基于漏洞严重程度分类的漏洞预测模型

高志伟[1],姚 尧[2],饶 飞[2],刘延钊[3],罗 平[2]
(1.石家庄铁道大学信息科学与技术学院,河北石家庄 050043;2.清华大学软件学院,北京 100084;3.中国信息安全测评中心,北京 100085)

摘 要: 软件漏洞预测模型有许多种,能预测软件中存在的漏洞总数以及发生的时间间隔,但不能预测软件漏洞的严重程度.然而在某些场合,如软件可信性,我们不仅要考虑软件漏洞发生的总数和时间间隔,而且也要考虑漏洞发生的严重程度对软件可信性的影响.既是在传统的软件安全性研究中,考虑漏洞发生的严重程度的影响,对软件的使用和风险控制也是很重要的.本文基于传统的马尔可夫模型,将软件漏洞按发生的严重程度进行分类,获得了一种新的软件漏洞预测数学模型.利用该模型不仅能够预测软件中存在的漏洞总数和时间间隔,而且同时也能预测每一类的漏洞总数和漏洞种类,试验表明有较好的准确度,这是其它漏洞预测模型所无法预测的.

关键词: 漏洞预测模型; 马尔科夫链; 漏洞严重程度; 分类预测
中图分类号: TN911.23 **文献标识码:** A **文章编号:** 0372-2112 (2013) 09-1784-04
电子学报 URL: http://www.ejournal.org.cn **DOI:** 10.3969/j.issn.0372-2112.2013.09.018

Predicting Model of Vulnerabilities Based on the Type of Vulnerability Severity

GAO Zhi-wei[1], YAO Yao[2], RAO Fei[2], LIU Yan-zhao[3], LUO Ping[2]
(1. *College of Information Science and Technology, Shijiazhuang Tiedao University, Shijiazhuang, Hebei* 050043, *China*; 2. *School of Software, Tsinghua University, Beijing* 100084, *China*; 3. *China Information Technology Security Evaluation Center, Beijing* 100085, *China*)

Abstract: There are many kinds of software vulnerability prediction models which are capable of predicting the total number and the time interval of occurrence of vulnerabilities in the software. But none of them can predict the severity of software vulnerabilities. However, in some cases, such as software credibility, we have to consider the total number of software vulnerabilities and time interval as well as the vulnerability severity affecting the trustworthiness of software. Considering the impact of the vulnerability severity, the application and risk control of software is also very important in the traditional software security. Based on the traditional Markov model, we classified the severity of software vulnerabilities occurrence, proposed a new software vulnerability prediction mathematical model. The model can not only predict the total number of software vulnerability and the time interval, but also the total number vulnerabilities of each class as well as the type of the vulnerabilities. Our tests showed that it has better accuracy, and the type of information that other prediction models can not offer.

Key words: predicting model of vulnerabilities; Markov chain; prediction with classificaiton; third party prediction of vulnerabilities

1 引言

随着网络的发展,软件漏洞对信息系统安全的威胁越来越大,因此对漏洞特点、出现、量化乃至预测的研究有了客观的需求.

Schultz[1]定义软件漏洞为"一种缺陷,使得入侵者可以绕过安全机制",Pfleeger[2]将漏洞定义为"安全系统中可被用来引起损失或危害的弱点",Y Shin[3]认为"软件漏洞是软件规格、开发或配置中缺陷的一个实例,它的运行违反潜在或外在的安全策略".

人们希望找到一种可以在一定程度上预测漏洞的方法.Alhazmi[4]讨论了软件漏洞的度量、分析和预测等问题的基本方面,Sridhara、Malaiya[5]讨论了漏洞的量化问题,Yonghee Shin、Chowdhury[6]等讨论了复杂度对漏洞以及安全性的影响,Caragea[7]、Williams等提出了一些基于经验的漏洞预测方法,Jinyoo Kim[8]等人研究了多版本软件中的漏洞的特性.在预测模型方面有热动力模型[9],它是最早应用于研究软件可靠性,由Anderson将

收稿日期:2012-11-23;修回日期:2013-05-08;责任编辑:李勇锋
基金项目:国家自然科学重点基金(No.90818021,No.9071803)

其引用到漏洞的动态预测中来;对数泊松模型[10]是一种传统对数泊松可靠性增长模型在漏洞预测研究中的应用;Rescorla 提出的两种趋势模型:二次模型[11]和指数模型[12],分别称为 RQ 和 RE;由 Alhazmi 和 Malaiya 提出的逻辑模型[4],讨论了软件漏洞的度量、分析和预测等一系列问题的基本方面陈恺等人[13]提出了多周期模型.

我们从第三方的角度出发,基于经验数据来研究漏洞的预测问题.所谓第三方,是指既非软件开发者,也非漏洞发掘者的评测机构.

2　漏洞与概率的关系

软件漏洞是软件缺陷的一种,与一般的软件缺陷相比,漏洞有如下特点:首先,漏洞是能够被人利用来进行入侵行为的缺陷;其次,漏洞的发现是人尝试攻击的结果,而不是机器运行的结果;最后,漏洞在软件的不同版本中的出现有一定的相似性.

由于我们不可能事先知道漏洞出现的时间、数量或者类型,因此漏洞的出现有着随机性,可以用一个数字量化它的可能性.无论是否被发现,漏洞都是客观存在的,我们的主观意志无法干涉漏洞出现与否.

通过以上分析,我们认为可以将漏洞的出现看做是一个概率问题.

3　漏洞发现概率的基本性质

为了简化讨论,我们做如下理想的假设:当软件的一个漏洞被发现后,它立刻被修补,并且不产生新的漏洞.这样,我们的起始状态就是软件没有发现漏洞的状态,设随机变量 ξ 为软件初始状态到发现漏洞所经历的时间,$F_\xi(t)$ 表示 ξ 的分布函数,t 表示任意时刻.$F_\xi(t)$ 即为 t 时刻发现漏洞的概率,我们设

$$R_\xi(t) = P_r\{\xi > t\} = 1 - F_\xi(t) \tag{1}$$

当 $t \to \infty$ 时 $F_\xi(t) \to 1$,$R_\xi(t) \to 0$.$R_\xi(t)$ 物理上表示不发现漏洞的概率.

我们定义漏洞发现率为软件在 t 时刻没有发现漏洞的情况下,在$(t, t+\Delta t)$时间内,当 Δt 很小时,单位时间内发现漏洞的概率,用 $\lambda(t)$表示.那么:

$$\begin{aligned}\lambda(t) &= \lim_{\Delta t \to 0} \frac{P_r\{t + \Delta t > \xi > t \mid \xi > t\}}{\Delta t} \\ &= \lim_{\Delta t \to 0} \frac{P_r\{t + \Delta t > \xi > t, \xi > t\}}{\Delta t \cdot Pr\{\xi > t\}} \\ &= \lim_{\Delta t \to 0} \frac{F_\xi(t + \Delta t) - F_\xi(t)}{\Delta t \cdot R(t)} \\ &= \frac{f(t)}{R(t)} = \frac{-R'(t)}{R(t)}\end{aligned} \tag{2}$$

其中,$f(t)$是随机变量 ξ 的密度函数.以初始条件 $R(0) = 1$解一阶常微分方程,假设漏洞发现率为常数,得到:

$$R(t) = e^{-\lambda t} \tag{3}$$

随机变量 ξ 的数学期望可以表示发现漏洞的平均时间间隔

$$\mathrm{ADTI} = \int_0^\infty t f(t)\,dt = \int_0^\infty \lambda t e^{-\lambda t} dt = \frac{1}{\lambda} \tag{4}$$

4　马尔可夫预测模型

马尔可夫链所反映的最本质的属性是马尔可夫性,又称为无后效性,它的直观含义是,在确切知道系统现在的状态的条件下,系统将来的状况与过去的状况无关.

如果漏洞的发现不满足马尔可夫性,则有将来的某个漏洞的发现依赖于以前的某个漏洞的发现,这与我们的假设矛盾.同时,对于第三方评测者来说,这类信息通常无法获得,也缺乏重要性.因此,我们认为,可以使用基于马尔科夫链的模型.

首先我们定义软件所处的一系列状态 $S = \{s_0, s_1, s_2, \cdots, s_n, s_{n+1}, \cdots\}$,其中 s_n 表示现有状态,即发现了最近的一个漏洞,s_{n+1}为发现下一个漏洞的状态,分别对应时间点 t_n 和 t_{n+1}.假设在 Δt 时间内,如果发现漏洞,那么只发现一个漏洞.

软件在 Δt 时间内发现一个漏洞的概率为 $\lambda\Delta t$.软件在 Δt 时间内不发现漏洞的概率为 $1 - \lambda\Delta t$.根据假设,Δt 时间内,只可能发现一个漏洞.又根据马尔可夫性,我们只讨论相邻两个状态之间的关系.所以,在 $t + \Delta t$ 时间内,软件只可能从 s_n 状态转移到s_{n+1}状态,不会从 s_{n+1}状态转移到 s_{n+2}状态.那么 s_{n+1}为吸收态.软件从 s_{n+1}状态转移到 s_n 状态的概率为0,软件从 s_{n+1}状态转移到 s_{n+1}状态的概率为1.状态转移图如下:

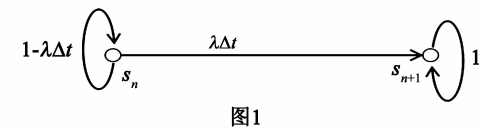

图1

设 t 时刻软件处于状态 s_n 的概率为 $P_n(t)$,处于状态 s_{n+1}的概率为 $P_{n+1}(t)$;设 $t + \Delta t$ 时刻软件处于状态 s_n 的概率为 $P_n(t + \Delta t)$,处于状态 s_{n+1}的概率为 $P_{n+1}(t + \Delta t)$.可得

$$\begin{cases} P_n(t + \Delta t) = (1 - \lambda\Delta t) P_n(t) + 0 \cdot P_{n+1}(t) \\ P_{n+1}(t + \Delta t) = \lambda\Delta t P_n(t) + 1 \cdot P_{n+1}(t) \end{cases} \tag{5}$$

当 $\Delta t \to 0$ 时,以初始条件 $P_n(0) = 1$ 解方程组,得:

$$\begin{cases} P_n(t) = e^{-\lambda t} = R(t) \\ P_{n+1}(t + \Delta t) = 1 - e^{-\lambda t} = F_\xi(t) \end{cases} \tag{6}$$

5　模型实现

我们使用 Jelinski-Moranda(J-M)模型进行计算.假

设软件中固有漏洞数 N_0 是一个未知的常数,漏洞的发现率在任意发现间隔时间内是常数,正比于软件的剩余漏洞数,在第 i 个时间间隔内,发现率为:

$$\lambda = \Phi(N_0 - i + 1) \tag{7}$$

其中 Φ 和 N_0 都是未知数,通过最大似然估计可求得它们的估计值.漏洞发现的时间间隔为

$$\mathrm{ADTI} = \int_0^{\infty} tf(t)\,\mathrm{d}t = \int_0^{\infty} \lambda t \mathrm{e}^{-\lambda t}\mathrm{d}t = \frac{1}{\lambda} = \frac{1}{\boxed{\Phi}\left(\boxed{N_0} - i + 1\right)} \tag{8}$$

6 基于错误分类的预测模型

在实际应用中,如果可以分类预测漏洞,将给漏洞的处理带来很大的方便.在下文中,我们将使用"传统漏洞预测模型"来指代上面的模型,它没有考虑漏洞的分类,我们将其拓展,进行漏洞分类预测.

我们假设按漏洞分为 k 类,其中 k 为正整数.设 $S = \{s_0, s_1, s_2, \cdots, s_k\}$,其中 s_0 为初始状态,即发现了最近一个漏洞的状态,s_i 为发现下一个漏洞为 i 类的状态,$i \in [1,k]$.

假设在 Δt 时间内只发现一个新漏洞,即只进行一次状态转换.设 i 类错误的发现率为 λ_i,$i \in [1,k]$.软件在 Δt 时间内发现 i 类漏洞的概率为 $\lambda_i \Delta t$,软件在 Δt 时间内不发现漏洞的概率为 $1 - \sum_{i=1}^{k} \lambda_i \Delta t$.由于我们只讨论相邻两个状态之间的问题,所以 s_i 到 s_0 或 s_j 的转移概率为 0,$j \in [1,k]$,$j \neq i$.s_i 到 s_i 的转移概率为 1.状态转移图如下:

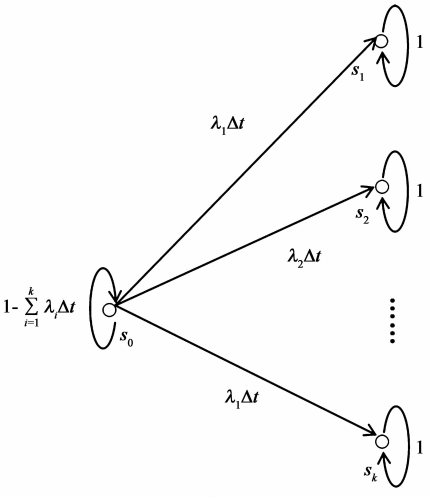

图2

设 t 时刻软件处于状态 s_0 的概率为 $P_0(t)$,处于状态 s_i 的概率为 $P_i(t)$;设 $t + \Delta t$ 时刻软件处于状态 s_0 的概率为 $P_n(t + \Delta t)$,处于状态 s_i 的概率为 $P_i(t + \Delta t)$,$i \in [1,k]$.可得到

$$P_0(t + \Delta t) = \left(1 - \sum_{i=1}^{k} \lambda_i \Delta t\right) P_0(t) + 0 \cdot \sum_{i=1}^{k} \lambda_i \Delta t P_i(t) \tag{9}$$

同理:

$$P_i(t + \Delta t) = \lambda_i \Delta t P_0(t) + 1 \cdot P_i(t) + 0 \cdot \sum_{j \in [1,k], j \neq i} P_j(t) \tag{10}$$

联立上述 $k+1$ 个方程:

$$\begin{cases} P_0(t) = \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t} \\ P_1(t) = \dfrac{\lambda_1}{\sum_{i=1}^{k}\lambda_i}\left(1 - \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t}\right) \\ \vdots \\ P_k(t) = \dfrac{\lambda_k}{\sum_{i=1}^{k}\lambda_i}\left(1 - \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t}\right) \end{cases} \tag{11}$$

$P_0(t)$的意义是从上一个漏洞被发现后到 t 时刻未发现新漏洞的概率,$P_i(t)$的意义是从上一个漏洞被发现后到 t 时刻发现一个第 r 类新漏洞的概率,实际上是软件发现第 i 类漏洞的概率分布函数,显然

$$\begin{cases} R(t) = P_0(t) = \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t} \\ F_r(t) = P_r(t) = \dfrac{\lambda_r}{\sum_{i=1}^{k}\lambda_i}\left(1 - \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t}\right) \end{cases} \tag{12}$$

发现第 r 类漏洞的平均时间间隔

$$\mathrm{ADTI}_r = \int_0^{\infty} tf_r(t)\mathrm{d}t = \int_0^{\infty} \lambda t \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t}\,\mathrm{d}t = \frac{\lambda_r}{\left(\sum_{i=1}^{k}\lambda_i\right)^2} \tag{13}$$

7 模型实现

我们将漏洞分为 k 类,假设软件中第 $r(r \in [1,k])$ 类固有漏洞数 N_r 是一个未知的常数,并且满足

$$\sum_{i=1}^{k} N_r = N_0 \tag{14}$$

假设第 r 类漏洞的发现率在任意给定的发现间隔时间内是常数,其数值正比于软件的第 r 类剩余漏洞数,在第 j_r 个时间间隔内,发现率为:

$$\lambda_r = \Phi_r(N_r - j_r + 1) \tag{15}$$

其中 Φ_r 和 N_r 都是未知数,通过最大似然估计可求得它们的估计值.

第 r 类漏洞发现的时间间隔为

$$\begin{aligned} \mathrm{ADTI}_{rj_r} &= \int_0^{\infty} tf_r(t)\,\mathrm{d}t = \int_0^{\infty} \boxed{\lambda_r}\, t \mathrm{e}^{-\sum_{i=1}^{k}\lambda_i t}\,\mathrm{d}t \\ &= \frac{\boxed{\lambda_r}}{\left(\sum_{i=1}^{k}\boxed{\lambda_r}\right)^2} \\ &= \frac{\boxed{\Phi_r}\left(\boxed{N_r} - j_r + 1\right)}{\left(\sum_{i=1}^{k}\boxed{\Phi_r}\left(\boxed{N_r} - j_r + 1\right)\right)^2} \end{aligned} \tag{16}$$

上式可以用来估算软件下一次发现第 r 类漏洞的间隔时间.为了得到所有种类漏洞的发现间隔时间,我们利用上式引入第 k 次发现第 r 类漏洞的绝对时间

$$\boxed{T_{rk}} = \sum_{i=1}^{k} \boxed{\mathrm{ADTI}_{\eta_r}} \tag{17}$$

将得到的所有种类漏洞的 $\boxed{T_{rk}}$ 进行排序,得到序列 $T_1, T_2, \cdots$,从而得到每个漏洞的绝对时间和种类.

8 实例分析

我们从中国国家信息安全漏洞库截取了从 2005 年 10 月 1 日到 2006 年 10 月 31 日的关于 Windows XP 操作系统的漏洞记录,共 65 条.取前 40 个数据作为样本.原始模型的标准差为 17.65,分类模型的标准差为 18.77.分类模型预测的准确率约为 75%.分类模型不但更准确,而且可以预测类别.

9 结论

传统的马尔可夫漏洞预测模型能够预测漏洞总数和发现时间间隔,分类预测模型能够预测每一种漏洞的总数和发现时间间隔,以及漏洞种类,准确度较高。当漏洞只被分为一类时,分类预测模型退化为传统马尔可夫漏洞预测模型。

参考文献

[1] Schultz Jr EE, Brown DS, Longstaff TA. Responding to Computer Security Incidents[OL]. Lawrence Livermore National Laboratory, ftp://ftp.cert.dfn.de/pub/docs/csir/ihg.ps.gz, 1990.

[2] Pfleeger Charles P. Security in Computing[M]. USA: Prentice - Hall, 1997.46 - 48.

[3] Shin Y, Williams L. Is complexity really the enemy of software security[A]. Proceedings of the Fourth ACM Workshop on Quality of Protection[C]. Alexandria, Virginia, USA: ACM, 2008.47 - 50.

[4] Alhazmi OH, Malaiya YK, Ray I. Measuring, analyzing and predicting security vulnerabilities in software systems[J]. Computers & Security, 2007, 26(3): 219 - 228.

[5] Alhazmi OH, Malaiya YK. Prediction capabilities of vulnerability discovery models[A]. Annual Reliability and Maintainability Symposium[C]. Newport Beach, CA: RAMS, 2006.86 - 91.

[6] Shin Y, Williams L. An empirical model to predict security vulnerabilities using code complexity metrics[A]. Proceedings of the Second ACM-IEEE InternationalSymposium on Empirical Software Engineering and Measurement[C]. Kaiserslautern, Germany: ACM, 2008.315 - 317.

[7] Zhang Su, Caragea D, Ou Xinming. An empirical study on using the national vulnerability database to predict software vulnerabilities[A]. Proceedings of the 22nd International Conference Database and Expert Systems Applications[C]. Toulouse, France: DEXA, 2011.217 - 231.

[8] Kim J, Malaiya YK, Ray I. Vulnerability discovery in multi-version software systems[A]. IEEE International Symposium on Software Reliability Engineering[C]. Seattle, Washington: IEEE CPS, 2008.299 - 300.

[9] Anderson R. Security in open VeTSUS closed systems-The dance of Boltzmann, Coase and Moore[A]. Proceedings of the Conference on Open Source Software Economics[C]. Cambridge: MIT Press, 2002.1 - 15.

[10] Musa J D, Iannino A, Okumoto K. Software Reliability Engineering[M]. NY: McGraw-Hill, 1999.193 - 223.

[11] Musa J D, Okumoto K. A logarithmic Poisson execution time model for software reliability measurement[A]. Proceedings of the 7th International Conference on Software Engineering[C]. Orlando: IEEE Press, 1984.230 - 238.

[12] Rescorla E. Is fining security holes a good idea[J]. IEEE Security and Privacy, 2005, 3(1): 14 - 19.

[13] 陈恺,冯登国,苏璞睿,等.一种多周期漏洞发布预测模型[J].软件学报,2010(9):2367 - 2375.
Chen Kai, Feng Deng-guo, Su Pu-rui, et al. Multi-cycle vulnerability discovery model for prediction[J]. Journal of Software, 2010(9): 2367 - 2375. (in Chinese)

作者简介

高志伟　男,1972 年 12 月出生,副教授、硕士生导师.北方交通大学计算机应用专业工学硕士.主要从事分布式计算、大型信息系统、网络安全研究工作.
E-mail: gao_zhiwe@163.com

姚　尧　男,1985 年 2 月出生.清华大学软件学院硕士.主要从事软件可信性、软件漏洞等方面的研究.
E-mail: yaoyaosuper@gmail.com