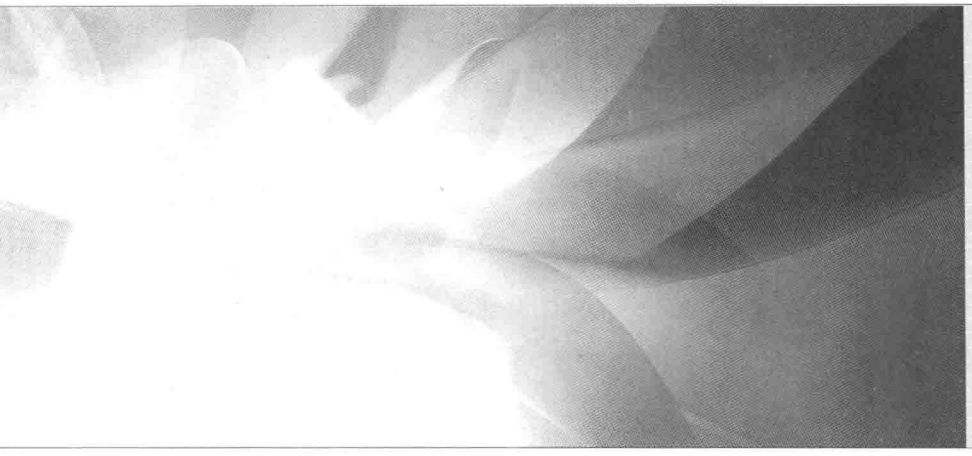
全国网络安全与执法专业丛书

弱点挖掘

徐云峰　史记　徐铎　编著

WUHAN UNIVERSITY PRESS
武汉大学出版社

全国网络安全与执法专业丛书



弱点挖掘

徐云峰　史记　徐铎　编著

WUHAN UNIVERSITY PRESS
武汉大学出版社

图书在版编目(CIP)数据

弱点挖掘/徐云峰,史记,徐铎编著. —武汉:武汉大学出版社,2014.1
全国网络安全与执法专业丛书
ISBN 978-7-307-12157-7

Ⅰ.弱… Ⅱ.①徐… ②史… ③徐… Ⅲ.信息系统—安全技术
Ⅳ.TP309

中国版本图书馆 CIP 数据核字(2013)第 275643 号

责任编辑:辛 凯 责任校对:鄢春梅 版式设计:马 佳

出版发行:**武汉大学出版社** (430072 武昌 珞珈山)
(电子邮件:cbs22@whu.edu.cn 网址:www.wdp.whu.edu.cn)
印刷:武汉理工大印刷厂
开本:787×1092 1/16 印张:19.25 字数:465 字 插页:1
版次:2014 年 1 月第 1 版 2014 年 1 月第 1 次印刷
ISBN 978-7-307-12157-7 定价:39.00 元

前　言

“知己知彼，百战不殆。”人类的生存离不开信息，社会和经济的发展对信息资源、信息技术、信息产业的依赖程度越来越大。当今世界，高新技术的竞争已经成为综合国力竞争胜负的关键。计算机网络的飞速发展推动着各行各业的变革，世界各国都在充分利用信息技术特别是计算机技术和网络系统所带来的巨大利益。随着全球信息共享速度的加快，世界各地发生危及网络系统安全事件的增多，信息安全问题日益突出并且受到越来越多人的关注，信息网络安全已成为信息社会化、社会信息化进程中事关国家安全的核心战略问题之一。目前，世界上已有不少国家制订了计算机间谍计划，因而，很多国家都在抓紧研究信息安全保密技术，组建信息安全保密工作机构，制定相关法规，提出信息安全保密方案和实施措施，以保护国家秘密信息的安全。为迎接高技术条件下窃密与反窃密斗争形势的挑战，应抓紧制定切实可行的对策，尽快建立起我们的信息安全防范体系，以保障我国社会主义现代化建设的顺利发展，保证我国在今后的国防竞争中立于不败之地。

2013 年 6 月，一名叫爱德华·斯诺登的美国互联网工程师，因为泄露美国监控互联网信息的“棱镜计划”而撩动了全球民众的神经。在这个信息时代，全球信息共享让每个人都深陷其中。网络信息入侵已经暴露在众目睽睽之下，信息安全再度成为全球关注焦点。信息主权的掌控、信息资源的争夺等问题，已成为当今国家安全领域面临的重大挑战，世界各国都打响了信息安全保卫战。世界上已有 50 多个国家和地区颁布了保护信息安全的法律，它们在维护信息安全、净化网络环境、保护信息消费者合法权益等方面重拳出击，经验值得借鉴。在全球视野中思考中国的信息安全建设，势在必行。

信息安全涉及的范围很广，大到国家机密安全，小到个人信息的泄露，我国目前信息安全危机具体体现在很多方面，从互联网相关设施、金融、电信等许多核心设备和存储、交换设备，到路由器、操作系统、服务器等主要产品大多存在安全漏洞、后门或隐蔽通道。加之互联网的开放性，给国家的信息安全带来了严重威胁。随着中国上网用户的激增，上网信息的多元化，不可避免地带来了泄密隐患，增加了国内信息网络被“入侵”、被窃密的机会和可能，因而必须提高信息安全认识。“棱镜”计划的曝光无疑是对我国信息安全技术发展的鞭策。我们要从国家安全角度重视信息安全，保证国家通信和计算机基础的安全，从国家和军队的最高利益出发，把信息安全工作作为一项重要工作来抓。

本书分为十章。第一章介绍弱点挖掘概论，主要使读者从整体上了解弱点挖掘；第二章介绍程序弱点挖掘，包括缓冲区溢出，异常信息处理机制等理论；第三章主要介绍操作系统弱点挖掘，其中包含了弱口令、存储访问控制、命令注入等；第四章介绍数据库弱点挖掘，主要涉及 SQL 注入、数据库提权、拒绝服务等内容；第五章主要介绍网站弱点挖掘；第六章简要介绍常见弱点数据库的特点以及设计原则；第七章介绍网络协议中可能会成为弱点的部分；第八、九章分别介绍被动分析以及高级逆向工程；第十章介绍漏洞检测

以及入侵监测的基本知识。最后在附录里面对相关标准进行了介绍。

本书由徐云峰、史记和徐铎主编，张士豪、张俊豪、郑义在本书编写过程中也给予了莫大的支持，一并感谢。

由于作者水平有限，书中可能会有不少谬误之处，敬请读者批评指正。

作 者

2013 年 9 月

http：//www. nirg. org

xu. lotus8340@ gmail. com

目　录

第1章 绪 论

弱点挖掘作为信息系统安全战略的一个重要组成部分，是网络攻防技术的重要实践。目前，互联网上存在的各类漏洞、弱点，加之以因特网的开放性，给国家的信息安全带来了严重威胁。在计算机领域当中，我们可能更加频繁地接触到“漏洞”，然而，“弱点”和“漏洞”是相同的概念吗？存在哪些区别？计算机弱点是如何分类的？目前有何种漏洞分析技术？这些都是一名合格信息安全工作者必须掌握的基本概念。本章简单地介绍了弱点挖掘概论，帮助读者从整体上了解弱点挖掘技术，以便后续深入地学习。

1.1 弱点的定义

弱点和漏洞在计算机取证方向是同义词，在很多文献中都表示同一个意思，但是各自也有不同的含义，在这里我们一一介绍：

了解弱点之前应该先了解一下漏洞：

漏洞在计算机中的含义就是某一种特定的缺陷，这种缺陷一般在硬件、软件、协议中都会存在。它是黑客可以利用的，是可以通过漏洞在未经授权的情况下对计算机进行非法访问或者对计算机进行破坏的。比如，在 Intel Pentium 芯片中，往往会存在逻辑错误，在计算机发展的初期，Sendmail 早期版本中就会出现编程错误，在 NFS 协议中，会存在认证方式上的漏洞，在 Unix 系统管理员设置匿名 Ftp 服务时配置不当的问题都可能被攻击者使用，威胁到系统的安全。因而，这些都可以认为是系统中存在的安全漏洞。只要这种缺陷一旦被发现，不法分子就可以通过这种缺陷对我们的系统发起攻击达到其不法目的。打个比方来说，“漏洞”的存在就像我们的房子没有关好门窗，久而久之总会有不法之徒潜入谋取不义之财。

总之，安全漏洞是指受限制的计算机、组件、应用程序或其他联机资源的无意中留下的不受保护的入口点。漏洞是硬件软件或使用策略上的缺陷，他们会使计算机遭受病毒和黑客攻击。

如果我们把计算机系统比作成一个建筑物的话，那么在这个建筑物中，操作系统(OS)是基础结构，为系统提供支持，为建筑物提供支撑作用。应用程序则是建筑物中的房间，或者是房间中的各种设施，都是在建筑搭建好的基础架构上实现的，系统用户是建筑物里的住户。门和窗是建筑物里各个房间之间的交互通道，它们是早就设定好的，而漏洞则是一些本不该存在的门、窗户，或者是墙上莫名出现的一个洞，更有可能是一些会破坏建筑物的危险材料或物品——这些缺陷和问题会使陌生人侵入建筑物，或者使建筑物遭遇安全威胁。这就是漏洞，对于系统来说，若出于安全的考虑，则必须最大限度地减少漏洞的存在，因为它会成为入侵者侵入系统和恶意软件植入的入口，影响我们系统用户的切

身利益。

了解完漏洞之后我们可以对比一下弱点的含义：

计算机弱点和漏洞在计算机安全方面其实就是一个意思，都是指计算机硬件、软件或策略上的缺陷，从而使攻击者可能在未授权的情况下访问系统。但是弱点涉及的范围很广，它不仅覆盖计算机系统，而且还将包括网络系统的各个环节，比如，路由器、防火墙、操作系统、客户和服务器软件等。对于计算机弱点的研究，已经历经 30 多年，在这一个过程中学者们根据自身的不同理解和应用需求对计算机弱点下了很多定义，但目前看来还没有一个统一的定义能够被人们广泛接受。其中，学者 KrsuI 曾经针对软件弱点作过一个比较有影响的定义，他将一个弱点定义为“软件规范(Specification)设计、开发或配置中一个错误的实例，它的执行能违犯安全策略”。后来，人们在修改和完善 KrsuI 定义的基础上对弱点进行了如下的定义：弱点(VuInerabiIity)是软件系统或软件组件中存在的缺陷，此缺陷若被发掘利用则会违犯安全策略，并对系统的机密性、真实性和可用性造成不良影响。

通过以上对漏洞和弱点的比较，我们可以发现漏洞和弱点都是攻击者可以利用的“工具”进而对计算机实施攻击。所以为了避免称呼上的混淆，本章将安全漏洞、脆弱性、脆弱点、安全弱点统一称作弱点，即上述名词表示同一概念。

1.2 弱点分类

对于弱点，如果能够彻底的了解其本质特征，那么就会对弱点信息能够进行很好的收集、存储和组织，以便于我们对弱点更加深入的了解和研究。通常，弱点的一个本质特征往往会通过一个分类属性进行表示，而对这个分类属性进行赋值的过程，就是给弱点在该维属性上分类的过程。对弱点的研究过程中我们常利用的弱点数据库，其在设计与实现中，人们常常用弱点的分类属性作为数据库表字段以便表达弱点的各方面性质。因此，对弱点进行分类将会是建构弱点数据库的基础，也同样是分析和评价弱点数据库的依据。由于弱点分类的意义十分重大，目前，已经有很多研究机构和研究人员都开展了弱点分类工作，并且也已经取得了很大的成效。就近十年来，国外比较有成效的工作有：

(1) Landwehr 提出了一个三维属性分类法：弱点起源、引入时间和位置的；

(2) Longstaff 进而又增加了对访问权限和易攻击性的分类；

(3) Power 根据危害性进行分类；

(4) Du 和 Mathur 的分类法描述了弱点起因、影响和修复属性；

(5) AsIam 和 KrsuI 根据 Unix 进行对弱点分类；

(6) Bishop 的六轴分类法；

(7) Knight 的四类型分类法和 Venter 的协调(harmonized)分类法。

在国内，比较成熟的分类法有李昀的基于星型网模型的分类法，单国栋的弱点分类研究，以及多维量化属性分类法等。

通过对上述分类法的分析，可以看出每种分类法的相同之处就是给出了弱点的不同分类属性。我们对这些分类属性进行了整理，并将它们简单地分成所属相关、攻击相关、因果相关、时间相关和其他属性等五类，整理结果如表 1.1 所示。

表 1.1　弱点的分类属性

属性类别	所属属性	攻击相关	因果相关	事件相关	其他属性
属性名称	生成厂商	易攻击性	起源成因	引入时间	弱点标志号
	OS 种类	攻击复杂性	后果影响	时间影响力	弱点名称
	应用程序	攻击来源	安全性威胁	发布时间	修复操作
	组件种类	攻击手段		更新时间	对象和影响速度
	出现位置	攻击所需权限			
	消息来源				

1.3　弱点挖掘分类

在通常情况下，我们知道若利用漏洞进行攻击的话，一般情况下可分为三个步骤：漏洞挖掘、漏洞分析和漏洞利用。其中，漏洞挖掘是其他两个步骤的前提和基础，所以漏洞挖掘对于网络的攻防具有重要的意义。漏洞挖掘，顾名思义就是寻找漏洞，主要是通过综合应用各种技术和工具，尽可能地找出软件中的潜在漏洞，然而这在很大程度将会依赖于个人经验，并不是一件十分容易的事。根据分析对象的不同，漏洞挖掘技术可以分为基于源码的漏洞挖掘技术和基于目标代码的漏洞挖掘技术。

目前，随着 Internet 和网页应用的普及，Web 漏洞挖掘技术也浮出水面。Web 漏洞主要出现在动态 Web 页面程序中，对于动态的 Web 页面，我们一般是无法获取到源码的，或者只能获取部分源码的，因此 Web 漏洞挖掘技术可以归类为基于目标代码的漏洞挖掘技术。基于源码的漏洞挖掘方法其实不难理解，主要就是将源码获取后，使用自动工具或手工检查的方法进行源代码分析，从而找到软件漏洞的技术。这类漏洞挖掘在软件生产中都会是重要的测试过程的一环，因为这可以提高软件发布后的安全性。

基于目标代码的漏洞挖掘技术与软件测试技术相近，分为白盒分析、黑盒分析和灰盒分析三种。

白盒分析主要采用逆向工程的方法将目标程序转换为二进制码或还原部分源代码。但是白盒分析有自身的缺陷：在一般情况下，程序员很难将目标程序完全转换为可读的源代码，尤其是当原作者采用了扰乱、加密措施后。所以这时候采用白盒分析就不会十分有效。

黑盒分析不同于白盒分析，它不对目标程序本身进行逆向工程，而是控制程序的输入，观察输出的一种方法。它相对白盒测试具有自身的优点：可以对某些上下文关联密切、有意义的代码进行汇聚，降低其复杂性，最后通过分析功能模块，来判断是否存在漏洞。但黑盒分析的过程需要分析者具有较高技术水平，否则很难在较短时间内找到可利用的漏洞。

灰盒分析则是将两种分析技术结合起来的方法，从而能够提高分析命中率和分析质量。

1.4 漏洞分析技术

随着计算机技术的发展，成熟的漏洞挖掘分析技术已经有多种，在实际应用中如果只运用单一的漏洞挖掘技术，是很难完成分析工作的，所以我们一般会将几种漏洞挖掘技术优化组合，去寻求效率和质量的均衡。下面我们列举几种常用的，熟悉的漏洞分析技术。

1.4.1 人工分析

人工分析是最基础的也是最早发展的一种漏洞分析技术，有时将其称为灰盒分析技术。它的工作原理大致如下：首先获得被分析的目标程序，然后可以手工的构造特殊输入条件，进而观察输出、目标状态等一系列的变化，最后获得漏洞的分析技术。其中输入包括有效的输入、无效的输入；输出包括正常的输出、非正常输出。如果发现有非正常输出，那么这就是存在漏洞的前兆，也可以说目标程序有存在漏洞的可能。另外若有非正常目标状态的变化，这也同样是发现漏洞的预兆，是研究者深入挖掘的方向。人工分析高度对于分析人员的经验和技巧要求较高，所以人工分析多用于有人机交互界面的目标程序，比如 Web 漏洞挖掘中多使用人工分析的方法。

1.4.2 Fuzzing 技术

Fuzzing 技术是一项比较成熟的技术，它是基于缺陷注入的自动软件测试技术，利用的主要方法就是黑盒测试，其工作原理如下：运用大量的半有效的数据作为应用程序的输入，以程序是否出现异常为标志，进而探索应用程序中可能存在的安全漏洞。其中，上面所说的半有效数据就是指被测目标程序所需的必要标志部分和大部分数据是有效的，另外有意构造的数据部分是无效的，这样应用程序在处理该数据时就有可能发生错误，可能导致应用程序的崩溃或者触发相应的安全漏洞。

根据分析目标的特点，Fuzzing 可以分为三类：

(1)动态 Web 页面 Fuzzing，针对 ASP、PHP、Java、Perl 等编写的网页程序，也包括使用这类技术构建的 B/S 架构应用程序，典型应用软件为 HTTP Fuzz；

(2)文件格式 Fuzzing，针对各种文档格式，典型应用软件为 PDF Fuzz；

(3)协议 Fuzzing，针对网络协议，典型应用软件为针对微软 RPC(远程过程调用)的 Fuzz。

早期的 Fuzzing 技术仅是一种简单的随机测试技术，但却有效地发现了许多程序中的错误。但是早期的程序中可能出现的错误也会比较简单，如代码中因没有对输入的字符串的长度进行检查，这样就会导致栈溢出。

由于早期的 Fuzzing 技术还不是很成熟，所以操作起来简单，而且其测试目标程序的关联性还不是很大，它的主要优点如下：

(1)可用性，不需要获得目标程序的源代码就可以测试；

(2)复用性，如测试 FTP(File Transfer Protocol)的 Fuzzing 程序可以用来测试任何 FTP 服务器；

(3)简单性，对于目标程序不必要过多的了解。但是，Fuzzing 技术会不可避免地带有

随机测试而产生大量冗余测试输入、覆盖率低导致发现软件缺陷概率低的缺点，同时，带有黑盒测试的低智能性的缺点，即黑盒测试只测试了程序的初始状态，而很多程序尤其是网络协议程序的很多错误是隐藏在程序的后序状态中的。

Fuzzing 技术已经经过了近 20 年的发展，我们知道 Fuzzing 技术是源于黑盒测试技术的，但是 Fuzzing 技术已经比黑盒测试更加成熟，主要表现在：①对于测试需求，着眼点已经不同。Fuzzing 的测试侧重于发现与软件安全性相关的错误，而黑盒测试却侧重于测试软件的功能的正确性。②Fuzzing 技术的测试用例已有很大的不同。由于测试需求的不同，Fuzzing 的测试用例大多数都是畸形的测试用例，而黑盒测试的用例刚好相反，大多数都是正确的测试用例。Fuzzing 为了能够提高测试用例的有效性，就必须提高测试用例的正确性，所以使用测试用例的畸形数据能够提炼出程序的潜在不安全点。③Fuzzing 和黑盒测试的测试用例的产生机理是不同的，Fuzzing 需要利用到有效的畸形数据，那么就必须要考虑到测试用例的数据格式、目标程序的结构流程和程序运行的中间状态；而黑盒测试只关心目标程序的外部接口和外部输入，如果就简单地从这个意义上讲，则现在的 Fuzzing 技术更接近于灰盒测试。

随着 Fuzzing 技术越来越成熟，有效性也得到验证之后，市场上出现了很多针对特定类型应用程序或者协议的 Fuzzing 工具，如针对浏览器的 mangleme、针对文件应用程序的 fileFuzzing 等，其中较为突出的是 2002 年出现的 SPIKE，主要就是针对网络协议的。后来，又出现了针对性更强、功能也更为单一的 Fuzzing 工具，如针对 IRC 协议的 ircfuzz，针对 DHCP 协议的 dhcpFuzz 等。当前的 Fuzz 工具主要是针对文件格式应用程序和网络协议应用程序，但也出现了可以测试浏览器、操作系统内核、Web 应用程序的 Fuzz 工具。

如图 1.1 所示，这是一个最为简单的 Fuzzing 架构，主要包含引擎和监视两个模块。

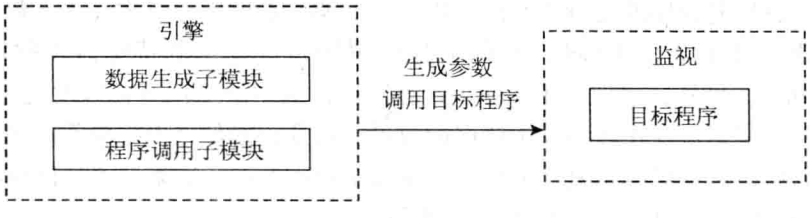


图 1.1　简单的 Fuzzing 架构

可以看出引擎模块包括两大功能：一是产生 Fuzzing 需要的数据；二是把数据发送到目标程序使之运行；监视模块的功能是监视目标程序的运行状态是否出错。这就是早期 Fuzzing。其中，监视模块功能的实现借助于简单的脚本来记录程序出错的信息。

Fuzzing 技术在得到的进一步的发展之后，又出现了如图 1.2 所示的架构设计。我们知道早期的 Fuzzing 会产生大量无效的测试数据，所以为了优化 Fuzzing，新架构中又增添了参数脚本和样本文件。其中，参数脚本给出了引擎生成的测试用例中的数据的格式、长度等与数据之间的一些关系，如 SPIKE、Sulley 使用的类 C 格式的脚本，Peach 使用的 XML 格式的脚本。样本文件是许多基于变异技术的数据生成方式的 Fuzzing 工具用来变异测试数据的基准。这样通过样本文件产生的测试数据有一个很大的特点就是可以大大提高

测试用例的有效性，可以提高测试的代码覆盖率，可以减轻测试用例构造的复杂度。Fuzzing 测试文件格式应用程序依据的样本文件是其相应格式(如 DOC、PDF 等)的样本文件(如 Peach)，Fuzzing 测试网络协议应用程序的样本文件是通过嗅探工具(如 Ethereal)的数据包转换的样本文件。

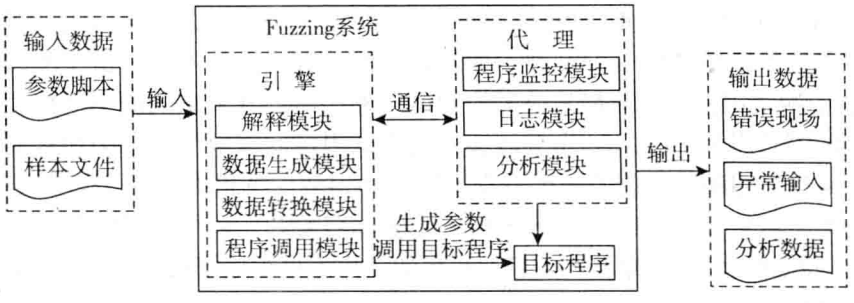


图 1.2 当前的 Fuzzing 架构

在和早期的 Fuzzing 框架对比中，我们可以发现在图 1.2 所示的架构中，Fuzzing 引擎的模块比之前划分粒度更为细化，分为了四个部分，其目的是为了加强代码的可复用性和整个 Fuzzing 构架的灵活性，这样用户就可以根据需求进而快速的制定适合其他多种协议的 Fuzzing 程序。Fuzzing 的监视模块较之以前也发生了很大的变化，内容转换为功能更为丰富的代理模块(如工具 Peach)，一方面，可以并行 Fuzzing 的过程以提高 Fuzzing 的效率；另一方面，这可以把引擎和代理分离开来，使各自在不同的机器上运行，这就可以对分布式应用程序进行 Fuzzing 测试。从图 1.2 可以发现代理都包括了程序监控子模块，这是用来监视并且控制程序的运行情况的。由于开发调试器的工作量大，大多数工具使用的是第三方的调试工具，如 Peach，有的工具使用的是可以定制特殊需求的调试器，如 Sulley、AutoDafe。另外，代理中还添加了日志模块，这主要是用来记录发生异常的现场，以协助用户进一步定位应用程序发生错误的位置，还要记录使之发生异常的测试用例或者程序输入，以便该错误可以恢复。最后，极少数工具如 Sulley 还添加了分析模块，以用来统计 Fuzzing 测试的结果信息，如代码覆盖率等。

早期的 Fuzzing 技术就是随机测试技术，测试数据多数是随机产生的畸形数据。随着 Fuzzing 技术的发展，人们对 Fuzzing 提出了更高的要求，都想能够提高产生数据的有效性。所以逐渐形成了下面两种产生测试数据的思想：

(1)将格式分析和程序理解进行结合进而产生数据的方法，相应的代表工具有很多，如 SPIKE、Peach、Sulley、AutoDafe 等。这种工具可以通过对文件和协议的理解，产生的数据可以有效地越过应用程序中对固定字段、校验和、长度的检查，这样就能够大大提高 Fuzzing 的测试数据的有效性。另外，该方法又可以分为基于生成技术的方法和基于变异技术的方法或两者相结合(如工具 Peach)的方法。基于生成技术的测试数据产生方法通常是给出文件格式或者网络协议具体的描述规则，然后依据此规则产生测试数据。该方法有自己的局限性。就是用户对格式或者协议要有非常深的了解，并且需要大量的人工参与。基于变异技术的数据产生方式一般是在对格式或者协议有所了解的前提下，然后对获得的

样本数据中的某些域进行稍微的变化，从而产生新的变异数据。这种方法很明显就是对初始值有着很强的依赖性，不同的初始值最终会带来差异很大的代码覆盖率，从而会产生差异很大的 Fuzzing 效果。还有一种工具如 AutoDafe 和 SPIKEProxy，这是完全自动化数据产生方式，核心技术是变异技术。该方法就是利用协议的自动分析技术然后实施对测试数据的自动生成。但由于协议自动分析技术不够成熟，其准确率还有待进一步提高，所以 Fuzzing 测试效果并不理想。

总之，基于格式分析和程序理解相结合的数据产生方法的优点是执行效率比较高、应用范围广、通用性强，缺点是仍然需要大量的人工参与来进行多种知识(如协议知识、数据格式知识、应用程序知识)的获取并实现这些知识到测试用例的转换。

(2)将静态分析与动态测试相结合来产生数据的方法。这种方法主要就是通过与静态分析技术、符号执行技术、具体执行技术等多种技术相结合，从而在达到一个较高的代码覆盖率的测试基础上进行 Fuzzing 测试。本质上来说，这种方法是白盒测试与 Fuzzing 测试技术的融合。该方法之所以能够使 Fuzzing 技术得到一个不错的代码覆盖率，是因为它借助了软件测试中的技术。但是该方法仍有自己的缺点就是仍然无法克服符号执行中的状态爆炸问题，也无法完全自动解决部分程序自带的高强度的程序检查(如校验和和加密数据)问题；另外，该方法每次执行都需要大量的时间，使用效率十分低，主要是因为它采用了类似于穷搜索的思路；而且每次执行都需要复杂的符号运算，从而消耗了大量的时间。

综上所述，与其他技术相比，Fuzzing 技术具有思想简单，容易理解、从发现漏洞到漏洞重现容易、不存在误报的优点。同时，它也存在黑盒分析的全部缺点，而且具有不通用、构造测试周期长等问题。

因为 Fuzzing 技术简单而且有效，得到了越来越多的关注，结合上述技术的不足和实际应用中的需求，在以后的研究方向中可以归纳为以下几个方面：

第一，Fuzzing 测试平台的通用性研究。随着发展，我们知道 Fuzzing 的测试对象开始越来越广泛，这样如何构建通用的、可扩展性强的通用平台这个大问题就摆到了我们的面前，因为这对于提高 Fuzzing 技术的整体发展十分必要。原则上通用的测试平台应该具备下面几个功能：①具备数据格式的解释功能，这样才能够产生适合多种数据格式的有效的畸形测试数据；②具备独立的、可定制的数据产生变异功能，以便可以产生多种类型的、针对性强的畸形数据；③具备可操作的跟踪调试功能以反馈运行时的多种信息；④具备高效的引擎，能够协调多个模块之间的自动化运行。

第二，逐步的提高知识获取的自动化程度。在实际的 Fuzzing 测试过程中，会发现绝大部分的时间都花费在输入数据格式、程序状态转换的人工分析上；所以若能够提高测试数据或通信协议的自动化分析或半自动化分析水平，那么就可以有效的提高 Fuzzing 的测试效率。

第三，向多维的 Fuzzing 测试用例生成技术方向进行研究。就目前来说，Fuzzing 测试用例的生成技术都是一维的，即每次仅仅变换一个输入元素，然而现实中许多漏洞都是由多个输入元素共同作用引起的。多维测试用例生成技术的研究可以有效扩展 Fuzzing 发现的漏洞范围，但是多维 Fuzzing 测试用例有一个瓶颈，就是会带来类似于组合测试中的状态爆炸问题，现有的组合测试理论成果对于解决 Fuzzing 多维测试中的状态爆炸问题有一