International Journal of Advanced Science and Technology
Vol.112 (2018), pp.157-166
http://dx.doi.org/10.14257/ijast.2018.112.15

An Analysis of Common Vulnerability and Exposure (CVE) of Software Products in the Year 2016

Afiq Bonandir[1] and Salman Yussof[2]

[1]*Tata Consultancy Services*
[2]*College of Computer Science and Information Technology*
Universiti Tenaga Nasional, Malaysia
afiq.bonandirTCS@tnb.com.my, salman@uniten.edu.my

Abstract

This research aims to analyze the common vulnerabilities and exposures of software products that have been discovered in 2016. The dataset used comes from Common Vulnerabilities and Exposures (CVE) database which is used today as an international standard for vulnerability numbering or identification. Some of the information in this research comes from National Vulnerability Database (NVD) at the National Institute of Standards and Technology (NIST). In order to find vulnerability metrics, normalization of the dataset has been done to reduce data redundancy and properly organize the attributes of the data. The data attribute will be structured based on the Serkan Ozkan method, who is the founder of the CVE details website. This research can help organizations to make an informed decision with respect to software security. Choosing software or vendor which are known to have less security vulnerabilities can help to improve information security in the organization. In this research, it has been discovered that most of the products that have critical and high severity comes from Adobe and Google.

Keywords: *Software vulnerability; CVSS; CVE data analytics.*

1. Introduction

Computer software is a general term that describes computer programs. Related terms such as software programs, applications, scripts, and instruction sets all fall under the category of computer software. By definition, a computer software is a set of statements or instructions to be used directly or indirectly in a computer in order to bring about a certain result. Software is often divided into application and system software [1]. Nowadays, software developer competes with each other to produce new product to the market. Software designer and architect face a huge pressure during the development of a new product. In a development phase, the developer should properly plan the system development life cycle of the product. By this way, they can minimize bugs and flaws of the software that may exist. As a result, more secure software can be built [2]. Vulnerabilities in a software product may cause it to be exploited, which could lead to various disastrous consequences to the users and their organizations. When this happens, it may cause the user and the system itself to be exposed to cyber security threats. The chances of user to be exposed to security threats such as virus and malware will be significantly increased [3]. A security vulnerability can be defined as a flaw in code or design that creates a potential point of security compromise for an endpoint or network [4]. Vulnerabilities create possible attack vectors, through which an intruder could run code or access a target system's memory. The means by which vulnerabilities are exploited vary and include code injection and buffer overruns; they may be conducted through hacking scripts, applications and free hand coding [4]. Security vulnerability of

Received (August 14, 2017), Review Result (November 20, 2017), Accepted (November 24, 2017)

ISSN: 2005-4238 IJAST

the software product that has been discovered can be classified into different types of vulnerabilities such as SQL injection, buffer overflow and memory corruption. The severity of the security vulnerability can be evaluated using Common Vulnerability Scoring System (CVSS). In this research, the security vulnerability was evaluated using CVSS v2 scoring system. Although the current version of CVSS is CVSS v3, CVSS v2 scoring system was used because CVSS v3 scoring system is a new version of CVSS scoring system and some of the vulnerabilities that exist cannot be evaluated by using CVSS v3 due to lack of data attribute and information on the software vulnerability itself, which has not been disclosed to the public.

2. CVSS v2 Scoring Metrics

The Common Vulnerability Scoring System (CVSS) is a free and open industry standard for assessing the severity of computer system security vulnerabilities. CVSS attempts to assign severity scores to vulnerabilities, allowing responders to prioritize responses and resources according to threat [5]. In computer security practice, this score can be used to rate security vulnerability and indicate how serious the vulnerability is. CVSS is composed of three metric groups: Base, Temporal, and Environmental, each consisting of a set of metrics, as shown in Figure 1 below [5].

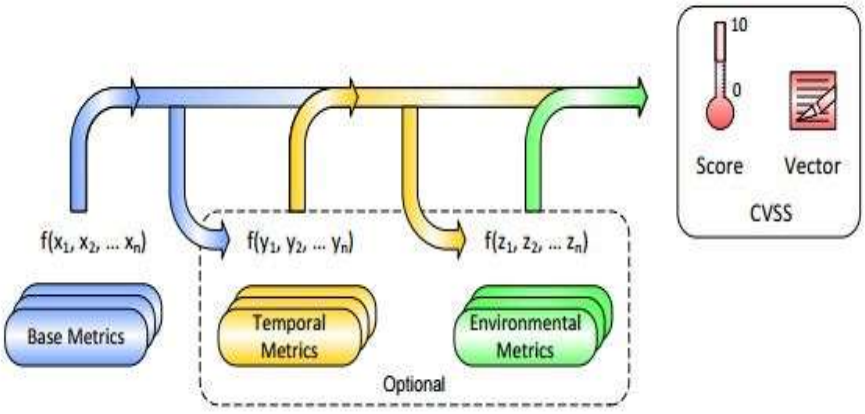

Figure 1. CVSS Metrics and Equations [5]

Base metrics represent the intrinsic and fundamental characteristics of a vulnerability that are constant over time and user environments. Temporal metrics represent the characteristics of a vulnerability that change over time but not among user environments. Environmental metrics represent the characteristics of a vulnerability that are relevant and unique to a particular user's environment [5].

As shown in Figure 1 above, only the base metrics are compulsory in order to evaluate the CVSS score, while the other two which are temporal metrics and environmental metrics are just optional. For this research, only the base metrics are considered. Temporal and environmental metrics are useful when evaluating the vulnerability and risk posed over time or when evaluating the vulnerability and risk for a particular user's environment. However, these do not apply to this research. Furthermore, there is insufficient data to determine the parameter values for the environmental and temporal metrics.

2.1. The Base Metrics

The Base Metrics consist of several components. These components and their description are given in Table 1.

Table 1. Base Metric Components [5]

Base Metrics Component	Explanation	Metric Value & Description
Access Vector (AV)	This metric reflects on how the vulnerability can be exploit. The more remote vulnerability can be exploited the higher the rating.	Network : Can be exploit remotely. Adjacent : Can be exploit remotely but limited to the same physical or logical network. Local : Not require network to exploit the vulnerability. It also can be exploit by physically access to vulnerability.
Access Complexity (AC)	This metric reflects the complexity of the attack required to exploit the vulnerability.	Low : Specialized condition does not exist and an attacker can expect repeatable success against the vulnerable component. Medium : Specialized condition somewhat exist and an attacker must spend some amount of effort to exploit the vulnerable component. High : Specialized condition exist that make attacker must spend some amount of effort to exploit the vulnerable component.
Authentication (Au)	This metric measures the number of times an attacker must authenticate to a target in order to exploit vulnerability.	None : Authentication does not required to exploit vulnerability. Single : Authentication require in order to exploit vulnerability. Multiple : Authentication required two or more in order to exploit vulnerability.
Confidentiality Impact (C)	This metric reflects information confidentiality of a successfully exploited vulnerability.	None : There is no loss of information confidentiality on the system.

		Partial : There is some loss of information confidentiality. High : There is total loss of information confidentiality.
Integrity Impact (I)	Integrity refers to the trustworthiness of the information. This metrics reflects integrity of a successfully exploited vulnerability.	None : Modification of system files is impossible. Partial : Modification of some system files is possible. High : Modification of entire system files is possible.
Availability (A)	Availability refers to on how information resources can be access. This metrics reflects availability of a successfully exploited vulnerability.	None : Performance and resources does not effected. Partial : Performance and resources reduced a little bit. Complete : Performance and resources can be control successfully by attacker.

3. Research Methodology

This research was conducted in two different phases. Each phase has a different objective. The output of the first phase contributes towards conducting the second phase.

PHASE 1: Producing vulnerability metric

The objective of this phase is to produce vulnerability metric from CVE-DETAILS dataset. In order to achieve the objective, the following steps are conducted:

STEP 1: Review the related works on common vulnerabilities and exposures
In step 1, a review on the past works that are related to common vulnerabilities and exposure issues are conducted. In this step, information such as types of security vulnerability will be collected to make sure the vulnerability that has been discovered can be classified properly.

STEP 2: Input data
Data from CVE-DETAILS database were collected. This data will be used to produce the vulnerability metric. Only data from the year 2016 were used.

STEP 3: Identify attributes
In this step, the attributes of the collected data were identified. Common vulnerabilities and exposures attributes of the dataset were identified based on the previous works of other researchers.

STEP 4: Extracting attributes
In this step, the attributes of the dataset that has been identified from the previous step were extracted. An analysis was conducted to find common vulnerabilities and exposures patterns. After that, the product vulnerability metric was produced.

PHASE 2: Vulnerability metrics visualization

The objective of this phase is to develop the visualization dashboard for the vulnerability metric produced in Phase 1 in order to better understand the result.

STEP 1: Input data
The input data that will be used for this step is the vulnerability metric that has been obtained from the previous phase, which is the vulnerability metric from CVE-DETAILS dataset.

STEP 2: The development of dashboard visualization
The dashboard visualization was developed based on the input data that has been obtained from the previous phase. The dashboard visualization was used to get a comprehensive view on the data in order to perform analysis and derive conclusion regarding the product vulnerabilities.

4. Analysis

Figure 2 shows the distribution of CVSS score on product vulnerability. Based on the figure, it can be seen that 55% of software products in 2016 have been discovered to have vulnerabilities with critical severity. This is followed by high severity with 27%, medium severity with 17% and low severity with only 1%. This shows that more than 80% of software products in 2016 contain vulnerabilities with high and critical severities.

Figure 3 shows the top vulnerability types in 2016. The top four software vulnerabilities found in 2016 are code execution with 25%, DoS with 21%, overflow with 20% and memory corruption with 16%. These four types of vulnerability contribute to more than 80% of the vulnerabilities found.

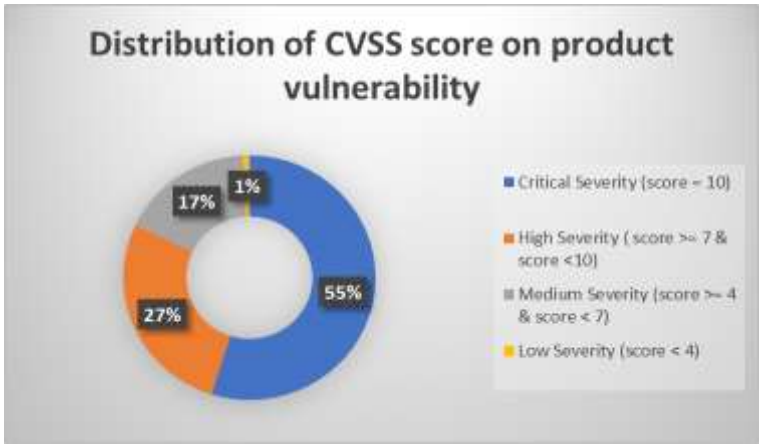

Figure 2. Distribution of CVSS Score on Product Vulnerability

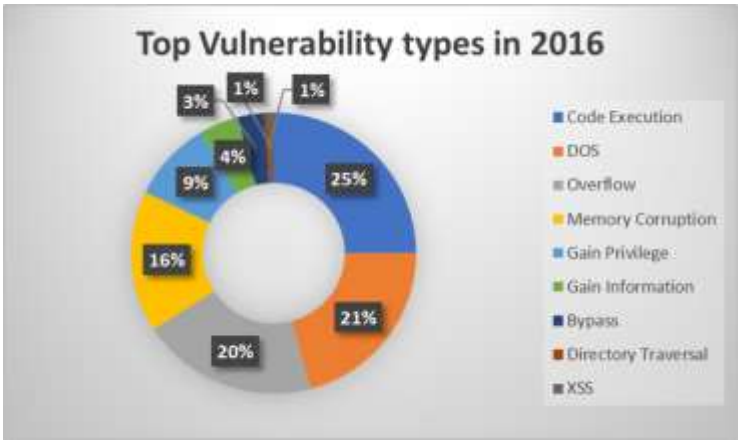

Figure 3. Top Vulnerability Types in 2016

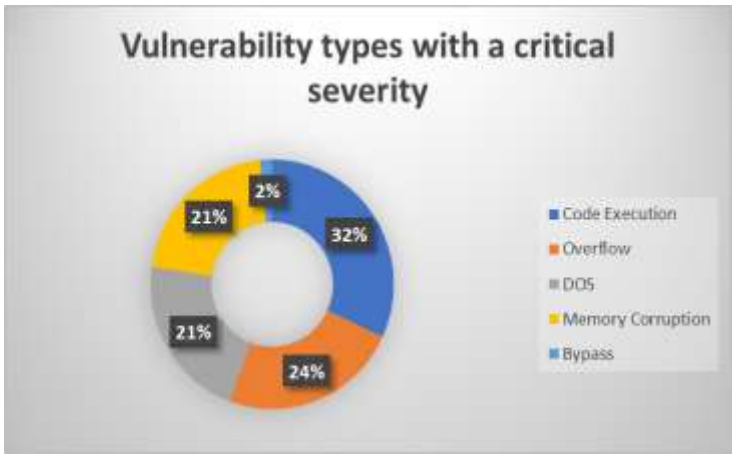

Figure 4. Top Vulnerability Types with Critical Severity in 2016

Figure 4 shows the top vulnerability types with critical severity. Based on the figure, there are four major types of vulnerability with high severity which are code execution with 32%, overflow with 24%, DoS with 21% and memory corruption with 21%. Together, they represent 98% of the vulnerabilities with critical severity.

Figure 5 shows the top vulnerability types with high severity. Based on the figure, there are four major types of vulnerability with high severity which are gain privilege with 28%, DoS with 21%, code execution with 19%, and overflow with 17%. Together, they represent 85% of the vulnerabilities with high severity.

Figure 6 shows the top vendors in terms of vulnerability in 2016. Based on the figure, it can be seen that 62% of software products that have been discovered to have vulnerabilities are Adobe product. This is followed by Google products with 24% and Apple product with 10%. Together, products from these three vendors contribute to 96% of the software that contain vulnerabilities.

Figure 7 shows the top vendors in terms of vulnerability with critical severity in 2016. Based on the figure, it can be seen that Adobe products dominate the list of software products that has been discovered to have vulnerabilities with critical severity with 93%. The second place goes to Google products with 6%, followed by Apply products with 1%.

Figure 8 shows the top vendors in terms of vulnerability with high severity in 2016. Based on the figure, it can be seen that 50% of software products that has been discovered to have vulnerabilities with high severity are Google products. This is followed by Adobe products with 24% and Apple products with 20%.

To get a more detail picture on software vulnerability, analysis on the actual products has also been conducted. Figure 9 shows the top products that have been discovered with vulnerabilities in 2016, arranged according to the number of reports received. Based on the figure, it can be seen that Android is the most vulnerable product with 704 vulnerability reports. This is followed by Acrobat Reader DC with 658 vulnerability reports and Acrobat with 655 vulnerability reports. Android is Google product while both Acrobat Reader DC and Acrobat are products from Adobe.

Figure 10 shows the top products that have been discovered to contain vulnerabilities with critical severity, arranged according to the number of reports received. Based on the figure, it can be seen that Acrobat Reader DC is the product with the highest number of vulnerabilities with critical severity, receiving 599 vulnerability reports. This is followed by Acrobat with 596 vulnerability reports and Air SDK & Compiler with 107 vulnerability reports.

Figure 11 shows the top products that have been discovered to contain vulnerabilities with high severity, arranged according to the number of reports received. Based on the figure, it can be seen that Android is the product with the highest number of vulnerabilities with high severity, receiving 704 vulnerability reports. This is followed by Acrobat Reader DC with 658 vulnerability reports and Acrobat with 655 vulnerability reports.

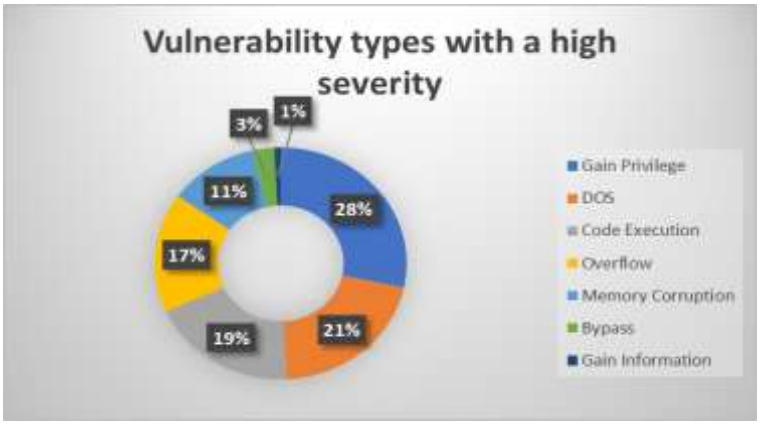

Figure 5. Top vulnerability Types with High Severity in 2016

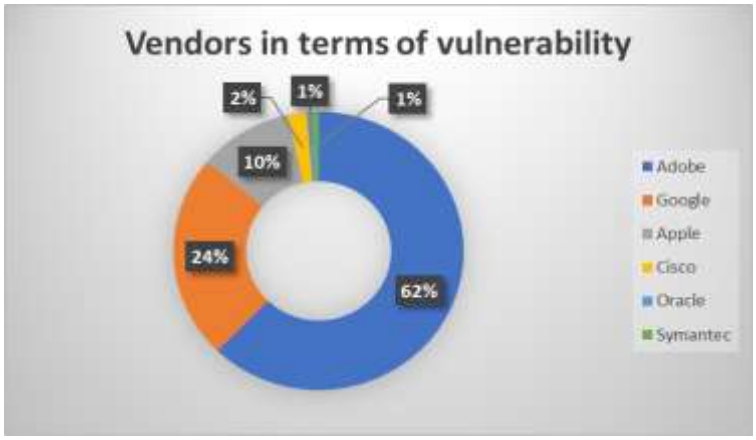

Figure 6. Top Vendors in Terms of Vulnerability in 2016

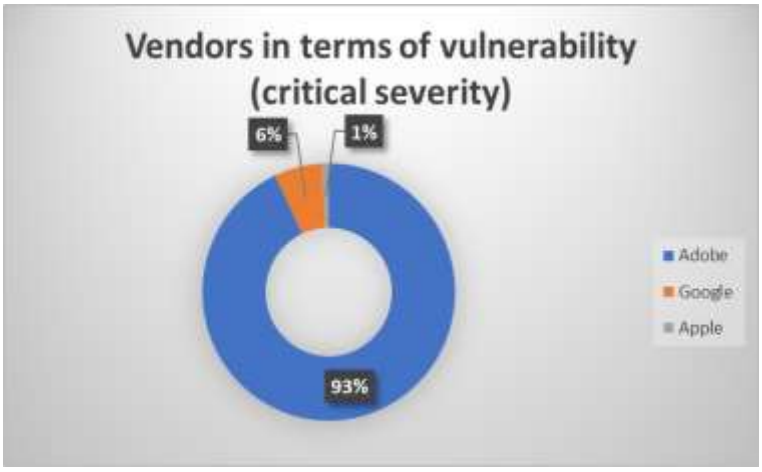

Figure 7. Top Vendors in Terms of Vulnerability with Critical Severity in 2016

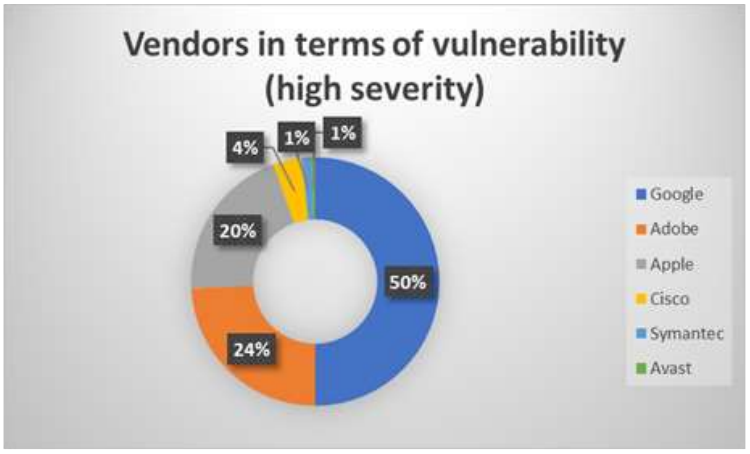

Figure 8. Top Vendors in Terms of Vulnerability with High Severity in 2016

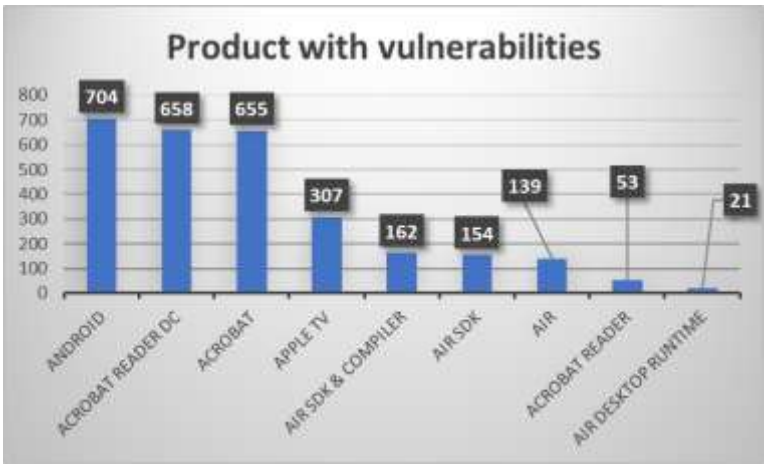

Figure 9. Top Products with Vulnerabilities in 2016

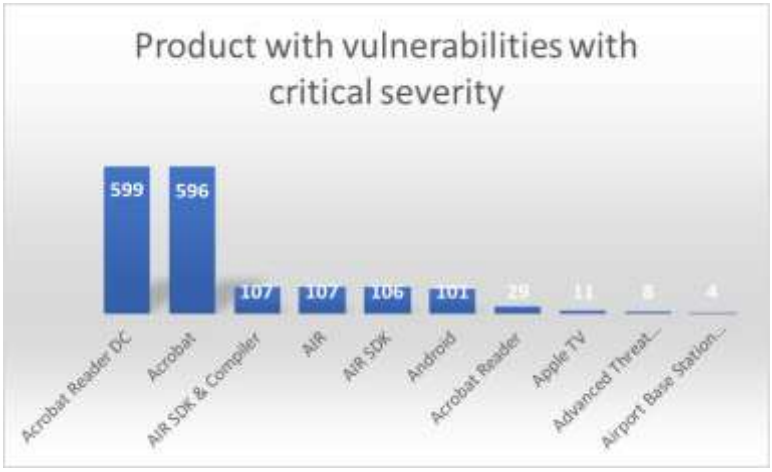

Figure 10. Top Products with Critical Severity Vulnerabilities in 2016

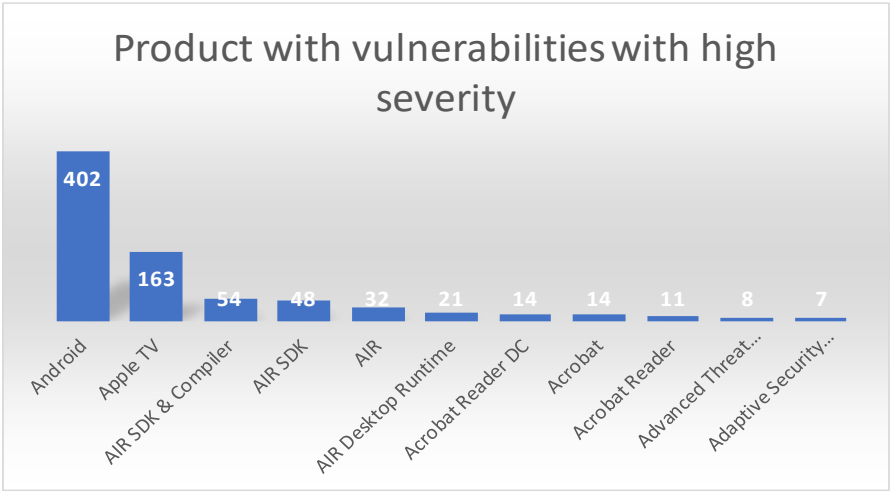

Figure 11. Top products with High Severity Vulnerabilities in 2016

5. Conclusion

In this paper, we had performed an analysis on the product vulnerabilities that have been discovered in 2016. It has been discovered that 82% of the vulnerable products that have been discovered in 2016 have vulnerabilities with either high or critical severity. The top vulnerability for critical severity is the code execution vulnerability while the top vulnerability for high severity is the gain privilege vulnerability. Both Adobe and Google top the list of products with critical and high severity. Given the popularity of products from these two vendors, many users could be at risk and it is important for users to apply patches and updates that have released for the products that they are using.

References

[1] J.J. Cardona and G. de Alaiza, Open Source, Free Software, and Contractual Issues. Tex. Intell. Prop. LJ., **(2006)**, 15, p. 157.
[2] C. Banerjee and S.K. Pandey, Software Security Rules, SDLC Perspective. International Journal of Science and Information Security, **(2009),** 6(1).
[3] A.M. Algarni and Y.K. Malaiya, Software Vulnerability Markets: Discoverers and Buyers. International Journal of Computer, Electrical, Automation, Control and Information Engineering. **(2014),** 8(3), pp. 71 – 81.

[4] A. Shostack and D. Allouch, Computer Security. U.S. Patent 6,298,445, October 2 **(2001)**.
[5] K. Scarfone and P. Mell, A complete guide to the common vulnerability scoring system version 2.0. In Published by FIRST-Forum of Incident Response and Security Teams, Jun **(2007)**, vol. 1, p. 23.

