ICVSS: A New Method for Vulnerability Quantitative Grading

Tiantian Tan[1, *], Baosheng Wang[1], Yong Tang[1], Xu Zhou[1] and Jingwen Han[2]

Abstract: Vulnerability technology is the basic of network security technology, vulnerability quantitative grading methods, such as CVSS, WIVSS, ICVSS, provide a reference to vulnerability management, but the problems of ignoring the risk elevation caused by a group of vulnerabilities and low accuracy of exploitable level evaluation exist in current vulnerability quantitative grading methods. To solve problems above in current network security quantitative evaluation methods, this paper verified the high relevance degree between type and exploitable score of vulnerability, proposed a new vulnerability quantitative grading method ICVSS, ICVSS can explore attack path using continuity level defined by privilege, add vulnerability type to measure indexes of exploitable metrics and use Analytic Hierarchy Process (AHP) to quantify the influence of vulnerability type on exploitable level. Compared with CVSS and WIVSS, ICVSS is proved that it can discover attack path consist of a sequence of vulnerabilities for network security situation evaluation, and has more accuracy and stability.

Keywords: Vulnerability dependency, exploitable level, measure index, quantitative grading.

1 Introduction

As increasingly requirements of privacy protection, privacy security is one of most important technology in network security. Some absolute network security mechanisms can be applied in edge network for data security, for some instances, for multi-monitor joint detection, Cai et al. [Cai, Chen, Chen et al. (2015)] proposed a solution with lower communication overhead; for DDoS attacks detection, Liu Yun et al. [Liu, Cai and Zhong (2011)] proposed a more robust approach based on Conditional Random Fields model; for Longest Prefix Matching (LPM), Policy Filtering (PF), and Content Filtering (CF), Cai et al. [Cai, Wang and Zheng (2015)] proposed a distributed Ternary Content Addressable Memory (TCAM) coprocessor architecture; for transaction rich applications, Gopinath et al. [Gopinath and Bhuvaneswaran (2018)] designed an ECC based secured cloud storage mechanism. The vulnerabilities are the key of network security [Tan, Wang, Zhou et al. (2018); Tan, Wang, Zhou et al. (2018)], and exist in the process of system design, implementation and operation, and unable to be eliminated completely. Vulnerability quantitative grading technology provides a method to obtain main features of vulnerabilities, quantify and rank them; it gives a reference for priority determination

[1] National University of Defense Technology, Deya Village, Changsha, China.

[2] University of British Columbia, Vancouver, Canada.

[*] Corresponding Author: Tiantian Tan.Email: happinesschild@126.com.

CMC. doi:10.32604/cmc.2019.06049

of vulnerability management. However, there are some problems in current vulnerability quantitative grading methods:

- Current vulnerability evaluation methods, such as CVSS, ignore the relevance among vulnerabilities, in reality, the attacker can successfully elevate privileges by a sequence of vulnerabilities, in this sequence, some vulnerabilities with low risk score are always ignored in vulnerability management. In addition, current vulnerability evaluation methods are short of detailed measure index, a lack of detailed measure index makes it low accuracy.
- Current network situation evaluation methods fully take the relevance among vulnerabilities into consideration, but compute the successful exploit rate by methods mentioned above, ignore the problems exist in current method, such as a lack of enough detailed measure index and low accuracy etc.

For problems mentioned above, this paper proposed ICVSS on the basis of CVSS, its major contributions include:

- Introduced continuity metric to measure dependency between vulnerabilities for attack path generation.
- Proved strong influence of vulnerability type on exploitability, added it to exploitable measure index, and quantified it through AHP algorithm.
- Proposed ICVSS, and proved it effectively detected attack path, and had a higher accuracy in exploitability quantification than CVSS and WIVSS.

2 Relevant research

Due to more comprehension and higher accuracy, CVSS is the most widely used vulnerability quantitative grading method, current research on quantitative grading mainly focus on the application or improvement of CVSS:

- Wen et al. [Wen and Zhang (2015)] proposed the risk assessment system CVSS PCA for CVSS measure index score determination and risk score distribution. Based on Expert System, Wen et al. [Wen and Zhang (2015)] proposed a CVSS correction. The method proposed by Liu et al. [Liu and Zhang (2012)] quantified the vulnerability risk index through the hierarchical analysis methods, ignored the vulnerability exploitable measure. Wang et al. [Wang and Gao (2011)] increased the vulnerability risk measure index, but lacked objective basis. Zhang et al. [Zhang, Fang (2015)] summarized android vulnerability detection method and discussed the theme of further research on android vulnerabilities risk. Liu et al. [Liu and Zhang (2011)] proposed a VRSS vulnerability assessment method, assessed vulnerabilities risk by combining qualitative and quantitative methods. WIVSS improved CVSS by re-determining CVSS measure index and further refining the index with the same score [Spanos and Sioziou (2011)], the result had more diversity, but lacked objectivity because of a lack of large samples during index optimization.
- Quantitative grading method using attack graph can access the network security situation. Multi-stage network attacks were formally described in Zhou et al. [Zhou, Watts and Aebersold (2001)], each vulnerability exploitable score was quantified as attack successful rate by the attack trees. Lai et al. [Lai and Hsia (2007)] proposed

three vulnerability analysis methods, used attack path to analyze network security score. The above study focus on attack graph generation ignored the study of exploitable quantitative evaluation. In 2014, Zhang et al. [Zhang and Feng (2014)] improved CVSS by increasing exploitable measure index, but did not give the reason and impact degree of new index. Frigault [Frigault (2008)] quantified the network security situation by Bayesian network attack graph, calculated successful rate and risk score by CVSS, and improved the accuracy rate. Lu et al. [Lu, Xia (2005)] proposed a quantitative fusion model for host security, analyzed vulnerability by the corresponding mathematical model, further evaluated host security performance; it gave a set of vulnerability exploitable measure indexes, but did not prove it.

3 Vulnerability continuity measurement

In reality, system risk assessment needs not only vulnerability risk level, but also the dependencies between vulnerabilities.

3.1 Privilege category

We hypothesize that the purpose of an attack is to elevate privilege [Montaigne, Coisne and Sosner (2015)]. Through the required privileges and risk levels of vulnerabilities, we have studied a large number of vulnerabilities and their effects, and summarized five privilege subsets, such as ACCESS, VISITOR, USER, ADMIN and SUPERADMIN. Take host T as an example, host T's privilege set *TP={TP1, TP2, TP3, TP4, TP5}*, the specific value includes:

- *TP1*=T_ACCESS. Visitors outside the firewall.
- *TP2*=T_VISITOR. Trusted system visitors.
- *TP3*=T_USER. Ordinary users with private storage space.
- *TP4*=T_ADMIN. Partial system privileges.
- *TP5*=T_SUPERADMIN. Root privileges of system.

3.2 Continuity of vulnerability

We introduce the vulnerability continuity metric to measure the dependency between vulnerabilities, and take the pre-exploit privilege and post-exploit privilege as the measurement indexes. Vulnerability continuity can be used to predict the attack path, and the much continuity vulnerability has, the higher risk the vulnerability has.

Definition 1 Pre-exploit privilege. If successful exploit of vulnerability *I* require system privilege PI_{Pre}, we define PI_{Pre} as the Pre-exploit privilege of vulnerability *I*.

Definition 2 Post-exploit privilege. If system privilege PI_{Pro} is obtained by a successful exploit of vulnerability *I*, we define PI_{Pro} as the Pro-exploit privilege of vulnerability *I*.

Definition 3 Continuity of vulnerability. During privilege elevation, if the pro-permission of vulnerability A is equal to the pre-permission of vulnerability B, namely, $PA_{Pro}=PB_{Pre}$, we think there exists a continuity between vulnerability A and vulnerability B, such as cve-2016-2207 and cve-2014-3390, the $P_{cve\text{-}2016\text{-}2207Pre}$="VISITOR",

$P_{cve\text{-}2016\text{-}2207Pro}$=$P_{cve\text{-}2014\text{-}3390Pre}$="USER", $P_{cve\text{-}2014\text{-}3390Pro}$="SUPERADMIN". Because cve-2014-3390 is unexploitable to system or network visitors, it is always considered as low risk. However, if an attacker successfully exploits cve-2016-2207 and obtains "USER" privilege, then cve-2014-3390 can be exploited to obtain "SUPERADMIN" privilege, we consider cve-2014-3390 have a continuity with cve-2016-2207.

Definition 4 Vulnerability sequence. For a group of vulnerabilities, if every vulnerability in the group only has one continuity with another unique vulnerability in the group, the group of vulnerabilities can be considered as an vulnerability sequence, such as cve-2008-0076, cve-2006-0026, cve-2006-6424, $P_{cve\text{-}2008\text{-}0076Pre}$="A_SUPERADMIN" in host A; $P_{cve\text{-}2008\text{-}0076Pro}$="B_SUPERADMIN"; $P_{cve\text{-}2006\text{-}0026Pre}$="B_SUPERADMIN" and $P_{cve\text{-}2006\text{-}0026Pro}$="C_SUPERADMIN"; $P_{cve\text{-}2006\text{-}6424Pre}$="C_SUPERADMIN" and $P_{cve\text{-}2006\text{-}6424Pro}$="D_SUPERADMIN". In this group, cve-2006-6424 only has a continuity with cve-2006-0026, cve-2006-0026 only has a continuity with cve-2008-0076, we consider this group as a vulnerability sequence.

3.3 Attack path generation using vulnerability continuity

Continuity of vulnerability can be used to generate attack path. In the process of privilege elevation, in a vulnerability sequence of a network system, rank the vulnerabilities of vulnerability sequence from attacker node to target node; in a vulnerability sequence of one host, rank vulnerabilities of vulnerability sequence from "ACCESS" to "SUPERADMIN", the obtained vulnerability sequence can be considered as an attack path, such as the group vulnerabilities mentioned in Definition 4, a vulnerability path is cve-2008-0076→cve-2006-0026→cve-2006-6424. Attackers in host A with "A_SUPERADMIN" may not directly exploit cve-2006-6424 in host C. However, if an attacker can successfully exploit the vulnerabilities in vulnerability path by order, that attacker will get "SUPERADMIN" in host C, we consider this vulnerability sequence as an attack path.

Vulnerability continuity can provide the efficient attack path for privilege elevation, and increase the accuracy of vulnerability priority determination.

4 Vulnerability exploitable measurement of ICVSS

4.1 Vulnerability exploitable measurement of CVSS

- Exploitable measure indexes of CVSS basic metric include attack vectors, attack complexity and identity authentication.
- Attack Vectors (*AV*).
  - Network(N). Successful exploit needs network layer, score is 1.0.
  - Adjacent(A). Successful exploit needs physical or logical networks rather than network layer, score is 0.646.
  - Local(L). Successful exploit needs read/write/execute functions in local system, score is 0.395.
- Attack Complexity (*AC*).
  - Low(L). None access conditions are required, score is 0.35.
  - Medium (M). Certain access conditions are required, score is 0.61.

  - High(H). Conditions beyond attackers control are required, score is 0.71.
- Authentication (*AU*).
  - None(N). Identity authentication is not required, score is 0.45.
  - One(O). One identity authentication is required, score is 0.56.
  - Multiple(M). Multiple authentications are required, score is 0.704.

The exploitable score can be calculated by the following equation:

$$ExploitableScore_{cvss}=20\times AV\times AC\times AU \quad (1)$$

4.2 Vulnerability type impact on quantitative grading

We used CWE classification standards on 41,815 vulnerabilities which have classification standards, and obtained 69 types of vulnerabilities. Because 47 CWE types have less than 20 samples which only accounted for 0.3% number of total samples, we ignored them for decreasing error and increasing efficiency. We calculated exploitable scores of 22 CWE types (41,727 vulnerabilities) and graded them as 2 levels: B1(harder to exploit)∈[1.2, 6.9], B2(easier to exploit)∈[7, 10].

As shown in Tab. 1, 18 CWE types is easier to exploit, account for 71.5%-97.9%, 3 CWE types is harder to exploit, account for 62.2%-89.30%, only one type of C16 has 58.4% samples distributed in easier to exploit and 41.6% samples distributed in harder to exploit, and C16 only account for 7% of total samples, this result can prove that vulnerability type has strong impact on exploitable level and is suitable as an exploitable measure index.

4.3 Vulnerability exploitable measurement of ICVSS

To introduce vulnerability type (*VT*) as a measurement index to Eq. (1), we used method in Miaoui et al. [Miaoui and Boudriga (2017)] to calculate the weight of *VT*=0.1, therefore, the weight of *AC, AV, AU* is 0.9, the ICVSS exploitable measure equation is:

$$ExploitableScore_{ICVSS}=18\times AV\times AC\times AU+VT \quad (2)$$

The specific steps of ICVSS include:

- According to CVE, obtain the vulnerability list of every system, use continuity metric to generate attack path set: Path=$\{path_0, path_1, ...path_n\}$, $n\in Z$.
- Use the exploitable metric to quantify and compute $ExploitableScore_I$ for every vulnerability in Path.
- According to $ExploitableScore_I$ of every vulnerability in Path, determine the selection probability S_i of each vulnerability [Zhang and Feng (2015)], compute the successful exploit probability P_i of every vulnerability.

  $$P_i=E_i\times S_i \quad (3)$$

- Compute the successful probability P_s of every element in Path.

  $$P_S=\prod_{i=1}^{n} P_i \quad (4)$$

4.4 Vulnerability type quantitative method

Analytic hierarchy process (AHP) is a multi-level analysis method for weight decision-making. It can extract decision-making problem into three levels: target level, rule level and policy level, and determine each weight of policy element to one target element by statistic weights of all lower layer elements to every upper level element, it need to establish a judgment matrix to find weight vector of lower layer to upper layer. We selected eigenvector method to determine weight vector by matrix consistency judgment for stable accuracy. AHP has 3 steps:

- Establish a 3-level hierarchical model.
- Construct pairwise comparison determination matrix, and compare indexes by pairwise. For example, take 2 factors x_i and x_j at each time, a_{ij} is used to represent the ratio of the influence of x_i and x_j on the corresponding element of upper layer. Saaty et al. gave a method for a_{ij} determination, use the number as the scale and 90 proportion rule to rank each specific type, according order of types, construct the judgment matrix A.
- Consistency check. Judging the order of matrix needs to construct a judgment matrix which is close to the consistency condition. Therefore, it is necessary to identify whether the judgment matrix deviates from the acceptable range, the specific steps are followed:
  - Use the following equation to calculate the consistency index *CI*:

    $$CI = \frac{\lambda_{max} - n}{n-1} \tag{5}$$

  - Find the corresponding average random consistency index *RI* in Aonso et al. [Aonso and Lamata (2006)].
  - Calculate the consistency ratio (*CR*) of judgment matrix:

    $$CR = \frac{CI}{RI} \tag{6}$$

  - If *CR*<0.10, consistency is acceptable; otherwise, judgment matrix should be corrected.

4.5 Using AHP to quantify VT

Based on vulnerability type, a 3-level AHP model should be constructed as Fig. 1. The target layer was vulnerability exploitable level, exploitable scores of 22 CWE types had been obtained by above analysis. To simplify calculation and improve efficiency, the 22×22 matrix was simplified to 22×4 matrix, the rule layer was 4 exploitable levels, B1∈[1, 4]: hard to exploit; B2∈(4, 6.9]: could exploit; B3∈(6.9, 9.9]: essay to exploit; B4∈(9.9, 10]: extremely easy to exploit, and the policy layer was 22 CWE types from id1 to id22.

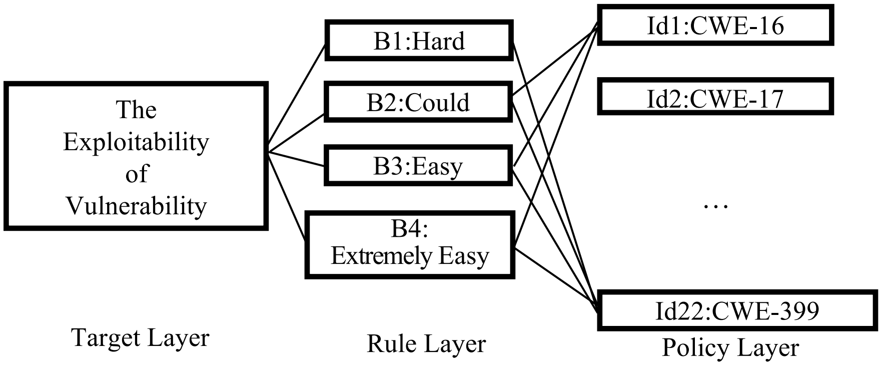


Figure 1: Hierarchy model based on vulnerability type

Construct the judgment matrix of rule layer to target layer:

$$\begin{array}{ccccc} & B1 & B2 & B3 & B4 \\ B1 & 1 & \frac{1}{3} & \frac{1}{5} & \frac{1}{9} \\ B2 & 3 & 1 & \frac{1}{3} & \frac{1}{6} \\ B3 & 5 & 3 & 1 & \frac{1}{4} \\ B4 & 9 & 6 & 4 & 1 \end{array}$$

The maximum feature vector of the matrix is calculated as:

$(0.0447, 0.1031, 0.1153, 0.6240)^{T}$

CR=0.046<0.1, judgment matrix consistency was acceptable.

According to vulnerability type, constructed judgment matrix of B1, the maximum feature vector is calculated by Matlab.

$(0.0559\ 0.0559\ 0.0203\ 0.0340\ 0.0218\ 0.2209\ 0.0340\ 0.0128\ 0.0128\ 0.0128\ 0.0203\ 0.0340\ 0.0340\ 0.0559\ 0.0340\ 0.0831\ 0.0340\ 0.0203\ 0.0203\ 0.0218\ 0.1451\ 0.0340)^{T}$.

Table 1: The exploitable level and successful exploit probability of 22 CWE types

ID	CWE-ID	Vulnerability Types(*VT*)	B1	$\frac{B1}{\sum_1^{22} C_i}\%$	B2	$\frac{B2}{\sum_1^{22} C_i}\%$
C1	CWE-16	Configuration	62	28	160	72.1
C2	CWE-17	Code Errors	45	28.5	113	71.5
C3	CWE-19	Data Processing	16	12.7	110	87.3
C4	CWE-20	Input Validation	730	19.3	3061	80.8
C5	CWE-22	Path Traversal	142	7.9	1651	92.1
C6	CWE-59	Link Following	359	89.3	43	10.7
C7	CWE-78	OS Command Injections	46	23.6	149	76.4
C8	CWE-79	Cross-site Scripting	1013	16.1	5302	84

C9	CWE-89	SQL Injection	81	2.1	3894	97.9
C10	CWE-94	Code Injection	125	6.9	1679	93.1
C11	CWE-119	Buffer Errors	823	12	6016	88
C12	CWE-134	Uncontrolled Format String	24	17.2	115	82.7
C13	CWE-189	Numeric Errors	196	15.4	1081	84.6
C14	CWE-200	Information Leak	670	24	2115	75.9
C15	CWE-254	Security Feature	51	21.7	184	77.6
C16	CWE-264	Permissions Control	1879	41.6	2637	58.4
C17	CWE-284	Access Control	102	25.2	304	74.9
C18	CWE-287	Authentication Issues	125	5.6	831	87
C19	CWE-310	Cryptographic Issues	1059	62.2	644	37.8
C20	CWE-352	Cross-site Request Forery	97	8.5	1047	91.5
C21	CWE-362	Race Conditions	260	70.1	111	29.9
C22	CWE-399	Resource Management	416	16.1	2159	83.9

CR=0.014<0.1, the consistency of the judgment matrix was acceptable. Using same method to construct judgment matrix of CWE type (CWE1-CWE22) and exploitable level (B2-B4), and carry out consistency detection to obtain the relevant weight sequence of each layer element to each element in adjacent upper layer. The final weight sequence is calculated by adding relevant weight sequence to weight sequence. According to the equation below, the average value is calculated as the weight of ignored vulnerability type mentioned at Section 4.2 to reduce the error.

$$VT = \frac{total\,weight - minimax\ \ weight}{maximum\,weight - total\,sequencing\,weight} \tag{7}$$

Table 2: The exploitable level and successful exploit probability of 22 CWE types

ID	B1 0.0477	B2 0.1031	B3 0.2253	B4 0.6240	Total Weight	*VT*
1	0.0559	0.0487	0.0171	0.053	0.0446	0.21
2	0.0559	0.027	0.0171	0.053	0.0424	0.19
3	0.0203	0.0161	0.0458	0.053	0.0460	0.22
4	0.034	0.0487	0.0458	0.0258	0.0331	0.12
5	0.0128	0.027	0.0275	0.0998	0.0719	0.44
6	0.2209	0.0161	0.0087	0.008	0.0191	0.00
7	0.034	0.0804	0.0458	0.0258	0.0363	0.14
8	0.0128	0.0804	0.1718	0.008	0.0526	0.28
9	0.0128	0.0161	0.0109	0.2147	0.1387	1.00
10	0.0128	0.027	0.0726	0.0258	0.0359	0.14
11	0.0203	0.027	0.0726	0.0258	0.0362	0.14
12	0.034	0.027	0.0275	0.053	0.0437	0.21
13	0.034	0.0161	0.0458	0.053	0.0467	0.23
14	0.0559	0.027	0.0275	0.053	0.0447	0.21
15	0.034	0.0487	0.0458	0.0258	0.0331	0.12
16	0.0831	0.0487	0.0275	0.0143	0.0241	0.04
17	0.034	0.0487	0.0458	0.0258	0.0331	0.12

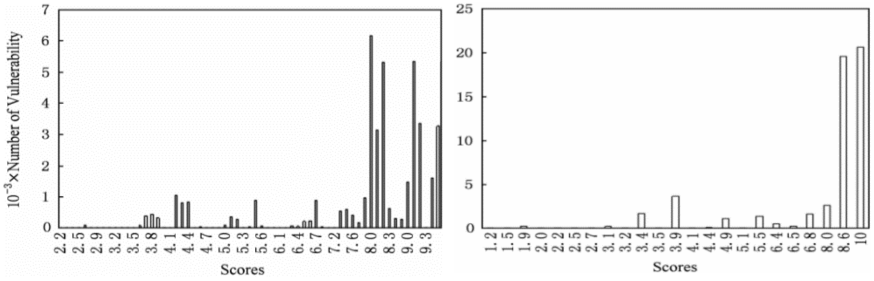
18	0.0203	0.0487	0.0171	0.0998	0.0721	0.44
19	0.0203	0.1969	0.0109	0.0143	0.0326	0.11
20	0.0128	0.0487	0.1718	0.008	0.0493	0.25
21	0.1451	0.0487	0.0171	0.008	0.0208	0.01
22	0.034	0.027	0.0275	0.053	0.0437	0.21

5 Evaluation

We did the experiment on 54,331 vulnerabilities mentioned in Section 4.2. The exploitable score distribution of ICVSS and CVSS are shown in Fig. 2.

5.1 The accuracy of vulnerability exploitable measurement

As shown in Fig. 2(b), there are 66 types of exploitable scores, from 1.4 to 10 points. Compared with the 23 scores obtained by CVSS (as shown in Fig. 2(a)), ICVSS has a finer quantitative granularity and can further classify the vulnerabilities which have the same CVSS exploitable score, but there are also some concentrating scores obtained by ICVSS. For some examples, 6142 samples are concentrated at 8 points, 5292 samples are concentrated at 8.2 points and 5336 samples are concentrated at 9.1 points, it is significantly optimized than that of CVSS.



(a) Exploitable score distribution using ICVSS (b) Exploitable score distribution using CVSS

Figure 2: The exploitable score distribution obtained by ICVSS and CVSS

5.2 Network security situation assessment using ICVSS

The network environment was set up as Fig. 3, it included:

- **Web server**, 192.168.3.2, provided IIS service, had 2 vulnerabilities: cve-2014-2130 (access as administrator) was represented by A; cve-2014-1443(any code can be executed) was represented by B;
- **FTP server**, 192.168.3.3, provided FTP service for external network, had 2 vulnerabilities: cve-2013-2193(get FTP user password) was represented by C; cve-2013-1091(any code can be executed) was represented by D;
- **User**, 192.168.2.2, had 2 vulnerabilities: cve-2006-3747(sensitive information available) was represented by E; cve-2009-4873(get administrator privilege) was represented by F. User communicated to the network, web server, SMTP server and the FTP server by RPCP, the servers is Linux, the user is Windows;

- **SMTP server**, 192.168.4.2, provided mail service, had vulnerability cve-2014-4076 (get administrator privilege) was represented by G;
- **Attacker**, 192.168.1.1, could communicated with FTP, Web server and User, needed to get a user's mail and SMTP server administrator privilege for obtaining STMP server information, SMTP server could only communicated through web server.

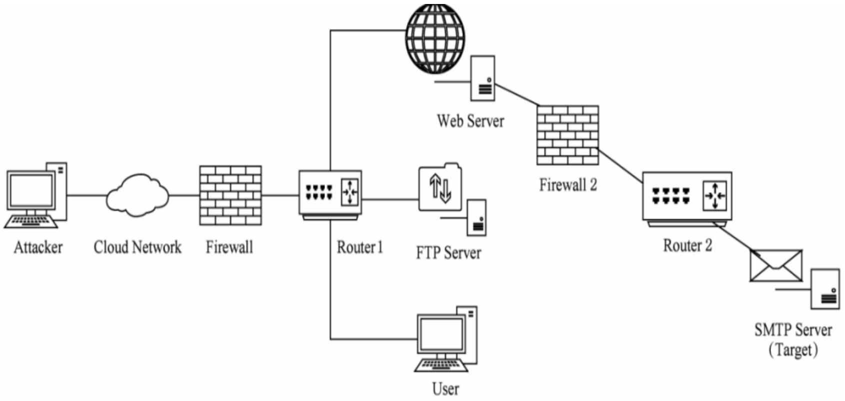


Figure 3: The network environment

According to cve database, we obtained the list of vulnerabilities of experimental environment. Five attack paths were obtained using continuity defined in Section 3.3:

Table 3: The exploitable level and successful exploit probability of vulnerabilities

Vulnerability	*E1*	*E2*	*E3*	*S*	*P1*	*P2*	*P3*
A	0.70	0.8	0.72	0.29	0.203	0.232	0.209
B	0.78	0.8	0.73	0.638	0.498	0.51	0.466
C	0.35	0.32	0.33	1	0.35	0.32	0.33
D	0.92	1	0.91	1	0.92	1	0.91
E	0.55	0.49	0.56	1	0.55	0.49	0.56
F	0.89	1	0.91	1	0.89	1	0.91
G	0.38	0.39	0.36	0.4	0.152	0.152	0.144

- R1: Start-E-A-G-Target;
- R2: Attacker-E-B-G-Target;
- R3: Attacker-F-C-B-G-Target;
- R4: Attacker-D-C-B-G-Target;
- R5: Attacker-B-G-Target;

The exploitable scores E_i were calculated by WIVSS, CVSS and ICVSS. According to Tab. 2, determined select factor S_i [Zhang and Feng (2015)] and calculated the success rate of attack by Eq. (3), and the exploitable scores and selection factor of each vulnerability is shown in Tab. 3.

We did a large number of simulation tests to get actual success probability, calculated the success rate of each attack path by Eq. (4), and calculated average accuracy and select factor of each attack path using WIVSS, CVSS, ICVSS in Tab. 4, it can be seen that ICVSS accuracy is the highest, standard deviation is the lowest, that proved the ICVSS method using continuity metric can effectively detect attack path, and have higher and more stable accuracy rate by adding CWE type as an measure index.

Table 4: Result and comparison

Method	*Route*					Accuracy	S
	R1	***R2***	***R3***	***R4***	***R5***		
WIVSS	0.0170	0.0416	0.0236	0.0224	0.0756	90.6	0.017
CVSS	0.0177	0.0390	0.0255	0.0255	0.0796	88.2	0.021
ICVSS	0.0168	0.0376	0.0201	0.0201	0.0671	92.5	0.010
Reality	0.0165	0.0362	0.0218	0.0223	0.0705		

6 Conclusions

Current quantitative grading methods only quantify the risk from the perspective of vulnerability, ignore the dependency between vulnerabilities, and have low accuracy in exploitable measure. This paper proposed ICVSS quantitative grading method, defined the continuity of vulnerability according to the privilege level to help user find attack path consist of a sequence of vulnerabilities, verified the strong impact of vulnerability type on exploitable level, used AHP algorithm to quantify the CWE type, we did the experiment on 54331 CVE vulnerabilities with 22 CWE types, the result proved that ICVSS can effectively find the attack path, it can solved the problem of the high concentration of CVSS vulnerability exploitable scores to a great extent, improved the accuracy rate of vulnerability exploitable quantitative grading and the accuracy of the network security situation assessment, it can help to provide vulnerability management with more accurate and stable determination.

Acknowledgement: This work is supported by the National Key R&D Program of China under Grants 2017YFB 0802300 and The National Natural Science Fund (No. 0901065614001). The authors would like to thank for Prof. Xingmin Sun, Zhaoqing Pan in Nanjing University of Information Science & Technology for recommendation, Prof. Xuejun Yang, Yuhua Tang, Huaimin Wang, Zhiping Cai in National University of Defense and Technology for tutoring.

References

Aonso, J. A.; Lamata, M. T. (2006): Consistency in the analytic hierarchy process: a new approach. *International Journal of Uncertainty, Fuzzing and Knowledge based Systems*, vol. 14, no. 4, pp. 445-459.

Cai, Z. P.; Chen, M.; Chen, S. G.; Qiao, Y. (2015): Searching for widespread events in large networked systems by cooperative monitoring. *International Conference on Network Protocols*, pp. 123-133.

Cai, Z. P.; Wang, Z. J.; Zheng, K. (2018): A distributed TCAM coprocessor architecture for integrated longest prefix matching, policy filtering, and content filtering. *IEEE Transactions on Computers*, vol. 62, no. 3, pp. 417-427.

Frigault, M. (2008): Measuring network security using Bayesian network-based attack graphs. *Computer Software and Applications,* vol. 22, no. 4, pp. 698-703.

Gopinath, V.; Bhuvaneswaran, R. S. (2018): Design of ECC based Secured Cloud Storage Mechanism for Transaction Rich Applications. *Computers, Materials & Continua*, vol. 57, no. 2, pp. 341-352.

Lai, Y. P.; Hsia, P. L. (2007): Using the vulnerability information of computer systems to improve the network security. *Computer Communications*, vol. 30, no. 9, pp. 2032-2047.

Liu, Q. X.; Zhang, Y. Q. (2012): Improving vrss-based vulnerability prioritization using analytic hierarchy process. *Journal of Systems and Software*, vol. 85, no. 8, pp. 1699-1708.

Liu, Q. X.; Zhang, Y. Q. (2011): Vrss: a new system for rating and scoring vulnerabilities. *Computer Communications*, vol. 34, no. 3 pp. 264-273.

Liu, Y.; Cai, Z. P.; Zhong P. (2011)**:** Detection approach of DDoS attacks based on conditional random fields. *Journal of Software*, vol. 22, no. 8, pp. 1897-1910.

Lu, Y. L.; Xia, Y. (2005): Research on target-computer secure quantitative fusion model. *Chinese Journal of Computers*, vol. 28, no. 5, pp. 914-920.

Miaoui, Y.; Boudriga, N. (2017): Enterprise security investment through time when facing different types of vulnerabilities. *Information Systems Frontiers*, vol. 20, no. 2, pp. 11-24.

Montaigne, D.; Coisne, A.; Sosner, P. (2015): Electrical atrial vulnerability and renal complications in type 2 diabetes. *Diabetologia*, vol. 59, no. 4, pp. 1-2.

Spanos, G.; Sioziou, A. (2013): Wivss: a new methodology for scoring information systems vulnerabilities. *Computer Communications*, vol. 34, no. 5, pp. 83-90.

Tan, T. T.; Wang, B. S.; Zhou, X.; Tang, Y. (2018)**:** The new progress in the research of binary vulnerability exploits. *Springer: Lecture Note in Computer Science*, vol. 11064, pp. 277-286.

Tan, T. T.; Wang, B. S.; Zhou, X.; Tang, Y. (2018): The new progress in the research of binary vulnerability analysis. *Springer: Lecture Note in Computer Science*, vol. 11064, pp. 265-276.

Wang, R. Y.; Gao, L. (2011): An improved cvss-based vulnerability scoring mechanism. *Proceeding of Third International Conference on Multimedia Information Networking and Security*, pp. 352-355.

Wen, T.; Zhang, Y. Q. (2015): ASVC: an automatic security vulnerability categorization framework based on novel features of vulnerability data. *Journal of Communications*, vol. 76, no. 8, pp. 823-895.

Wen, T.; Zhang, Y. (2015): A novel automatic severity vulnerability assessment framework. *Journal of Communications*, vol. 10, no. 5, pp. 786-798.

Zhang, F. L.; Feng, B. (2015): Vulnerability assessment based on correlation.

Application research of Computers, vol. 31, no. 3, pp. 811-814.

Zhang, Y. Q.; Fang, Z. J. (2015): Survey of android vulnerability detection. *Journal of Computer Research and Development*, vol. 52, no. 10, pp. 2167-2177.

Zhou, H.; Watts, J. D.; Aebersold, R. (2001): A systematic approach to the analysis of protein phosphorylation. *Nature Biotechnology*, vol. 10, no. 5, pp. 357-378.