A MULTI-LAYER TREE MODEL FOR ENTERPRISE VULNERABILITY MANAGEMENT

Bin Wu and Andy Ju An Wang
Southern Polytechnic State University, Marietta, GA, U.S.A.

Keywords: Enterprise vulnerability, Multi-level tree model, Assessment, EVMAT, NVD.

Abstract: Conducting enterprise-wide vulnerability assessment (VA) on a regular basis plays an important role in assessing an enterprise's information system security status. However, an enterprise network is always very complex, separated into different types of zones, and consisting hundreds of hosts in the networks. The complexity of IT system makes VA an extremely time-consuming task for security professionals. They are seeking for an automated tool that helps monitor and manage the overall vulnerability of an enterprise. This paper presents a novel methodology that provides a dashboard solution for managing enterprise level vulnerability. In our methodology, we develop a multi-layer tree based model to describe enterprise vulnerability topology. Then we apply a client/server structure to gather vulnerability information from enterprise resources automatically. Finally a set of well-defined metric formulas is applied to produce a normalized vulnerability score for the whole enterprise. We also developed the implementation of our methodology, EVMAT, and Enterprise Vulnerability Management and Assessment Tool, to test our method. Experiments on a small E-commerce company and a small IT company demonstrate the great potentials of our tool for enterprise-level security.

1 INTRODUCTION

Conducting enterprise-wide vulnerability assessment on a regular basis plays an important role in assessing an enterprise's information security status. However, the inherent complexity of information systems and the rapid emergence of new vulnerabilities make it extremely time-consuming task for security professionals. It is common for a moderate enterprise to have hundreds of different IT resources such as computer hardware and software, distributed in different zones of enterprise. When the IT resources scale up, it is increasingly challenging for a centralized manager to scan the vulnerability information against each IT resource in the enterprise networks and further quantify the overall vulnerability status thus create corresponding security mechanisms. Security professionals are seeking an automated tool to help monitor and manage the complex IT resources. In this paper, we present a new multi-layer tree model-based approach to support the centralized management of enterprise vulnerability, which is essential for enterprise risk management.

Our methodology firstly provides an efficient model to describe an enterprise vulnerability topology. Here we abstract an enterprise as a collection of business goals, such as E-commerce, customer services, and financial accounting. These business goals are established by senior managers of the enterprise. Only by meeting these business goals, can a company maintain its competitiveness in the business and market. In an enterprise, there are a large number of IT resources contributing to business goals. For instance, an HTTP server and database play the core roles of E-commerce business goal. In general, we assume that we have a formal description of an enterprise IT resources with respect to the device, weight, and their functioning roles to the business goals. To support the modeling of an enterprise vulnerability topology, we implemented EVMAT, which provides a user-friendly GUI help construct the model using the above three elements.

With the initial model of an enterprise vulnerability topology, we assign different weights and interests to all business goals and resources to identify the importance of a business goal/ resource to the enterprise. Then we utilize Common Vulnerability Scoring System (CVSS, 2006) to calculate the vulnerability scores for all leaf business goals. Thirdly, we provide a multi-layer C/S structure tool to gather and extract system characteristics from each resource in enterprise network based on the Open Vulnerability and Assessment Language (OVAL, 2009) standards and further retrieves vulnerability data from National Vulnerability Database (NVD, 2010) in order to evaluate software vulnerability scores corresponding to those resources. Then we rank the weaknesses of each product installed in a resource to support decision making of security professionals. Fifthly we calculate the overall vulnerability score of a resource and the corresponding impact score to the business goals it contributes to. Finally, we produce a normalized vulnerability score for the whole enterprise based on a set of well-defined metric formulas.

2 RELATED WORK

Existing literature such as Zhang et al.(2008), Adnerson et al. (2005) and Shi et al. (2008) provides different models to describe enterprise security. In particular, Adnerson et al. (2005) provides a formal enterprise level model of security used for canonical representation, identification of components that need to be measured. Shi et al. (2008) provides another modeling methodology to manage the network security in Enterprise. However, both of them did not provide any methodology to measure the security level of an enterprise.

A number of research papers Lee et al. (2001), Liao, Striegel and Chawla (2010), Homer (2009), and Chen et al. (2009) focus on the evaluation and management of enterprise network security. Myerson, Judith M. (2002) indentifies vulnerabilities in an enterprise network environment. Chen et al. addressed a comprehensive approach to enterprise network security management.

There are some researches utilizing the enormous vulnerability data from NVD to evaluate vulnerability of a software product. Wang, J., Wang, H., Guo, M., Zhou, L., Camargo, J. (2010) provides a set of security metrics to rank attacks based on vulnerability analysis. Wang, J. and Guo, M. (2010) proposes a novel methodology of using Bayesian networks to automating the categorization of software security vulnerabilities based on standardized vulnerability data.

3 ENTERPRISE VULNERABILITY TOPOLOGY

In this section we discuss our model of enterprise vulnerability topology for calculating the overall vulnerability score of an enterprise. There are three principles in our modeling method:

An enterprise is a collection of business goals. These business goals form a tree of business goals with each node a business goal associated with a different interest (weight). The root business goal must be the top of the business existence. For each business goal, it may have multiple children business goals.

Each leaf business goal utilizes a number of IT resources to reach its goal. A resource can contribute to one or more business goals. For a pair of resource and business goal, weight is used to measure the importance of a resource contributing to that business object.

Each leaf business goal should have a vulnerability score using the transferred CVSS base metrics. This quantitative score describes the characteristics and impacts on that business goal when it becomes vulnerable due to its internal defects and external threats.

3.1 Model Demonstration

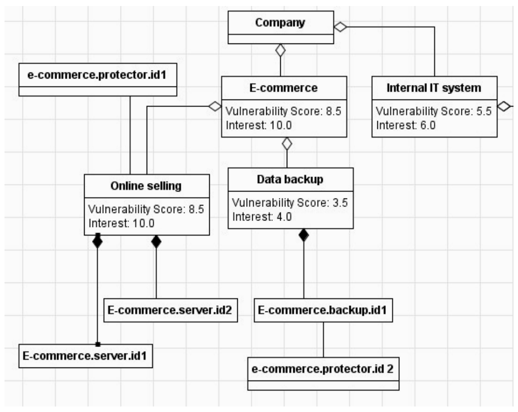


Figure 1: Sample enterprise vulnerability topology.

In Figure 1, ◇ means the relationship between a business goal and its children business goals. ◆ means the relationship between a business goal and the resources it utilizes. From this figure, it is evident that the root business goal is the *Company* node. It has two business goals: one is *E-commerce* and the other is *Internal IT system*. Also *E-commerce* has two children business goals: *Online selling* and *Data backup*. Two servers are used for reaching the goal of

Online selling: *e-commerce.server.id1* and *e-commerce.server.id2*.

Then we need to determine the interest (weight) of a business goal related to its parent business goal. The same as resource, we need to determine the weight (range from 0 -10) of a resource for a business object.

3.2 Multi-layer Tree Model

The model in section 3.1 is useful when modeling a small company. However, when the enterprise scales up, consisting of tens of business goals and hundreds of IT resources, it is not so convenient to put all things in a single chart. Here we introduce the multi-layer tree model to hide details of low layer elements. The top layer tree model can be only main business goals and for the second layer, the root is one of main business goals and the business goal can extend to some sub-business goals. For the lower layers, business goals are connected with resource groups. The lowest level only consists of leaf business goals and resources. Figure 2 demonstrates a sample multi-layer tree model of an IT company.

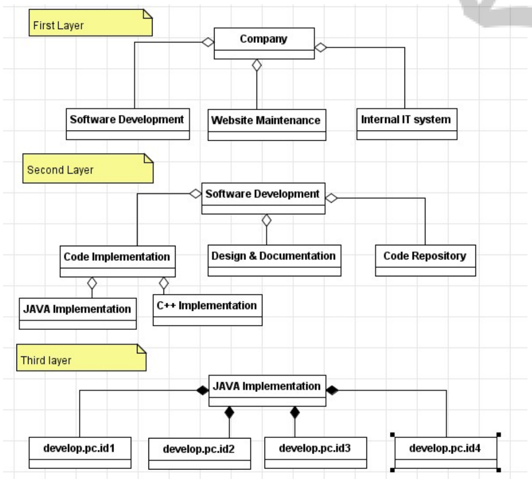


Figure 2: Multi-layer tree model of an IT company.

In figure 2, the top layer of constructed model only consists of three main business goals: *Software Development, Website Maintenance* and *Internal IT system*. The second layer extends *Software Development* business goal to three sub business goals. The third level describes the relationship of *Java Implementation* business goal and the IT resources contributing to it. In this way, our multi-layer model hides details of low layer resources from senior security managers and allows primary security professionals focus on the IT resources belongs to them.

4 TECHNOLOGY DETAILS

In this section, we discuss the algorithm and technical details of our approach based on the discussion above.

4.1 Vulnerability of a Business Goal

In our model, we evaluate the vulnerability score of leaf business goal. CVSS is a suite of standard measurement systems for industries, organizations and governments that need accurate and consistent vulnerability impact scores. It is most used for evaluating individual IT vulnerabilities. In our methodology, we transfer the calculation of base metrics in CVSS to compute the vulnerability of a business goal. The scoring formula of base metrics can be found in (A complete guide to CVSS, 2010).

4.2 Vulnerability of a Software Product

For each machine, we use OVAL Interpreter to attain the system characteristics XML files from different resources, then a product vulnerability calculator extracts the Common Platform Enumeration (CPE, 2010) information of every product and then calculates the vulnerability score based on the data retrieved from NVD. NVD is the U.S. government repository of standards based vulnerability management data represented using the Security Content Automation Protocol (SCAP, 2010) For a record in the NVD, it contains CVE-id (Common Vulnerabilities and Exposures, 2010).vulnerability-configuration, vulnerable-software-list, published date and time, CVSS base metrics and scores, Common Weakness Enumeration (CWE, 2010) summary of a vulnerability and so on. We extract all vulnerability data that has impact on the input product and then compute the software vulnerability score. We adopt the security metrics in Wang et al. (2009) with modification that takes time phase into consideration.

Rigorous measurement of software security can provide substantial help in the evaluation and improvement of software products and processes. However, little agreement exists about the meaning of software security and how to define software security. We define software security metrics based on the representative weakness of the software as shown in the formulas below:

$$SM(s) = \sum_{n=1}^{m} (P_n \times W_n) \qquad (1)$$

Where *SM(s)* stands for the security metrics for the software *s*, anW_id (i = 1, 2, …, m) are the severity of those representative weakness in the software *s*.

Note that a software product may have many weaknesses and flaws. Here "representative" refers to those weaknesses that lead most vulnerabilities that may be exploited by attackers. Suppose the weakness corresponding to W_n has k vulnerabilities and their corresponding CVSS base scores are $V_1, V_2, \ldots, V_k$. The severity of this weakness, W_n, is defined as the average score of them, as demonstrated in the formula (2) below.

$$W_n = \frac{\sum_{i=1}^{K} V_i}{K} \qquad (2)$$

In formula (1), each P_n (n = 1, 2, …, m) represents the risk of the corresponding weakness. We use the percentage each representative weakness occurs in the overall weakness occurrences to calculate P_n as the formula (3) below.

$$P_n = \frac{R_i}{\sum_{i=1}^{m} R_i} \qquad (3)$$

Where R_n is the frequency of occurrences for each representative weakness over a span of time in months, as illustrated in formula (4) below, where K is the number of vulnerabilities related to each representative weakness, and M is the number of months.

$$R_n = \frac{K}{M} \qquad (4)$$

To make the value of software security metrics *SM(s)* to range from 0 to 10, we require that the following formula (5) hold for *Pn*.

As shown in the formulas above, we define software security metrics based on the *representative* weaknesses of the software. For a given piece of software, we first find out those typical weaknesses reported in Common Weakness Enumeration (CWE) related to the software and calculate the number of vulnerabilities caused by these weaknesses. Some weakness causes more vulnerabilities than others. We pick up those weaknesses that cause most vulnerabilities as our "representative weaknesses". After identifying the representative weaknesses for the software, we incorporate the severity of representative weaknesses into the security metrics. The severity of a vulnerability is captured by calculating the percentage of occurrences of this vulnerability compared with the total occurrences of all vulnerabilities.

$$\sum_{n=1}^{m} P_n = 1 \qquad (5)$$

4.3 Vulnerability of a B-R Relationship

Before calculating the vulnerability score of a B-R relationship, users can determine the important level of a specified software product or allow EVMAT tool to assign the important level. For instance, for a resource performs as an HTTP server, the *apache server* and *MySQL* are the core components to achieve its functionality while a browser like Firefox seems to be unrelated. There are three kinds of important level: *core*, *related* and *unrelated*, describing how a product influences the functionality of a resource. After assigning these values, we can calculate the overall vulnerability of a resource.

$$\mathrm{S_R} = \frac{\sum_{\mathrm{i=1}}^{\mathrm{m}} \mathrm{V_i} \times \mathrm{L_i}}{\mathrm{m}}, (1 < i < m) \qquad (6)$$

In formula (6), *Vi* is the vulnerability score of a product installed in the resource. *Li* is the important level of that product. Currently we have Core=1.0, Related=0.5 and Unrelated =0.0. In formula (6), *m* stands for the number of most vulnerable products to a B-R relationship. In our implementation of EVMAT, m=5. This formula first finds m largest $\mathrm{V \times L}$ products and then calculates the overall vulnerable score of a B-R relationship.

4.4 Computing the Weight Tree

The usage of a weight tree is to determine the importance of a resource to the whole enterprise. For instance, in an E-commerce company, the server used for online selling is far important than a personal PC used by an employee. Since it is impractical that companies always apply vulnerability patches immediately for all machines because of the cost of maintenance procedure. Computing the weight tree could help security administrators focus on the most important resources and delay those not so important.

In section 3, we have already modeled the enterprise vulnerability topology. Each business goal has an interest (weight) value related to its parent and each resource has a weight value related to the business goal it contributes. Now we use formula 8 to calculate the weight of a resource to the whole enterprise.

$$\mathrm{W_i} = \mathrm{W_p} \times \frac{\mathrm{w_i}}{\sum_{\mathrm{i=1}}^{\mathrm{m}} \mathrm{w_i}}, (1 < i < m) \qquad (7)$$

In formula (7), $\mathrm{W_p}$ is the weight of i's parent, *m* is the children number of p. The weight of root is 10. Formula (7) iterates from the root to a resource node to calculate the overall weight of the resource to the whole enterprise.

4.5 Overall Vulnerability of an Enterprise

Finally, we calculate the overall vulnerability score of an enterprise. Assume a leaf business goal has vulnerability score sb (calculated by CVSS) and it has n resources weighted $(\mathrm{wr}_1, \mathrm{wr}_2 \ldots, \mathrm{wr}_n)$ and the vulnerability scores are $(\mathrm{sr}_1, \mathrm{sr}_2 \ldots, \mathrm{sr}_n)$. The contributed vulnerability score for that business goal is:

$$\mathrm{s} = \sum_{i=1}^{n} \mathrm{sb} \times \mathrm{wr}_i \times \mathrm{sr}_i \qquad (8)$$

Then we sum up all leaf business goals and normalize the score into (0-100).

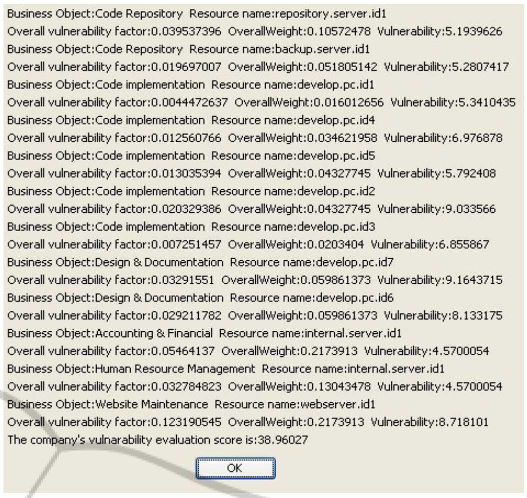
$$\mathrm{es} = \sum_{i=1}^{m} \mathrm{s}_i \times 10 \qquad (9)$$

es is the overall vulnerability score of an enterprise.

5 EXPERIMENT DEMONSTRATION

In this section, we model an IT company vulnerability topology and calculate the overall vulnerability score of that company. The full model is shown in Figure 4 and all resource entities have already read the scored products file it installed. The result of enterprise vulnerability analysis is shown in Figure 3.

From figure 3 we can see the four servers play the core role of the whole enterprise. The company's vulnerability score is 38.96, which is relatively high (The range is 0-100).



```
Business Object:Code Repository  Resource name:repository.server.id1
Overall vulnerability factor:0.039537396  OverallWeight:0.10572478  Vulnerability:5.1939626
Business Object:Code Repository  Resource name:backup.server.id1
Overall vulnerability factor:0.019697007  OverallWeight:0.051805142  Vulnerability:5.2807417
Business Object:Code implementation  Resource name:develop.pc.id1
Overall vulnerability factor:0.0044472637  OverallWeight:0.016012656  Vulnerability:5.3410435
Business Object:Code implementation  Resource name:develop.pc.id4
Overall vulnerability factor:0.012560766  OverallWeight:0.034621958  Vulnerability:6.976878
Business Object:Code implementation  Resource name:develop.pc.id5
Overall vulnerability factor:0.013035394  OverallWeight:0.04327745  Vulnerability:5.792408
Business Object:Code implementation  Resource name:develop.pc.id2
Overall vulnerability factor:0.020329386  OverallWeight:0.04327745  Vulnerability:9.033566
Business Object:Code implementation  Resource name:develop.pc.id3
Overall vulnerability factor:0.007251457  OverallWeight:0.0203404  Vulnerability:6.855867
Business Object:Design & Documentation  Resource name:develop.pc.id7
Overall vulnerability factor:0.03291551  OverallWeight:0.059861373  Vulnerability:9.1643715
Business Object:Design & Documentation  Resource name:develop.pc.id6
Overall vulnerability factor:0.029211782  OverallWeight:0.059861373  Vulnerability:8.133175
Business Object:Accounting & Financial  Resource name:internal.server.id1
Overall vulnerability factor:0.05464137  OverallWeight:0.2173913  Vulnerability:4.5700054
Business Object:Human Resource Management  Resource name:internal.server.id1
Overall vulnerability factor:0.032784823  OverallWeight:0.13043478  Vulnerability:4.5700054
Business Object:Website Maintenance  Resource name:webserver.id1
Overall vulnerability factor:0.123190545  OverallWeight:0.2173913  Vulnerability:8.718101
The company's vulnerability evaluation score is:38.96027
```

Figure 3: Analysis result of a small IT company.

The analysis result implies that the security professionals need to pay more attention to reduce vulnerability score.

Then we assume that due to delayed update of MySQL installed on server *internal.server.id1*, the vulnerability of MySQL becomes 10.0 (previously it was 5.6). The overall vulnerability score increases to 44.06 and the vulnerability score of that server increases to 7.23. Both two examples show the advantages of using our tool to manage IT resources.

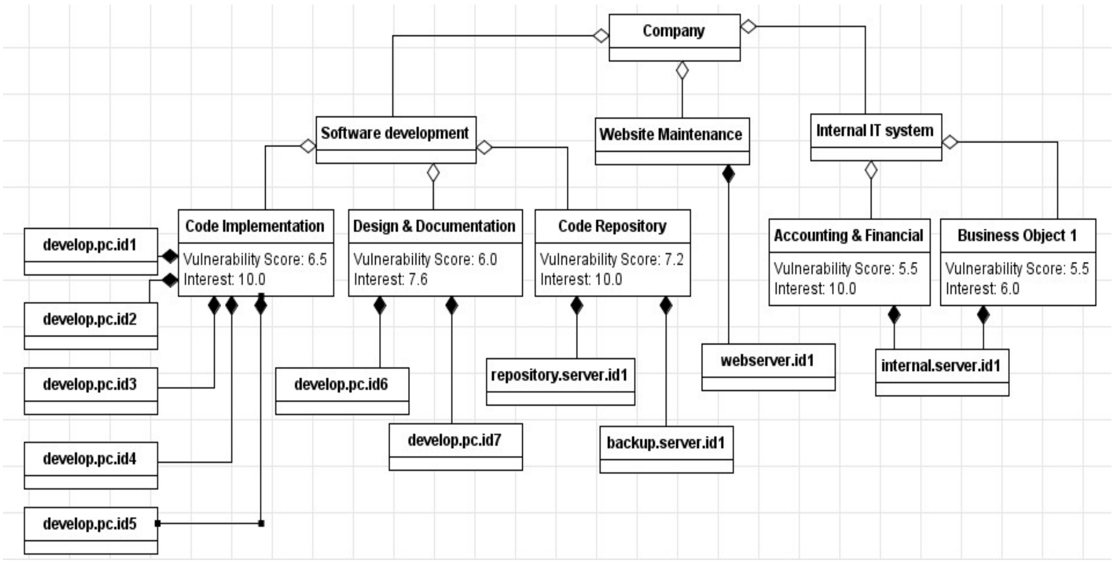


Figure 4: An IT company vulnerability topology model.

6 CONCLUSIONS

This paper presents a model-based automated approach to quantify the overall vulnerability score of a company. Our developed tool, EVMAT 1) provides a user interface to model the enterprise vulnerability topology, 2) automatically gathers system characteristics based on OVAL and further evaluates software vulnerabilities installed in a computer resource based on the vulnerability data retrieved from NVD; 3) Rank the weaknesses of software product to help security administrators decide the product that fits their secure demand most and 4) quantitatively measures the overall vulnerability of an enterprise. The experiment of modeling a small IT company using our tool demonstrates the potentials of this tool.

ACKNOWLEDGEMENTS

This paper is based upon work supported by the National Science Foundation under Grant No. 0722157 and 0941900. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. The authors would like to express their gratitude to anonymous reviewers of this paper for their thoughtful comments and suggestions.

REFERENCES

Mell Peter and Scarfone Karen and Romanosky Sasha.Common Vulnerability Scoring System.*IEE Security and Privary*, 4(6):85-89, 2006.

OVAL, Open Vulnerability and Assessment Language. http://oval.mitre.org/

NVD, National Vulnerability Database. http://nvd.nist.gov/

Shi, Fuqian and Xu, Hongbiao and Wang, Haining. A Representative Management Model of Network Security in Enterprise Informatization. *Proceedings of the 2008 International Conference on Information Management,* volume 2: 304-307, 2008

Zhang, Zonghua and Nat-Abdesselam, Farid and Lin, Xiaodong and Ho, Pin-Han. A model-based semi-quantitative approach for evaluating security of enterprise networks. *Proceedings of the 2008 ACM symposium on Applied computing*, 1069-1074, 2008.

Anderson, Evan and Choobineh, Joobin and Grimaila, Michael R. An Enterprise Level Security Requirements Specification Model. Proceedings of the *Proceedings of the 38th Annual Hawaii International Conference on System Sciences,* 186.3--, 2005

Lee, Jae Seung and Kim, Sang-Choon and Sohn, Seung Won. A Design of the Security Evaluation System for Decision Support in the Enterprise Network Security Management. *Proceedings of the Third International Conference on Information Security and Cryptology*, 246-260, 2001

Liao, Qi and Striegel, Aaron and Chawla, Nitesh. Visualizing graph dynamics and similarity for enterprise network security and management. *Proceedings of the Seventh International Symposium on Visualization for Cyber Security*, 34-45, 2010

Homer, John. *A comprehensive approach to enterprise network security management*. Phd thesis, Kansas State University, 2009

Chen, Xiuzhen and Zheng, Qinghua and Guan, Xiaohong. An OVAL-based active vulnerability assessment system for enterprise computer network. *Information System Frontiers*, 10(5): 573-588, 2009.

Myerson, Judith M. Identifying enterprise network vulnerabilities. *Int. J. Netw. Manag*., 12(3): 135-144, 2002.

Wang, Ju An and Wang, Hao and Guo, Minzhe and Zhou, Linfeng and Camargo, Jairo. Ranking Attacks Based on Vulnerability Analysis. *Proceedings of the 2010 43rd Hawaii International Conference on System Sciences*, 1-10, 2010

Wang, Ju An and Guo, Minzhe. Vulnerability categorization using Bayesian networks. *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*, 29:1—29:4, 2010.

OVAL Interpreter. http://oval.mitre.org/ language/ interpreter.html.

A complete guide to CVSS. http://www.first.org/cvss/cvss-guide.html

CPE, Common Platform Enumeration. http://cpe.mitre.org/

SCAP, Security Content Automation Protocol. http://scap.nist.gov/

CVE, Common Vulnerabilities and Exposures. http://cve.mitre.org/

CWE, Common Weakness Enumeration. http://cwe.mitre.org/

Wang, Ju An and Wang, Hao and Guo, Minzhe and Xia, Min. Security metrics for software systems. *Proceedings of the 47th Annual Southeast Regional Conference*, 47:1—47:6, 2009

CERT, Computer Emergency Response Team at Carnegie Mellon University's Software Engineering Institute. http://www.cert.org/stats/