IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

A Review on A Study of Vulnerabilities of Open-Source Software System

Mr. Sudheer Shetty[1], Adithya Tejaswi D[2], Afiza A[3], Aishwarya Salimath[4], Akash Devadiga[5]
Department of Information Science and Engineering[1,2,3,4,5]
Alva's Institute of Engineering and Technology, Mijar, Karnataka, India

Abstract: *The issue of open-source software system vulnerabilities is covered in this essay. Open Sources are used because, due to flexibility, cost-effectiveness, accessibility. A binary program can reveal whether the components of the open-source system with vulnerabilities are reused and cannot recognize the location of vulnerabilities. For this purpose, they addressed and designed BMVul which is a identification on a software modularization method, in which binary programs are oriented. With the creation of an open-source component function of vulnerabilities set using function signatures, BMVul finds vulnerabilities in binary modules that make use of such components. Most businesses lack a trustworthy a direct and rapid method of notification when zero-day vulnerabilities are discovered and when updates are released. Hence, open-source attack vectors are more prevalent than necessary. It is now believed that while procedures can be used to efficiently supplement professional human inspection of OSS, it may not be possible to completely replace it. Open-source software systems has more importance in selection of detection method and cybersecurity ecosystem. Reusing vulnerable open-source components can result in security issues. The developers of mobile application are heavily depending upon open-source system software to unload(offload) functionalists which are common such as media format playback and implementation of protocols. Several vulnerabilities were found in popular open-source libraries such as FFMPEG and OpenSSL. These weaknesses are carried over onto mobile applications that use these libraries, making them susceptible.*

Keywords: *Open Source*

I. INTRODUCTION

Open-source software is developed in the public domain collaboratively with a license that grants rights to the base of user which are reserved usually for copyright holders. A famed open-source license is GNU General Public License which permits free distribution under which the remaining or further developments development are also free.

Security issues and flaws in proprietary software are identified, and open-source software is suggested as a potential remedy. Open-source software in which license is provided with unlimited access on the source code, the source code is modified and examined according to customer's wish. According to Wall Street Journal article, the open-source software supporters, mainly Linux a PC based system, this software is viewed as a less vulnerable to worms and viruses, making OSS as a suitable choice over the other operating system, like a Windows XP.

Although a bug or flaw are system flaws that may (or may not) lead to a vulnerability, the vulnerability is described as a fault in an information system, internal controls,system security policies or implementations that a threat source may use against them [1]. The recent technologies in computing power, the data availability, new algorithms had major-break throughs in artificial intelligence (AI) and machine Learning (ML)in the last decade. Many applications in AI/ML had become important in everyday life, ranging from automation and natural language processing to forecasting and privacy issues [2][3].

The open-source supply chain been advanced, the open-source vulnerabilities which is also limelight. According to the "State of the Software Supply Chain Report" ,29% of open-source software (OSS) projects exists at least one known vulnerability that may become attack surface to entire system. The cyberattacks focused at OSS projects have aggressively or exponentially grown [4].

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

The software society is connected globally, the amount of development work is correctly crowdsourced to the OSS developer's community with little understanding of security problems this is created. Due to the Heartbleed flaw in OpenSSL2, third-party libraries speed up development at the expense of increased risk.

The relevance of OSS vulnerability identification as well as the security dangers related to utilising OSS libraries in software development. It draws attention to the ubiquity of OSS and the possible drawbacks of skipping crucial security maintenance chores like vulnerability scanning [5].

The study suggests enhancing communication between the open-source community, security researchers, and repository providers and highlights the significance of bug reward programmes and vulnerability disclosure procedures to encourage security disclosures [6].

The shortcomings of current methods to stop open-source the shortcomings of current methods for preventing open-source vulnerabilities and suggests a fresh strategy termed xVDB (Extended Vulnerability Database), which makes use of several data sources to build a sizable patch database. The evaluation of the xVDB database reveals that it has strong patch collection coverage, gathering at least four times as many security fixes than current methods. To help secure the software ecosystem, it is also used to find copies of insecure code on open websites [7]. Because IT systems are so prevalent, it is critical to immediately discover and evaluate vulnerabilities in order to avoid serious effects. This is made easier by publicly accessible databases like NVD and CVE, but data may also be accessed via other channels like social media and blogs. Common Vulnerability Scoring System (CVSS) is used to categorise and score vulnerabilities, and Open-Source Intelligence is often utilised by IT security professionals to conduct vulnerability assessments (OSINT). To expedite the process and order suggestions, automated machine learning techniques to vulnerability assessment have been suggested [8][7].

II. LITERATURE REVIEW

Author	Approach	Literature review on
Shoshitaishvili [5]	KPatch	application binaries that contain vulnerable open-source software are patched. KPatch employs symbolic execution to find and fix dangerous software code pathways.
Shen [6]	Leveraging Transitive Dependencies	To find and patch vulnerable software versions, LTD combines programme analysis and machine learning techniques.
Rebert [6]	VMT	VMT offers a variety of methods, such as dynamic analysis, static analysis, and vulnerability modelling, for evaluating the security of software applications.
Wang [5]	VulHunter	The authors tested VulHunter on several well-known OSS apps and demonstrated its accuracy in identifying vulnerabilities.
Cheng [8]	CVSS prediction	A CVSS prediction model based on the Convolutional Neural Network (CNN) algorithm was suggested by (2021).
Rahman [9]	OWASP	evaluated four open-source web application vulnerability scanners: OpenVAS, Wapiti, Arachni, and Vega.
Song [10]	ASan (Address Sanitizer) and Memory Sanitizer (MSan)	The evaluation was based on a dataset of programs with known vulnerabilities, and the evaluation criteria included the detection rate, false positive rate, and runtime overhead

III. THE VULNERABILITY OPEN-SOURCE PHENOMENON

As scientists and researchers shared their source code and built on one another's inventions in the 1960s and 1970s, the open-source software movement was born [7]. The Free Software Foundation was established by Richard Stallman to support computer users' rights to study, copy, alter, and redistribute software after MIT granted a commercial corporation a licence to use code developed by its scientists in the 1980s. To ensure that open-source software remained

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

cost-free and accessible to everyone, Stallman developed the GPL licence [8]. It is challenging to comprehend the open-source movement from an economic standpoint because the concept of free and open-source software was developed in the academic community [18]. Academic institutions have a distinct history from businesses that exist to make money for their owners and shareholders. Almost one-third of survey participants who were asked about their motivations for contributing to open source said they felt obligated to contribute back to the community [17].

The open-source movement, however, is not restricted to the gift economy, as evidenced by recent events. For instance, IBM's integrated development environment Eclipse was worth $40 million until it was made available as open source. IBM, a multibillion-dollar corporation, is undoubtedly regarded as a component of the commodity economy, and the theory of the gift economy cannot fully account for their contribution to open source [10][12].

The open-source movement, however, is not restricted to the gift economy, as evidenced by recent events. For instance, IBM's integrated development environment Eclipse was worth $40 million until it was made available as open source [9]. IBM, a multibillion-dollar corporation, is undoubtedly regarded as a component of the commodity economy, and the theory of the gift economy cannot fully account for their contribution to open source.

IV. CHARACTERISTICS OF OPEN-SOURCE WEB

Tool	OWASP ZAP	Paros
Type	Web Application Penetration Testing Tool	Web Application Vulnerability Scanner
License	Free and Open Source	Open source with free
Platform	Windows, Linux, Mac	Windows, Linux, Mac
Active Development	Yes	No (Discontinued)
User Interface	GUI, CLI	GUI
Scanning Techniques	Active and Passive Scanning	Active Scanning
Supported Protocols	HTTP/HTTPS, WebSocket, FTP, SOAP, REST, and more	HTTP/HTTPS
Extensibility	Extensible via Add-ons	Not Extensible
Documentation	Extensive Documentation and User Guide	Limited Documentation

Web application penetration testing may be done using OWASP ZAP and Paros, two open-source web application vulnerability scanners. While Paros is a defunct programme that only supports active scanning, OWASP ZAP is an active tool that supports both active and passive scanning. Several protocols are supported, OWASP ZAP has a GUI and CLI interface, and it can be expanded with add-ons. Paros, on the other hand, only supports HTTP/HTTPS protocols and has a GUI interface.

In terms of documentation, Paros has little to no documentation while OWASP ZAP offers rich documentation and user manuals. It is advised for web application penetration testing to use OWASP ZAP instead than Paros since it is a more potent and adaptable tool overall **[11].**

HOW EFFECTIVE IN TERMS OF PRECISION, RECALL, AND ACCURACY ARE THESE DETECTORS?

Depending on the instrument and the kinds of vulnerabilities it is intended to discover, vulnerability detectors' performance varies. Nonetheless, these instruments have generally performed well in terms of recall, accuracy, and precision. Depending on the instrument and the kind of vulnerability, some studies claim that the accuracy of vulnerability detectors for open-source software can range from 60% to 95%. On the other hand, the recall might be anything between 30% and 90%. These instruments' accuracy can vary, but it is often greater than 80% [13].

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

HOW EFFECTIVE ARE THE DETECTORS IN FINDING VULNERABILITIES IN TERMS OF THEIR PRICE?

When compared to manual approaches, the cost of employing open-source vulnerability detectors is quite inexpensive. As many of these tools are open source and free, anybody may download and use them for nothing. There are also several commercial vulnerability scanners that offer supplementary capabilities and assistance, although they can be expensive. The cost of employing open-source vulnerability detectors, however, may also differ based on the tools' operating requirements, such as memory and computing capacity [13].

HOW ACCURATE ARE THESE DETECTORS' DETECTION RATES IN COMPARISON?

Open-source software has a wide variety of vulnerability detectors, and each tool's precision varies. Yet, several research that examined the effectiveness of various vulnerability detectors discovered that certain solutions outperformed others. For instance, while comparing six well-known open-source vulnerability scanners, researchers discovered that three of them outperformed the others by a large margin. One commercial vulnerability scanner surpassed numerous open-source scanners in terms of accuracy and speed, according to another research. In general, it's critical to compare several vulnerability detectors in order to decide which instrument is most appropriate for a certain task or project [13].

V. EXTRACTION OF CORE VULNERABILITY INFORMATION

A technique called Core Vulnerability Information Extraction (CVE) is used to find and monitor well-known security flaws in software and other systems. The MITRE Company, which maintains the CVE system, offers a standardised naming scheme for vulnerabilities as well as comprehensive information on each one, including how it may be exploited and how to reduce the risk.Core Vulnerability Information Extraction [14][16].

To locate and keep track of well-known security holes in software and other systems, a method called Core Vulnerability Information Extraction (CVE) is utilised. The MITRE Corporation, which oversees the CVE system, provides full information on each vulnerability, along with details on how it may be exploited and what steps can be taken to lower the risk.[14]

VI. VULNERABILITY OPEN-SOURCE PREDICTION.

In the realm of software security, it is customary to characterize vulnerability prediction as a binary classification problem. By examining the characteristics and patterns in the source code, it is intended to train a machine learning model to discriminate between susceptible and non-vulnerable code [14].

The dataset utilized in this study for testing and performance assessment came from a repository of function-level source code from multiple open-source projects that was made accessible to the public. Given the existence or absence of particular preset vulnerabilities, the dataset includes both susceptible and non-vulnerable code that is tagged accordingly [12].

The source code was first preprocessed to extract pertinent features and convert them into a format appropriate for input into the model before the machine learning model was trained. Code metrics like function length, complexity, and depth, as well as more intricate aspects like control flow graphs and data flow graphs, were among the elements that were extracted [14].

Support vector machines, random forests, and neural networks are just a few of the machine learning techniques that were examined. Standard performance criteria including accuracy, precision, recall, and F1 score were used to evaluate each algorithm's performance [15].

The study's findings demonstrated that, depending on the technique utilized, machine learning models may accurately forecast vulnerabilities in open-source software with accuracy rates ranging from 70% to 90%. The study also emphasized the need of rigorous feature selection and preprocessing, as well as the necessity of continual model review and improvement over time to consider modifications to the programmer environment and the dynamic nature of software vulnerabilities.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

VII. DATASET OPEN-SOURCE

The original Draper VDISC dataset, which includes a sizable number of function-level source codes gathered from numerous open-source projects, including the Debian Linux distribution, open git repositories on GitHub, and synthetic codes from the SATE IV Juliet Test Suite of NIST's Samite project, was divided into several subsets that we were able to extract. These function-level codes were carefully annotated by the dataset's authors in accordance with conclusions from three distinct static analyzers that indicated possible exploits, and they were grouped into five different categories of CWE vulnerabilities [13].

In order to avoid include a duplicate of the training sample in the test dataset, they divided the entire dataset into three subparts: training (80%), validation (10%), and test (10%) sets. There are much less positive (susceptible) samples in the Draper VDISC Dataset than there Due to its real-world nature, negative (non-vulnerable) samples were obtained [18].

We maintained this imbalance in our trials to objectively assess the efficacy of our suggested approach, but we also created balanced subgroups in a few of them to assess the discernibility of other vulnerability categories. For our tests, we only employed C language functions that could be parsed by the Picoparsec parser that we used in our implementation.

VIII. IMPLEMENTATION DETAILS

To implement the proposed source code representation method, we utilized the Picoparsec library,which is a parser for the C language (C99), to generate ASTs of the source codes. In addition, we adapted some open-source implementations to our case to convert a regular AST into a complete binary AST.

Then, using the grammar of the language C to determine 48 key token kinds, we gave distinct values to each token type, encoding the nodes of the AST into numerical values. We also assigned different values for encoding auxiliary information (e.g., char, int, float, short, signed, +, −, ==, >, <, etc.) of tokens when required.

These values were represented as numerical tuples, with the first number indicating the token type and the second and third numbers representing auxiliary information [15].

For the machine learning implementations, we used both the Scikit-learn and TensorFlow libraries. In Scikit-learn, we implemented and executed the Multi-layer Perceptron (MLP) algorithm, while in TensorFlow, we implemented and ran the Convolutional Neural Network (CNN) algorithm.

WHILE DETECTING VULNERABILITIES IN OPEN-SOURCE SOFTWARE, AST ENABLES US TO MANAGE THE DIMENSION OF THE INPUT CHARACTERISTICS (OSS).

The accuracy of vulnerability detection algorithms can be significantly influenced by the depth of the AST. Deeper ASTs can offer additional context information, such as the connections between various code segments, which can be helpful in locating vulnerabilities. However bigger feature vectors produced by deeper ASTs can also result in overfitting and slower training rates.

Deeper ASTs are typically more successful in finding vulnerabilities in OSS, where the codebase is frequently huge and complicated. Yet, it's crucial to strike a balance between the depth of the AST and other elements like the volume of training data and the difficulty of the machine learning model [14].

Moreover, lowering the AST at a certain level may cause the loss of crucial data and degrade the model's accuracy. As a result, it is crucial to carefully assess how the AST depth affects model performance and select a suitable cut-off depth based on the issue at hand.

Overall, there are several variables that affect and complicate the effect of AST depth on OSS vulnerability identification. Nonetheless, accurate and efficient vulnerability detection models may be created by carefully balancing the depth of the AST with other factors.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

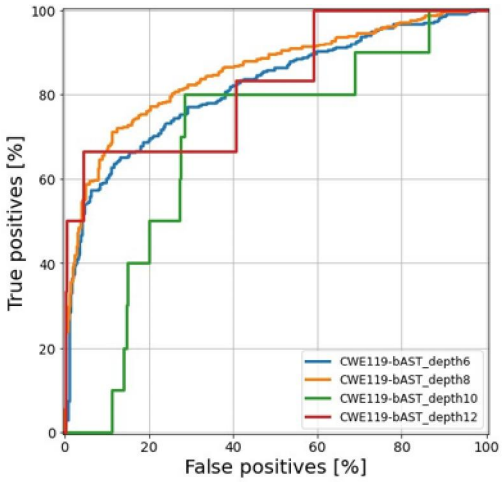

Fig.1 ROC curves for various levels of binary AST

Increasing the dimensionality of the input features can cause overfitting and worse model performance, a phenomenon known as the "curse of dimensionality." This is especially true when there is a lack of training data or when it is noisy [19].

The similar phenomenon can be true when it comes to OSS vulnerability detection. While deeper ASTs may include more information, this may not always result in better model performance if the feature vectors' higher dimensionality causes overfitting or other problems [16].

The amount of the training dataset, the complexity of the machine learning model, and the vulnerabilities being addressed may all affect the best AST depth for OSS vulnerability detection. It is crucial to thoroughly assess how AST depth affects model performance and choose an acceptable depth decision considering these factors. The trials mentioned in the text above serve as an excellent illustration of how such an evaluation may be carried out [17].

IX. PROBLEM STATEMENT AND SOLUTION

Issue 1: Open-Source Software Vulnerabilities Caused by Lack of Maintenance

SOL: Because it receives little maintenance, open-source software is susceptible to assaults. For their projects, many developers use open-source software, which makes it possible for attackers to take advantage of any vulnerabilities found in libraries or dependencies. This may result in malware attacks, data breaches, and other security problems.

Frequent upkeep and security updates are the solution.

Open-source software needs to be routinely maintained if this issue is to be reduced. Developers should routinely scan their code for vulnerabilities and install any necessary security patches and updates. Also, while choosing open-source software, developers should exercise caution and choose programmes with vibrant and helpful communities.

Issue 2: Open-Source Software Vulnerabilities Owing to Code Flaws

SOL: Many developers who may not be knowledgeable about secure coding techniques sometimes work together to create open-source software. Defects in the code may develop as a result, creating vulnerabilities that attackers may exploit.

Review of the code and security audits are the remedy.

It is critical to do regular code reviews and security audits in order to prevent vulnerabilities in open-source software caused by coding errors. Before attackers can take advantage of them, code reviews help find coding faults and other security problems. Aside from that, security audits can spot flaws that code reviews might miss as well as locations where the code can be strengthened to avoid flaws in the future [20].

X. FUTURE PERSPECTIVE

There has been and probably will continue to be worry about how vulnerable open-source software is. While open-source software has numerous advantages, like openness and community engagement, it is also more prone to security vulnerabilities because of this.

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

Future developments in a few major areas are probably going to affect how vulnerable open-source software is:

- **Adoption on the rise**: More and more businesses are embracing open-source software as it gains popularity. As a result, attackers will have a bigger attack surface to exploit.
- **Emphasis on security:** As the value of open-source software rises, security will receive more attention. To make their software safe and attack-resistant, developers will need to take more aggressive precautions.
- **Automation:** The sophistication of automation tools is increasing, and this development is expected to continue. Automation can assist in locating weaknesses and enhancing the security of open-source software.
- **Collaboration:** It will become more crucial for developers and security experts to work together. This will aid in locating weaknesses and the creation of stronger open-source software security measures.
- **Regulation:** To safeguard the security of open-source software, governments and business associations may implement restrictions. As a result, developers may be subject to more scrutiny and responsibility, which might enhance open-source software's overall security.

However, there are several promising trends that might assist to reduce this risk even if open-source software's vulnerability will likely continue to be a problem in the future. Developers may contribute to ensuring that open-source software is a useful and safe tool for years to come by implementing proactive security steps and working with others.

XI. CONCLUSION

Our review paper's conclusion emphasises the significance of researching open-source software's vulnerabilities. Due to its accessibility and low cost, open-source software has become a crucial component of contemporary software development. Open-source software development is collaborative, hence there is a chance that there might be security flaws. The sorts of vulnerabilities, the causes of them, and the possible effects they can have on software systems and their users are just a few of the topics covered in our review article on open-source software vulnerabilities.We have also covered the techniques for locating, categorising, and correcting these flaws, including as vulnerability scanning tools, code reviews, and patching. Our analysis shows that reducing possible security threats requires a proactive strategy for resolving vulnerabilities in open-source software. We have also emphasised the necessity for continued research in this field to deepen our knowledge of open-source software vulnerabilities and provide more efficient tools for their detection and remediation. In conclusion, this review article shows the need of maintaining a watchful approach to software security and stresses the crucial role that open-source software vulnerability research may play in attaining this aim.

REFERENCES

[1] A. M. Delaitre, B. C. Stivalet, P. E. Black, V. Okun, T. S. Cohen, and A. Ribeiro, ‘‘Sate V report: Ten years of static analysis tool expositions,’’ NIST, Gaithersburg, MD, USA, Tech. Rep. SP-500-326, 2018.

[2] E. Ustundag Soykan, Z. Bilgin, M. A. Ersoy, and E. Tomur, ‘‘Differentially private deep learning for load forecasting on smart grid,’’ in Proc. IEEE Globecom Workshops (GC Wkshps), Dec. 2019, pp. 1

[3] Z. Bilgin, E. Tomur, M. A. Ersoy, and E. U. Soykan, ‘‘Statistical appliance inference in the smart grid by machine learning,’’ in Proc. IEEE 30th Int. Symp. Pers., Indoor Mobile Radio Common. (PIMRC Workshops), Sep. 2019, pp. 1–7.

[4] Sonatype. (2021). State of the Software Supply Chain. [Online]. Available: https://www.sonatype.com/resources/state-of-the-software-supply-chain2022

[5] Brandon Carlson, Kevin Leach, Darko Marinov, Meiyappan Nagappan, Atul Prakash. Open-Source Vulnerability Notification. 15th IFIP International Conference on Open-Source Systems (OSS), May 2019 Montreal.

[6]Vulnerability Detection in Open Source Software: An Introduction Stuart Millar, Rapid7 LLC, stuart millar@rapid7.comarXiv:2203.16428v1 [cs.CR] 6 Mar 2022

[7] HYUNJI HONG, SEUNGHOON WOO, EUNJIN CHOI, JIHYUNCHOI, AND HEEJO LEE, (Member, IEEE) Department of Computer Science and Engineering, Korea University, Seoul 02841, South Korea Corresponding author: Heejo Lee (heejo@korea.ac.kr)

IJARSCT
ISSN (Online) 2581-9429
International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
Impact Factor: 7.301
Volume 3, Issue 12, May 2023

[8] Philip KühnRelke, David N.kuehn@peasec.tu-darmstadt.de Christian Reuter Science and Technology for Peace and Security (PEASEC), Technical University of Darmstadt Darmstadt, Germany david.relke@stud.tu-darmstadt.de reuter@peasec.tu-darmstadt.de arXiv:2210.02143v1 [cs.CR] 5 Oct 2022

[9] Hilmi S. Abdullah Academic Journal of Nawroz University (AJNU) Volume 9, No 1 (2020).
Regular research paper : Published 17 Feb 2020 Corresponding author's e-mail : hilmi.salih@gmail.com Copyright ©2018. This is an open access article distributed under the Creative Commons Attribution License.

[10] Yu Nong Washington State University, Pullman, USA yu.nong@wsu.edu Haipeng Cai Washington State University, Pullman, USA haipeng.cai@wsu.edu

[11] Vulnerability Advisories onto their Fix Commits in Open-Source Repositories}, author = {Homeroom, Daan and Sabetha, Antonino and Coppola, Bonaventura and Tamburri, Damian A.}, year = {2021}, month = {March}

[12] Dongyu Mengdmeng@ucsb.eduUCSantaBarbaraMicheleGuerrieromichele.guerriero@polimi.it Politecnico di Milano Aravind Machiryamachiry @purdue.edu Purdue University Hojjat Aghakhani hojjat@ ucsb.edu UC Santa Barbara Priyanka. Santa Barbara Andrea Centinela. Centinela @ utwente.nl University of Twente Christopher Kruegel chris@cs.ucsb.edu AsiaCCS'21, June 07–11, 2021, Hong Kong, China ASIA CCS '21, June 7–11, 2021, Hong Kong, Hong Kong

[13] Mekala Lokesh Kumar, Kandukuri Sai Krishna, Dr Deepika Badampudi, vulnerability of open source libraries Master of Science in Software Engineering January 2023

[14] Security Vulnerability Impact on Open Source: A Social Media Exploration Mohammad, Haytham and El-Gayar, Omar, "Security Vulnerability Impact on Open Source: A Social Media Exploration" (2022). Faculty Research & Publications. 288. https://scholar.dsu.edu/bispapers/288

[15] ZEKI BILGIN, (Member, IEEE), MEHMET AKIF ERSOY, ELIF USTUNDAG SOYKAN, EMRAH TOMUR, PINAR ÇOMAK, AND LEYLI KARAÇAY Received July 28, 2020, Vulnerability Prediction from Source Code Using Machine Learning.

[16] Jiang, Y., Jeusfeld, M A., Ding, J. (2021) Evaluating the Data Inconsistency of Open-Source Vulnerability Repositories In: ARES 2021: The 16th International Conference on Availability, Reliability and Security, 86 (pp. 1-10). Association for Computing Machinery (ACM)

[17] Matthieu Jimenez University of Luxembourg matthieu.jimenez@uni.luRenaud Rwemalika University of Luxembourg renaud.rwemalika@uni.lu Mike Papadakis University of Luxembourg. michail papadakis@uni.lu ESEC/FSE '19, August 26–30, 2019, Tallinn, Estonia

[18]Richard Amankwah, Jinfu Chen, Patrick Kwaku Kudjo, Dave Towey Faculty of Science and Engineering, University of Nottingham Ningbo China, 199 Taikang East Road, Ningbo, 315100, Zhejiang, China,2020

[19] Valentina Piantadosi University of Molise, Italy. piantadosi@studenti. unimol.it Valentina Piantadosi University of Molise, Italy v. Piantadosi @studenti.unimol.it Valentina Piantadosi University of Molise, Italy v.piantadosi@studenti.unimol.it

[20] Vulnerability Detection in Open Source Software: An Introduction Stuart Millar, Rapid7 LLC, *staurtmillar@rapid7.com*arXiv:2203.16428v1 [cs.CR] 6 Mar 2022