their contributions and innovations to the software. Meyer and Seliger [70] in 1998 had worked on increasing the efficiency of the software. Grant [71] in 1996 and Simon [72] in 1991 had worked in certain areas of development. Glaser and Strauss [73] in 1967, Meyers [74] in 1997, Strauss and Corbin [75] in 1990 had studied that OSS process of innovation had requested an approach for developing propositions and analytical processes. Oram [76] in 2000 had studied that workers in the community of Freenet are busy for innovation and development in the (DFS) Distributed Filesystem Software. Stake [77] in 1995 and Yin [78] in 1994 had studied the specialization of Freenet. Fenton [80] in 1995 had worked on the ISO standard which was implemented by many companies in the industry of software. Feller [82] in 2000 had produced a framework for research then analyzed the procedure. In 2001 Hars [83] had studied that Open Source developers and had given for the involvement in their works. In 2002 Scacchi [84] had extensively studied the technical and social process that is related to Open Source development practices. In 2002 Wolf [85] has released the outcomes from a survey of 526 Open Source developers from Source Forge and reported to OSS developers. Watts [86] in 1999 had claimed that in the network of open source it requires 25% of developers at Source Forge. Strauss and Corbin [87] in 1990 had the development list for e-mails. Jorgenson [88] in 1989 had studied the modifications on the source code. Banker [89] in 1994 had found non-linear effects in OSS. Kemerer and Slaughter [90] in 1999 noted that there are challenging changes to the software architecture. Glass [91] in 1992 had discussed the alleged innovation in software distribution. Wolf [92] in 2002 had studied and said that it is essential for measuring the privacy of the software – whether it is closed source or open source software. Csikszentmihalyi [93] [94] [95] in 1975, 1990 and 1996 had explored the work and tasks regarding with the software. Kollock [96] in 1999 had discussed 4 motivations for contributing public goods.

Challenges faced by software Systems and their protection:-

The software faces several challenges during its software development life cycle, these are mentioned in this section:-

Malicious Code:- It is used for describing the code in any part of a software system which can definitely produce some undesired effects, security damage to a system and security breaches. It is an application security threat that cannot be controlled effectively by any standalone antivirus software. A back door is a malicious code.

Bugs:- A bug is a flaw, failure, error or mistake in a system or program that can produce anunexpected or wrong result. They also behave in differentways. There is problem in causing a code to crash or produce invalid outputs.

Open-Source Software:-It issoftware where the source code is available with a license where the copyright carrier gives the right to change, distribute for studying the software to everyone and for some purposes. OSS can be developed in many terms.

Cryptography:- We are presently staying in the information age. Data transmission is veryimportant and it is very necessary in today's world. How we exchangeprivate messages,commercial secrets or business. We should keep away our information which is confidential from our boss,hackers, and secret processes using cryptography. It is shell integrated so these encryption operations are easier.

Black Box:- Black Box is a testing that is done without first knowledge, just without knowing.

Hacker:- They can be novice or expert, bad or good. A person who can trick or exploit a system.

Internet Presence:- It is the thin veil which separates information, systems and services between the internet and a network.

Invasive:- It is a procedure of attaching, probing by trespassing to the persons who are non public parts of a network or system.

Passive:- It is the collection of data not only by attaching or probing to the persons who are non-public of any network or system.

Red Team:- It is someone or a group of people who are conducting anethical hacking engagement or a black box penetration test.

White Box:- It is a testing which can be completed with sufficient knowledge for getting the source code of a program during testing.

Vulnerability:- It is a problem that allows a hacker to suppress the information of any system information assurance. It is the intersection of three elements: a system susceptibility or flaw, an attacker can access the flaw and an attacker has the capability to exploit the flaw.

Firewalls:- It is called as a device which works on network and stops some specific kind of network traffic.It forms a barrier between a trusted and a corrupt network. Firewall also blocks the spread of computer attacks. It checks the flow of every incoming and outgoing packets possibly across networks.

Redundancy:- It is a system design in which a component is duplicated if it fails there will be a backup. Redundancy helps in security of storage of memory in our computer. It actually refers to replication.

Intrusion Detection System(IDS):- IDS is a software application or device thatcan monitor a system or network for malicious activity or policy violations. Any detected activity or violation is typically reported either to an administrator or collected centrally using a security information and event management system(SIEM).

Intrusion Prevention System(IPS):- IPS is a threat prevention technology and network security which examines the network traffic flows to detect and prevent vulnerability exploits. It provides a complementary layer of analysis that negatively selects for dangerous contents.

Virtual Private Network(VPN):- A VPN is used by a public network that is the Internet for enabling its remote users and the sites to be securely connected to the modern network.

Ethical Hacker:- It is a networking expert or a computer who can systematically attempts to penetrate a computer system or network on behalf of its owners for the purpose of finding security vulnerabilities that a malicious hacker could potentially exploit. An ethical hacking is a technique by which a person has the knowledge of hacking but he will not focus on hacking systems but he will focus on how to secure systems.

Trojan horse:- A Trojan horse is a program which will appear that it will not produce any harm but it's a malicious computer program that can be used to hack into a computer by misleading users of its true intent. It is a malicious code which eats away all the data of any computer.

Viruses and Worms:- Viruses are similar to worms by which worms can make functional copies of their own and can create the similar type of damage. Viruses always spread the host file that is infected, they occur only with human intervention. Worms are software which is standalone and they do not need any human help or host program to propagate.

Challenges Faced By Open Source Software Systems:-

This section describes various challenges faced by open source software systems and its possible solutions, some of these are mentioned in Table 1.

Modification in the source code:- If some attacker studies the source code and make some changes on his own then it will be difficult for any person to work on it. There will be no privacy left at that time.

Inserting a malicious code in software:- If any hacker inserts the malicious code in the software from a back door then no one can identify that the source code is modified and it is being used for malicious purposes.

Cloud Computing:- As we all know that everything is virtually migrated to the cloud. We know that Open source software powers the cloud but the cloud computing architecture restricts its users the freedom they get from Open source software. As Cloud Computing has a strong privacy which the open source lacks.

The Internet of Things (IOT) :- IOT has several challenges for the Open Source Software despite of being similar to the features of cloud.

Corporate Control:- As we know that the companies are investing very much for their development. But this change is not so much in the corporate control over open source code.

Apple:- Open source code is published by Apple. But Apple's products are closed and are super-proprietary.

Security:- Developers have made the source code to reuse it again and again which can make their work easier. But due to insertion of malicious codes it has become vulnerable for us. So we can say that the security is lacking in open source software.

Updates:- Most of the open source software allows the user to install the updates bythemselves. But there are some different versions of the software to download the same application that will require some different versions of the same software which will lead to compatibility issues, dissatisfaction and lack of performance.

License Issuing:- There are very popular GPL License that requires the source code to be distributed only under GPL license which will become a problematic situation for the enterprises.

Too many contributors:- There are many developers who are contributing to open source software which is generally good but this success is having an issue. As due to more software are attracting more contributors so it is increasingly difficult to keep a track of what is happening with the source code and to ensure that the level of quality remains good.

Table 1:- Challenges of Open Sources Software Security

S.No	Author Name	Challenges of open source software	Possible Solutions
1	Raymond(2000)	Open Source software is a big flaw for bug problems which is very challenging for a developer.	Programmers have searched for bugs and by the bug report the developer can fix this problem in the code. All bugs are shallow. By this the quality of software increases.
2	Garfinkel(1999)	The TCP wrapper s/w which was contained by the FTP sitewas hacked also the hackers changed the OSS code to include a back door.	Then the code containing error was invented and was rectified in a day.
3	Moody(1997)	Many systems over Internet were crashed and attacked by malicious users.	By releasing appropriate patches for the software, bugs were possibly fixed for the Linux and BSD Operating System.
4	Schneier(2000a) Schneier(2000b)	Vulnerabilities are found in the Phase 1 of Window Exposure. This became very harder for users to cope up with it.	These vulnerabilities were then rectified by studying the patches in the code.
5	Viega(2000) Garfinkel(1999)	In the GNU Mailman Program the bugs were cited and were	The Sentmail STMP security flaws were corrected and improved after a company

		for three years instead of correcting it. The STMP server had flaws	developed a commercial version of the software.
6	Payne(1999)	Security flaws and malicious code was inserted in the Open BSD, Sun Microsystems, Debian GNU software. Due to which it was very tedious to work on it.	The three Unix based Operating Systems were studied carefully and it was corrected by the programmers and then a new version of the software were released after a quantitative assessment.
7	Pfleeger(1997) Garfinkel(1996) Spafford(1996)	The logging mechanism hasbecame a tough task to maintain security of the users.	It was prevented by keeping the confidential data secured by using the cryptography and other file protection mechanism by the U.S Department of Defence(DOD).
8	McConnell(1999)	The open source development procedure was not defined properly. Due to this reason there are a number of software bugs which reduces the quality of the software.	It was further corrected and then new modified versions of the software were released. Further there were new development actions which are debugging and coding, the companies are more dedicated to them instead of system testing and documentation.
9	Bollinger et al.,(1999)	The most important requirement for open source code is that it should be self explanatory, modular and should be self contained.	This need was fully fulfilled by the big organizations of these open source software by which the user can easily access and can detect whether there are any flaws in the source code or not.
10	Pighin(1997) Zamolo(1997)	They statistically studied and analyzed approximately 350000 lines of code to study the characteristics of the code and they found some loopholes in them.	These loopholes in the source code by the assessment of statistically analyzing the source code it gave the users to rectify the code and then producing proper form of bug free software to the industry.
11	Drake(1996)	There were then also some traces of bug reports which were reported by the programmers to change the source code.	Then the US National Security Agency(NSA) had reported after 3 long years of quality assurance activities and measurement , they analyzed the results from more than 100000000 lines of code and they produced a high quality software totally bug free.
12	Fenton et al.,(1995)	There were some standards which were lacking by the open source software due to which it was becoming very easy for an attacker to replicate the source code and then to modify it for malicious purposes.	These standards were given by the International Standards Organization(ISO) in 1991 which includes four criteria they are:- simplicity, testability, readability and ability to describe on its own.
13	Fenton(1997) Pfleeger(1997)	The quality of the software was degraded and was not as it was required. It created bugs in the source code which became very easy for an hacker to hack the source code.	This disadvantage was corrected by using the Logiscope by the Telelogic in 2000. In which user defined programming standards were introduced. The data from this source code were collected and it was given 10 metrics for measuring the component quality.
14	Boehm(1998)	There was a lack of software development process for some time being due to some	Then by the intensive programmers and Boehm developed the software for further development in the software industry and

		reasons. Due to all of these reasons it was becoming increasingly harder for users to modify the source code in order to prevent itform attackers.	they released the intensive spiral model in which there was no risk associated with the source code and it allowed the developers to release that software in the market for its users.
15	Mockus et al.,(2000)	After checking the source code of the software there were some flaws which lead to corruption of software during that time.	Then after testing the source code they checked the defeat density, productivity, core team size and the risks were eliminated for further development in the software. Then it became difficult for an attacker to replicate code.

Conclusion:-

In this section, we can conclude that by keeping the source code opened for any system. It first increases the exposure and then there lies more information regarding vulnerabilities which are available to the attackers. But the exposure appears to be low with closed source systems, but the exposure actually will be more. As for the open source software, only interested parties can access the openness of any system. In the future by keeping the source open the security increases. Open source helps us to make any modification to the OSS source code. We will modify, redistribute, or make changes in the source code. Here, we also see by keeping the source code, open does not make the software more secure but easily allows an attacker to bypass the code and allow a backdoor to exploit the software. Openness is also keeping the developers and programmers away from innovative development. We have some companies like the Sun Microsystems and Apple who have taken the first proposal to publish the source code freely and these companies have much greater security as compared to other software. These companies have enhanced greatly in producing hybrid projects these days. The OSS movement can take the total security to a greater level in the software industry. The aim of this paper is to identify the flaws and security issues related to the Open source software, which can be beneficiary to us and our surroundings.

References:-

1. Alhazmi, O., Malaiya, Y. & Ray, I. (2005) Security Vulnerabilities, in Software Systems: A Qunatitative Perspective in Data and Applications Security 2005, LCNS 3654, 281-294.
2. Auguste Kerckhoffs. (1883) La cryptographiemilitarie.Journal des sciences militaries, IX, 1983. pp. 5-8, Jan 1883, pp. 161-191, Feb. 1883.Diyachenko, T. (2015). STATISTICAL ANALYSIS OF THE UNIFORMITY OF CRYPTOGRAMS IN THE DYNAMIC CRYPTOSYSTEMS.
3. Anderson, R. (2001) Why Information Security is Hard – An Economic Perspective, in Proceedings of the Seventeenth Computer Security Applications Conference, New Orleans, December 10-14, 358-365.
4. Banker, R.D., Chang, H., Kemerer, C.F., 1994. Evidence of economies of scale in software development. Information and Software Technology 36 (5), 275-282.
5. Boehm, B. (1988) A spiral model for software development and enhancement. IEEE Computer, 21 (5), 61-72.
6. Bollinger, T. "Linuxand Open-Source Success: Interview with Eric. S. Raymond," IEEE Computer, 1999, pp. 85-89.
7. CERT Advisor CA-2001-01 Interbase Server Contains Compiled in Black Door Account (2001),URL: http://www.cert.org/advisories/CA-2001-01.html. (Accessed on 22/08/2016).
8. Clarke, I., 1999. A distributed decentralized information storage and retrieval system, Unpublished Masters Thesis. University of Edinburgh, Edinburgh.
9. Callhoun, C., 1986. The radicalism of tradition: community strength or venerable disguise and borrowed language? American Journal of Sociology 88 (6), 886-924.
10. Clarke, I., Sandberg, O., Wiley, B., Hong, T.W., 2000.Freenet: a distributed anonymous information storage and retrieval system. In: Proceedings of the Paper Presented at the Designing Privacy Enhancing Technologies in International Workshop on Design Issues in Anonymity and Unobservability. Berkley, CA.
11. Chowdhry,P.(1999)Opensourcemeetsthe'Baywatch,URL:http://www.zdnet.com/eweek/stories/general/0,11011, 2352305,00.html. (Accessed on 28/08/2016)
12. Csikszentmihalyi, M. (1996). Creativity: Flow and the Psychology of Discovery and Invention. New York, HarperCollins.,Csikszentmihalyi B. (2007). Creativity support tools: Accelerating discovery and innovation5. Communications of the ACM, 50(12), 20-32.

13. DiBona, C., Ockman, S. & Stone, M. (eds) (1999) Open Sources: Voices from the Open Source Revolution. O'Reilly & Associates, Sebastapol, California.
14. Friedrichs, O. (2000) Secure Programming,URL: http://www.securityfocus.com/forums/secprog/secure-programming.html. (Accessed on 02/09/2016).
15. Fenton, N. &Pfleeger, S.L. (1997) Software Metrics: a Rigorous and Practical Approach. 2nd edn. International Thomson Computer Press, London.
16. Fenton, N. & Neil, M. (1999) A critique of software defect prediction models. IEEE Transactions on Software Engineering, 25 (5), 675-689.
17. Fenton, N., lizuka, Y. &Whitty, R. (eds). (1995) Software Quality Assurance and Measurement: A Worldwide Perspective. International Thomson Computer Press, London.
18. Feller, J.F., B. "A Framework Analysis of the Open Source Software Development Paradigm, " Proceedings of the ICIS 2000, Brisbane, Australia, 2000, pp. 58-69.
19. Fielding, R. (1998). "Shared Leadership in the Apache Project." Communications of the ACM 42(4): 42-43.
20. Fichman, R.G., Kemerer, D.F., 1997. The assimilation of software process innovations: an organizational learning perspective. Management Science 43 (10), 1345-1363.
21. Godfrey, M. &Tu, Q. (2000) Evolution in open source software: a case study. Proceedings IEEE International Conference on Software Maintenance.
22. Garfinkel, S. (1999) Open source: how to secure?,URL: http://www.wideopen.com/story/101.html. (Accessed on 10/09/2016)
23. Garfinkel, S. &Spafford, E. (1996) Practical Unix and Internet Security, 2nd edn. O'Reilly & Associates, Sebastapol, California.
24. Gross,G.(2000) Panel:open source security needs to be a priority,URL:http://www.newsforge.com/ article.pl?sid=00/10/17/1830254. (Accessed on 10/09/2016).
25. Grant, R.M., 1996. Towards a knowledge-based theory of the firm. Strategic Management Journal 17, 109-123.
26. Glaser, B., Strauss, A., 1967. The discovery of grounded theory: strategies for qualitative research. Aldine de Gruyter, New York, NY. Glaser, B. (2017). Discovery of grounded theory: Strategies for qualitative research. Routledge.
27. Glass, R.L., Vessey, I., Conger, S.A. 1992. Software tasks: intellectual or clerical. Information and Management 23 (4), 183-192.
28. Hatton, L. (1997) Re-examining the fault density-component size connection. IEEE Software, 14 (2), 89-98.
29. Harrison, W. (2001) Editorial: Open Source and Empirical Software Engineering. Empirical Software Engineering, 6, 193-194.
30. Hatton, L. (1997) Re-examining the fault density-component size connection. IEEE Software, 14 (2), 89-98.
31. Hars, A., Ou, S. "Working for free? Motivations of participating in Open Source Projects,"b Proceedings of the Hawaii International Conference on Systems Sciences, 2001.
32. International Standards Organization (1991) Information Technology-Software Production Evaluation: Quality Characteristics and Guidelines for their Use. ISO/IEC IS 9126, Geneva.
33. Jorgenson, D.L., 1989. Participant Observation: A Methodology for Human Studies. Sage, Newbury Park, CA.
34. Kemerer, C.F., Slaughter, S., 1999. An empirical approach to studying software evolution. IEEE Transactions on Software Engineering 25 (4), 493-509.
35. Kollock, P. (1999). The Economies of Online Cooperation: Gifts and Public Goods in Cyberspace. Chapter 7, Communities in Cyberspace, M. A. P Smith and. Kollock eds. London, Routledge.
36. Kohanski, D., 1998. Moths in the machine. St. Martin's Press, New York.
37. Kohanski, D., 2000. Moths in the machine.The Power and Perils of Programming, 2nd Ed. St. Martin's Press, New York.
38. Lettice, J. (2001) German armed forces ban MS software, citing NSA snooping, URL: http://www.theregister.co.uk/content/4/17679.html. (Accessed on 17/09/2016)
39. Levy, E. (1996) Smashing the stack for fun and profit.
40. Levy, E. (2000 a) Wide open source, URL: http://www.securityfocus.com/commentry/19. (Accessed on 20/09/2016)
41. Lerner, J., Tirole, J., 2002. The Simple Economics of Open Source, NBER Working Paper Series, WP 7600 .Harvard University, Cambridge, MA.
42. Levy, S., 1984. Hackers. Anchors/Doubleday, New York.
43. Moody, G., 2001. Rebel Code. Perseus Publishing, Cambridge, MA.

44. McAllister, N. (2001) The spy who hacked me: will open source be the hero of International Security.URL:http://www.sfgate.com/cgibin/article.cgi?file=/technology/archive/2001/03/15/china.dtl. (Accessed on 29/09/2016)
45. Moody, G. (1997) The greatest OS that never was, URL: http://www.wired.com/wired/5.08/linux_pr.html. (Accessed on 01/10/2016)
46. McCabe, T.(1976) A complexity measure. IEEE Transactions on Software Engineering, 2 (4), 308-320.
47. McConnell, S. (1999) Open source methodology: ready for prime time? IEEE Software, 16 (4), 6-8.
48. Meyer, M.H., Seliger, R., (1998). Product platforms in software development. Sloan Management Review 40 (1), 61-74.
49. Meyers, J.D., (1997). Qualitative research in information-systems. MIS Quarterly 21 (2), 241-242.Myers, M. D., &Avison, D. (Eds.). (2002). Qualitative research in information systems: a reader. Sage.
50. Mockus, A., Fielding (2002) A case study of open source software development: the Apache Server. Proceedings of the International Conference on Software Engineering.
51. Netcraft Web Server Survey (2001),URL: http://www.netcraft.com/survey/. (Accessed on 10/10/2016)
52. Neumann, B. C. &Ts'o, T. (1994) Kerberos: An authentication service for computer networks. IEEE Communications.
53. Norin, L. &Stockel, F. (1998) Open-source software development methodology, URL: http://www.ludd.luth.se/users/no/mssc_abstract.html. (Accessed on 16/10/2016)
54. NSA Security-Enhanced Linux (2000),URL : http://www.nsa.gov/selinux/. (Accessed on 25/10/2016)
55. O'Reilly, T. (1999) Lessons from open source software development. Communications of the ACM, 42 (4), 33-37.
56. Open Source Initiative (OSI) (2006) The Open Source Definition. URL: http://www.opensource.org/docs/osd. (Accessed on 02/01/2017).
57. Oram, A., 2000. Gnutella and Freenet Represent True Technological Innovation, URL: http://www.openp2p.com/pub/a/20805/12/2000. (Accessed on 22/02/2017)
58. Ozment, A. (2005) The Likelihood of Vulnerability Rediscovery and the Social Utility of Vulnerability Hunting, in Proceedings of the Fourth Workshop on the Economics of Information Security, Harvard University, June 2-3, Cambridge, Massachusetts, 1-21.
59. Payne, C. (2002) On the security of open source software, in Information Systems Journal, 12,1,61-78.
60. Pighin, M. &Zamolo, R. (1997) A predictive metric based on discriminant statistical analysis. Proceedings ACM ICSE '97, 262-269.
61. Payne, C. (1999) Security through design as a paradigm for systems development. Murdoch University, Perth, Western Australia.
62. Payne, C. (2000) The role of development process in operating system security. In: Information Security: Third International Workshop, ISW 2000, Vol. 1975 of Lecture Notes in Computer Science. Pieprzyk, J., Okamoto, E. &Seberry, J. (eds), pp. 277-291 Springer, Germany.
63. Pfleeger, C. (1997) Security in Computing. Prentice-Hall, Upper Saddle River, New Jersey.
64. Pliskin, N., Balaila, I., Kenigstein, I., 1991. The knowledge contribution of engineers to software development: a case study. IEEE Transactions on Engineering Management 38 (4), 344-348.
65. Raymond, E. (2000) The Cathedral and the Bazaar, URL: http://www.tuxedo.org/esr/writings/cathedral-bazaar/. (Accessed on 04/11/2016)
66. Raymond, E., 1999. The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary. O'Reilly, Sebastopol, CA.
67. Russell, D. &Gangemi Sr., G. (1992) Computer Security Basics. O'Reilly & Associates, USA.
68. Raymond, E.S. (2001) The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, O'Reilly, Beijing, China.
69. Rescorla, E. (2004) Is finding security holes a good idea? , in Proceedings of the Third Annual Workshop on Economics and Information Security, University of Minnesota, May 13-14.
70. Schwarz, M. and Takhteyev, Y. (2008) Half a Century of Public Software Institutions: Open Source as a Solution to Hold Up Problem.Schwarz, M., &Takhteyev, Y. (2010). Half a Century of Public Software Institutions: Open Source as a Solution to Hold-Up Problem. Journal of Public Economic Theory, 12(4), 609-639.
71. Schneier, B. (2000a) Closing the window of exposure: reflections on the future of security,URL: http://www.securityfocus.com/templates/-forum_mesage.html?forum=2&head=3384&id=3384. (Accessed on 09/11/2016)

72. Scacchi, W. "Understanding the requirements for Developing Open Source Software Systems," IEE Proceedings - Software (In press), 2002.
73. Schneier, B. (2000b) Full disclosure and the window of exposure. Crypto-Gram,URL: http://www.counterpare.com/crypto-gram-0009.html#1. (Accessed on 22/11/2016)
74. Simpson, S. (1999) PGP DHvs RSA FAQ, URL: http://www.scramdisk.clara.net/pgpfaq.html. (Accessed on 29/11/2016).
75. Simon, H., 1991. Bounded rationally and organizational learning. Organization Science 2 (1), 125-134.
76. Strauss, A., Corbin, J., 1990. Basics of Qualitative Research.Sage, Thousand Oaks, CA. Corbin, J., Strauss, A., & Strauss, A. L. (2014). Basics of qualitative research.Sage.
77. Stake, R.E., 1995. Case studies. In: Denzin, N.K., Lincoln, Y.S. (Eds.), Handbook of Qualitative Research. Sage, Thousand Oaks, CA, pp. 236-247.Denzin, N. K., & Lincoln, Y. S. (Eds.).(2011). The Sage handbook of qualitative research.Sage.
78. Strauss, A., Corbin, J., 1990. Basics of Qualitative Research.Sage, Thousand Oaks, CA. Corbin, J., Strauss, A., & Strauss, A. L. (2014). Basics of qualitative research.Sage.
79. SSH1Session Key Retrieval Vulnerability (2001) ,URL: http://www.securityfocus.com/vdb/bottom.html?vid=2344. (Accessed on 04/12/2016)
80. Thompson, K. (1984) Reflections on trusting trust. Communications of ACM, 27.
81. Telelogic (2000) Logiscope User's Manual, V3.1. Telelogic, Paris.
82. Taylor, M., Singleton, S., 1993. The communal resource: transaction cost and the solution to collective action problems. Politics and Society 21 (2), 195-214.
83. U.S. Department of Defence (DOD) (1985) Trusted computer system evaluation criteria. DOD 5200.28-STD.
84. Viega, J. (2000) The myth of open source security,URL: http://developer.earthweb.com/ journal/techfocus/052600_security.html(Accessed on 12/12/2016)
85. Von Hippel, E. (1998) Sources of Innovation, New York, Oxford University Press
86. Von Hippel, E., von Krogh, G., 2003. The private-collective innovation in open source software development.Organization Science, in press.
87. Wilson, G. (1999) Is the open source community setting a bad example? IEEE Software, 16 (1), 23-25.
88. Wayner, P., 2000. Free For All: How Linux and the Free Software Movement Undercuts the High-Tech Titans. HarperBusiness, New York.
89. Wolf, B., Karim, R.L., Bates, J. "Hacker Survey," Boston Consulting Group, 2002.
90. Watts, D. Small Worlds, Princeton University Press, Princeton, 1999.
91. Ylonen, T. (1996) SSH – secure login connections over the Internet. Proceedings of the 6th USENIX UNIX Security Syposium.
92. Yin, R.K., 1994. Case Study Research: Design and Methods, second ed. Sage, Thousand Oaks, CA.