Automated Software Vulnerability Management

Seminar Paper
MA-INF 3317 Selected Topics in IT Security
Version 1.1

December 15, 2016

Ehab Qadah

Supervisor: Luis Alberto Benthin Sanguino

Table of Contents

Abstract. One of the main concerning areas for most organizations is software vulnerability analysis. To avoid cyber-attacks organizations continuously try to identify the vulnerable software. In this paper, we discuss techniques and systems to automatically monitor software vulnerabilities using open standards and public vulnerability information repositories or alternative sources such as social media and developer blogs.

1 Introduction

One of the main concerning areas for most organizations is software vulnerability analysis to ensure certain level of security, this area is important because of the continuous discovery of new software vulnerabilities (on daily basis) that open the doors for cyber-attacks. All organizations must be aware of the public known software vulnerabilities to perform the needed actions to avoid any kind of theft or damage for their computing assets. These actions include installing the corresponding patches if available. Information about reported software vulnerabilities can be retrieved from public repositories like the National Vulnerability Database (NVD)[1].

Finding the vulnerability information related to the organization's IT assets is usually time and resource consuming task, du the fact of required manual retrieving and processing of vulnerability information by human resources. This issue motivates to build automated software vulnerability management systems.

1.1 Software Vulnerability Management

Vulnerability is defect or weakness in system leads to security incident or violation such as inappropriate access to the system, that could be exploited by attacker or accidentally triggered defined by G. Stonebumer et al. in [1].

The Software Vulnerability Management concept can be defined as the process of identifying related vulnerabilities in software that is installed inside an organization. The process first requires managing the inventory of the organization's IT assets and periodically search for the related vulnerabilities. In case of the discovery of a vulnerability, a certain actions should be performed such as alerting the system administrators or trying to install the corresponding patches to avoid possible threats based on those known vulnerabilities.

2 Automated Software Vulnerability Management System

This section presents the proposed technique and system by Takahashi et al. in [3] to automatically monitor vulnerabilities of computing assets inside the IT infrastructure of an organization. Their main contribution is to automate the process of vulnerability management using open standards and tools. They

[1] `https://nvd.nist.gov/`

used open standards and data to make their system usable by a wide range of organizations.

The proposed system first collects information about the list of IT assets inside an organization, the system stores the collected information about the organization's IT assets and uses it to determine the corresponding standards identifiers (CPE-IDs)[2] for each IT asset. Then the system utilizes these identifiers to check the existence of related vulnerabilities by querying vulnerability repositories and using computed identifiers as keys. Finally an alert about the identified security defects will be sent to the system administrator by the proposed system.

2.1 System Components

The system proposed in [3] is composed by 4 elements, which are described in the following:

1. **Terminals**: This element includes all electronic devices used by the organization's employees to perform their job activities. In most cases, an agent is installed on a terminal to collect information about the installed IT assets on it. The collected information is then sent to the asset management server.
2. **Asset Management Server**: this component is responsible to communicate with the installed agents on the terminals to collect information about the IT assets, and in order to collect information about the terminals without installed agent the organization's network traffic is monitored and analyzed by the asset management server. The collected information is organized and stored using defined system's schema. Also asset management server determines the IT assets standard identifiers using the collected information and append these identifier to the stored IT assets information. In addition this element checks for the presence vulnerability by querying the vulnerability knowledge base.
3. **Vulnerability Knowledge base**: is the local system's database for the vulnerability information to compose data from different vulnerability repositories like NVD.
4. **Administrator terminal**: is the console used by the system administrator which receives the notification alert about the discovered security vulnerabilities from the asset management server.

[2] Common Platform Enumeration CPE-IDs are explained in section 3.

2.2 System Workflow

In this section the workflow of the proposed system is described.

1. **Compile IT assets information**: the system starts with the process of collecting the information on the organization's IT assets. This is achieved by sending requests to the agents installed on the terminals, or by monitoring the network. The agents gather the information of the installed software, and then, an XML document is generated. This document contains, for instance, the software name, version, publisher, and installation date, as shown in Figure 1. [3]

```
<?xml version="1.0" encoding="UTF-8"
    standalone="yes"?>
<assetInfo version="1">
    <!-- SNIP -->
    <installedSoftwareInfo version="1">
        <softwareInfo>
            <name>Adobe Flash Player 23.0.0.205</name>
            <version>23.0.0.205</version>
            <publisher> Adobe Systems Software </publisher>
            <size>0x24e23</size>
            <installationDate>20161122</installationDate>
        </softwareInfo>
        <!-- SNIP -->
    </installedSoftwareInfo>
</assetInfo>
```

Fig. 1. An example of collected information by the proposed system for an installed software on some terminal.

2. **Resolving of IT asset identifier**: the system determines the CPE-IDs for the IT assets using the stored information from the first stage to uniquely identify them. The CPE dictionary [4] is obtained from NVD repository and used as reference CPE-IDs for the proposed system. The basic algorithm builds a query from the collect software information like name, version and owner, then a matching rate (percentage of the similar characters between the query and CPE-ID) for all CPE-IDs in the CPE dictionary is calculated. '

[3] Based on figure 4 in [3] with omitting some details
[4] `https://nvd.nist.gov/cpe.cfm`

The CP-ID with the highest matching rate is selected and added to asset stored information (represented by XML field cpe) as seen in Figure 2.

```
<cpe id="cpe:/a:adobe:flash_player:23.0.0.205" matchingRate="7.654244"/>
```

Fig. 2. An example of calculated CPE-ID with the matching rate value.

3. **Finding Vulnerability Information**: The system uses the determined CPE-IDs to query the vulnerability knowledge base for related vulnerabilities and in case of discovering a vulnerability the system administrator is notified by alert containing the CPE and CVE data.

3 Standards

This section introduces the most related standard for the software vulnerability management, which is the Security Content Automation Protocol (SCAP).[5]

3.1 SCAP

SCAP is a collection of open standards and enumerations for the security related software flaws and configuration issues, the exchange of these standards offers the ability to automate vulnerability management[2].

It is provided and maintained by the National Institute of Standards and Technology (NIST) [6]. The repository of the SCAP content is the National Vulnerability Database (NVD), that provides a data feeds for each SCAP standard which can be publicly accessed by the security community. Also NVD is managed by NSIT.

The SCAP Standard is composed by six components. In the following we describe the components that can be utilized by vulnerability analysis systems.

1. **Common Vulnerabilities and Exposures (CVE)**[7] : is a list of known security software venerabilities (available in XML format) With unique standard identifiers (e.g. CVE-2016-7892) is assigned. The CVE standard identifiers allow the data exchange between security solutions and vulnerability repositories. The CVE list is managed by The MITRE Corporation[8].

[5] https://scap.nist.gov/
[6] https://www.nist.gov/
[7] https://cve.mitre.org/index.html
[8] https://www.mitre.org/

The CVE list is not a vulnerability repository[9], it is just a list of identifiers for the known vulnerabilities with basic information such as standard identifier and description. It designed to allow the linking between different vulnerability repositories. The NVD provides a XML vulnerability feed that built based on the CVE list. The CEV entries enriched with additional information by NVD such as corresponding CVSS base score and CPE mapping.

2. **Common Platform Enumeration (CPE)**: is the identifiers dictionary for all software products and applications,operating systems and hardware devices. The official CPE dictionary[10] provided in XML format by NVD.
3. **Common Vulnerability Scoring System (CVSS)** [11] : is a scoring system for the software vulnerabilities, which provides a relative severity for the vulnerability.

4 Alternatives to the NVD Repository

This section describes alternative vulnerability data source to the traditional vulnerability repositories such as NVD for the software vulnerability monitoring task. Due the regular delay of revelation of security related defects by the typical vulnerability information sources detection of zero day venerabilities is difficult. In order to solve this issue a technique to collect the security information from completely new data source which is the Social Media like Twitter to monitor software vulnerabilities was proposed in [4].

The proposed system takes the advantage of getting informed about security vulnerability information earlier than the normal software vulnerabilities repositories, and that because of the widespread usage of social media platforms and technical blogs to discuss the security software bugs and issues between the software developers and experts communities. On the other hand the classical information sources (such as NVD) wait the availability of patches from the software vendor before publishing the vulnerability information.

The introduced system is called SMASH (Social Media Analysis for security on HANA) consist of two subsystems namely data collection and processing. The data collection subsystem is responsible to gather the security information from the social media (Twitter [12]) by searching the Twitter stream content to related security information (by using security associated terms e.g exploit) and store it in the local database to be utilized by the system later. The system also keeps a copy of the software vulnerabilities form NVD (XML vulnerability feed) to be used in recognizing the new from already published vulnerabilities.

The data processing subsystem is performing the extraction of security information by analyzing the stored Twitter content using various data mining

[9] The list of vulnerability databases list can be found at `https://cve.mitre.org/compatible/product_type.html#VulnerabilityDatabase`

[10] The Official CPE Dictionary available at `https://nvd.nist.gov/cpe.cfm`

[11] `https://www.first.org/cvss`

[12] Twitter has been used in the proposed system prototype, but the technique is also applicable for other social media platforms

techniques. The extracted security information includes zero-day vulnerabilities which are not published yet, and requests to create new CVE or update old entry.

The proposed system offers the functionality of monitoring certain software components selected by the system's users, then the discovered vulnerabilities by the system are displayed to the users.

5 Discussion

The proposed technique and system in Section 2 mainly focuses on the automation of software vulnerabilities management using open standards and public vulnerability repositories. The performed action by the proposed system in case of a vulnerability discovery can be argued as simple, which does not follow the system requirement of avoiding the manual operation. The system just sends an alert for the system's administrator who is responsible to perform the needed action based on the provided vulnerability information from the system. To achieve the goal of having acceptable level of automation, the system should execute fast and immediate actions such as blocking the corresponding terminal or even the vulnerable software to ensure security inside the organization. Furthermore, the accuracy of the system's outcome mainly depends on the precision of the determination IT assets identifiers (CPE-IDs). The proposed system finds the CPE-ID for an IT asset based on a numerical method of calculating a matching rate, this approach does not always give an accurate matching[13]. Also the determined CPE-IDs must be validated by the system before moving to the further steps to avoid useless processing or invalid alerts.

The proposed technique and system in Section 4 is trying to detect the vulnerable softwares as fast as possible by utilizing a new different vulnerability information source which is the social media (Twitter). The system analyzes the social media content to extract the security vulnerability information. This approach offers the opportunity to detect zero-day vulnerabilities which is not feasible using classical channels such as NVD. Despite that this approach may utilize non validated or wrong information to identify security software vulnerabilities, which forces the proposed system to have a trust module of the extracted information from the social media data, which means the accuracy of the system's output fundamentally affected by the strength of the users trust model.

[13] The authors indicate that the match rate approach does not always give accurate results.

6 Conclusion

The importance of automatic detection and monitoring of vulnerable softwares inside the organizations to avoid cyber-attacks introduces the need software vulnerability management systems. The two proposed systems allow the organizations to get informed about software vulnerabilities. But they do not replace the normal security solutions and polices but they just complete their work and ensure an efficient level of security inside the organization. The role of vulnerability information repositories like NVD and the open standards of SCAP protocol is the main actor in building the software vulnerability management systems.

References

1. G Stoneburner, A Goguen, and A Feringa, Risk Management Guide for Information Technology Systems, NIST Special Publication 800- 30, July 2002 http://csrc.nist.gov/publications/nistpubs/800-30/sp800-30.pdf
2. The Technical Specification for the Security Content Automation Protocol (SCAP) NIST Special Publication 800-126 Revision 3.
3. Takahashi, Takeshi, Daisuke Miyamoto, and Koji Nakao. "Toward automated vulnerability monitoring using open information and standardized tools." 2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops). IEEE, 2016.
4. Trabelsi, Slim, et al. "Mining social networks for software vulnerabilities monitoring." 2015 7th International Conference on New Technologies, Mobility and Security (NTMS). IEEE, 2015.