

# **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

1	Introduction.....	3
1.1	Software Vulnerability Management .....	3
2	Automated Software Vulnerability Management System.....	3
2.1	System Components .....	4
2.2	System Workflow.....	5
3	Standards.....	5
3.1	SCAP .....	5
4	Alternatives to the NVD Repository.....	7
5	Discussion .....	9
6	Conclusion .....	9

**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)<sup>1</sup>.

Finding the vulnerability information related to the organization's IT assets is usually time and resource consuming task, that 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, and to make available in a wide range of organizations.

---

<sup>1</sup> <https://nvd.nist.gov/>

The proposed system first collects a list of IT assets, the system uses the collected and stored information about the organization's IT assets to find the corresponding standards identifiers (CPE-IDs)<sup>2</sup>, then the system utilizes those identifiers to check the existence of related vulnerabilities by querying the vulnerability repositories. 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 computing assets of the organization, and to collect information about the terminals without installed agent the asset management server monitors and analysis the organization's network traffic. The asset management server determines the IT assets standards identifiers using the collected information. Afterward this element checks for the vulnerabilities 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 found security vulnerabilities from the asset management server.

---

<sup>2</sup> Common Platform Enumeration (CPE) IDs 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.<sup>3</sup>
2. **Resolving of IT assets identifiers:** the system determines the CPE-IDs for the IT assets using the collected information from the first stage, the CPE dictionary<sup>5</sup> obtained from NVD repository is used as reference CPE-IDs for the proposed system. The basic algorithm to extract the corresponding CPE-IDs for the IT-assets builds a query from the collect information like name, version and owner and calculate the matching rate (percentage of the similar characters) for each CPE-ID record in the CPE dictionary and select the CP-IDs with the highest matching rate and these CPE-IDs added to assets stored information as seen in Figure 1.
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).<sup>6</sup>

### 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)<sup>7</sup>. The repository of the SCAP content is the National Vulnerability Database (NVD), that provides a data feeds for each SCAP standard

<sup>3</sup> based on figure 4 in [3]

<sup>4</sup> The information in Figure 1 also include cpe and cve which generated by the system in later stages

<sup>5</sup> <https://nvd.nist.gov/cpe.cfm>

<sup>6</sup> <https://scap.nist.gov/>

<sup>7</sup> <https://www.nist.gov/>

```

<?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>
      <cpe id="cpe:/a:adobe:flash-player:23.0.0.205"
matchingRate="7.654244"/>
      <cve id="CVE-2016-4273"/>
      <cve id="CVE-2016-6982"/>
    </softwareInfo>
  <!-- SNIP -->
</installedSoftwareInfo>
<networkInfo version="1">
  <hostName>ehab-qadah-pc</hostName>
  <openPorts>
    <port>444</port>
    <!-- SNIP -->
  </openPorts>
  <nicInfoList>
    <gateWay>131.220.207.254</gateWay>
    <ipAddress>131.220.198.146</ipAddress>
    <macAddress>255.255.240.0</macAddress>
    <subnetMask>255.255.252.0</subnetMask>
    <nicName>Intel(R) 82578DM Gigabit Network Connection</nicName>
  </nicInfoList>
  <!-- SNIP -->
</networkInfo>
</assetInfo>

```

**Fig. 1.** An example of the collected information by the proposed system for a terminal.

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[2].

1. **Common Vulnerabilities and Exposures (CVE)**<sup>8</sup>: is a list of known security software vulnerabilities (available in XML format) With unique standard identifiers (e.g. CVE-2008-2948) 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<sup>9</sup>. The CVE list is not a vulnerability repository<sup>10</sup>, 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 the XML vulnerability feed that built based on the CVE IDs. CEV entries enriched with additional information by NVD such as corresponding CVSS base score and CPE mapping.
2. **Common Platform Enumeration (CPE)**: is identifiers dictionary for all software products and applications, operating systems and hardware devices. The official dictionary provided in XML format by NVD.<sup>11</sup>
3. **Common Vulnerability Scoring System (CVSS)**<sup>12</sup> : is a scoring system for the software vulnerabilities, which provides a relative severity for the vulnerabilities.

## 4 Alternatives to the NVD Repository

This section describe 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 vulnerabilities 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 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 before publishing the vulnerability information.

<sup>8</sup> <https://cve.mitre.org/index.html>

<sup>9</sup> <https://www.mitre.org/>

<sup>10</sup> The list of vulnerability databases list can be found at [https://cve.mitre.org/compatible/product\\_type.htmlDatabase#Vulnerability](https://cve.mitre.org/compatible/product_type.htmlDatabase#Vulnerability)

<sup>11</sup> The Official CPE Dictionary available at <https://nvd.nist.gov/cpe.cfm>

<sup>12</sup> <https://www.first.org/cvss>

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 <sup>13</sup>) by searching the Twitter stream content to related security information 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 the already published vulnerabilities.

The data processing subsystem is performing the extraction of security information from the stored Twitter content to detect the zero-day vulnerabilities which are not published yet, by using various data mining techniques.

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.

---

<sup>13</sup> Twitter has been used in authors prototype, but the technique is also applicable for other social media platforms



## 5 Discussion

The proposed technique and system in Section 2 mainly focuses on the automation of software vulnerabilities monitoring using the open standards and public vulnerability repositories. The performed action by the proposed system in case of vulnerability discovery can be argued as simple, it 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 need action based on the provided vulnerability information. To achieve the goal of full automation level, the system should execute fast and immediate actions such as blocking the corresponding terminal or even the software component to ensure security inside the organization. Furthermore, the accuracy of the system's outcome mainly depends on the accuracy of the determination of CPE-IDs for all IT assets inside the organization, the proposed systems find the CPE-IDs based on a numerical method of calculating the match rate, this approach does not always give an accurate matching<sup>14</sup>. Also the determined CPE-IDs should be validated by the system before moving to the further steps to avoid useless processing and 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 like Twitter. The system analyzes the social media content to extract the security vulnerability information. This approach offers the opportunity to detect vulnerable softwares before publishing the vulnerability information in the classical channels such as NVD, but this approach may utilize non validated or wrong information identify security software vulnerabilities, which forces the proposed system to have trust module of the extracted information from the social media data, that means the accuracy of the system's output fundamentally affect by the strength of the users trust model.

## 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 discussed two software vulnerability management systems do not replace the normal security solutions and producers but they just complete their work and ensure an efficient level of security. 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.

---

<sup>14</sup> The authors indicate that the match rate approach does not always give accurate results.

## 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.