University of Bamberg

Distributed Systems Group

Master Thesis

in the degree programme International Software Systems Science
at the Faculty of Information Systems and Applied Computer Sciences,
University of Bamberg

Topic:

Open-Source Software Discovery and Vulnerability Analysis

Author:
Rajendran, Hari Prashanth

Reviewer:
Prof. Dr. Guido Wirtz

Date of submission:
01.10.2021

Contents

List of Figures

List of Tables

Listings

Abbreviations

DSG	Distributed Systems Group
OSS	Open-source Software
NVD	National Vulnerability Database
CVE	Common Vulnerabilities and Exposure

1 Introduction

Insert the text of your DSG thesis here.

1.1 Motivation

Nowadays, firms and individuals are using different forms of computer software which is running on different platforms from a simple mobile application to a sophisticated distributed enterprise system. Even though the software is created using different methodologies based on a wide variety of technologies, still each has its own advantages and disadvantages[1][2]. Software security is an essential concern in the software development process because not only to reduce the additional cost but also it can cause severe damage for developers or organizations[3]. In some recent years, there have been many incidents that happened in which software vulnerability imposed vital damage to companies and individuals. To make a clear idea of this issue, I have mentioned incidents that happened in recent years. A prominent example of open-source software vulnerability is the Equifax breach in 2017, which literally exposed 145 million users' data due to outdated open-source software. The outcome of this incident relies on openness because the same code has been seen by all users which includes the attackers[4]. Software vulnerabilities are imposed by the importance of the threats which depend on the factors like exploitation complexity and attack-surface[8]. Likewise, there have been a lot of incidents before in which the companies and individuals have been affected by software vulnerability with significant damage [1]. In addition, vulnerabilities are common and fundamental. Therefore, choosing software is important to risk management for information security. Selecting software packages with strong security features can reduce risk management to individuals or organizations. OSS has been referred to as a potential answer for software security issues and vulnerabilities because most of the open-source software licenses give full access to any part of the code to examine and modify[5].

1.2 Problem Statement

In general open-source software is used massively in modern Software Development which says around 96% of the application in the enterprise market uses open-source software. Today standard components are not re-written; instead they are shared as packages around the world. Open Source Portals like Github and Gitlab make it very easy to share those components. The advantage of this ”Shared Code Culture” is also the biggest disadvantage because the whole Community and therefore also the components are constantly changing. Releases of components are often pushed daily if the project is active. But if the community switches or the main Authors are leaving, those projects become deprecated very fast. And due to rising IT security threats, a component can become insecure if not patched frequently. Whether an OSS component will become critical also depends on the context where the component will be used. The context will define which actions need to be taken to use an OSS component within the Project. As opposed to these, I propose to build a scanner to extract all the possible open-source software components and their dependencies from software projects and perform data analysis with help of CVE and NVD to find the severity of each vulnerable component[6]. The very first challenge for building this scanner will be scanning for open-source software components and that should happen on the client-side of the system due to the source code privacy of the projects. Along with the scanning, the system should extract the open-source component meta information from a variety of software projects by using the best text analysis approach. After the extraction of the open-source software components, the next challenge will be assessing the vulnerability and its security issues of each component by choosing the right predictive analysis model. The final output of this scanning will be a PDF report which gives a clear detail of each open-source software component used inside the project.

2 Background

2.1 Open-source Software

At a fundamental level, it's a software code that's available for all users to inspect, modify, copy and use in almost any way they choose. The first evolution of open-source software happened in the 1970's by Richard Stallman[10]. Commercial software often called proprietary software also has additional licensing that further prohibits people from attempting to reverse engineer or modify this process but in open-source software, the original source code is made available alongside the final executable program. This means any developers or programmers can modify the application to improve or customize it. The students also can study how the programs were written and the programs can be easily copied and distributed over the internet with the help of cloud repositories[11]. Supporters of open-source believe open collaboration allows the software to evolve via the contribution of many users and also they believe people should have the right to use their software in whatever way they want without licensing restrictions. The term open-source has made many impacts in computer science practising whereas a lot of technologies have been built and distributed by a permissive license generally. The open-source software focuses more on the term "free" which helps to build open and transparent software systems. It also helps to improve the project becoming reliable and scalable so that it will help the digital economy to grow[11].

2.2 National Vulnerability Database

The NVD is one of the largest and most effective databases which is used to report all known vulnerabilities for both commercial and open- source components. It was governed and operated under the US National Institute of Standards and Technology(NIST) from 2005. This organization is sponsored by the Department of Homeland Security's National Cybersecurity and Communications integration center and Network Security deployment[13]. The information which has been provided by the NVD database helps the developer or security member to help track the software security. The information from NVD helps to analyse the software security vulnerability of an OSS component. It also helps the user to know what type of issues are in it and also helps the user to move further. A section of NVD also provides information on CVE as well as the information source, which is usually from the MITRE corporation[12]. Figure 1.2 will show us how far an OSS component has impacted all these years. This information is gathered by NVD with the help of the CVSS V2 and CVSS V3 data. The NVD also gives us the history of all OSS components with the help of MITRE's CVE dictionary[13]. The vulnerabilities of each component are discovered by the security researchers or organization and they will report this vulnerability to CVE. Once the rectified vulnerability is reported to the product owner or open-source project community to resolve it , the report will stay private for 60 to 90 days with CVE before going public. The information from the NVD is very useful to the consumers who use it so that they can protect themselves from the vulnerability but at the same time a hacker can also see the vulnerability of software before the consumer does. Once the threat of an OSS component is received by the CVE then this information is passed on to the NVD database and will be available in search.

2.3 Common Vulnerabilities & Exposure

The CVE is built for one main purpose which is to identify, describe and catalog the cybersecurity vulnerabilities. Each Vulnerability will have one CVE record in the catalog. All these vulnerabilities are discovered by the security expert who works for an organization or independent which is partnered with CVE. The CVE program was found in 1999 first and it is runned by the Department of Homeland Security(DHS) and Cybersecurity and Infrastructure Security Agency(CISA). The CVE and NVD sound like they are the same but they are totally two different programs. Whereas the CVE record contains the id number, description and a public reference of a known vulnerability, NVD is a database which is used to list all the CVE records of each vulnerability. The sponsors of the CVE program decided to leave this information publicly accessible. The CVE can also be used to integrate with security tools and services by an individual or organization. The users have all access to copy the record, reference the information or analyse the record without modifying it. The users can also link a CVE record to another vulnerability under some terms and conditions. The CVE provides a generic identifier for vulnerabilities which helps the user to identify the information to access accurately and easily[14].

2.4 Data Scraping

Data exists everywhere in different formats like web pages to printed materials and as we established before there's a lot of value that can be found in the right set of data. Data scraping is a process that extracts data from tables or other structured document formats and converts them into a format that can be easily processed or analyzed. It is useful because not all data sources provide ready-to-analyze, pre-formatted data. The data scraping plays a part in unlocking this value. Data scraping refers to the process of retrieving data from one format into a more useful format for further processing or analysing. In some scenarios the data might be extracted from similar data sets from two different sources. In that case the data has to be reviewed and processed to make sure they are both formatted equally. When it comes to different types of data sources there are endless ways in which data can be formatted. The two of the biggest categories of data sources are digital and physical sources[15].

Data extraction can be categorized in two types: Logical and Physical extraction. Logical extraction will extract less information because it will consume less time and is a lot easier and it is categorized into two types: which is Full and Incremental extraction. Physical extraction is the opposite of logical extraction where it consumes more time but the outcome of this extraction will give you more information. Physical extraction is categorized in two types: Online and Offline extraction[16].

2.5 Vulnerability Analysis

As businesses today increase their dependence on information technology including the cloud, IoT devices, mobile and social, their cyber risk continues to rise. However, a vulnerability management program can help to identify the weakness before it creates any major issues. Stats say 95% of all cyber attacks exploit known vulnerabilities and on an average of 15,000 new vulnerabilities are discovered each year. So constant vigilance is required to evaluate IT security posture, discover weaknesses and respond appropriately. The key to responding to this more dangerous threat environment is a robust vulnerability analysis program. A Formal process that identifies and quantifies the security weakness including your application software, hardware and network. A vulnerability analysis should give a clean and clear report of what your environment needs attention and where on the list of priorities it lies. Identifying vulnerabilities is important because unlike the targeted attacks which dominated the landscape previously. Today's advanced attacks are programmed to search for vulnerabilities in a system and automatically start their attack process. Therefore it is critical to defend even if your organization is not a high priority target.The vulnerabilities can be scored based on the risk, impact and potential exploitation of the weakness.

The vulnerability analysis for open-source software should be imposed because software selection is essential in terms of information security. The quality of software selection should be strong so that we can reduce the risk of information security[5]. Vulnerability is also known as vulnerability assessment, "it is a mechanism that defines, identifies and classifies the security holes"[9]. The vulnerability assessment will also provide deep insight, knowledge and threats about the software to an organization or individual. There are several types of vulnerability analysis: Network-based analysis, Host-based analysis, Wireless network analysis, Application analysis and Database analysis.

2.6 Dependency Manager

A good design of software design concerns the software to be built from smaller, single-purpose modules with well defined interfaces and top of that keeping with the concept of software reuse. As a result of the broad adoption of open-source software, most of the softwares built now is dependent on softwares which is built by others. So to monitor all these dependencies used inside a software requires a dependency manager. A dependency is an external software module where it can contain one single file or a group of files clubbed together and can be used for performing specific tasks. Software modules known as dependency managers coordinate the integration of external libraries or packages into bigger applications. All dependency managers use a specific configuration file which consists of: dependency name, dependency version and repository of the dependency.The most common dependency manager's configuration files are composer.json, package.json, build.gradle and pom.xml. All the initialised dependencies of the project are fetched from the repository by using their name and version. In some cases there are few dependency managers that have their own repository like maven central for maven and gradle projects, npm for npm based projects and packagist for composer projects[18]. The dependency manager is required for two main reasons which is: i,to give a confirmation that both the development and production environment uses the same dependency and its version. ii, to keep the dependency up to date[17].

2.6.1 Components of Dependency Manager

The following are typical components of a dependency management system[19]:

- **Module:** This component can be used in all sorts of projects and these can be called packages or libraries based on the programming language. The modules come with the information of what dependencies are required for a specific module.
- **Manifest file:** This file records all the dependencies of the project. It contains all the meta information of the project.
- **Lock file:** This file usually captures all the meta information of the dependencies and converts into a dependency source tree and also makes the versions immutable.
- **Repository:** In most cases each dependency has its own repository where the dependencies will be fetched directly from the repository to the project. For example, maven and gradle project dependencies are fetched fromMaven Central.
- **Dependency Constraint:** There will be a dependency constraint in the manifest file where it allows only the required version of the project.
- **Resolution Rule:** This is a rule where every dependency manager has to select the correct dependency version to the project.

3 Literature Survey

This is the last chapter of this thesis [Tur38].

3.1 Evolution of Vulnerability Analysis

Vulnerability analysis is an important operation to perform in all domains which also includes the software product. One single vulnerability can cause a catastrophe in an active system and which can be an open gateway to the attackers to exploit the system. A good vulnerability assessment should define, identify and categorize the issues in the component. Neumann and Parker say that the new vulnerabilities of IT systems are evolved from the attacks that happened in the past by using long-known techniques[21]. Therefore building a strong and secure application is merely an impossible task or will be more expensive. Earlier the vulnerability assessment happens manually in all organizations where this is a huge and overwhelming work for the IT security teams.

All the newly identified vulnerabilities should be stored in excel or a CSV file for later use. When compared to the no. of vulnerabilities today, the 90's and late 2000 had very less number of vulnerabilities. The early vulnerability scanning software just gives a simple report of found vulnerabilities in the system and later this report has to be given to the IT security team to analyse the possible threats of the vulnerability. Then the report is sent to higher authority for a review and approval. This is such a manual process which has been used earlier to detect vulnerabilities in an active system[22]. Manual scanning and repair strategies would soon become impractical as the number of vulnerabilities grew in following years and the necessity of vulnerability management became more apparent to companies. Now the future of vulnerability analysis is focussing on fully automated assessment.

3.1.1 Benefits

Lutz Lowis describes that attackers may usually repeat their exploits by reusing a susceptible service[20]. Despite many old and new threats there are few advantages to performing vulnerability analysis operations. Here are some advantages that can be achieved through a vulnerability assessment[20][22][23]:

- A vulnerability analysis can identify all the possible vulnerabilities in the system where this can be identified by both organization and the attacker(hacker) if they use the same software for scanning. The organization has a higher advantage by fixing this issue before the attackers initiate.
- If the IT security person conducts a regular scanning of the system, the scanning can give the level of risk available in the system. So this helps to figure out the health of the overall system.
- Vulnerability assessment will save money and time because if an organization or individual fails to complete the vulnerability analysis procedure, there is a greater

possibility that an attacker will be able to exploit the system, resulting in the system having to be rebuilt.

- The previous evaluation reports will be useful in improving the present system in the future.

3.1.2 Vulnerability Attacks

A vulnerability is an attribute of a software component that has a potential of exploiting or damaging an active system. Most of the vulnerabilities have a high tendency of causing damage to security policy by internal or external persons(hacker or insider). There is a past event which happened in 2017 where a big data breach occurred in Equifax. This data breach caused more than 100 million user data to be leaked[4]. In this vast area of IT there are still new and unknown vulnerabilities emerging everyday.

Ryohei Koizumi and Ryoichi Sasaki have found that whenever a vulnerability assessment operation takes place the IT security team should always use the latest scanning tool because sometimes the old softwares is effective against some small virus threat but it does not exploit the vulnerability[24]. There are some vulnerability attacks which have to be focused more because these type attacks are listed as high in risk level. Here are some major attack types that are focused by few researchers[25]:

- **Configuration-based:** This type of vulnerability occurs when there is a misconfiguration in the system or running any unwanted services in the background. The configuration-based vulnerability has a major weak point where the hackers can easily get into the organization network and try to find the active system by any form of misconfiguration. This type of vulnerability is based on weak management protocols, weak permissions and weak encryption.

- **Security patches:** A security patch is an important update for all active systems because a role of a security patches is to fix or remove the flaw or issue which is found from the vulnerability report. The software patches play a vital role in rectifying and fixing the components for both commercial software and open-source software components. Generally the security patches will be released every month for example Microsoft sends security patches to its windows operating system every month. A system which fails to update their security patches will lead to major security breaches. As everyone knows, the best example of security patches failures is ransomware which is called wannacry[26].

- **Zero-Day Vulnerability:** This is the most difficult vulnerability to rectify and fix the issue, this is because the vulnerability is new and unknown to the organization. The attackers will take advantage of this loophole and will cause a major security risk. The term “Zero-day” is used because the vendor was unaware of the threat which affected the software and the vendors had “0” days worked on the security patch or fixin the vulnerability.

- **Faulty Open-Source Package:** This the vulnerability where hackers were using it for several years. Generally the hackers used to inject a credential sniffer inside a very useful common library or package. This kind of attack happens when the vendor doesn't update the open-source packages regularly. This threat will stay in the package for several days or until it is found by the vendor.

3.1.3 Types of Vulnerability Analysis

The vulnerability analysis is also called vulnerability assessment where the main intended purpose of this is to keep the organization safe from the digital threats. This is a methodology which is used to find the IT application and the infrastructure. It also involves intense scanning by the security expert or team of the organization. There are few types of vulnerability analysis which are used exclusively for some part of IT:

Network-based Analysis: A network-based analysis is a mechanism to identify the network defects and issues in the network. They mostly scan and analyze the network endpoint and device network for security issues. The failure of vulnerability analysis will give an opportunity to the hackers to take advantage of the network issue. The organization will invest more time to improve their existing framework which is used for network vulnerability analyses. In 2008 Hai L Vu etl, developed a vulnerability analysis framework for scanning network vulnerabilities and along with that they have also proposed a scalable algorithm. Both framework and algorithm are used to evaluate the network vulnerabilities without generating a full-scale graph[27]. This figure gives the results of each network component used in the network by using the framework. The results come with a brief description about the effects of the listed vulnerability. These information about the vulnerabilities have been taken with the help of vulnerability databases[27].

Host-based Analysis: Sometimes a vulnerability can be found in the vendor's resources and with this vulnerability there are high chances where even an insider can be an attacker for the system. The attackers mostly cause damage by making an improper configuration setting in the host.This vulnerability assessment takes place in servers, workstations or other network hosts. The host-based analysis will give a detailed insight of configuration settings in the network, patches and update history. The insight which is gathered from the analysis will also give us the potential damage caused by the attackers or intruders. Anil Sharma et. al. created a software tool "Ferret" by using perl language that simply identifies the vulnerabilities present in the host[28]. This software tool helps the system administrator to identify the vulnerabilities and take action based on the threat. The host vulnerability is checked by using a different plug-in module and the end output will also mention which plug-in module is used for the assessment.

Database Analysis: Misconfigurations occur often in databases and Big Data systems. Database vulnerability analysis is mainly used for identifying the available risk in the databases. The most common risks are missing patches, weak passwords and default vendor accounts[29]. Sartaj Singh described the importance of inherent dangers of the database like how data theft is happening in the internet era and he also mentioned that the existing encryption methods are not fool proof for the high end professionals[31]. A vulnerability attack can exploit file permission, database configuration files and also have potential to steal sensitive information like credit card details, personal details, etc. There is still research going to secure the database more effectively and efficiently. In 2008 Ghassan Jabbour and Daniel A. Menasce presented a framework that provides a self protection to the database from unauthorized or intensional security parameter changes and also they proposed that this framework can be implemented in an Oracle 10g Release 2 database[32].

Application Analysis: Vulnerabilities are frequently identified in third-party apps that are built and managed. The vulnerability assessment is important to an organization's security team because they have to identify the vulnerability in the application before it exploits the system. This process is used to identify vulnerabilities which are misconfiguration in applications, outdated software packages and weak authentication. Sultan S. Alqahtani[33] has researched a modeling approach that improves traceability and trust in software products by linking the security knowledge with the software artifacts. He also introduced a scanner called Semantic Global Problem Scanner(SE-GPS) which is created by integrating the modeling approach and with the modeling approach the tool can now link the NVD[13] security database to the maven build repository.

The application vulnerability analysis is an important security process to be considered by the organization. The process's main goal is to identify the vulnerability and report it to the security authority to mitigate it before the vulnerability exploits the system. Most application vulnerability analysis tools use the proper guidelines to make a good scanning tool. Here is the main guideline that a good vulnerability scanner tool should use[34]:

- **Setup:** The setup should begin with a proper documentation about the application, with perfect security permissions and configuration tools.
- **Test Execution:** Run the scanner tool which can be an existing tool or a self created one. The scanner should identify all the software packages and its dependencies used inside the software product.
- **Vulnerability Analysis:** Once the execution is finished, the extracted software packages and its dependencies should be analysed by using any existing vulnerability databases like NVD, ISS(Internet Security Systems) , etc. each software package is searched in the database by using its name and version.
- **Reporting & Remediation:** After the previous process, the analysis will give a proper report of each software package. The report includes the risk level of each package which is categorized as LOW , MEDIUM and HIGH. Sometimes the report will also provide a brief description of the threat. The remediation action is taken by the security department of the organization. Mostly the remediation will have two possibilities which is to change the version of the package or to find an alternate software package.

3.1.4 Vulnerability Databases

3.2 Data Scraping

3.2.1 Types of Data Scraping

3.2.2 Challenges

3.2.3 Data Scraping Methods

3.3 String Distance Metrics

3.3.1 Hamming Distance

3.3.2 Levenshtein distance

3.3.3 Damerau-Levenshtein distance

3.3.4 Jaro Distance

4 Design & Architecture

This is the last chapter of this thesis [Tur38].

5 Implementation

This is the last chapter of this thesis [Tur38].

6 Results & Discussion

This is the last chapter of this thesis [Tur38].

7 Conclusion

This is the last chapter of this thesis [Tur38].

8 Future Work

This is the last chapter of this thesis [Tur38].

References

[Tur38] Alan Mathison Turing. *Systems of Logic Based on Ordinals: a Dissertation.* Ph.D. dissertation, Cambridge University, Cambridge, UK, 1938.

In accordance with § 9 Para. 12 APO, I hereby declare that I wrote the preceding master's thesis independently and did not use any sources or aids other than those indicated. Furthermore, I declare that the digital version corresponds without exception in content and wording to the printed copy of the master's thesis and that I am aware that this digital version may be subjected to a software-aided, anonymised plagiarism inspection.

Bamberg, 16.11.2012 Alan Turing