A Software Security Assessment System Based On Analysis of Vulnerabilities

Chenmeng Sui, Yanzhao Liu, Yun Liu

China Information Technology Security Evaluation Center, Beijing, China, liuyz@itsec.gov.cn
Corresponding Author Key Laboratory of Communication and Information Systems, Beijing Jiaotong University, Beijing, China, 10120220@bjtu.edu.cn, liuyun@bjtu.edu.cn

Abstract

In recent years, software security plays an important role in verifying system safety and avoiding the casualties and property losses, but it is difficult to assess system security in traditional software engineering and software test. Focusing on the software security assessment system, this paper based on the vulnerability analysis method, which takes the advantage of both qualitative analysis and quantitative analysis to assess the security of software. The major methods of domestic and foreign software security measurement are discussed. Through the comparative analysis of existing vulnerability rating system, especially the CVSS and VRSS, this paper discovers their respective advantages and proposes a more accurate rating system to obtain the final security score of the software system. An example application is provided in the paper, which shows that the assessment system we proposed is practical.

Keywords: software security, vulnerability analysis, CVSS, VRSS, Quantitative scoring, SSAS

1. Introduction

Software security refers to the software within the stipulated time will cause harm to the system itself and systems outside probability, such hazards include physical security, substantial property damage and great expectations of the event.[1,2] As one of the most important attributes for trustworthy software, security can protect people's property and interests will not be threatened. Therefore, the trustworthiness of the software referred to the security is attracted high attention by researchers.

Because of the importance of software security, many security experts at home and abroad are working on software security evaluation. For example, the business transaction information is based on computer network security, create a relatively safe, reliable, and convenient E-business applications, the information to provide better and reliable protection[3]. Because of the importance of software security, many security experts at home and abroad are working on software security evaluation. At present, the most frequently used network security evaluating methods are risk evaluating method and leak scanning tool software to test and evaluate the security degree of the system[4]. However, measuring security is hard because the discipline itself is still in the early stage of development. According to the different characteristics, and the commonly used method of software security testing at home and abroad are divided into 3 categories: based on reliability analysis method, based on formal models method and based on software security testing method[5]. These methods have their advantages and disadvantages, but they do not specifically quantify a degree of security software. So the focus of our research is to put forward a software security assessment system that can give a specific score of the security.

Along with the software scalability, connectivity and complexity increases, more and more software flaw and vulnerabilities exposed. Attackers can exploit vulnerabilities in software to jeopardize intellectual property, consumer trust, and business operations and services. So vulnerability is the root causes of software security and analyzing and rating them is our first step to establish this security assessment system.
Over the past several years, a number of large computer security vendors and non-profit organizations have developed, promoted, and implemented procedures to analyze and rate the vulnerabilities. We combine the advantages of the CVSS (Common Vulnerability Scoring System), a vulnerability scoring system designed to provide an open and standardized method for rating software vulnerabilities[6], and the VRSS, a method for rating and scoring vulnerabilities, which combines historical vulnerability rating systems’ advantages[7], in our research. By improving and expanding these two systems, we propose a new method to score the vulnerability. The score of each vulnerability is eventually used in software security assessment system and the final security score can reflect the security of software.

The paper is organized as follows. Section 2 introduces the related content of software vulnerability, including the vulnerability properties and classification. Section 3 analyzes the rating systems of vulnerabilities, especially the CVSS and VRSS. Section 4 presents our improvement and proposes our new software security assessment system. Section 5 gives an example application to prove our system is practical. Finally, Section 6 is the conclusion of the paper and some future works will also be shown.

2. SOFTWARE VULNERABILITIES

In software security, vulnerabilities play a special role. A vulnerability means a bug, a flaw, a weakness, or an exposure of an application, system, device, or service which could lead to a failure of confidentiality, integrity, or availability [8,9]. RFC 2828 defining vulnerability as system design and flaws or weaknesses in the management or operation, can be exploited to violate system security policy. As Cheswick and Bellovin noted, “Any software, regardless of whether it looks how safe it is, of which hides a vulnerability”[10]. So understanding of the nature of the vulnerability is very helpful to our research on software security. First of all, we need to categorize vulnerabilities and analyze their attributes, so as to evaluate the impact of vulnerabilities on software security.

2.1. Vulnerability Classification

There are a great variety of different vulnerabilities existing for different kinds of software. So how to classify these vulnerabilities validly is a very practical problem. At present, the vulnerability classification mainly used in computer systems security assessment areas such as intrusion detection and software engineering[11]. Nearly 30 years, many research institutions and researchers have carried out work in this area. Comparison of productive work are: Landwehr proposed a three-dimensional property classification including vulnerability origin, time and location; Longstaff increased the classification of access and easy offensive; Power proposed a classification based on hazard; Du and Mathur’s method describes the causes, impact and restoration property of vulnerabilities; and Bishop’s 6-axis classification, Knight’s Four types classification and Venter’s harmonized classification[12]. Through the analysis of various classifications above, we can see that each of these classifications is in fact based on the vulnerability of certain attributes to classify. So effects of different types of vulnerabilities on software security in the end also were caused by their attributes.

2.2. Vulnerability Attributes

Now that we select vulnerabilities as metrics to evaluate software security, then the key question becomes the property of each category of vulnerability analysis and their specific impact on software security. Through research on security vulnerabilities in the past, we can find these basic attributes include: vulnerability types, causes, consequences and locations, and so on. In the related studies in the literature [12], professor Fang divided vulnerability attributes into 5 main categories: belonging-related, attack-related, cause-related, time-related and other attributes. And table 1 shows these five types of attributes subdivide again 22 specific attributes.
Not every vulnerability attribute is useful for the security assessment. So, choosing which properties to analyze and how to quantify these properties is what we have to focus on. Only through in-depth studies on various attributes of the vulnerability, so as to develop the most suitable analysis method and evaluation criteria of the project. In the following section, we will introduce two vulnerability rating systems.

3. CVSS and VRSS

Over the past several years, a number of large computer security vendors and non-profit organizations have developed, promoted, and implemented procedures to rank information system vulnerabilities. The United States computer emergency readiness team (US-CERT) vulnerability scoring system assigns a score between 0 and 180 for each vulnerability. The SANS Institute has a vulnerability scoring system that is used by a group of security researchers to assign severity ratings to certain vulnerabilities. Microsoft has a vulnerability rating system known as the Microsoft Security Response Center Security Bulletin Severity Rating System[13]. A plan for the new system, called the Common Vulnerability Scoring System (CVSS), was unveiled at the RSA Conference in San Francisco in February, 2005. National Computer Network Intrusion Protection Center recently proposed a new vulnerability rating and scoring system named VRSS for qualitative rating and quantitative scoring vulnerabilities. All the methods of evaluation, only the CVSS and VRSS can give a specific vulnerability score and the calculation details. Next, we focused on both of these methods.

3.1. CVSS

The Common Vulnerability Scoring System (CVSS) was introduced by the National Infrastructure Advisory Council (NIAC) and is now managed by the Forum of Incident Response and Security Teams (FIRST). CVSS provides an open framework for communicating the characteristics and impacts of IT vulnerabilities[9]. Since CVSS was published, it has been widely adopted by a variety of organizations.

CVSS metrics for vulnerabilities are divided into three groups: Base metrics measure the intrinsic and fundamental characteristics of vulnerabilities that do not change over time or in different environments. Temporal metrics measure those attributes of vulnerabilities that change over time but do not change among user environments. Environmental metrics measure those vulnerability characteristics that are relevant and unique to a particular user’s environment. Each group produces a numeric score ranging from 0 to 10, and a Vector, a compressed textual representation that reflects the values used to derive the score. The base metrics are necessary to the assessment system, while the temporal and environmental metrics are not required. So we mainly focus on the base metrics and Table 2 lists the base metrics and the possible values for each metric.
### Table.2 CVSS base metrics

<table>
<thead>
<tr>
<th>Base Metric</th>
<th>Metric value</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrity</td>
<td>None(N)/Partial(P)/Complete(C)</td>
<td>0/0.275/0.660</td>
</tr>
<tr>
<td>Availability</td>
<td>None(N)/Partial(P)/Complete(C)</td>
<td>0/0.275/0.660</td>
</tr>
<tr>
<td>Confidentiality</td>
<td>None(N)/Partial(P)/Complete(C)</td>
<td>0/0.275/0.660</td>
</tr>
<tr>
<td>Access Vector</td>
<td>Local(L)/Adjacent network(A)/Network (N)</td>
<td>0.395/0.646/1.0</td>
</tr>
<tr>
<td>Access Complexity</td>
<td>High(H)/Medium(M)/Low (L)</td>
<td>0.35/0.61/0.71</td>
</tr>
<tr>
<td>Authentication</td>
<td>None (N)/Single(S)/Multiple (M)</td>
<td>0.704/0.56/0.45</td>
</tr>
</tbody>
</table>

The base equation is the foundation of CVSS scoring. The base equation is:

\[
\text{BaseScore} = \text{round to 1 decimal}(((0.6 \times \text{Impact}) + (0.4 \times \text{Exploitability}) - 1.5) \times f(\text{Impact}))
\]

\[
\text{Impact} = 10.41 \times (1 - (1 - \text{ConfImpact}) \times (1 - \text{IntegImpact}) \times (1 - \text{AvailImpact}))
\]

\[
\text{Exploitability} = 20 \times \text{Access Vector} \times \text{Access Complexity} \times \text{Authentication}
\]

The scoring process first calculates the base metrics according to the base equation, which delivers a score ranging from 0 to 10, and creates a vector. This score can visually show an impact on basic security properties of software from a vulnerability. CVSS provides a common assessment method, helps security form a unified standard. So we apply its base metrics to our system.

### 3.2. VRSS

VRSS is a new vulnerability rating system and scoring system for qualitative rating and quantitative scoring vulnerabilities, which can combine respective advantages of all kinds of vulnerability rating systems. VRSS has two important steps. One is rating method and the other is scoring method. Figure.1 shows the framework of VRSS.

![Figure 1. The framework of VRSS](image)

According to the CVSS base metrics, VRSS use confidentiality, integrity, and availability properties to evaluate the risk level of vulnerabilities and the possible values for each of these three security properties are None, Partial and Complete. Each of the three metrics has three different values, so considering these three metric, there are 27 kinds of cases through combination of the three metrics. Ultimately there are 10 possible impact score and it is an intermediate result which will be used in the scoring method. In the second step, VRSS use exploitability metrics to calculate quantitative score. The exploitability metrics capture how the vulnerability is accessed and whether or not extra conditions are required to exploit it and the metric values are totally reference CVSS base metrics. The equations are listed as follows:

\[
\text{Quantitative Score} = \text{Impact Score} + \text{Exploitability Score};
\]

\[
\text{Exploitability Score} = 2 \times \text{Access Vector} \times \text{Access Complexity} \times \text{Authentication}
\]

The result of VRSS, which is similar with the methods of X-Force and Vupen, is more consistent with the normal distribution than CVSS v2[7].

### 4. Software Security Assessment System—SSAS
In the previous chapter, we have already presented and analyzed the attributes, types and evaluation methods of vulnerabilities. In order to give a quantitative characterization to software security, we make some improvements to CVSS and VRSS, so as to propose SSAS in this section, a system that based on analysis of vulnerabilities. We will introduce SSAS systematically in the following parts.

4.1. Improvement of CVSS and VRSS

4.1.1. CVSS

As an international standard, CVSS provides an open framework and a tool to quantify the severity and risk of vulnerabilities. In addition, CVSS provides a common assessment method, which could help security field form a unified standard. Thus, relatively speaking, CVSS has wider development prospect. But on the other hand, no matter how scientific and perfect standard-setting, in order to work in the real application, the standard must be combined with specific application environments. CVSS use confidentiality, integrity, and availability properties to evaluate the base impact and the possible values for each of the three security properties are none, partial and complete. The classification of the possible values is not enough for our system. So according to the classification and quantitative criteria in reference [14], we give our specific quantitative criteria in Table.3, Table.4 and Table.5.

<table>
<thead>
<tr>
<th>Level-Score</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₁ – 10</td>
<td>Implant root Trojan horse, can append, modify or create any file content</td>
</tr>
<tr>
<td>I₂ – 8</td>
<td>Can use ESP to append, modify, delete or create some of the contents of the file or the root processes memory space’s data structure.</td>
</tr>
<tr>
<td>I₃ – 5</td>
<td>Can use garbage or ESP to append, modify delete or create some specific file content or processes memory space’s data structure of several normal users</td>
</tr>
<tr>
<td>I₄ – 2</td>
<td>Can use garbage or ESP to append, modify, delete or create some files or processes space’s contents of normal users</td>
</tr>
<tr>
<td>I₅ – 0</td>
<td>No effect on system integrity</td>
</tr>
</tbody>
</table>

Table.4 Quantitative criteria of availability

<table>
<thead>
<tr>
<th>Level-Score</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>A₁ – 10</td>
<td>System of unrecoverable crash</td>
</tr>
<tr>
<td>A₂ – 8</td>
<td>Some files, components or processes of system are not available, the system is suspended or restarted</td>
</tr>
<tr>
<td>A₃ – 5</td>
<td>Some data files or processes memory space’s data structure of multiple users is covered, modified or deleted and result in a crash or not available</td>
</tr>
<tr>
<td>A₄ – 2</td>
<td>Some data files or processes memory space’s data structure of a user is covered, modified or deleted and result in a crash or not available</td>
</tr>
<tr>
<td>A₅ – 1</td>
<td>In the equivalence attack conditions, a service process or system load increases or decrease performance</td>
</tr>
<tr>
<td>A₆ – 0</td>
<td>No effect on system availability</td>
</tr>
</tbody>
</table>

Table.5 Quantitative criteria of confidentiality

<table>
<thead>
<tr>
<th>Level-Score</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>C₁ – 10</td>
<td>Read the contents of arbitrary files, monitor system or network activity</td>
</tr>
<tr>
<td>C₂ – 8</td>
<td>Read some contents of the system files or system processes and kernel space</td>
</tr>
<tr>
<td>C₃ – 5</td>
<td>Read the contents of some specific files or memory space’s data of multiple normal users</td>
</tr>
<tr>
<td>C₄ – 2</td>
<td>Read some specific normal files of non-sensitive memory space’s data of a normal users</td>
</tr>
<tr>
<td>C₅ – 1</td>
<td>Confirm OS type, version num and application</td>
</tr>
<tr>
<td>C₆ – 0</td>
<td>No effect on system confidentiality</td>
</tr>
</tbody>
</table>

In the next part we will use these data to calculate the base impact of the vulnerability.
4.1.2. VRSS

The quantitative score of VRSS is composed of impact score and exploitability score. The value of impact score is range from 0 to 9, while the other is range from 0 to 1. So the latter’s contribution to the overall score is very small and not enough to show its true impact on security. Take this into consideration, we treat exploitability as a probability and multiply by the base impact score.

4.2. The framework

Figure 2 shows the framework of the SSAS. The system has three important steps: data acquisition, data analysis and data calculation. In the following part, we will introduce these three steps in details.

![Figure 2. The framework of SSAS](image)

Nessus is a great tool designed to automate the testing and discovery of known security problems. Nessus is designed to help identify and solve these known problems, before a hacker takes advantage of them[15]. In General, Nessus is a powerful and practical software and we choose it as our vulnerability scanner in step 1.

After analysis and summary of the scan report, we get a vulnerability list, which including three levels of vulnerability. Through the specific judgment on the impact properties, we choose the corresponding score of each item and give the weight to them. In the whole, the second step focuses on analyzing the impact of the vulnerabilities by reading the descriptions of them.

The last step is calculating the data we obtained from the previous step. We need three indirect values to obtain the final results. The specific equations will be explained in the next part.

4.3. The equations

Firstly we calculate the base impact of a vulnerability on the software security. According to the description in Tab.4.1.1, we choose the corresponding value of each attribute and its weight. If the software has a higher requirement for one of the attributes, the weight for the higher one is 0.5 and the others’ weight is 0.25. If the software has an average requirement for the 3 attributes, the weight is 0.33 for each one.

\[
\text{Base Impact} = I_x \times W_i + A_y \times W_A + C_z \times W_C \tag{1}
\]

As is shown in Eq.(1), x, y and z stands for corresponding level of the attributes. By analyzing the specific effects on the three security properties of software from a vulnerability, we can get the value of x, y, z and their respective weights. And the base impact is an intermediate result ranging from 0 to 10.

Secondly the exploitability metrics capture how the vulnerability is accessed and whether or not extra conditions are required to exploit it. The corresponding values of AV, AC and AU could be got in Tab.3.1. Exploitability is a probability and its value is range from 0 to 1. The equation is as follows:

\[
\text{Exploitability} = AV \times AC \times AU \tag{2}
\]
Thirdly, as is shown in Eq.(3), we use BI and EX to calculate the final impact score ranging from 0 to 10, which reflects the severity of the vulnerability quantitatively.

\[
\text{Impact Score} = BI \times EX
\]  

Our ultimate goal is to assess the security of the software and our research is based on the vulnerabilities. So finally, we define software security metrics based on the representative vulnerabilities of the software as shown in the equation below:

\[
SRS = \sum_{j=1}^{3} \sum_{i=1}^{n_j} \left( \frac{IS_i}{n_j} \times W_j \right)
\]

SRS is the abbreviation for the security risk score and it stands for the final security state of the software. In the early stages we use NESSUS to scan the vulnerabilities and this tool divides vulnerabilities into 4 risk levels. We select the levels which have an impact on the security. In formula(4), \( j \) stands for the risk level and the value of 1, 2, 3 represents low, medium, high. \( n_j \) stands for the vulnerabilities’ number of corresponding levels. For example, \( n_1 \) stands for the number of vulnerabilities in low risk. \( W_j \) is the weight of corresponding levels and it stands for the impact bias of the vulnerabilities. We give them the values as follows:

\[
W_1 = 0.7 \quad W_2 = 0.8 \quad W_3 = 0.9
\]

By calculating the average scores of the vulnerabilities in each level, we get the sum of 3 average scores and we define it as the security risk score that can reflect the security of the software quantitatively.

5. Example Application

In order to put our SSAS into practical application, we implement a security assessment platform with java. And we choose the wisdom subway platform developed by our lab as our test object. Firstly, we use NESSUS to scan this software and collect the useful vulnerabilities for the latter calculation. Secondly, by analyzing the vulnerabilities’ attributes and impacts, we enter the corresponding values of each vulnerability into the platform. Figure.3 shows the input interface and Figure.4 shows the entries have been added. After these preparatory works, we can easily work out the final software security score with our SSAS. Figure.5 shows the result of the calculation. As the score increases, the system risks are increasing. The result showed in Figure.4 presents the software we developed is secure relatively. And according to the numbers and severity of the vulnerabilities, we can prioritize these vulnerabilities and remediate those that pose the greatest risk.
6. Conclusion and future work

In this paper, we have introduced the classifications and attributes of vulnerabilities and analyzed the existing vulnerability rating methods. By taking full advantage of the existing systems’ metrics and improving the details of some aspects, we propose our SSAS to assess the security quantitatively. Finally, we prove SSAS can be put into practice through the platform we have developed.

As possible future works, we plan to assess the software security from a multiple perspective. In addition, we have analyzed just an individual impact on software security from a vulnerability, without taking into account the multiple consequences of vulnerability at the same time, so we will study on vulnerability relevance in the future. Furthermore, we will continue to pay attention to SSAS and make it more accurate, flexible and usable.

7. Acknowledgement

The authors are grateful for a variety of valuable comments from the anonymous reviewers. This work is supported by National Natural Science Foundation of China under Grant 90818021.

8. References