A Software Assessment Method Based on Relevance Vulnerability

Xudong Miao, Yongchun Wang*
Operation Software and Simulation Research Institute
Dalian Naval Academy
Dalian, China
miao3399@sina.com, woaiwojia728@126.com

Xingchen Cao,Binbin Qu,Sheng Jiang,Feng Fang
School Of Computer Science & Technology
Huazhong University Of Science And Technology
Wuhan, China
1250338142@qq.com

***Abstract*—At present, most of the software security assessment system can only evaluate the potential impact of a single vulnerability on the system which ignore the impact of the multiple vulnerabilities. Therefore, we introduce the concept of relevance vulnerability pattern and design a relevance vulnerability pattern library taking consider of the potential impact caused by multiple vulnerabilities. After that, a software assessment method is given based on relevance vulnerability. Experimental results show that the evaluation results are comprehensive and objective.**

Keywords-relevance vulnerability; software security assessment; quantitative assessment

I. INTRODUCTION

Software vulnerability includes three elements [1,2]: system weakness, exposed attack points, attack vector.

According to the Internet Security Threat Report issued by CNCERT / CC (National Internet Emergency Response Center), the number of terminals infected by virus was as many as 2.76 million and the number of websites attacked by hacker was 10256 in December 2014. The number of security vulnerabilities collected by National Information Security Vulnerabilities Sharing Platform (CNVD) was 619. Among that, there were 206 high-risk vulnerabilities and 575 vulnerabilities that can be exploited to implement remote attack. It means that dozens of vulnerabilities may be found per day.

To reduce the losses caused by vulnerabilities, system administrators must consider different hardware and software platforms in the process of assessment. The vulnerabilities can be divided into different levels. Administrators can fix bugs that belong to high risk level. However, vulnerabilities have different characteristics and different risk levels. It's particularly important that the way we extract information that can be used to assess system.

This paper introduces the conception of the relevance vulnerability model and relevance vulnerability, and designs a pattern library to store relevance vulnerability pattern, and presents a new kind of security vulnerability assessment method based on relevance vulnerability.

II. RELATED WORKS

IBM ISS X-Force [3] is a database that collects thousands of software defects and software vulnerabilities. It already contained 4000 unique software vulnerabilities, software defects and security policies, and most of them were found from Internet or the research results from the first X-Force system and IBM ISS system. X-Force adopts a qualitative classification method and calculates a security risk level for each bug to describe the potential hazard. Due to the interference of subjective factors, the qualitative assessment cannot be accurately calculated.

CVSS [4,5] (Common Vulnerability Scoring System) is an open framework for assessment system. Many members of institutes apply CVSS to its own software system, but CVSS cannot properly assess the potential harm caused by multiple vulnerabilities.

Path Graph [6,7,8] was first proposed by Cunningham in 1985. Attack Path Graph presents a series of penetration scenes that possibly exist in the computer network. Each scene is action sequences that are implemented by attackers. Attack Path Graph could reflect the network status and dependencies between the attack actions. It could assess the potentially harmful impact caused by multiple vulnerabilities objectively as well. Currently, Attack Path Graph is widely used in network assessment, but few scholars study the role in the system assessment.

In summary, the current assessment system only considers the risk of assessing a single vulnerability. Therefore, it has important research value to study the method of software assessment for the related vulnerabilities.

III. RELEVANCE VULNERABILITY

Factually, attackers may use multi-step attack method to enhance their competence or implement high-level attack through a few low-level software bugs, so as to achieve the purpose of damaging or controlling the system. In order to assess the impact of multi-step attack, this paper presents the concept of related vulnerabilities, and applies them to the assessment system.

Definition 1: If attackers attack a target with a bug set named ∂, using multi-step attack method, the set ∂ can be regarded as relevance vulnerability.

Attacker exploits a few vulnerabilities rather than just one bug. Thus, relevance vulnerability defined usually includes more than one bug.

Definition 2: Relevance vulnerability pattern is a tuple $RM = (V, R, v_0)$, where V is a finite set of states, R is a state conversion function on the set V, and v_0 is the initial state.

Relevance vulnerability pattern is an abstract concept that describes the characteristics of bugs which must satisfy as a member of relevance vulnerability. Relevance

978-1-4673-8660-9/15 $31.00 © 2015 IEEE
DOI 10.1109/CIS.2015.108

vulnerability pattern is also a concept related with system, it describes how many bugs could be exploited in a multi-step attack. The same pattern corresponds to several sets of relevance vulnerabilities in the system.

Definition 3: The element R of the state transition function is a tuple T = (srcV, destV, cond), where srcV represents the initial state, destV represents the target state, and cond represents conditions that must satisfy during state srcV to state destV.

Relevance vulnerability pattern includes two parts: the sets of vulnerability and conditions. The former describes the vulnerabilities exploited in a multi-step attack; the latter describes when the vulnerabilities can constitute relevance vulnerability. The conditions can be divided into two cases: Exist and Reach.

1. Exist: the vulnerabilities can constitute relevance vulnerability as long as they exist in the same system.

For example, a web application includes two bugs: CSRF [9] and reset the security verification problems vulnerability. The attacker links his own email to the victim's account through CSRF vulnerability, and then clicks the button "Reset Password" which could send a letter to the attacker's email. Generally, reset password requires user to complete security verification problems. But, if an attacker exploits the reset security verification problems vulnerability, he can bypass the security verification and reset password directly. In this example, the set contains CRSF and reset the security verification problems vulnerability, and the conditions can be described as Exist.

2. Reach: A set of vulnerabilities must be executed on the same path. The path here refers to the path mentioned in the control flow graph (CFG). It means the bugs constituted the relevance vulnerability exist in same execution path. In TABLE I, for example, the statement 4 can reach to statement 5, and statement 5 can reach to statement 6, but statement 7 cannot reach statement 10 because there is not an executable path from 7 to 10.

TABLE I. A SIMPLE EXAMPLE PROGRAM FOR JAVA

```
1.public class Test {
2.
3.      public void CalFun(double n) {
4.            int          a = 1;
5.            double     b = a * n;
6.            if (b < 0) {
7.                 System.out.println("true:" + a);
8.            }
9.            else {
10.                System.out.println("false:" + b);
11.           }
12.     }
13.  }
```

IV. SYSTEM ASSESSMENT

Software security assessment method proposed in this paper is a kind of quantitative analysis method. It adopts metrics of Common Vulnerability Scoring System (CVSS) as assessment metric groups. In order to facilitate the description, a brief description should be given first.

A. CVSS

CVSS consists of three metric groups: Basic Metrics, Temporal Metrics and Environmental Metrics. Basic metrics concern the indexes which don't change with time or environment, such as the features of bug and the impacts caused by these features. It consists of six indexes: Access Vector, Access Complexity, Authentication, Confidentiality Impact, Integrity Impact and Availability Impact. The result calculated by Temporal Metrics is a value between Basic Metrics and Environmental Metrics. Temporal Metrics consist of two indexes: Exploitability and Report Confidence. Even the same vulnerability represents different harm level in the different environment. Thus, the assessment system must consider the impact caused by environment. Environmental Metrics consist of five indexes: Collateral Damage Potential, Target Distribution, Confidentiality Requirement, Integrity Requirement and Availability Requirement. Fig.1 represents the flow of CVSS system.

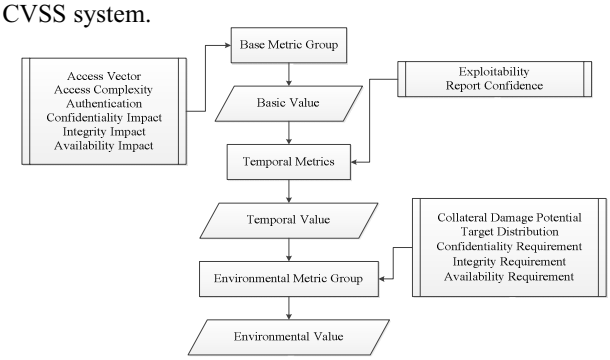

Figure 1. CVSS process flow

In CVSS, the calculation steps as follows:

First, calculate basic assessment factor according to the Basic Metrics and Environmental Metrics as shown in formula (1):

$$impact = \min\{10, 10.41*(1-(1-C*CR)*(1-I*IR)*(1-A*AR))\} \quad (1)$$

Then, calculate the basic assessment exploitable factor, as shown in formula (2):

$$Expl = 20*AV*AC*Au \quad (2)$$

Calculate the Basic Value through formula (3):

$$BV = ((0.6*impact + 0.4*Expl) - 1.5)*f \quad (3)$$

If impact= 0, then f=0, otherwise, f=1.176.

Calculate the Temporal Value combine formula (3) and Temporal Metrics, as shown in formula (4):

$$TV = BV*E*RC \quad (4)$$

At last, calculate the Environmental Value combine formula (4) and Environmental Metrics, as shown in formula (5):

$$EV = (TV + (10 - TV)*CDP)*TD \quad (5)$$

B. System Assessment Method

According to the relevance vulnerability pattern, the vulnerability list can be divided into two parts. Vulnerabilities corresponding to a kind of pattern should be added into relevant sets and vulnerabilities which don't

match any patterns still exist in the original list of bug reports. For the latter, we still use CVSS assessment. For the former, we propose a new evaluation method.

As mentioned above, the potential impact caused by multiple vulnerabilities exploited jointly should be larger than that caused by just one bug. In this way, the evaluation results are comprehensively and objectively. In this paper, a quantitative assessment method is studied. How to quantitatively assess the system becomes a key point. We propose a factor (Relevance Degree) which is used to measure the significant degree of correlation between bugs, on the basis of combination of CVSS and relevance vulnerability. It is defined as follows:

Definition 4: The value of Relevance Degree can be calculated by formula (6) and formula (7):

$$VT = \frac{1}{Count} * \sum_{i=1}^{Count} \frac{EV_i}{10} \quad (6)$$

$$RD = VT / Count \quad (7)$$

Count in formula (6) means the number of vulnerabilities in relevance vulnerability , EV_i represents the environment value of the bug whose sequence number is i. In formula (7), The RD gets higher with fewer bugs and higher VT. The indicator can effectively measure the possibility that the relevance vulnerability can be used by attackers.

The ultimate goal is not calculating a certain bug but considering the impact on the system caused by all the bugs, the score which can describe the potential security threats of the system. We think the impact caused by relevance vulnerability should be much serious than that cause by a few uncorrelated bugs, so we must take the RD of relevance vulnerability into consideration during the process of calculating the system assessment score (SV). The specific formulas of SV are as follows:

$$RV = \sum_{k=1}^{l} EV_k \quad (8)$$

$$SV = \sum_{i=1}^{m} EV_i + \sum_{j=1}^{n} RV_j * (1 + RD_j) \quad (9)$$

Assuming that the bug report list is divided into two sets: A and B. Set A collects m non-relevance vulnerabilities while set B collects n relevance vulnerabilities. In formula (8), l represents the number of vulnerabilities in relevance vulnerability and EV_k means the environment value whose bug ID is k. In formula (9), EV_i means the environment value whose bug ID is i, RV_j represents related value of relevance vulnerability whose relevance ID is j and RD_j represents the relevance degree whose relevance ID is j.

The final result SV is a comprehensive score of the system and it describes the potential threat. We introduce the method not only consider the impact caused by a certain bug but also consider harm caused by multi-step attack.

Assuming that there are six bugs A to F, there exists two relevance vulnerabilities among them. F is non-relevance vulnerability. Specific information is shown in TABLE II.

TABLE II. ENVIRONMENTAL VALUE OF VULNERABILITIES

	Vulnerability Name	EV
Relevance Vulnerability 1	A	3.5
	B	5.0
Relevance Vulnerability 2	C	6.4
	D	7.2
	E	2.7
Non-Relevance Vulnerability	F	3.5

Take the EV of bug A and bug B into formula (6):

$$VT = \frac{1}{2} * \frac{3.5+5}{10} = 0.425$$

Take the VT above into formula (7):

$$RD = \frac{0.425}{2} = 0.2125$$

RD is the related factor of relevance vulnerability 1. Then take it into formula (8):

$$RV = 3.5 + 5.0 = 8.5$$

In the same way, the relevance vulnerability's RD is 0.1811 and its RV is 16.3.

Take the results into formula (9):

$$SV = 3.5 + 8.5 * (1 + 0.2125) + 16.3 * (1 + 0.1811) = 33.1$$

The assessment score of the system is 33.1.

V. SYSTEM FRAMEWORK

In order to verify the effectiveness of the assessment method brought in this paper, we design a system named SSAS (Software Security Assessment System). It is based on source code static analysis and quantitative assessment framework CVSS. Systematic evaluation requires two steps. First, it analyzes the source code and finds bugs. Then, it uses relevance mode library to partition the bug report list and assesses the system according to the classification of bugs. System framework is shown in Fig2.

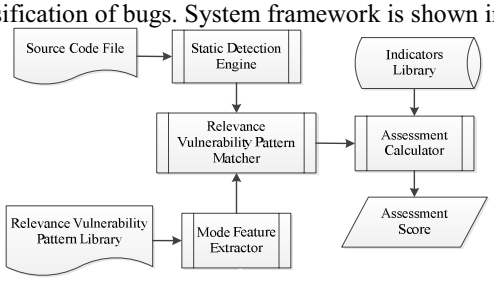

Figure 2. System Framework

Static Detection Engine: Use Java source code static analyzer generated by ANTLR to collect code information, and store the information in Intermediate Representation (IR) which is a data structure similar to Jimple. Then, use flow analysis module to generate control dependence graph (CDG), data dependency graph (DDG) and call dependency graph (CG). At last, find security vulnerabilities according to users' requirements.

Relevance Vulnerability Pattern Library: It offers XML file recorded relevance vulnerability pattern and user interface that can manage the XML file. The user can add or remove models to pattern library according to his actual needs.

Mode feature extractor: Read Xml files and extract the relevance features from the mode library. Then, store the features into IR and provide interfaces for subsequent analysis.

Relevance vulnerability Pattern Matcher: Analyze bug reports according to the extracted pattern features, identify relevance vulnerabilities in line with a certain mode combine with the call dependency graph.

Indicators Library: Record CVSS metric value. Store basic metrics, temporal metrics and environment metrics into the database and provide interfaces for subsequent analysis.

Assessment Calculator: Use CVSS evaluation algorithm to assess the non-relevance vulnerabilities directly. For each group of relevance vulnerabilities, calculate the significant degree factor (RD) and then calculate relevance vulnerability's score. At last, sum the values calculated above. Then we can obtain a system evaluation value.

VI. Experiment Analysis

A. Experiment Environment

The experiments are carried out in the following environments: Intel Core (TM) i3-2100 CPU, 2.92GB Memory, Windows 7 Ultimate (32bit), Eclipse 4.2 SR2 (Juno), JDK 1.6_45, ANTLR Works 1.4.2, ANTLR lib 4.3.

We use an open source project named Marketplace. It is micro-electric business open platform based on J2EE. The system contains three bugs, 31 source files, 31 classes, 271 methods and 3325 lines of source code.

B. Experiment Result Analysis

In order to verify the objectivity and comprehensiveness of the assessment method, this experiment compares with SSAS and CVSS system. Statistics are shown in TABLE III.

TABLE III. Test Statistics

	Vulnerability Name	Assessment Value	RD	System Value
SSAS	Null Pointer	4.4	0	15.3
	Hardcoded	2.1	0.2234	
	SQL Injection	6.8		
CVSS	Null Pointer	4.4	None	13.3
	Hardcoded	2.1		
	SQL Injection	6.8		

In the test set, "Hardcoded" and "SQL Injection" constitute a group of relevance vulnerability. And its related factor RD is 0.2234. "Null Pointer" is non-relevance vulnerability and should be assessed alone. The result is shown as follows.

SV=4.4+ (2.1+6.8)*(1+0.2234) =15.3

The CVSS system can only assess the harm caused by a single vulnerability on the system, rather than caused by relevance vulnerability.

SV=4.4+2.1+6.8=13.3

From the above calculations, we can see CVSS sums the three bugs' value simply while SSAS calculates the score according to the related factor RD. So, the latter makes the final result more comprehensive.

VII. Conclusion

For most of the security assessment system assess the impact of a single vulnerability. This paper presents the concept of relevance vulnerability in order to assess the related vulnerabilities. Therefore we give a method to extract the association features from the related vulnerability pattern database and find the related vulnerabilities from the vulnerability report list. The proposed solution can accurately evaluate the potential impact of multiple vulnerabilities on the system, which makes the evaluation system more objective and comprehensive.

In the future, we can focus on the following aspects: 1. the characteristics of software vulnerabilities are not invariable and the maintenance of CVSS has a feedback mechanism. We should adjust the evaluation algorithm of the system in accordance with documents released by FIRST to make the assess results more accurate. 2. Relevance vulnerability is a concept of vulnerability assessment based on multi-step attack. The attack techniques are constantly changing. Therefore, we can analyze emerging attack techniques, abstracts relevance conditions and add into the relevance pattern to make the pattern library more comprehensive.

Acknowledgment

The research work presented in this paper is supported by the Natural Science Foundation of Hubei Province of China under Grant No.2014CFB1006 and the Independent Innovation Fund of Huazhong University of Science and Technology under Grant No.2015QN062.

References

[1] Oliveira D, Rosenthal M, Morin N, et al, "It's the psychology stupid: how heuristics explain software vulnerabilities and how priming can illuminate developer's blind spots," Proceedings of the 30th Annual Computer Security Applications Conference. ACM,2014, pp.296-305.

[2] The Three Tenents of Cyber Security. http://www.spi.dod.mil/tenets.htm.

[3] Liu Q, Zhang Y. VRSS, "A new system for rating and scoring vulnerabilities," Computer Communications, 2011, 34(3), pp.264-273.

[4] Allodi L, Massacci F, "Comparing Vulnerability Severity and Exploits Using Case-Control Studies," ACM Transactions on Information and System Security (TISSEC), 2014, 17(1), p 1.

[5] Spanos G, Sioziou A, Angelis L. WIVSS, "a new methodology for scoring information systems vulnerabilities," Proceedings of the 17th Panhellenic Conference on Informatics, ACM, 2013, pp. 83-90.

[6] Gallon L, Bascou J J, "Cvss attack graphs," Signal-Image Technology and Internet-Based Systems (SITIS), 2011 Seventh International Conference on. IEEE, 2011, pp. 24-31.

[7] Fang X, Zhai L, Jia Z, et al, "A Game Model for Predicting the Attack Path of APT," Dependable, Autonomic and Secure Computing (DASC), 2014 IEEE 12th International Conference on. IEEE, 2014, pp. 491-495.

[8] Gallon L, Bascou J J, "Using CVSS in attack graphs," Availability, Reliability and Security (ARES), 2011 Sixth International Conference on. IEEE, 2011, pp. 59-66.

[9] You J, Guo F, "Improved CSRFGuard for CSRF attacks defense on Java EE platform," Computer Science & Education (ICCSE), 2014 9th International Conference on. IEEE, 2014, pp. 1115-1120.