International Journal of Computer Science & Communication (ISSN: 0973-7391)
Volume 11 • Issue 2 pp. 47-51 April 2020 - Sept 2020 www.csjournals.com

Security Issues in Open-Source Software

Sumandeep Kaur
Assistant professor in Govind National College Narangwal ,Ldh,Punjab India

Abstract: Open-source software (OSS) dictates that the source code of an open-source project is publicly accessible, and may be redistributed and modified by a community of developers. Open source projects embrace strong values of community, collaboration, and transparency, for the mutual benefit of the platform and its users. This commitment to community pushes developers to constantly contribute new features and to ensure old ones perform properly. As a result, popular OSS projects are often on the cutting edge of technology. The open exchange of information is fundamental to open source projects and allows them to be more cost-effective, flexible, and secure. This paper deals with the advantages and disadvantages of open-source software as well as various security issues related to OSS.
Keywords:- Flexibility, Community, Cost, vulnerabilities.

INTRODUCTION:-

Open source software is computer software that has a source code available to the general public for use as is or with modifications. This software typically does not require a license fee. OSS has become very popular and there are OSS applications for a variety of different uses such as office automation, web design, content management, operating systems, and communications. Open-source technologies helped establish much of the internet. Furthermore, many of the programs in use every day are based on open-source technologies.. The key difference between OSS and proprietary software is its license. As copyright material, software is almost always licensed. The license indicates how the software may be used. OSS is unique in that it is always released under a license that has been certified to meet the criteria of the Open Source Definition. These criteria include the right to :Redistribute the software without restriction; modify computer programs, Access the source code ,Modify the source code ,Distribute the modified version of the software.

EXAMPLES OF OPEN SOURCE SOFTWARE

- Mozilla's Firefox web browser
- Thunderbird email client
- PHP scripting language
- Python programming language
- Apache HTTP web server
- The Oxford English Dictionary, 1857
- Human Genome Project, one year before Linux
- Linux, 18 million users
- Google uses Linux (1000 queries per second!)
- Apache,OS webserver
- Open Office,a complete office suite of programs, compatible with Microsoft's Office, available in 21 different languages

ADVANTAGES OF USING OPEN SOURCE IN THE ENTERPRISE

1. FLEXIBILITY AND AGILITY

IT leaders must fundamentally provide flexibility and agility for their enterprise. If you can't compete on agility, you're going to get left behind by the competition. Open source enables technology agility, typically offering multiple ways to solve problems. Open source helps keep your IT organization from getting blocked because a particular capability isn't available from a vendor. Instead of waiting for the vendor to deliver that capability, you can create it yourself.

2. SPEED

Your enterprise will soon be competing on speed, if it isn't already. Open source enables speed. A great advantage of open source is the ability to take the community versions, get started, understand whether they can

solve your business problem, and begin to deliver value right away. Once you make that determination, professional support and services are increasingly available for open source products, especially those supported by Red Hat.
This allows you to get the best of both worlds — flexibility, agility, and the ability to get started quickly and inexpensively, with the ability to mature to a large scale, fully supported, enterprise-grade implementation, and you don't have to go over proprietary licensing hurdles to get there.

3. COST-EFFECTIVENESS
Open source is generally much more cost-effective than a proprietary solution. Not only are open source solutions typically much more inexpensive in an enterprise environment for equivalent or superior capability, but they also give enterprises the ability to start small and scale (more on that coming up). Given that enterprises are often budget challenged, it just makes financial sense to explore open source solutions.

4. SOLID INFORMATION SECURITY
Commercial open source has a solid information security record in a dangerous world. Obviously, it's difficult to claim security superiority for any solution and it's a challenging environment for all of us, but the responsiveness of the open source community and vendors relative to information security problems has been very good. The fact that we've had eyes on code that in some cases is decades old, and we were able to identify and fix problems when they became apparent, rather than have the code molder in a proprietary environment where few knew about the exposure but some were exploiting it, is an advantage of open source.

5. SHARE MAINTENANCE COSTS
You can solve your enterprise problems while effectively sharing some of the maintenance costs. One of the fundamental advantages of open source is community involvement. Rather than writing an application and having to sustain it yourself, you can share the cost of maintaining and sustaining applications among multiple parties.

6. COMMUNITY.
Open source solutions geared toward the enterprise often have thriving communities around them, bound by a common drive to support and improve a solution that both the enterprise and the community benefit from (and believe in). The global communities united around improving these solutions introduce new concepts and capabilities faster, better, and more effectively than internal teams working on proprietary solutions.
7. TRANSPARENCY
Open source code means just that—you get full visibility into the code base, as well as all discussions about how the community develops features and addresses bugs. In contrast, proprietary code produced in secrecy may come with unforeseen limitations and other unwelcome surprises. With open source, you're protected against lock-in risks and can see exactly what you're getting.

8. MERIT-BASED.
With open source code, the sole motivation behind decision making around the direction of a solution is to make the best, most useful product possible. Corporations making proprietary code usually put the bottom line foremost, which is not always ideal. When choosing a technology integral to your business, it's best to ensure its agenda supports your own interests.

9. FASTER TIME TO MARKET.
Because open source solutions are openly available and can be explored for free, it's often much faster to investigate options and get solutions off the ground.

DISADVANTAGES OF OSS

1.CUSTOMIZATION CAN JEOPARDIZE SUPPORT:
However, the ability to modify the source code leads to one of the key disadvantages as well, which is that you need to have dedicated people to support the code. It's your responsibility especially if you have modified it, which means you might end up spending more money than you initially planned.

2. LEVEL OF SUPPORT:
Unlike proprietary software, most open source initiatives do not come with any structured support. They often even rely on contacting the developer directly.

3. INTELLECTUAL PROPERTY CONCERNS
Many people also expressed intellectual property concerns. It can be difficult to and expensive to ensure full compliance with all of the different open source licensing terms, and that can lead to litigation. Although, that is not an issue unknown in the telecom world.

4. USABILITY
Compared to closed software, in most cases, open-source software is not as user-friendly. The main point of criticism is that open-source software is more oriented towards the needs of the developer and not the "unskilled" end user. Let's face it, regular users will never even look at the source code, let alone to tamper with it.
5. SECURITY
Open-source software is not developed in a controlled environment. With hundreds of developers working on the software, there is a chance that some of them could have malicious intentions. All it takes for a disaster is a single programmer to incorporate some malware into the software. In the case of closed software, only the vendor developers can see and edit the source code. That's why closed software is seen as safer, although the risk of hidden backdoor Trojans is always a possibility.

6. LONG-TERM COST
Sometime down the road costs for FOSS software can stack up. If any problem occurs that needs immediate attention it's up to you to put out the fire and those costs. You can't yank your vendor's chain to fix the issue because there is none. Instead, you have to either deal with it in-house or hire external help. Also, consider the costs of the implementation and staff training associated with introducing new software in the office.

OPEN SOURCE SECURITY RISKS AND VULNERABILITIES TO BE AWARE OF
1. THE PUBLIC NATURE OF EXPLOITS
The nature of the open source model is that open source projects make their code available to anybody. This has the advantage that the open source community can flag potential exploits they find in the code and give open source project managers time to fix the issues before publicly revealing information on vulnerabilities.
However, eventually such exploits are made publicly available on the National Vulnerability Database (NVD) for anyone to view. Hackers can use the publicity of these exploits to their advantage by targeting organizations that are slow to patch the applications that may be dependant on open source projects with recent vulnerabilities.
Dealing with this risk from the organization perspective means recognizing that open source exploits are made vulnerable and that hackers stand to gain a lot from attempting to breach services that use vulnerable components. Update as quickly as possible or pay the consequences.

2. POTENTIAL INFRINGEMENT RISKS
Open source components can create intellectual property infringement risks, as these projects do not have standard commercial controls. Proprietary code may, therefore, be able to make its way into open source projects.This risk is evident in the real-world case of SCO Group, who contended that IBM stole part of the UnixWare source code and used it for their Project Monterey and sought billions of dollars in damages. Appropriate due diligence into open source projects can flag up potential infringement risks.

3. OPERATIONAL ISSUES
A key area of risk faced by an enterprise using open source components is the operational inefficiencies of the organization. Of grave concern, from an operational standpoint, is an organization's failure to track open source components and to update these components, in keeping with new versions. One of the main sources of risks when using open source components in the enterprise comes from operational inefficiencies. Of primary concern from an operational standpoint is the failure to track open source components and update those components as

new versions become available. These updates often address high-risk security vulnerabilities, and delays can cause a catastrophe, as was the case in the Equifax breach.
It's vital, therefore, to keep an inventory of your open source usage across all your development teams, not only to ensure visibility and transparency, but to avoid different teams using different versions of the same component. Keeping an inventory needs to become part of a dedicated policy on open source usage, and software composition analysis tools provide a means to enforce this practice in an automated, easily manageable way without manually updating spreadsheets.

4. MALPRACTICE OF DEVELOPERS

Developer's malpractices including copy and pasting code from open source libraries. Copying and pasting are problematic as developers copy any vulnerabilities that could exist in the code of the project when they do it. Also, there is no way to update or track a code piece once a developer had added it to the codebase of your organization. This makes the application of the organization prone to vulnerabilities that could occur in the future. Another malpractice that can take place is the manual transfer via email of open source components. Some security risks arise due to developer malpractices, such as copying and pasting code from open source libraries. Copying and pasting is an issue firstly because you copy any vulnerabilities that may exist in the project's code when you do it, and secondly because there is no way to track and update a code snippet once it's added to your codebase, making your applications susceptible to potential future vulnerabilities that arise. You can avoid this issue by creating an open source policy that specifically forbids copying and pasting snippets directly from projects to your application codebases.
Another malpractice that can occur is the manual transfer via email of open source components across teams. This is opposed to the recommended best practice which is to use a binary repository manager or a shared, secure network

WHAT CAN DEVELOPERS DO TO SECURE OPEN SOURCE COMPONENTS?

Treat everything as code, including compliance. Doing so will help ensure that known regulations including those for the payment card industry (PCI) or healthcare (HIPAA) information privacy are followed. This will also make it easier to ensure that patches are universally applied.

1. EMBRACE AUTOMATION.

Staying up to date on vulnerabilities logged in online sources such as the National Vulnerability Database or postings on project home pages is necessary, but also time-consuming. Put some frontline tools in place to help catch the obvious things (there are some great commercial and open source Dynamic Application Security Testing (DAST) solutions available) and employ monitoring tools to keep up with what's happening in real time. Tools such as SumoLogic are great and serve as a modern alternative to a Security Information and Event Management (SIEM). At a minimum, static code analysis must be part of the CI/CD process, which provides automated, early detection of security issues to complement peer reviews.

2. BRING DEVELOPERS AND SECURITY TOGETHER.

Enlist your security teams to train developers to drive thorough understanding of security and the latest trends. An initial secure coding workshop held in partnership with the security team is a great way to kick things off. Invite them to design reviews and include them in sessions when high-risk changes are being made.

Build a security-first culture. Your organization must focus on more than just bringing developers and security together, but also ensure that effective security practices are built into everything you do. The best fixes and the best alerting mechanisms in the world cannot resolve poor security practices. The Equifax breach, for example, attributed to vulnerable versions of the open source software Adobe Struts, is a case in point. Since the well-publicized breach in 2017, companies are still downloading the vulnerable versions of the package despite the fact that a patch is available. (The patch was also available two months before the Equifax breach and has been issued multiple times since.) In DevOps culture, security discussions must happen early and often throughout the software development lifecycle and beyond. If you're using open source components, it's your responsibility to be aware of the updates and to actually apply them yourselves.
There are also free tools for assessing the risks in open source software and containers. Many open source software packages utilize free static analysis scanners and the results are available for everyone to inspect.

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.Payne, C. (2002) On the security of open source software, in Information Systems Journal, 12, 1, 61-78.

4. Raymond, E.S. (2001) The Cathedral and the Bazaar: Musings on Linux and Open Source by an Accidental Revolutionary, O'Reilly, Beijing, China.
5. 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
6. https://www.zivtech.com/blog/benefits-open-source-software
7. https://www.academia.edu/18931966/Free_and_Open_Source_Software_Evolution_Benefits_and_Characteristics
8. https://www.onebusiness.ca/sites/default/files/MEDI_Booklet_Open_Source_Software_accessible_E.pdf
9. https://www2.cs.siu.edu/~carver/talks/foss.pdf
10. https://www.researchgate.net/publication/28810296_Open_Source_Software_and_Libraries
11. https://www.researchgate.net/publication/264037934_Role_of_Free_and_Open_Source_Software_in_Computer_and_Internet_Security
12. https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/78959/All_About_Open_Source_v2_0.pdf
13. https://tavaana.org/sites/default/files/introduction_to_opensource.pdf
14. https://enterprisersproject.com/article/2015/1/top-advantages-open-source-offers-over-proprietary-solutions
15. https://entrepreneurhandbook.co.uk/open-source-software/
16. https://www.aoe.com/en/open-source.html
17. https://opensource.com/article/17/8/enterprise-open-source-advantages
18. https://www.esds.co.in/blog/tag/disadvantages-of-open-source-software/#sthash.smLlQCWF.dpbs
19. https://blog.dialogic.com/blog/key-advantages-and-disadvantages-of-open-source-software
20. https://medium.com/4thought-studios/the-pros-and-cons-of-open-source-software-d498304f2a95
21. https://opensource.org/blog
22. https://www.csoonline.com/article/3157377/open-source-software-security-challenges-persist.html
23. https://www.sentinelone.com/blog/open-source-security/
24. https://www.computerweekly.com/feature/Open-source-software-security
25. https://www.xfive.co/blog/5-open-source-security-risks/
26. https://cloudacademy.com/blog/open-source-software-security-risks-and-best-practices/
27. https://blog.bitsrc.io/open-source-security-risks-and-vulnerabilities-to-know-in-2019-8354058f6ad3
28. https://dzone.com/articles/open-source-software-security-risks-and-best-pract
29. https://www.xfive.co/blog/5-open-source-security-risks/
30. https://www.thebalancecareers.com/what-is-open-source-software-2071941

.