软件安全性缺陷测试需求获取与定位

彭会斌,费 琪
(江苏自动化研究所,江苏 连云港 222061)

摘 要:近年来,软件安全性事件层出不穷,涉及的领域也越来越广,造成的危害也越来越大。现有的缺陷数据库包含的安全性漏洞数量非常庞大,如果对其逐个进行针对性测试,则测试成本难以承受。因此,文中首先从影响软件安全性的缺陷引入原因维、危险后果维以及可能导致缺陷被激活的操作方式维三个维度对安全性缺陷进行分类。这种三维结构综合分类法,可以弥补单一分类法的不足,为测试人员分析安全性缺陷提供了更为准确细致的描述手段;其次,通过数据流图结合数据交互边界提出一种可行的基于数据交互边界的软件安全性缺陷确定技术;最后,通过对 DREAD 模型的改进,提出一种软件安全性缺陷优先级度量模型,从而解决了软件安全性缺陷定位问题和软件安全性缺陷优先级确定问题。

关键词:软件安全缺陷;安全缺陷需求获取;安全缺陷定位;优先级度量模型
中图分类号:TP31 **文献标识码**:A **文章编号**:1673-629X(2019)08-0107-06
doi:10.3969/j.issn.1673-629X.2019.08.021

Requirement Acquisition and Location for Software Security Defect Testing

PENG Hui-bin,FEI Qi
(Jiangsu Institute of Automation,Lianyungang 222061,China)

Abstract:In recent years,software security incidents emerge in an endless stream,involving more and more fields and causing more and more harm. The existing defect database contains a large number of security vulnerabilities. If the targeted tests are carried out one by one,the test cost is unbearable. Therefore,firstly the security defects from three dimensions:the reason dimension,the dangerous consequence dimension and the operation mode dimension which may cause the defects to be activated are classified. This three dimensional structured comprehensive classification can make up for the single classification and provide a more accurate and detailed description for testers to analyze security defects. Secondly,a feasible software security defect determination technique based on data flow graph and data interaction boundary is proposed. Finally,by improving the DREAD model,a software security defect priority measurement model is proposed to solve the problem of software security defect location and software security defect priority determination.

Key words:software security defects;security defect requirement acquisition;security defect location;priority measurement model

0 引 言

随着面向对象、构件软件、分布式软件等新技术的兴起,软件安全性变得日益重要,并成为制约软件技术发展与应用的一个重要因素[1]。特别是在一些涉及国家和军事机密的应用场合,软件的安全性更是被认为是软件产品的首要质量属性。近年来,软件安全性事件层出不穷,涉及的领域也越来越广,造成的危害也越来越大,而且这种趋势的增长速度十分惊人。

在实际的安全测试过程中,由于测试本身的局限性,不可能做到理论上的“完全”测试[2-4],即不能对软件所有的运行场景进行测试和验证,因此也就不能检测到所有可能的异常操作。而这些异常操作往往是非法用户精心构造,能够利用软件本身存在的缺陷进行的,是一种在实际正常使用中操作概率非常小的操作。同时,实际使用中,用户在关心软件能够做什么的同时,也更看重软件系统可以避免怎样的攻击[5-6]。由于现行安全性测试标准的可行性不高、测试人员对安全机制的忽视以及军用软件等关键软件对安全性的苛刻要求,导致传统的软件安全性功能测试并不能完全适应军用软件安全性定型测试[7];相比之下,从软件

收稿日期:2018-08-22 **修回日期**:2018-12-26 **网络出版时间**:2019-03-27
基金项目:国防科工局技术基础科研(JSZL2017207B013)
作者简介:彭会斌(1974-),男,研究方向为软件可靠性、软件安全性、软件工程与质量。
网络出版地址:http://kns.cnki.net/kcms/detail/61.1450.TP.20190327.1620.006.html

可能具有的典型安全性缺陷出发,通过分析用户的反向使用行为可能触发的软件缺陷来获取测试需求则具有更强的针对性。也就是说,软件安全性缺陷测试需求是从非法用户的视角确定软件不应该具有的非法功能和缺陷,是一种反向测试需求。

软件安全性测试作为保证软件安全性的一种重要途径,对软件产品进行安全性评价具有重要意义[8-9]。根据国家规划,软件测试技术将作为保证软件产品安全性的重要手段,力争使测试越早介入软件开发,从而提高软件抵御潜在风险的能力,降低软件的安全效率。

软件安全性测试需求的获取是生成软件安全性测试用例进行实际测试的基础。目前在提取软件安全性需求方面主要采取的是安全性用例技术,通过描述遐想的异常场景,进而人工分析提出可能存在的安全性缺陷。然而这种获取安全性测试需求方法的缺点是:需要专业的软件安全需求分析人员,并有从事软件安全性分析的大量经验;获取软件安全性需求的人为主观性较大;获取软件安全性需求时容易造成溢漏或重复;不能对需求进行优先级确定。

软件安全性测试需求的准确、全面获取一直是软件测试所追求的目标之一,但这是一个极为复杂的问题,就目前的研究成果而言,离全面准确还存在相当大的距离。但其成功解决对提高软件质量,缩短软件测试时间都具有十分重要的理论价值和理论意义。

文中首先从影响软件安全性的缺陷引入原因维、危险后果维以及可能导致缺陷被激活的操作方式维三个维度对安全性缺陷进行分类;其次,通过数据流图结合数据交互边界提出一种可行的基于数据交互边界的软件安全性缺陷确定技术;最后,通过对 DREAD 模型的改进,提出一种软件安全性缺陷优先级度量模型,从而解决了软件安全性缺陷定位问题和软件安全性缺陷优先级确定问题。

1 基于测试的软件安全性缺陷三维结构表示

1.1 软件安全性缺陷引入原因维

软件安全性缺陷的产生主要是由于程序员不正确和不安全编程引起的[10-12]。大多数程序员在编程初始就没有考虑到安全问题。在后期,由于用户不正确的使用以及不恰当的配置都可能引入安全性缺陷[13-14]。分析安全性缺陷产生原因的目的是减少安全缺陷的产生。通过分类、统计分析,软件安全性缺陷产生的原因不外乎以下几种:

(1)对象不正确交互。

对象包括:独立构件、模块、程序、进程或者系统。交互引用包括:数据发送、数据接收。

(2)危险资源管理方式。

软件系统中,对重要系统资源进行不正确的创建、使用、转移或者破坏。

(3)不正确的安全防护措施。

相关防护或检测技术被滥用、误用或者被简单忽视。

1.2 软件安全缺陷行为导致软件产生误操作方式维

软件安全性缺陷在具体被测软件中被非法利用,是通过一定的使用行为体现出具体的执行过程的,而这种具体行为称之为对软件产生影响的具体方式,这在一定程度上能够指导具体的基于软件安全性缺陷的测试行为。

这里的实体包括构件、类、函数以及数据库中的项等。如基于构件的软件系统中,通常这种实体就是指被测软件中的构件;而在面向对象的软件中,通常是指类的删除;对于传统的面向过程的软件系统中,通常实体就是指函数,这种函数包括用户自定义的函数,也包括软件中预定义或引用的其他函数;在数据库系统中,这种实体通常就是指数据库中的一条记录。

(1)非法删除软件实体。

这种缺陷是指非法用户通过恶意删除软件系统中的实体,从而达到其相应的非法使用目的。通常这种误操作方式可以不只是针对单一实体,也可能是针对多个实体的删除操作。

(2)非法增加新实体。

这种缺陷是指非法用户通过在被测软件中非法地增加新实体,从而达到其相应的非法使用目的。同理在采用不同实现技术的被测软件中,实体的具体形式有所不同。

(3)非法修改实体。

相对于前两类非法修改方式,通过非法修改实体从而达到非法用户相应目的的方式最为常见。

1.3 软件安全性缺陷产生危险后果维

软件安全性缺陷被非法利用,即非法用户通过攻击行为使用,从其可能造成的危险后果的视角来看,可以分为以下四类:

(1)非法执行目标代码。

非法用户利用被测软件系统中特定的软件安全性缺陷,可以达到执行目标代码的目的。这可能只是非法用户初步的目的,并不是最终产生的最为危险的后果。如对于缓冲区溢出缺陷,虽然其最终目的可能是非法访问数据或者修改目标对象,但是对于非法用户最初的目的就是按照其用意非法执行特定目标代码。

(2)非法修改目标对象。

这里的目标对象也包括不同粒度的对象,但是这并不同于误操作分类方面的修改实体。这里的非法修

改目标对象是一种软件特定执行的不可预期的结果,而并不是基于软件安全性缺陷的特定执行方式。如对于缓冲区溢出缺陷,非法用户利用缺陷进行攻击的目的可能就是由溢出造成对内存数据进行修改,而为了达到这种目的,可以通过软件系统中相应的输入构件。这里的修改也包括对目标对象的删除。

(3)非法访问数据对象。

数据的非法访问是大多数软件安全性缺陷被利用造成的危险后果。对于关键软件系统的安全性,重点就是保护关键数据信息的安全性,而数据信息安全性的首要保证就是防止非法访问。

(4)造成拒绝服务。

拒绝服务主要是针对软件系统结构设计上的缺陷,从而可能造成的拒绝服务的危险后果。

分别从软件安全性缺陷引入原因、软件安全性缺陷行为导致软件产生的误操作方式以及可能产生的危险后果三个视角对安全性缺陷进行描述,可以对后续的基于软件安全性缺陷的测试提供信息。软件安全性缺陷的引入原因可以用软件安全性缺陷测试需求的获取,即被测软件中可能存在的缺陷的初步定位;软件安全性缺陷行为导致软件产生的误操作方式的分类有助于指导具体的测试执行;而软件安全性缺陷产生的危险后果就是缺陷被利用最终产生的超出软件设计者预期的输出。

其中三维度中的每个类型还可以随着以后知识的积累进行扩充,这样便于以后对软件安全性缺陷进行更加细致的划分。图 1 给出了这种分类方法的形象描述。

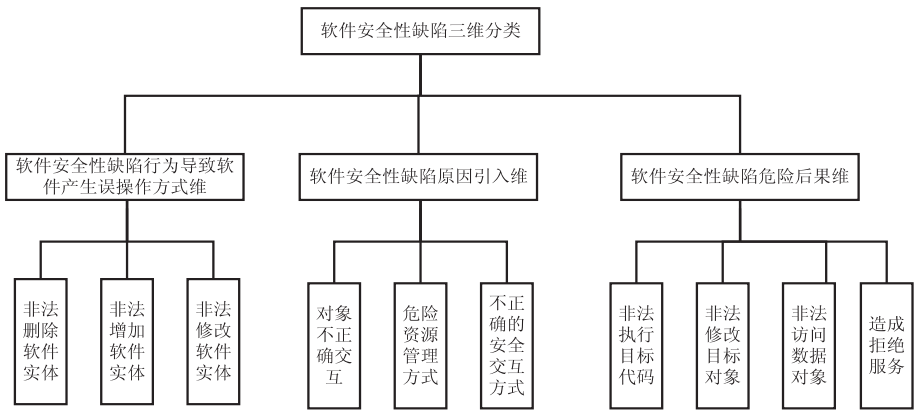

图 1　软件安全性缺陷三维结构分类模型

对于软件安全性缺陷,首先从引入的原因进行分析,然后再对其进行 Destructive 测试,就可以仿真出可能的危害,根据破坏效果也就可以对安全性缺陷进行分类。最后在对软件系统进行回归修补时,不可避免地会涉及到修改方法,所以基于此也可以对缺陷进行分类总结。这种三维结构综合分类法,可以弥补单一分类法不够完善的不足,为测试人员提供一定的支持。

2　基于数据交互边界的软件安全性缺陷定位技术

2.1　确定软件安全性缺陷可能存在的交互路径

确定软件安全性缺陷可能存在的路径,就是确定最有可能存在安全性缺陷的领域,并确定软件应该采用何种避免措施或保护措施。如果没有采用相应的措施,则就确定为软件安全性缺陷。

(1)需要确定数据信息来源。在软件系统中,所有的外部数据信息的来源可以概括为以下几种:用户输入、环境变量输入、文件系统输入、网络输入、进程输入。根据测试的级别不同,可以对数据信息的来源进行裁减。而在这些数据源中,环境变量输入、文件系统输入以及进程输入都是本地计算机系统的输入,一般情况下,认为其较为可信;而用户输入和网络输入由于其用户的不可控性,认为其风险等级较高。将风险等级用三维数组(a,b,c)表示,其中 a 取值为本地输入或者外地输入, b 取值为具体的数据输入类型, c 为风险等级。所以软件系统风险等级列表为:(本地输入,环境变量输入,非常低);(本地输入,进程输入,低);(本地输入,文件系统输入,中);(外部输入,用户输入,高);(外部输入,网络输入,非常高)。

(2)需要确定各种用户访问类别。每个应用程序至少都有匿名的访问方式,即使这样也会执行身份鉴别。但是,虽然这个用户要经过身份鉴别,但他和应用程序之间的交互是以匿名方式进行的。将用户风险等级用三维数组(a,b,c)表示,其中 a 取值为远程用户或者本地用户, b 取值为具体的用户类型, c 为风险等级。用户风险等级列表为:(远程用户,匿名用户,非常高);(远程用户,经身份鉴别的用户,中);(远程用户,具有文件操作能力的远程用户,高);(本地用户,具有执行权限的用户,低);(本地用户,管理用户,非常低)。

(3)绘制软件系统的顶层带有交互边界的数据流图。图 2 所示为一个 Web 应用系统的典型例图。其中用户表示进程输入、文件系统输入、用户输入和网络输入的统称,是一种抽象表示,后续不作特殊说明保持不变。

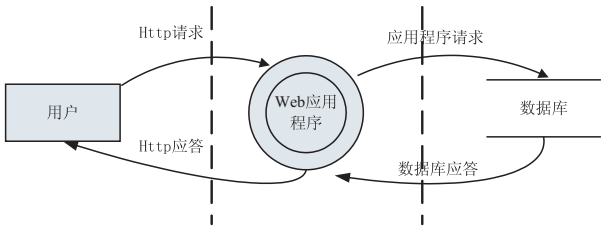

图 2 软件系统顶层数据流交互边界

(4)根据软件实际结构绘制第 0 层带有数据交互边界的数据流图,即抽取出每种操作过程。通常的基于 B/S 或者 C/S 的软件系统都存在典型的登录过程和注册过程,如图 3 所示。

(5)确定安全性缺陷可能存在的路径。安全缺陷路径使用二维数组(a,b)表示,其中 a 表示访问类型, b 表示可能存在缺陷的交互路径。安全缺陷可能存在的路径有如下可能:(匿名远程用户,身份证明信息);(远程用户,注册信息);(经身份鉴别的远程用户,启动信任信息)。

2.2 确定可能存在的软件安全性缺陷

通过上一小节,已经确定了安全性缺陷可能存在的路径,在此基础上对多个数据处理进程进行分解,最终确定可能存在的缺陷。

(1)需要对第 0 级基于数据交互边界的数据流图进行分解和详细描述,抽取第 1 层基于数据交互边界的数据流图。根据图 2,对用户登录过程进行分解,如图 4 所示。

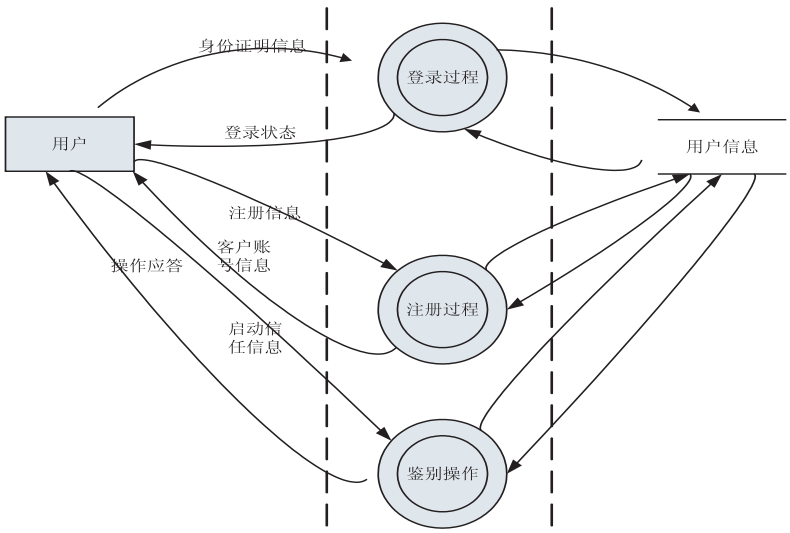

图 3 软件系统第 0 层数据流交互边界

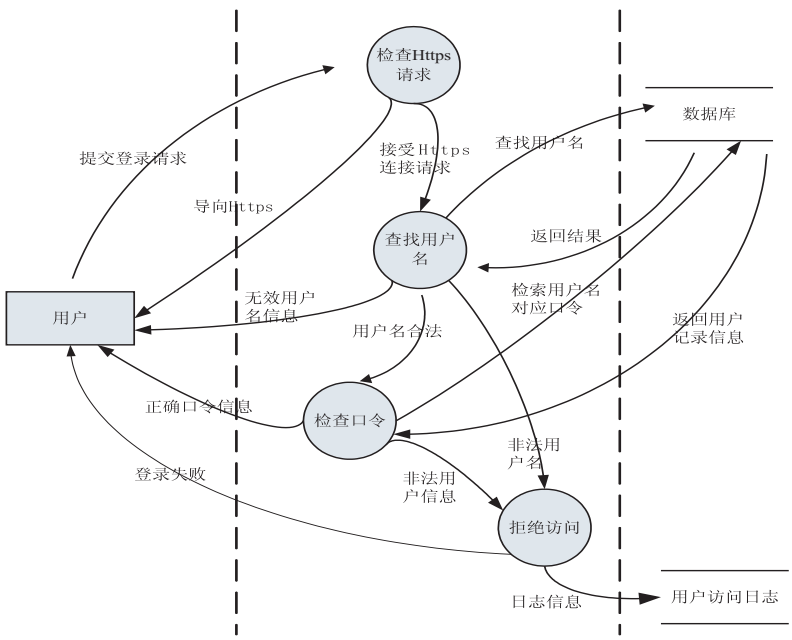

图 4 登录过程的第 1 层数据流交互边界

(2)基于典型软件安全性缺陷,对穿越数据交互边界的数据流信息进行分析,确定可能存在的缺陷。对于图 3 登录过程中可能存在的缺陷为:登录数据信息中包含的安全缺陷为元字符缺陷、命令注入缺陷、鉴别与授权混淆缺陷、访问控制缺陷;返回无效用户数据信息可能存在的缺陷为:异常错误处理缺陷;正确口令信息可能存在的缺陷为:随机数缺陷;登录失败数据信息可能存在的缺陷为:异常错误处理缺陷。

3　基于改进 DREAD 模型的软件安全性缺陷优先级度量

3.1　确定影响软件安全性缺陷优先级的属性

DREAD 模型是 Michael Howard 和 David Leblanc 提出的一种定性度量模型[8-9],模型是用于分出软件安全性威胁的严重程度级别的一种有效技术。同时考虑到了威胁的五种属性:

(1)潜在的破坏性:如果这个漏洞被利用,产生的破坏程度;

(2)再现性:探测并利用这个漏洞需要的时间;

(3)可利用性:漏洞被利用的可能性;

(4)受影响的用户:漏洞利用的影响会有多大;

(5)可发现性:漏洞被发现的可能性。

这是一种主观模糊评价模型,当需要修补的漏洞较多时,就根据模型确定的等级进行。

DREAD 模型提出的属性对软件安全性缺陷并不是完全适用,针对软件安全性缺陷特点,提取了以下五种软件安全性缺陷属性。

(1)普遍性/流行性(prevalence,PE):也就是缺陷在当前同类软件中可能存在的可能性。

对于普遍性的度量,采用交叉比较的方法:

· 相关漏洞在 CWE 漏洞数据库中的排名(CWE Top_25CWE/SANS Top 25 Most Dangerous Programming Errors)。

· 软件安全性缺陷相关威胁在 OWASP 威胁数据库中的排名(OWASP Top_10 Top Ten Cyber Security Menaces for 2008)。

· 软件安全性缺陷相关漏洞是否在 Microsoft 已发布的漏洞数据库中。

· 相应软件安全性缺陷在以往的测试过程中出现的次数。(初始设为 0)

PE 分类分为 4 个等级:广泛、高、普通、有限。

广泛:这种缺陷在实际项目中最为常见,限制这种类型的不超过四类。

高:这种缺陷经常会遇到,但是并不是最常见的。

普通:这种缺陷偶尔也会在实际中遇到。

有限:这种缺陷在实际项目中很少或者几乎没有被发现过。

(2)危害性(damage potential,DP):如果软件安全性缺陷被非法执行,则会对软件系统或者用户的使用产生的破坏程度;危害等级包含:关键、高、中等及低。

关键:这种缺陷的危害等级最高,所以应该立即进行修正。如果这种缺陷被非法利用,则会造成严重的破坏。在选择或者相互比较确定典型缺陷列表时,通常设定危害等级为关键的一般限制在四个以内。

高:这种缺陷的危害等级也是很高,但是相对于关键类型较低。

中等:这种缺陷也是一定要上报的,但是并不是非常紧要的。

低:这种缺陷不一定需要上报,或者并不需要进行及时的修改。

(3)可探测性(detectable,DE):软件中软件安全性缺陷可能被非法用户探测到的可能性,容易探测到,意味着缺陷容易被利用,容易产生攻击。可探测性等级包含:容易、一般及困难。

容易:已存在自动化工具或者技术可以自动化探测缺陷,或者通过简单的操作就可以被发现。

一般:仅仅通过自动化工具或者成熟技术不能发现缺陷,软件安全性缺陷的探测可能需要对程序的逻辑结构有一定的了解;或者通过现有的攻击工具,只能在少数情况下发现缺陷。

困难:发现这种缺陷需要花费大量的时间、手工执行的方法,以及智能的半自动化工具的支持和专业的攻击技术。

(4)影响用户级别(affected users,AU):软件安全性缺陷影响到用户的级别,影响的用户分别包含:管理员、多个普通用户、特殊配置用户(单个)及 Guest 用户。

管理员:缺陷被非法利用时,管理员以及大部分用户的正常使用受到影响甚至限制,系统处于瘫痪状态。

多个普通用户:缺陷一旦被利用,多种普通用户(如图书馆信息系统中除管理员外的 VIP 用户,一般学生会员,图书信息上载员,图书信息审核员等)的使用受到严重影响,但是系统管理员的基本使用功能可以正常进行。

特殊配置用户(单个):这种缺陷一旦被利用,则不会影响大多数类型的用户,往往只能影响到特殊类型用户。如图书馆信息系统中一般的学生会员的借阅,或者图书信息员的上载。这种缺陷和上一级区别别在于,这种缺陷被非法利用,只会影响到一种除管理员以外的任何用户的正常使用,而不会同时影响多个类型用户。

Guest 用户:这种缺陷被非法用户利用,则只会影

响到类似 Guest 用户的使用,或者匿名用户的使用。

(5)可靠性(reliability,RE):软件安全性缺陷被非法用户利用后产生的结果的可预期性。按照缺陷后果可分为 4 个等级:危险、一般、受限及低。

危险:非法用户利用软件安全性缺陷使用相同的方法,多次执行相同的攻击用例,可以产生相同的结果。这种缺陷对于软件系统来说,一旦被非法用户发现,就容易多次重复攻击。

一般:相对于上一级,这种缺陷被非法用户发现后,如果非法用户需要重复攻击行为,需要重新对攻击场景进行一定的修改。

受限:这种缺陷被非法用户发现后,如果非法用户需要重复攻击行为,则需要一定的辅助攻击手段或其他资源。

低:对于这种缺陷,测试人员基本可以忽略,因为就算非法用户曾经利用这种缺陷对软件系统进行了攻击,但是缺陷被重复利用的可能性极低,甚至只是理论上。

这种分级模型考虑到的影响因素都是定性因素,只能对缺陷进行定性判断,为了定量分析优先级,得到软件安全性缺陷优先级影响因素度量表,如表 1 所示,分别设定相应的定量权值为 4、3、2、1。

表 1 软件安全性缺陷优先级影响因素度量

属性	1	2	3	4
PE	有限	普通	高	广泛
DP	低	中等	高	关键
DE	困难	一般	-	容易
AU	Guest 用户	特殊配置用户(单个)	多个普通用户	管理员
RE	危险	一般	受限	低

3.2 软件安全性缺陷优先级(PRI)的计算

通过表 1 可以确定软件安全性缺陷各个影响因素的标度值,考虑到不同的被测件对这些影响属性的要求不同,所以设每种属性的权重集为 W:

$$W = (W_{PE}, W_{DP}, W_{DE}, W_{AU}, W_{RE})$$

影响因素集为 F:

$$F = \begin{pmatrix} PE \\ DP \\ DE \\ AU \\ RE \end{pmatrix}$$

所以可得 PRI:

$$PRI_i = W_{DP}\, PE_i + W_{OP}\, DP_i + W_{RP}\, DE_i + W_{AU}\, AU_i + W_{RE}\, RE_i$$

其中,i 表示软件安全性缺陷的编号。

4 结束语

软件安全性缺陷测试需求获取与定位为测试人员提供了一种获取软件安全性缺陷的手段及方法。利用软件安全性缺陷测试需求获取与定位方法,测试人员能快速提取到相关软件潜在的缺陷,并对潜在的缺陷进行优先级排序,从而能够更充分及更快速地发现软件安全性缺陷。

参考文献:

[1] DAVIS N, HUMPHREY W, REDWINE S T, et al. Processes for producing secure software[J]. IEEE Security & Privacy, 2004, 2(3): 18-25.

[2] 李舟军,张俊贤,廖相科,等. 软件安全漏洞检测技术[J]. 计算机学报,2015,38(4):717-732.

[3] 刘 璇. 计算机软件安全漏洞检测技术[J]. 电子技术与软件工程,2016,14(20):210.

[4] 陈 峰,李伟华. 软件安全分析的有穷自动机模型[J]. 西北大学学报:自然科学版,2011,41(1):22-26.

[5] 王维静,王树明,陈 震,等. 软件安全的多指标综合评测[J]. 计算机工程与应用,2006,42(11):94-96.

[6] 陈 峰,李伟华,陈 昊,等. 采用模型检测器的软件安全模型验证方法[J]. 西安交通大学学报,2011,45(2):15-20.

[7] HOWARD M, LEBLANC D. Writing secure code[M]. 2nd ed. USA: Mircosoft Press, 2005.

[8] WYSOPAL C, NELSON L, ZOVI D D, et al. The art of software security testing[M]. [s. l.]: [s. n.], 2014.

[9] DU Wenliang, MATHUR A P. Testing for software vulnerability using environment perturbation[C]//Proceeding of international conference on dependable systems and networks. New York, NY, USA: IEEE, 2000: 603-612.

[10] 胡 昌. 基于知识库的安全需求获取方法[D]. 天津:天津大学,2012.

[11] RAHMAN M A, AL-SHAER E. A formal approach for network security management based on qualitative risk analysis[C]//2013 IFIP/IEEE international symposium on integrated network management. Ghent, Belgium: IEEE, 2013: 244-251.

[12] 危胜军,何 涛,胡昌振,等. 基于组件依赖图的软件安全漏洞预测方法[J]. 北京理工大学学报,2018,38(5):525-530.

[13] NYRE A A, JAATUN M G. Seeking risks: towards a quantitative risk perception measure[C]//International conference on availability, reliability, and security. Berlin: Springer, 2015: 256-271.

[14] 杜经农,卢炎生. 一种 Web 软件安全漏洞分类方法[J]. 计算机工程与应用,2009,45(25):10-14.