doi :10.3969/j.issn.1671-1122.2012.01.021

计算机安全漏洞检测技术综述

宋超臣，黄俊强，王大萌，段志鸣

（黑龙江省电子信息产品监督检验院，黑龙江哈尔滨 150090）

摘 要：由于信息系统已被广泛应用到国家的各个领域，信息系统的安全显得尤为重要。计算机安全漏洞已经成为信息系统的最主要的威胁之一，因此发现信息系统的漏洞检测技术也成为了目前的研究热点。文章对目前漏洞检测技术中的静态检测技术、动态检测技术和混合检测技术进行了概述，并对各种技术的优缺点进行了比较分析。

关键词：漏洞检测；静态检测；动态检测；混合检测

中图分类号：TP393.08 **文献标识码：**A **文章编号：**1671-1122（2012）01-0077-03

A Survey on Detecting Techniques of Computer Security Vulnerability

SONG Chao-chen, HUANG Jun-qiang, Wang Da-meng, Duan Zhi-ming

(*HLJ Province Electronic & Information Products Supervision Inspection Institute, Harbin Heilongjiang* 150090*, China*)

Abstract: Information system has been widely used in every field of country, it is very important to the security of information system. Computer security vulnerability is one of the primary threats, so how to detect Computer security vulnerability become a research focus. This paper summarize the detecting techniques of computer security vulnerability, and analyses the advantages and disadvantages of these techniques.

Key words: vulnerability detection; static detection; dynamic detection; hybrid detection

0 引言

随着信息技术的发展，软件功能越来越强大，随之而来的是庞大的源代码数量。导致消除源代码中所存在的设计漏洞或实现漏洞越来越困难，黑客利用漏洞可以对系统进行破坏甚至入侵系统。信息安全已成为信息系统的重要问题，据 CNCERT/CC2010 统计，近年来漏洞数量呈现明显上升趋势。黑客攻击目的也逐渐转向经济利益，攻击手段层出不穷，因此加大对漏洞挖掘技术的研究力度才能有效地保障信息安全。

1 漏洞的定义与特点

可以将漏洞定义为存在于一个系统内的弱点或缺陷，这些弱点或缺陷导致系统对某一特定的威胁攻击或危险事件具有敏感性，或具有进行攻击威胁的可能性[1]。软件漏洞产生通常是由于在软件的设计和实现中由于开发人员有意或无意的失误造成是系统潜在的不安全性。漏洞可以划分为功能性逻辑漏洞和安全性逻辑漏洞。功能性逻辑漏洞是指影响软件的正常功能，例如执行结果错误、执行流程错误等。安全性逻辑漏洞是指通常情况下不影响软件的正常功能，但如果漏洞被攻击者成功利用后，有可能造成软件运行错误甚至执行恶意代码，例如缓冲区溢出漏洞、网站中的跨站脚本漏洞（XSS）、SQL 注入漏洞等[2]。

漏洞具有以下特点：1）编程过程中出现逻辑错误是很普遍的现象，这些错误绝大多数都是由于疏忽造成的；2）数据处理（例如对变量赋值）比数值计算更容易出现逻辑错误，过小和过大的程序模块都比中等程序模块更容易出现错误；3）漏洞和具体的系统环境密切相关。在不同种类的软、硬件设备中，同种设备的不同版本之间，由不同设备构成的不同系统之间，以及同种系统在不同的设置条件下，都会存在各自不同的安全漏洞问题；4）漏洞问题与时间紧密相关。随着时间的推移，旧的漏洞会不断得到修补或纠正，新的漏洞会不断出现，因而漏洞问题会长期存在[3]。

收稿时间：2011-12-12

作者简介：宋超臣（1979-），男，黑龙江，工程师，硕士研究生，主要研究方向：信息安全、Web 服务；黄俊强（1974-），男，黑龙江，信息安全测评中心主任，高级工程师，主要研究方向：网络与信息安全、风险评估与等级保护测评；王大萌（1978-），男，黑龙江，工程师，硕士研究生，主要研究方向：信息安全、Web 服务；段志鸣（1974-），男，辽宁，工程师，硕士研究生，主要研究方向：网络安全。

2 漏洞研究技术分类

根据研究对象的不同，漏洞挖掘技术可以分为基于源代码的漏洞挖掘技术、基于目标代码的漏洞挖掘技术和混合漏洞挖掘技术三大类。基于源代码的漏洞挖掘又称为静态检测，是通过对源代码的分析，找到软件中存在的漏洞。基于目标代码的漏洞挖掘又称为动态检测，首先将要分析的目标程序进行反汇编，得到汇编代码；然后对汇编代码进行分析，来判断是否存在漏洞。混合漏洞挖掘技术是结合静态检测和动态检测的优点，对目标程序进行漏洞挖掘。

2.1 静态检测技术

静态检测技术可以通过手工或源代码分析工具辅助完成，主要针对源代码结构、跳转条件、边界条件、调用函数等进行分析，查找目标代码中的不安全因素。例如，可以检查源代码中的 Printf 之类函数，检查是否对其使用条件进行了限定，是否对边界条件进行了检查。由于现代软件源代码数量庞大，不可能完全对其进行人工审计。静态分析技术具有以下特点[4]：首先，通过工具对源代码进行扫描，静态检测技术效率高，分析速度快；其次，静态检测技术可以通过设置不同的测试条件实现对代码的全面扫描；再次，静态检测技术可以在一些开源项目或在项目开发阶段进行挖掘，及时修复系统中存在的漏洞。但这种方法必须获得源代码，并且需对目标代码进行分析、编译等。另一方面静态检测技术是通过对源代码依据一定规则分析来实现，因此要建立源代码的特征库和规则库。随着漏洞数量的增加，特征库也随之不断扩大，继而带来检测效率不断降低。由于静态检测技术是依据特征库来进行分析判断，因此存在误报和漏报的情况。

文献 [5] 中，将静态检测技术分为词法分析、规则检查和类型推导。词法分析技术最早出现于 BMAT 技术中，这种方法只对语法进行检查，判断词法中是否存在漏洞，如果存在则根据知识库进行进一步判断。规则检查通过检查程序编制规则来判断程序是否存在漏洞，如 C 语言中是否对变量进行了初始化。规则检查将这些规则以特定语法描述，然后再将程序行为进行比对检测。类型推导通过推导程序中变量和函数类型，来判断变量和函数的访问是否符合类型规则。基于类型推导的静态分析方法适用于控制流无关分析，但对于控制流相关的特性则需要引入类型限定词和子类型[6]的概念来扩展源语言的类型系统，使得新类型系统在源语言的数据类型上加以扩展并表示出类型之间的关系。

针对现有漏洞静态检测方法中存在的误报率和漏报率较高问题，文献 [7] 提出了一种基于数据安全状态跟踪和检查的安全漏洞静态检测方法。该方法扩展了漏洞状态模型的状态空间，设定多个安全属性，通过安全属性描述安全状态。同时，对漏洞状态机中进行合法性校验，识别误报的可能性。通过建立非可信数据识别系统中的漏报情况。

2.2 动态检测技术

由于在实际检测过程中，除开源软件以外，很难获得被测系统的源代码信息，因此限制了静态检测技术的应用。动态检测技术是通过构造非标准输入数据，调试运行软件系统，根据系统功能或数据流向，检查运行结果的异常，以判断被测软件系统是否存在漏洞。动态检测技术通常以输入接口或者运行环境为入手点，检测系统中存在的漏洞。虽然动态检测技术具有准确率高的优点，但是其效率却十分低下，因为各种软件系统本身的功能和流程有所不同，所以不能像静态检测技术那样进行统一的扫描，而应该针对软件系统的功能进行动态检测，这样就势必造成了效率的降低[4]。并且动态检测只能确定漏洞可能存在的范围，需进行进一步的跟踪和分析，通过经验才能确定安全漏洞的类型和利用，因此难以应用到大型软件当中。

文献 [8] 提出了一种动态与静态技术相结合的二进制漏洞挖掘方法。该方法通过对二进制文件反编译，得到伪源代码，然后设计了虚拟执行环境 VM。然后，通过 VM，考察指令执行状态，跟踪寄存器的变化，解决了指针别名的问题。通过记录 VM 中虚拟内存每条指令访存地址，最后统计计算出每条访存指令实际访问的变量地址，解决了变量精确识别的问题。文献 [9] 提出利用全系统模拟器作为动态执行环境，通过追踪输入数据处理路径检测溢出漏洞。该方面面向可执行代码，通过构建全系统模拟器来进行漏洞检测，将可执行代码动态转换成形式统一、便于分析追踪的元指令表示形式。为解决动态检测代码覆盖率的问题，该方法制定了溢出漏洞的判定规则，归纳溢出形成条件，并结合系统状态回退完成多路径漏洞搜索，提高了检测的覆盖率。

2.3 混合检测技术

由于静态检测技术需要目标程序源代码，并具有检测规模大、误报率高的缺陷，而动态检测技术又存在覆盖率低、效率低的缺陷。因此，近几年，混合检测技术发展迅速，它有效的结合了静态检测和动态检测的优点，避免了静态检测和动态检测的缺点，有效地提高了检测效率和准确率。混合检测技术表现为与动态检测技术相似的形式，然而测试者根据程序的先验知识，在测试过程中有针对性的设计测试用例。这种测试可以直接针对数据流中感兴趣的边界情况进行测试，从而比动态检测更高效。

目前，混合检测技术主要通过自动化的漏洞挖掘器即 Fuzzing 检测技术实现。漏洞挖掘器首先分析目标软件的运行环境、功能和接口等，构造畸形数据，生成测试用例。然后，通过接口传递测试用例，运行程序。最后，使用监控程序监视

程序运行，如果程序运行出现异常则记录程序运行环境和输入数据进一步对异常信息进行分析。不同漏洞挖掘器由于挖掘对象的不同，其结构、挖掘方法都有很大的不同。目前主要有文件类型漏洞挖掘器、FTP 漏洞挖掘器、Web 漏洞挖掘器、操作系统漏洞挖掘器等。根据挖掘器构造测试用例方式的不同，Fuzzing 技术可以分为两类 [4]，Dumb Fuzzing 和 Intelligent Fuzzing。Dumb Fuzzing 检测技术类似于黑盒测试完全根据随机的输入去发现问题。这种方法实现简单，容易快速的触发漏洞的错误位置。由于没有针对性，因此其效率低下。Intelligent Fuzzing 检测技术通过研究目标软件的协议、输入、文件格式等方面内容，有针对性的构造测试用例，能够提高自动化检测的效率，因此这种方法能够更加有效的进行软件安全漏洞挖掘。

3 结束语

因为不同漏洞研究技术针对的研究目标不同，所以具有不同的优缺点。静态检测技术具有分析速度快，覆盖全面的特点，但其误报率却很高；而动态检测技术准确率高，但其不能覆盖程序的各个流程，具有较大的随机性；混合检测技术避免了上述两种方法的缺点，但由于目前检测技术的局限性，仍有很多工作需要完成：

1）后门及访问控制缺陷检测困难。由于后门及访问控制缺陷不会引起程序异常，并在逻辑上很难区分，所以常规漏洞检测方法难以发现后门及逻辑上的缺陷；

2）检测过程难以全部自动化。目前，Fuzzing 技术需要根据经验创建测试用例，并且自动化的检测结果需要根据经验进行进一步分析，以确定漏洞是否存在以及如何利用。（责编 程斌）

参考文献：

[1] 徐欣民. 一种缓冲区溢出漏洞自动挖掘及漏洞定位技术 [D]. 湖北：华中科技大学 ,2008.

[2] 胡文涛. 基于 Fuzzing 的网络服务型软件的漏洞挖掘研究 [D]. 上海：上海交通大学，2008.

[3] 单国栋，戴英侠，王航. 计算机漏洞分类研究 [J]. 计算机工程，2002，(10)：3-6.

[4] 张晛譞. 基于 windows 平台的软件安全漏洞发掘技术研究 [D]. 四川：电子科技大学，2007.

[5] 张林，曾庆凯. 软件安全漏洞的静态检测技术 [J]. 计算机工程，2008，(12)：157-159.

[6] Foster J S, Fghndrich M, Aiken A. A Theory of Type Qualifiers[J]. ACM SIGPLAN Notices, 1999, 34(05): 192-203.

[7] 梁彬，侯看看，石文昌，梁朝晖. 一种基于安全状态跟踪检查的漏洞静态检测方法研究与实施 [J]. 计算机学报，2009，(05)：1-14.

[8] 夏超，邱卫东. 二进制环境下的缓冲区溢出漏洞动态检测 [J]. 计算机工程，2008，34(22)：94.

[9] 马俊. 基于模拟器的缓冲区溢出漏洞动态检测技术研究 [D]. 湖南：国防科学技术大学，2008.

上接第 73 页

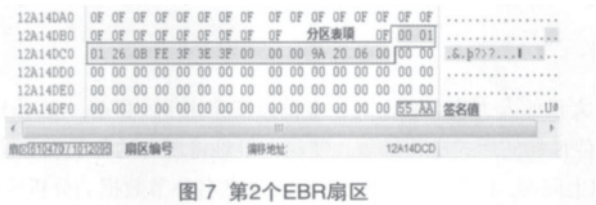

图 7 第2个EBR扇区

总扇区数为 1012095，扇区最大编号为 1012094，因此后面已经没有分区，最后的分区结构如图 11 所示。

3.5 重建主分区表

根据前面 4 步的分析，得到 2 个主分区和扩展分区的分区信息，形成如表 4 所示的主分区表，将 3 个分区表项填入 MBR 的分区表区域，并在 MBR 的最后 2 个字节填入签名值“55AA”，即完成主分区表的恢复。

表 4 MBR的分区表项

分区表项序号	分区类型(16 进制)	起始扇区(16 进制)	总扇区数(16 进制)
1	FAT32（0B）	63(3F)	112392(1B70B)
2	NTFS（07）	112455(1B747)	208845(32FCD)
3	扩展分区（05）	321300(4E714)	690795(A8A6B)

在本次实验中，各个分区是连续的，因此只要找到第一个 DBR 或 EBR 扇区，便可以依次计算出各个分区的起始扇区和总扇区数等信息。若硬盘存在未分区的空闲空间、各个分区存在不连续的情况，那么需要搜索每一个扇区的最后两个字节，找到“55AA”签名的扇区，因为具有“55AA”签名的扇区，除了 DBR 扇区和 EBR 扇区外，FSNIFO 扇区、DBR 备份扇区等也有“55AA”签名，需要进行分析和鉴别以确定是否是 DBR 或 EBR 扇区。

4 结束语

本文介绍了一种在《电子取证》课程中使用的手工恢复主分区表信息的方法，利用 DBR 扇区和 EBR 扇区中存储的有关分区情况的冗余信息，通过分析提取出分区的起始扇区、总扇区数和分区类型等信息，生成分区表项，从而重建主分表。对分区连续的硬盘的主分区表的恢复实验，成功重建了主分区表，验证该方法是可行的。下一步的研究重点将针对分区不连续、主分区表损坏再进行分区的情况下，通过寻找原分区的真实 DBR 扇区和 EBR 扇区，来重建主分区表。

参考文献：

[1] 戴士剑，涂彦辉 . 数据恢复技术（第 2 版）[M]. 北京：电子工业出版社，2005.3.

[2] 马林 . 数据重现：文件系统原理精解与数据恢复最佳实践 [M]. 北京：清华大学出版社，2009.4.

[3] NTFS 文件系统 [EB/OL]. http://www.ntfs.com/mbr-damaged.htm，2011-12-01.