Software Vulnerability and Application Security Risk

Jianping Peng, Sun Yat-Sen University, Guang Zhou, China
Meiwen Guo, Xinhua College of Sun Yet-Sen University, Guang Zhou, China
Jing Quan, Salisbury University, Salisbury, USA

ABSTRACT

This research investigates the software vendor-based relationships between software vulnerability and application security risk. The data is obtained from the China National Vulnerability Database of Information Security (CNNVD). At first, we use the latent class model to classify the software vendors into three categories, and then employ regression models to estimate relationships between software vulnerability and application security risk for each of the three categories of the software vendors. The results show the relationships vary across the software vendors. The findings suggest that an IT vendor should learn specific vulnerability features according to its type to effectively avoid vulnerability generation on their products.

KEYWORDS

Information Security, Security Vulnerability, Software Process, Software Risk

1. INTRODUCTION

In spite of advances in software testing and quality assurance (Lewis, 2016), it is impossible to completely avoid security vulnerabilities in software product, due to the high degree of the complexity of software development. In fact, software vulnerabilities are growing steadily in recent years. Software security vulnerability is one of the important causes for software quality risks. Moreover, the security breaches often result in monetary losses (Jerman-Blažič, 2008).

One of the reasons for the rise of software vulnerabilities is the high cost for bug hunting and vulnerability discovery. Another reason is due to the externality of software products, which allows software vendors to take no direct responsibility for the losses caused by security risks. As a result, software vendors have little incentives for improving the safety of their software products (Byung, 2009).

To encourage software vendors to actively reduce software security vulnerabilities and improve the overall software quality, many countries have taken steps to build national software security vulnerability databases (Wu and Zhang, 2012). The databases will register any verified vulnerability in a vendor’s software product and record will be published in the public domain to remind users of the security issue. In the meantime, the vendor is urged to publish software patches to fix the security vulnerability and reduce the risks of software applications. In theory, software vendors can learn

DOI: 10.4018/IRMJ.2019010103
from this process and improve the overall quality of their products. But in reality, little research has been done on how to investigate of the learning process.

This research aims to fill the gap by focusing the relationship between the type of software vendors and vulnerability features. Based on the data from the China National Vulnerability Database of Information Security (CNNVD), we attempt to answer the following three research questions:

**RQ1:** What is the best classification of vendors in CNNVD?

**RQ2:** What are vital vulnerability features influencing risks of different vendors’ software products?

**RQ3:** What is the learning mechanism for different software vendors for reducing the risks in their software applications?

### 2. LITERATURE REVIEW

Software vulnerability is one of the important reasons for poor security. Software vulnerability analysis mainly covers fundamental research on the generation, discovery, utilization, management, and reduction of vulnerabilities. The focus of the current vulnerability research is on the study of vulnerability classification, dynamic and static analysis of code vulnerability, rules and standards of vulnerability, and vulnerabilities fixing (Wu, 2009). Software security vulnerabilities are usually caused by design defects in system. For example, a vulnerability may be generated due to a design flaw of failing to detect the size of the data buffer, which leads to a memory stack overflow and forces the computer to execute the codes supplied by the attacker (Kuperman, Brodley, Ozdoganoglu, Vijaykumar, and Jalote, 2005). From a different angel, Syed, Rahafroz & Keisler (2018) study how social media attends software vulnerability information. They argue that a higher volume of retweets of vulnerabilities is an indication of public attention to such information. They ran a negative binomial regression to predict retweet count based on tweet content categories, source, and technical features of tweets as well as the features of software vulnerabilities.

Vulnerability classification is one of the important areas in the research of software vulnerabilities (Mell, Scarfone, & Romanosky, 2006). It describes the vulnerabilities from various perspectives including the cause of the vulnerability, the utilization of vulnerability technique, and vulnerability function and scope. For the purpose of systematically supervising and classifying vulnerability information, many countries have set up Vulnerability Databases to collect and save vulnerability information features and solutions (Wu and Zhang, 2012). In China, the government set up China National Vulnerability Database of Information Security (CNNVD) in 2009. The CBBVD aims to provide software vulnerability analysis, early warning and risk evaluation services. Wu and Zhang (2012) carried out a comparative study of vulnerability data from the USA and China and found that the CNNVD was highly dependent on foreign databases and was deficient in the application related to behaviour mining on software vendors.

The economic loss related to software vulnerabilities is well documented. For example, Telang and Wattal (2007) found that one vulnerability announcement would result in a deduction of 0.6% of the software vendors’ stock share prices. To limit the potential losses, researchers propose many methodologies to improve software quality and reduce vulnerability occurrence. For instance, Aberdeen (2010) proposed three different countermeasures for enterprises to lessen security threats and latent vulnerabilities in application software: discovery and repairing, maintenance, and protection of the source code.

Software quality control methodologies, including software testing model, control framework in R&D process, and new vulnerability testing software, are successful for software quality improvement at various degrees (Nie, Zhao, Chen & Han, 2011; Evans & Mahanti, 2012; Sune & Nielsen, 2013). However, because of complexity and economical externality of software products, it is impossible to eliminate software security vulnerabilities completely (Wei, Wang & Zou, 2009). Arora, Krishnan,
Process Batch Offloading Method for Mobile-Cloud Computing Platform
Pawan Kumar Thakur and Amandeep Verma (2015). Teaching Cases Collection (pp. 1-13).
www.igi-global.com/article/process-batch-offloading-method-for-mobile-cloud-computing-platform/148161?camid=4v1a

Digital Knowledge Management Artifacts and the Growing Digital Divide: A New Research Agenda
www.igi-global.com/chapter/digital-knowledge-management-artifacts-growing/13718?camid=4v1a