Poster: System thinking of the Software Vulnerability Market via Complex Network Theory

Keman Huang[1,2], Zhiyong Feng[1,2], Jianqiang Li[3], Xiaohong Li[1,2]
[1]Tianjin Key Laboratory of Cognitive Computing and Application, Tianjin 300072, China
[2]School of Computer Science and Technology, Tianjin University, Tianjin 300072, China
[3]School of Software Engineering, Beijing University of Technology, Beijing 100086, China
keman.huang@tju.edu.cn, zyfeng@tju.edu.cn, lijianqiang@bjut.edu.cn, xiaohongli@tju.edu.cn

***Abstract*—Uncovering the patterns of the software vulnerabilities can be helpful for the policy making to remove or reduce the effect of software vulnerabilities. In order to understand the emerging pattern in the software vulnerabilities, taking advantage of the complex network theory, this paper considers the software vulnerability market as a dynamic complex system and then proposes a heterogeneous network model to represent its evolution over time. The preliminary empirical result uncovers the universal scale-free property, which inspires the further complex network study for the software vulnerability market.**

Keywords— Software Vulnerability Market, Complex Network, System Thinking, Heterogeneous Network Model, Scale-free

I. INTRODUCTION

Information security has been increasingly important in the current business environment supported by the IT system [1]. Due to the inherent vulnerable characteristic of the software running on these systems [2], these vulnerabilities maybe exploited by the attackers which will compromise the IT systems and consequent huge economy lost [3]. As the vulnerability disclosure can effectively promote the vulnerability patching and improve the information security for the software products [4], vulnerability markets are emerging recent years . Agencies such as the Computer Emergency Response Team (CERT), Security Focus etc. will disclose vulnerabilities and the National Vulnerability Database (NVD) consolidates these reported vulnerabilities into a single database for easy access and tracking. In order to guarantee the security of the software products as well as the IT system, the security service providers and the software vendors will offer security software or software patching to reduce or remove the effect of software vulnerabilities.

Recently, some models based on the statistical analysis have been presented to study the patterns in the software vulnerability market intending to understand the patching and disclosure behaviors which can be helpful for security policy development [5]. However, few methodologies consider the software vulnerability market as a complex system that they the emerging properties over time are somehow overlooked. Therefore, in order to analyze the characteristic emerging over time in the software vulnerability market, as well as provide needed diversity in research methods [6], we take advantage of the complex network theory [7] and propose the Vendor-Product-Vulnerability heterogeneous network model, intending to shed valuable insights into the vulnerability software market.

II. SYSTEM THINKING OF SOFTWARE VULNERABILITY MARKET

A. System Thinking of Software Vulnerability Market

In the software vulnerability market, the software vendors release software products to fulfill the consumers requirement. Due to the inherent vulnerable characteristic of the software, the software's vulnerabilities may be uncovered by the discovers who could be the software vendors, the third-party software analyzers, or the hackers. Then the vulnerabilities will be publicly disclosed and the information is freely available to everyone. The hacker community may develop or release exploitations, causing economic loss or damage for the system running the software product. In order to guarantee the software's security, the software vendor need to developed patches so that the consumers can employ an update to remove the vulnerability. Even if a patch is not available or installed, specific countermeasures can provide partial protection against attacks. Hence, the software vulnerability has a life cycle consisting of its discovery, disclosure, exploitation and patching. Furthermore, as time goes by, different software vulnerabilities will go through different life cycle phases driven by behaviors of the vendors, discovers, hackers in the market. Therefore, the software vulnerability market can be considered as a dynamic complex system.

Definition 1 (*Software Vulnerability Market*) : Software vulnerability market is a dynamic complex system consisting of software vendors, software products, software vulnerabilities, vulnerability discovers, hackers and specific countermeasures. Each vulnerability will go through its life cycle phases including discovery, disclosure, exploitation and patching over time.

B. Heterogeneous Network Model

Obviously, the software vendors, software products and software vulnerabilities play the core roles in the software vulnerability market. Based on the relations among them, we can formally define the heterogeneous network model for the software vulnerability market as follow:

Definition 2 (*Vulnerability-Product-Vendor Heterogeneous Network Model for Software Vulnerability Market, VPV*) : $VPV = \{Vu, Pr, Ve, R_{cp}, R_{pv}\}$ where $Vu = \{vu_i\}$ refers to the set of the related vulnerabilities, $Pr = \{pr_j\}$ refers to the related products, $Ve = \{ve_k\}$ refers to the related vendors, $R_{cp} = \{< vu_i, pr_j, t_{cp} >\}$ represent the relations among the vulnerabilities and products and t_{cp} is the time when the vulnerability is public disclosed into the market; $R_{pv} = \{< pr_j, ve_k, t_{pv} >\}$ represent the relations among the products and the vendors, and t_{pv} is the time that product is released into the market.

For each vulnerability, we should consider how it transfers between different phases, how serious the vulnerability is and what type the vulnerability belongs to. Therefore, we can further define the vulnerability as follow:

$$vu_i =< vun_i, vnt_i, vnl_i, vns_i > \tag{1}$$

where vun_i is the name of the vulnerability; vnt_i is the vulnerability's type; vnl_i is the lifecycle of the vulnerability; and vns_i refers to the severity of the vulnerability.

As the vulnerability discovered date t_{dis} is mostly unavailable for the public and the product released date t_r is not straightly related for the vulnerability, in this paper, we focus on the disclosure date t_d , the exploit date t_e and the solution date t_s and the other two are out of scope.

$$vnl =< t_d, t_e, t_s > \tag{2}$$

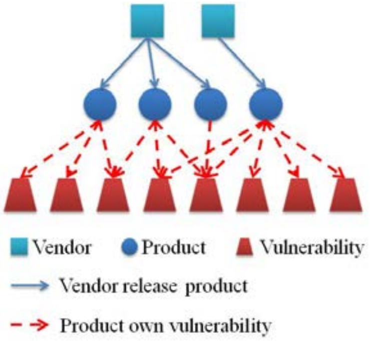

Fig. 1. Vendor-Product-Vulnerability Hetergeous Network Model.

Straightforwardly, given a series of consecutive time intervals $t_1, t_2, \ldots t_n$, we can get the vulnerabilities disclosure in each interval $t_i, 1 \le i \le n$ to construct the snap *VPV* network $VPV(t_i)$. Therefore, the evolution of the software vulnerability market can be formally defined as the series of *VPV* network $\{VPV(t_i), 1 \le i \le n\}$.

III. Scale-free Nature of Networks

The National Vulnerability Database (NVD) is the most comprehensive public database consisting of the historical disclosed vulnerabilities nowadays. Each vulnerability in the NVD is allocated with the Common Vulnerability and Exposures Identifier (CVE-ID) as the index, the Common Vulnerability Scoring System (CVSS) measuring the severity level and the Common Weakness Enumeration Specification (CWE) representing the vulnerability type. Therefore, we collect all the historical disclosed vulnerabilities from NVD since 2008. And then considering the time interval as one year, we group the vulnerabilities based on their disclosure date so that we can get a series of VPV networks to represent the evolution of the software vulnerability market over time. Similar to [8], for each snap *VPV* network, we calculate the *vulnerability's in-degree* representing the affected products of each given vulnerability, the *product's out-degree* representing the number of vulnerabilities for the given product, the *vendor's out-degree* representing the number of released products. We find that all these three distributions for each snap *VPV* network meet the long-tail power-law distribution[1], which means that the software vulnerability market is actually a complex system with the universal scale-free property [9] but only a random market. Only few vendors release larger number of products in the market and they occupy the core position in the market. While most products only own a few discovered vulnerabilities, some products have a lot of disclosed vulnerabilities. Most of them are operation system relevant products in the *software ecosystem* and they attract many attentions from vendors, third-party researchers and hackers. Therefore, the potential vulnerabilities have a larger possibility to be discovered. Similarly, most vulnerabilities are specific for products while some are common for most of the products. Obviously, focusing on fixing these vulnerabilities could gain most security welfare for the whole market.

IV. Conclusion

In order to understand the evolution of the software vulnerability market, we consider it as a complex, dynamic system and then the heterogeneous network model is presented to quantify its emerging pattern over time. The preliminary empirical study based on the NVD software vulnerability market shows that the software vulnerability market is actually a complex system with scale-free property, which is a universal architectures in many real networks. Hence, in the future, we will further go deep into this property and uncover the mechanism for these emerging patterns along time.

References:

[1] R. Anderson and T. Moore, "The economics of information security," Science, vol. 314, pp. 610-613, 2006.

[2] M. Shahzad, M. Z. Shafiq and A. X. Liu, "A large scale exploratory analysis of software vulnerability life cycles," in Proceedings of the 34th International Conference on Software Engineering, Zurich, Switzerland, 2012, pp. 771-781.

[3] S. Ransbotham, S. Mitra and J. Ramsey, "Are markets for vulnerabilities effective?" MIS Quarterly, vol. 36, pp. 43-64, 2012.

[4] A. Arora, R. Krishnan, R. Telang, and Y. Yang, "An empirical analysis of software vendors' patch release behavior: impact of vulnerability disclosure," Information Systems Research, vol. 21, pp. 115-132, 2010.

[5] T. August and M. F. Niculescu, "The Influence of Software Process Maturity and Customer Error Reporting on Software Release and Pricing," Management Science, vol. 59, pp. 2702-2726, 2013.

[6] M. A. Mahmood, M. Siponen, D. Straub, H. R. Rao, and T. S. Raghu, "Moving toward black hat research in information systems security: an editorial introduction to the special issue," Mis Quarterly, vol. 34, pp. 431-433, 2010.

[7] M. Newman, Networks: an introduction: Oxford University Press, 2010.

[8] K. Huang, Y. Fan and W. Tan, "Recommendation in an Evolving Service Ecosystem Based on Network Prediction," IEEE Transactions on Automation Science and Engineering, vol. 11, pp. 906-920, 2014.

[9] A. Barabási, "Scale-free networks: a decade and beyond," Science, vol. 325, p. 412, 2009.

[1] Due to the space limitation, we will not include the figures in this paper. However, these supported figures will be available in our website: colmanzf.com/colman/