



Vulnerability Management

漏洞管理

(美) Park Foreman 著
吴世忠 郭涛 董国伟 张普含 译



机械工业出版社
China Machine Press




vulnerability Management

 **CRC Press**
Taylor & Francis Group

客服热线: (010) 88378991, 88361066
购书热线: (010) 68326294, 88379649, 68995259
投稿热线: (010) 88379604
读者信箱: hzsj@hzbook.com

华章网站 <http://www.hzbook.com>

 网上购书: www.china-pub.com

上架指导: 计算机/信息安全

ISBN 978-7-111-40137-7



9 787111 401377

定价: 69.00元



vulnerability management

漏洞管理

(美) Park Foreman 著

吴世忠 郭涛 董国伟 张普含 译



机械工业出版社
China Machine Press

本书是资深安全漏洞管理专家、信息安全战略专家兼国际安全顾问 20 余年跨国工作经验的总结，以创新的方法从多个角度全面讲解了漏洞管理的理论、方法与最佳实践！结合大量实际案例深入阐述了安全漏洞防范的战略视野和实施方式，旨在帮助读者从技术、流程和管理角度全面了解漏洞管理，从而掌握评估和减弱内外部漏洞的行之有效的方法。

本书共分 10 章：第 1 章介绍了风险管理、漏洞管理、安全产业现状等；第 2 章讲解漏洞产生过程、漏洞程序的作用，并结合实际案例讲解漏洞管理程序故障问题；第 3 章讲解漏洞管理计划的参与者、漏洞管理策略及合规性；第 4 章侧重于漏洞扫描的总体架构，并涵盖当前流行的漏洞管理技术，以及漏洞测试相关的数据、评价、技术标准和漏洞管理扫描程序 Nessus；第 5 章阐述了如何选择漏洞管理产品，包括总体要求、实施过程的自动化、体系结构、如何进行用户定制与整合、评分和部署方法、访问控制等相关技术；第 6 章讲解漏洞管理流程，包括与漏洞管理相关的 ITIL-ITSM 过程和 IAVA 过程，以及该流程中的数据分级和风险评估等重要步骤；第 7 章介绍了一系列与执行、汇报、分析相关的文档，如发现报告、审计报告、合规性报告等；第 8 章提供了一些建议，引导读者从制定检查表、工程规划和实施策略等方面逐步了解如何在一个大型的公司里开发一个完整的漏洞管理项目；第 9 章从一个更宏观的、策略性的层面来研究漏洞的呈现形式及修复方法；第 10 章对上述内容进行了概括性总结。

Vulnerability Management by Park Foreman (ISBN 978-1-4398-0150-5)

Copyright © 2010 by Taylor and Francis Group, LLC.

Authorized translation from the English language edition published by CRC Press, part of Taylor & Francis Group LLC; All rights reserved; 本书原版由 Taylor & Francis 出版集团旗下 CRC 出版公司出版，并经授权翻译出版。版权所有，侵权必究。

China Machine Press is authorized to publish and distribute exclusively the Chinese (Simplified Characters) language edition. This edition is authorized for sale in the People's Republic of China only (excluding Hong Kong, Macao SAR and Taiwan). No part of this publication may be reproduced or distributed in any form or by any means, or stored in a database or retrieval system, without the prior written permission of the publisher. 本书中文简体字翻译版权由机械工业出版社独家出版并限在中国大陆地区销售。未经出版者书面许可，不得以任何方式复制或抄袭本书的任何内容。

Copies of this book sold without a Taylor & Francis sticker on the cover are unauthorized and illegal. 本书封面贴有 Taylor & Francis 公司防伪标签，无标签者不得销售。

封底无防伪标均为盗版

版权所有，侵权必究

本书法律顾问 北京市展达律师事务所

本书版权登记号：图字：01-2012-6633

图书在版编目（CIP）数据

漏洞管理 / (美) 福尔曼 (Foreman, P.) 著；吴世忠等译. —北京：机械工业出版社，2012.12

书名原文：Vulnerability Management

ISBN 978-7-111-40137-7

I . 漏… II . ① 福… ② 吴… III . 企业管理—风险管理 IV . F270

中国版本图书馆 CIP 数据核字 (2012) 第 248180 号

机械工业出版社 (北京市西城区百万庄大街 22 号 邮政编码 100037)

责任编辑：高婧雅

北京京师印务有限公司印刷

2013 年 1 月第 1 版第 1 次印刷

186mm×240mm·15.75 印张

标准书号：ISBN 978-7-111-40137-7

定价：69.00 元

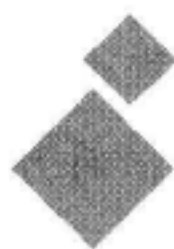
凡购本书，如有缺页、倒页、脱页，由本社发行部调换

客服热线：(010) 88378991；88361066

购书热线：(010) 68326294；88379649；68995259

投稿热线：(010) 88379604

读者信箱：hzjsj@hzbook.com



译者序

随着信息技术的飞速发展，互联网日益成为人们生活中不可缺少的一部分，社交网络、微博、移动互联网、云计算、物联网等各种新技术、新应用层出不穷。但不管是 Facebook、Twitter 等新兴互联网公司的迅速崛起，还是 Android 日益成为智能手机市场的主流操作系统，信息安全一直都是永恒的话题。“震网病毒”和“火焰病毒”事件凸显了网络武器的实战破坏能力，关键信息基础设施保护已成为世界各国网络空间防御的新重点；“维基泄密”事件彰显网络空间攻防双方的不对称性，“百密难免一疏”成为保密防范永远的痛；究其根源，所有这些信息安全事件都存在一个共同点——信息系统或软件自身存在可被利用的漏洞。因而，漏洞分析和风险评估日益成为信息安全领域理论研究和实践工作的焦点，越来越引起世界各国的关注与重视。

为推动国内的漏洞分析和风险评估工作，提高国家信息安全保障能力和防御水平，中国信息安全测评中心长期跟踪和关注相关领域的理论进展和技术进步，有针对性地精选一些优秀书籍译成中文，供国内读者参考借鉴。

本书从基本概念、重要作用、关键技术、流程管理等多个角度深入阐述了漏洞管理的运作及其发挥的作用。基本概念部分以风险管理的作用和漏洞管理的起源为切入点，分析了信息安全产业目前存在的缺陷及挑战，以及漏洞的诸多来源，并通过具体实例展现了失败的漏洞管理带来的巨大损失，进而说明了漏洞管理对于企业的重要

作用；关键技术部分介绍了多种实用的漏洞管理技术，包括主动和被动扫描、漏洞测试数据标准、漏洞严重等级标准、美国国家漏洞库（NVD）等，这些技术可以有效辅助漏洞管理工作的开展，极大提高管理效率；流程管理部分，阐明了以信息技术基础架构库-IT 服务管理（ITIL-ITSM）流程和保障漏洞预警（IAVA）流程为基础的漏洞管理过程，以组织和规范漏洞管理工作，并深入介绍了此过程中形成的各类报告的形式和内容，最终说明了更高层面的策略性漏洞的管理方法。

本书翻译工作还得到了中国信息安全测评中心的章磊、王眉林、贾依真、吴健雄、张翀斌等同志的支持和帮助，在此深表感谢。

本书得到中国信息安全测评中心“漏洞分析与风险评估”专项工程、国家自然科学基金项目（90818021、61100047、61272493）的支持。



前言

漏洞管理 (Vulnerability Management, VM) 已经有了上千年历史, 城市、部落、国家和企业都会触及该学科的知识。漏洞会让潜在的攻击者有机可乘, 任何机构的成功管理和运作都依赖于对漏洞进行检测和修复的能力。以往, 人们修筑城堡, 在城市中建造防御设施和高级预警系统, 这些都是他们意识到自身的脆弱性并为了抵御危害而采取的各种措施。如今, 我们检测到在软件系统、基础设施以及企业战略中也都存在一些漏洞, 这些漏洞通常需要从多个角度、以创新性的方法来解决。

本书是一本信息安全从业人员的指导手册, 读者包括安全工程师、网络工程师、安全部门的官员或首席信息主管 (CIO) 等在内的安全行业从业人员, 系统地介绍了什么是漏洞管理及其在组织机构中的作用。本书涵盖了漏洞管理的各个重点领域以满足不同读者的需求。技术章节从宏观视角介绍了漏洞相关内容, 不打算太纠结于技术细节的决策者也可以读懂这部分内容。其他有关流程和策略的章节, 也能为领导层提供一定的参考, 但主要是从工程师或安全主管的角度介绍了漏洞管理技术及其流程在企业中所起的作用。

作者建议对漏洞管理领域感兴趣的读者阅读相关章节, 并略读其余章节。如果不能以长远的眼光全面理解漏洞管理的各个方面, 将难以有效参与漏洞管理的任何一个环节。通常, 员工们会担心他们在某个过程中承担的工作看起来毫无意义。希望本书介绍的内容能够在一定程度上减轻这种焦虑。

致谢

nCircle 网络安全公司的 Tim Erlin 先生着重从完整性和准确性角度审阅了技术相关章节，他所提的建议见解深刻，对我有很大的帮助。Ben Rothke 先生着重从清晰性角度协助审阅了原稿。



目 录

译者序
前言

第 1 章 绪论 /1

- 1.1 风险管理的作用 /2
- 1.2 漏洞管理的起源 /3
- 1.3 安全产业及其缺陷介绍 /4
- 1.4 来自政府和产业的挑战 /5
- 1.5 漏洞的来源 /5
- 1.6 有缺陷的漏洞管理示例 /5
- 1.7 漏洞管理的重要性 /6

第 2 章 漏洞体验 /7

- 2.1 简介 /8
- 2.2 漏洞产生过程 /8
 - 2.2.1 复杂性 /9
 - 2.2.2 连通性 /10
 - 2.2.3 互操作性 /10
- 2.3 创建漏洞：一个例子 /11
- 2.4 使用漏洞管理程序的理由 /13
 - 2.4.1 网络过度开放 /13
 - 2.4.2 安全系统配置标准缺失 /14
 - 2.4.3 重大经济损失风险 /14
 - 2.4.4 收益损失 /15
 - 2.4.5 生产力损失 /15

2.5 漏洞管理程序故障 /16

- 2.5.1 案例研究 1：获得组织的支持 /16
- 2.5.2 案例研究 2：技术集成的挑战 /22

第 3 章 计划和组织 /33

- 3.1 概述：计划结构 /34
- 3.2 漏洞管理计划和技术开发 /36
- 3.3 参与者 /37
 - 3.3.1 操作者角色 /37
 - 3.3.2 贡献者角色 /39
- 3.4 策略和信息流 /40
 - 3.4.1 现行策略 /40
 - 3.4.2 新策略 /41
 - 3.4.3 合规和统辖 /42
- 3.5 小结 /44

第 4 章 漏洞管理技术 /45

- 4.1 简介 /46
- 4.2 总体架构 /47
 - 4.2.1 硬件模式 /47
 - 4.2.2 用户提供的硬件和虚拟化 /49
- 4.3 代理 /50

- 4.3.1 代理架构 /50
 - 4.3.2 优点与缺点 /52
 - 4.3.3 检测方法 /53
 - 4.4 被动网络分析 /53
 - 4.4.1 优点与缺点 /56
 - 4.4.2 检测方法 /57
 - 4.4.3 物理层 /57
 - 4.4.4 数据链路层 /58
 - 4.4.5 网络层 /58
 - 4.4.6 4 至 7 层 /58
 - 4.5 主动扫描技术 /58
 - 4.5.1 优点与缺点 /59
 - 4.5.2 检测方法 /59
 - 4.6 混合方法 /82
 - 4.7 推理扫描 /83
 - 4.8 CVE/83
 - 4.8.1 结构 /84
 - 4.8.2 CVE 的局限 /86
 - 4.9 漏洞测试数据标准 /86
 - 4.9.1 架构定义 /87
 - 4.9.2 系统特征架构 /88
 - 4.9.3 结果架构 /88
 - 4.9.4 测试描述 /88
 - 4.10 漏洞危害程度评价标准 /92
 - 4.11 美国国家漏洞库 /98
 - 4.11.1 CPE /98
 - 4.11.2 XCCDF/100
 - 4.12 SCAP/101
 - 4.13 Nessus/102
 - 4.13.1 优点与缺点 /103
 - 4.13.2 扫描模型 /103
 - 4.13.3 使用 Nessus/104
- 第 5 章 选择技术 /107**
- 5.1 概述 /108
 - 5.2 总体需求 /108
 - 5.2.1 责任分担 /108
 - 5.2.2 时间表 /110
 - 5.2.3 标准 /112
 - 5.2.4 报告 /113
 - 5.2.5 高级报告 /115
 - 5.3 自动化 /116
 - 5.3.1 标签生成 /116
 - 5.3.2 流程整合 /117
 - 5.3.3 流程和系统的灵活性 /117
 - 5.3.4 补丁管理支持 /118
 - 5.4 体系结构 /118
 - 5.4.1 被动的体系结构 /119
 - 5.4.2 基于代理的体系结构 /119
 - 5.4.3 主动扫描的体系结构 /120
 - 5.4.4 保证平台安全 /124
 - 5.4.5 系统整合 /125
 - 5.5 定制与整合 /126
 - 5.6 评分方法 /127
 - 5.7 访问控制 /129
 - 5.7.1 活动目录 /129
 - 5.7.2 RADIUS 和 TACACS+ /130
 - 5.7.3 授权 /130
 - 5.8 部署方法 /131

- 5.8.1 主动扫描器部署 :
物理部署 /132
- 5.8.2 虚拟扫描器 /133
- 5.8.3 被动分析器的部署 /133
- 5.8.4 代理部署 /134

5.9 小结 /135

第6章 过程 /137

- 6.1 介绍 /138
- 6.2 漏洞管理过程 /138
 - 6.2.1 准备 /139
 - 6.2.2 发现 /140
 - 6.2.3 轮廓 /140
 - 6.2.4 审计 /141
 - 6.2.5 修复 /141
 - 6.2.6 监控和调整 /141
 - 6.2.7 管理 /142
- 6.3 基准 /142
- 6.4 ITIL-ITSM 流程 /144
 - 6.4.1 服务支持 /144
 - 6.4.2 服务台 /146
 - 6.4.3 事件管理 /146
 - 6.4.4 服务交付 /148
 - 6.4.5 其他方面 /149
- 6.5 IAVA 流程 /149
- 6.6 数据分级 /152
 - 6.6.1 案例研究 : Big Tyre Corporation/153
 - 6.6.2 数据分级流程 /154
- 6.7 风险评估 /154
 - 6.7.1 信息收集 /155

- 6.7.2 安全控制评估 /156
- 6.7.3 业务需求 /157
- 6.7.4 资产估值 /158
- 6.7.5 漏洞评估 /159
- 6.7.6 安全控制措施有效性
评估 /160

6.8 小结 /160

第7章 执行、汇报与分析 /161

- 7.1 介绍 /162
- 7.2 发现报告 /162
- 7.3 评估报告 /165
- 7.4 框架报告 /168
- 7.5 审计报告 /171
 - 7.5.1 主动扫描审计报告 /171
 - 7.5.2 被动扫描审计报告 /172
 - 7.5.3 审计趋势分析 /174
- 7.6 主动扫描 : 时间安排与
资源 /177
 - 7.6.1 审计参数 /177
 - 7.6.2 时间安排 /180
- 7.7 审计趋势与性能报告 /180
 - 7.7.1 基本报告 /180
 - 7.7.2 高级报告 : 控制图 /184
 - 7.7.3 介绍漏洞群 : 控制
性能报告 /187
- 7.8 合规性报告 /190
 - 7.8.1 系统合规性报告 /190
 - 7.8.2 合规性执行总结 /192
- 7.9 小结 /193

第8章 规划 /195

- 8.1 介绍 /196
- 8.2 章程制定 /197
 - 8.2.1 介绍：业务价值 /197
 - 8.2.2 目的和目标 /197
 - 8.2.3 范围 /198
 - 8.2.4 假设 /198
- 8.3 业务用例 /199
- 8.4 需求文档 /199
- 8.5 安全架构建议 /201
- 8.6 RFP/202
- 8.7 实施计划 /202
- 8.8 操作流程文档 /204
- 8.9 资产估价指南 /205
- 8.10 漏洞管理策略 /205
- 8.11 部署策略 /206
 - 8.11.1 基本策略 /206
 - 8.11.2 基于风险的策略 /207
 - 8.11.3 改进的时间表 /208
- 8.12 部署标准与进展报告 /209
- 8.13 小结 /209

第9章 策略性漏洞 /211

- 9.1 介绍 /212
- 9.2 操作环境 /215
- 9.3 管理外部因素 /216
- 9.4 控制内部漏洞 /217
 - 9.4.1 业务模式 /218
 - 9.4.2 业务程序 /218

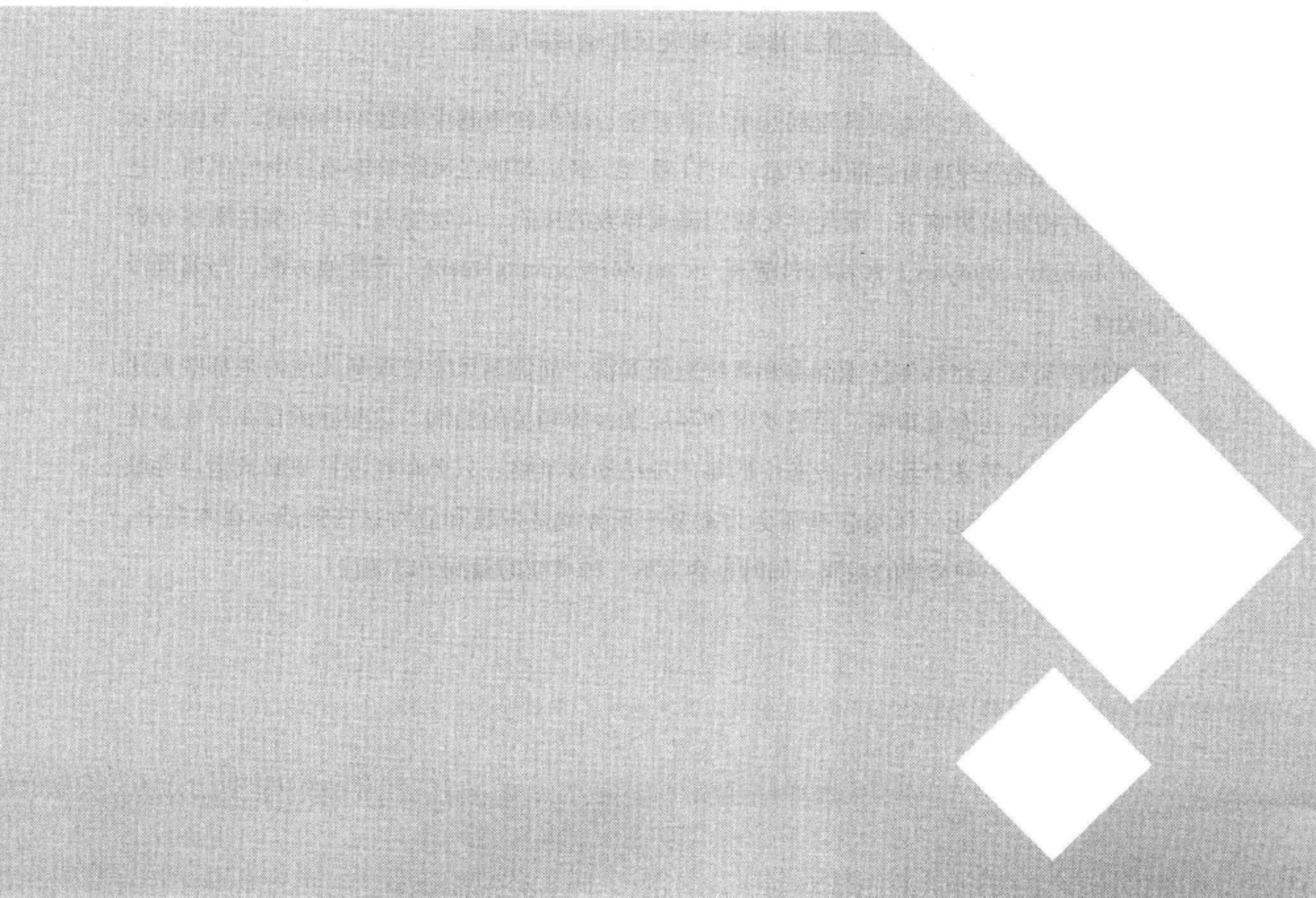
- 9.4.3 复杂性 /219
- 9.4.4 反应方案 /219
- 9.4.5 漏洞方法论与变更 /220
- 9.4.6 复杂性 /222
- 9.5 规避原则 /223
- 9.6 了解对手 /225
 - 9.6.1 优点与缺点 /225
 - 9.6.2 现实事件 /226
 - 9.6.3 目的与目标的对比 /227
 - 9.6.4 时间放大效应 /228
 - 9.6.5 政治环境加剧
攻击 /229
- 9.7 小结 /229

第10章 总结 /231

- 10.1 介绍 /232
- 10.2 跨领域机会 /233
- 10.3 跨技术机会 /234
 - 10.3.1 代理 /234
 - 10.3.2 补丁管理 /235
 - 10.3.3 应用渗透测试 /235
- 10.4 流程缺陷 /236
- 10.5 运行环境的变化 /238
 - 10.5.1 省时 /238
 - 10.5.2 节电 /238
 - 10.5.3 分布式计算 /239
- 10.6 报告 /241
- 10.7 服务水平协议 /241
- 10.8 小结 /241

第1章

绪 论



漏洞管理（Vulnerability Management, VM）是对漏洞进行检测、分类、修复和消解的一种周期性活动。这是从公司或政府部门的角度所给定的比较宽泛的定义，在本书中将做进一步的讨论。漏洞管理并不是一门新兴的学科，也不是一门新兴的技术，其重要功能已被军事部门和私营企业普遍应用，主要用于对信息系统、流程和策略方面的漏洞进行检测并加强防御。随着组织的日益复杂，有必要将漏洞管理的功能完整地抽取出来，辅之以相关的支撑工具以使其成为一项专门的业务。这样一来，漏洞管理作为风险管理的一个部分将获得更为精确的定义。

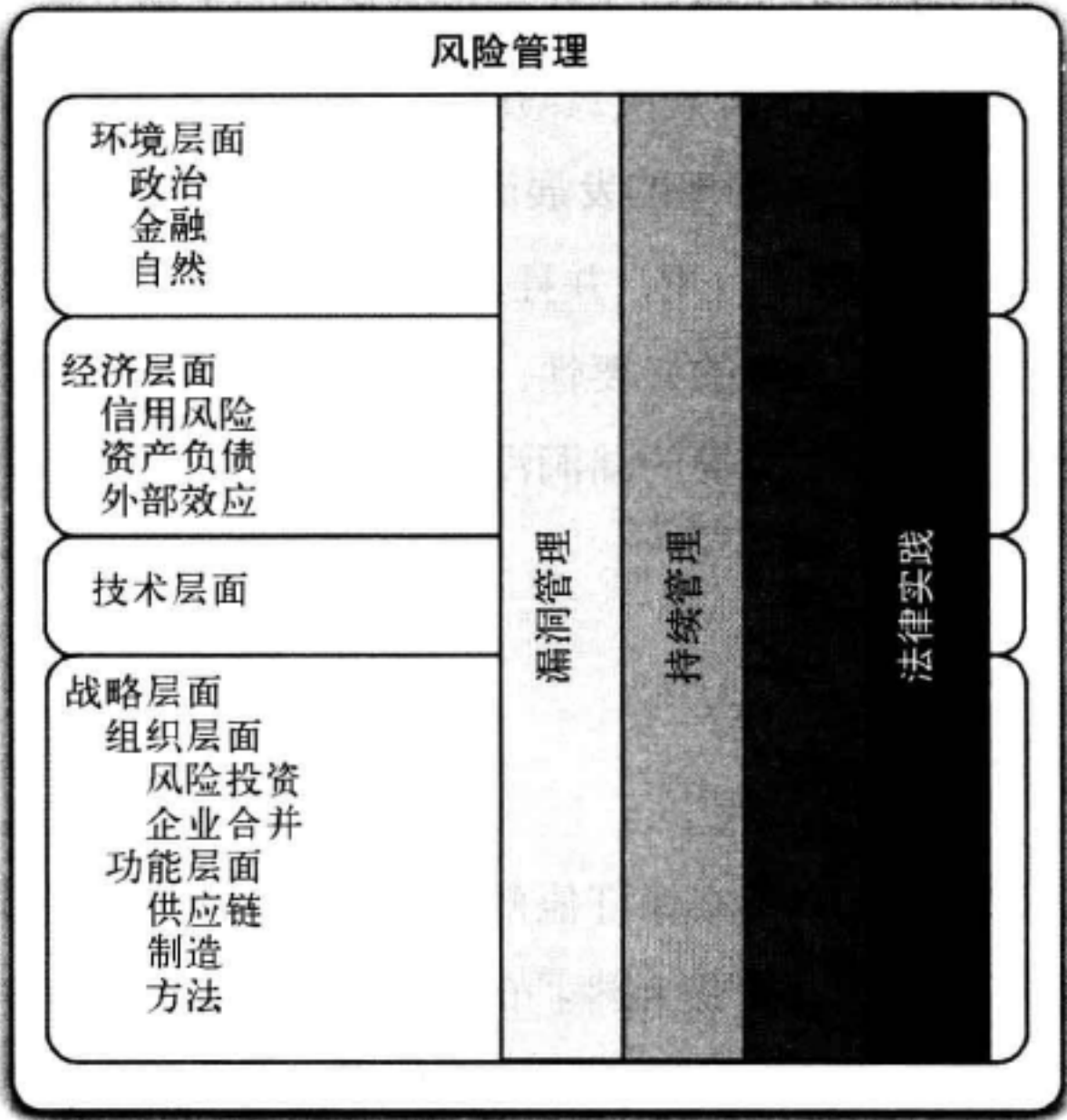
1.1 风险管理的作用

风险管理旨在检测出可能发生损失的情况或事件，并获取相应的风险应对方法。有以下几种应对风险的方式：

- 接受风险，即什么也不做，让它发生。众所周知，这是养虎为患。
- 防范风险，即采取措施防止风险发生。
- 降低风险，即采取相应的保护措施来减轻风险造成的后果。

在漏洞管理中，我们看到风险都是由信息系统、流程和策略中的缺陷导致的。下图显示了漏洞管理与风险管理项目之间的关系。可以看出，漏洞管理在风险管理项目中的作用，是为了在组织的控制或影响下，发现并化解由漏洞导致的风险。风险管理中有关事件概率分析（event probability analysis）和持续性管理（continuity management）等其他方面，与漏洞没有直接关联。

漏洞管理通常关注软件技术漏洞和系统配置漏洞。而需要风险管理师关注的是那些无法被自动检测到的，与企业策略、经济状况和环境条件等相关的漏洞。这些漏洞存在于业务流程、策略和供应链等各个环节。业务中的每一项活动或策略，只要存在设计缺陷或适应性缺陷，就能被利用。因此，风险管理师更为重要的任务就是发现和应对这些挑战。在本书中，我们将主要讨论第一种类型的漏洞，同时也会对第二种类型的漏洞予以关注。



漏洞管理在风险管理体系中的角色

1.2 漏洞管理的起源

漏洞管理已经存在了很长时间但鲜有人关注，直到最近才引起人们的重视。长期以来军队都非常明白漏洞管理的重要性，并一直通过训练，不断完善漏洞管理。从组织和策略部署到各个士兵及武器的防御检查，其目的与审计是一样的。反复的训练、装备以及防御重组都是一种修复和改进。但如果不了解敌情，所有这些活动都无从谈起。

一个学过军事历史的学生能够很容易认识到交战的一方是如何利用对方的弱点和策略失误打败对方的。人们往往倾向于将这些胜利者誉为天才，而不会认为是失败者缺乏能力。例如，在坎尼战役中，汉尼拔撤退中线兵力，包围了罗马军队，从而从四面发起进攻，最终击败了罗马军队。由于这一经典战术，汉尼拔被视为战争天才。然而，人们还可以把这场战役看成汉尼拔的对手——罗马执政官之一的瓦罗（Varro）的策略失误。瓦罗原本相信罗马军队能够从中部突破汉尼拔部队的前线部队，从而将敌人的整个防线击退到他们身后的河流处，谁料汉尼拔竟然会改变前线阵型，假装撤退部队中部以诱敌。在战争中，保持部队行进统一是一项基本的军事纪律，但是瓦罗完全没有考虑这些，而这是一个漏洞。

然而，在商业领域里，人们总是倾向于认为应对风险的失败是能力不足的体现，尤其是当公司强大、富裕，能投入足够的资源解决风险时，这种观点尤为突出。

作为 IT 领域的一个学科，漏洞管理的发展尚不够成熟，用户也缺乏应用经验，这是因为之前一直没有强大的、企业级的技术可用；并且，以前人们也未充分认识到一个完整的、集成的、有着精确的流程定义的解决方案的必要性。虽然在企业环境下军事化的纪律可能不是必需的，但缺乏规范将可能导致某个关键的漏洞没有发现或没有补救，并可能最终导致灾难性的损失。

1.3 安全产业及其缺陷介绍

企业和政府一样都是依靠新产品来保证他们的网络安全，这种情况并不少见。安全产业因此一直致力于销售需要不断进行升级和维护的产品和服务。当某个安全问题出现端倪时，供应商早已开发出相应的解决方案。当用户开始滥用网络端口登录远程服务器时，供应商为我们提供了防火墙。当病毒成为一个不容忽视的问题时，供应商立刻又提供了反病毒软件和服务。当类似震荡波的蠕虫病毒出现后，供应商又在反病毒软件中增加了更多的网络防病毒功能。当企业内部的应用程序成为受攻击对象后，应用级防火墙又应运而生了。

不幸的是，这些解决方案似乎都治标不治本。大多数安全问题都是由于没有以安全的方式进行编码、修复、配置或设计而导致的。这就好比军队缺乏指挥官的监管、训练和充足的武器装备。技术供应商不断为我们提供产品化的解决方案，就好比将可以买到的全部武器都交给部队，但敌人并不会把武器作为攻击目标。由此可以购买安全产品是一种失败的策略。

我并非有意贬低各种安全技术产品的使用。安全技术产品是一个完整的安全策略的重要组成部分。但是，当各种安全问题发生时，很少有人会认真去检测和修复被利用的漏洞，而这些安全技术中没有一项能够完全补救诸如没有使用强密码或没有为所购买和安装的套装软件打补丁之类的漏洞带来的风险。

大多数网络安全产品的价值在于在没有出现更持久、更可靠的风险应对方案时，这些安全产品能够暂时地降低风险。安装反病毒产品是一个不错的选择，只要你进行及时、正确地更新即可。当新型病毒出现时，产品应当迅速做好准备防止该病毒入侵，直到病毒所攻击的软件供应商提供补丁。否则，最终病毒将寻找到攻入组织、突破防御的方法。因此，重要的是在这些发生之前对漏洞进行永久性修复。

1.4 来自政府和产业的挑战

IT 企业面临着来自世界各国政府的挑战。不同的国家立法标准都不尽相同，使得企业在法律中常常遇到“雷区”，给一些操作带来了挑战，而跨国公司面临的挑战尤其巨大。在一些国家中，监管机构和工会组织，认为它可能会侵犯隐私。在一些国家，采集指定用户的上网行为记录是强制性的，并必须按照要求提供给政府。一些模糊而繁杂的规定，如美国的 Sarbanes-Oxley (SOX) 法案，已经形成了众多见效甚微但投入费用可观的安全控制措施。这使得基于全球管理软件包的网络安全主动防御面临更大的挑战，因为安全管理者现在只能从那些真正安全的行动中挑出符合当地法规的部分予以实施。

有关安全控制及相关认证和审计的行业标准越来越多，这些行业标准包括：SAS 70、SOX § 404、ISO 17799、ISO 27001、PCI、FIPS、HIPAA、GLB、IEEE P1074、EAL。标准和认证固然重要，但它们往往让我们忽略了最核心的问题：存在漏洞的软件、架构和策略。没有什么能长期替代基于良好编程、充分测试、合理配置并经有效实践后部署的软件是无法代替的。

1.5 漏洞的来源

对于软件购买者而言，软件公司自身就是一个真正的挑战：它们的编码水平和基础设计能力都应该持续改进。一些公司希望销售更多的产品，所以他们不断推出更多的功能，而不是去提高前期代码产品的安全性。他们可能开发了一种新的电子通讯协议或一项使用该协议的新应用，但他们却从没有在一开始就试着确保该协议的安全。在及时将漏洞告知用户和发布补丁方面，软件供应商的表现也并不令人满意。这是由于在软件供应商看来，打补丁是一件得不偿失的工作，因为没有人会为这些额外的开发工作埋单。由于缺乏内在动力，上述问题难以解决。在某些情况下，部分供应商可能市场中处于绝对的垄断地位，消费者很少有其他选择。在这种情况下，更换软件制造商代价巨大，因为公司可能已在数千个结点部署了该软件，数百名训练有素的技术支持工程师也已经只熟悉该软件。

1.6 有缺陷的漏洞管理示例

当人们在落实漏洞管理措施时，通常不太认真。例如，某公司为符合支付卡行业

(Payment Card Industry, PCI) 标准的规定, 打算在整个企业中部署漏洞管理代理软件。该公司之所以这么做仅仅是因为审计人员告诉他们应该这么做, 于是他们就做了, 完全不考虑这项措施能够带来的收益和实施的效果。这样一来, 唯一的有形要求就是这项已部署的技术, 但没有人思考接下来的事情, 例如这些显而易见的问题: “我们应该在哪些主机上安装漏洞管理代理软件?” “首先应该修复什么样的漏洞?” 等。我把这种做法称为复选框安全策略 (check box security strategy), 即经公司授权的某个人向公司提供一份任务清单, 然后公司就对照执行, 每完成一项就在上面打钩。

复选框安全策略的另一个明显问题在于, 能解决造成众多漏洞的根源的某个工具却没有正式的负责人。在上述例子中就体现为, 代理软件和服务器安装后却没有指定任何人负责维护它们。不管用什么办法, 系统是无法自行维护的。所以需要有人查看系统报告, 修复或重装组件或代理软件, 并确保报表服务器处于良好的运行状态。还需要有人负责监控整个系统, 确保系统能达到预定目标。这和军队部署是类似的, 如果没有指挥官的监管, 部队就难以协调一致地行动。

1.7 漏洞管理的重要性

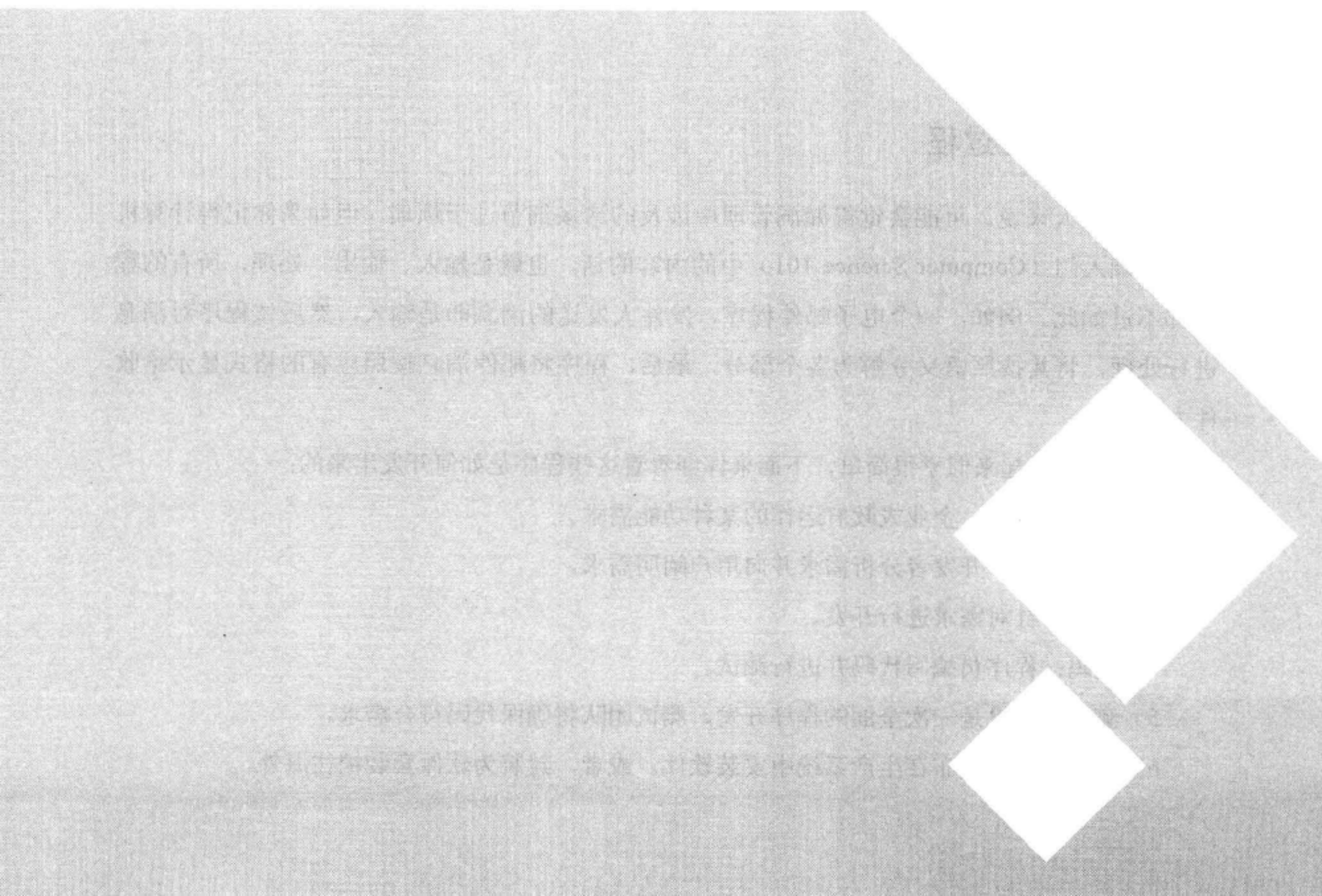
对于企业而言, 资源是有限的, 不可能取消在风险管理上投入太多的资源, 因此前期的风险分析非常重要。但是, 这不应该成为不进行漏洞管理的借口。当未实施漏洞管理时, 在入侵检测或安全事件管理上花费有限的经费似乎难以自圆其说。尽管漏洞管理涉及更复杂的流程和系统, 但能十分有效地降低企业面临的风险, 而当企业需要防范的致命风险减少时, 企业面临的风险状况将完全不同。

本书并不仅仅只是简单地介绍了漏洞管理技术和一些操作技巧, 而且还从技术角度和流程角度深入讲解了漏洞管理是如何运作和发挥作用的, 这两个方面相辅相成, 缺一不可。技术工具在漏洞管理过程中起到推动作用, 本书将花大量篇幅让读者体会到这一点。当在公司环境中实施了漏洞管理后你会发现, 要实现安全真正需要的是那些愿意并且能够严格保证公司基础设施稳固性和安全性的人, 除此之外的一切都是在浪费金钱和时间。

在本书中, 你还将深入理解漏洞以及漏洞控制的策略意义。漏洞不仅存在于单个主机或是网络设备中, 还可以存在于其他层面, 这些层面中的漏洞必须通过调整技术战略才能予以解决。处于组织和行业层面的漏洞管理则属于风险管理的范畴了, 已经超出了任何技术领域。

第2章

漏洞体验



2.1 简介

漏洞管理这一学科，能够很好地与其他诸如信息技术基础架构库（Information Technology Infrastructure Library, ITIL）、ISO 17799 及 ISO 27001 等标准框架中的管理规范相融合。在业务活动中经常会出现各种情况，并且可能会引发一些比较固定的工业处理流程，这些管理规范就是为了将这些工业处理流程标准化而制定的。例如，ITIL 中的事件管理规范，就是针对 IT 基础设施的故障排查操作流程而制定的。这些故障不是基础设施和基本服务的正常运作，而是运作过程中产生的一些异常现象。故障排查操作是一系列由人力和技术支持的过程，而事件管理规范正是为了将这一过程及其他类似的工作流程标准化。

同样，漏洞管理是为了规范由不良的设计、开发、配置或质量控制导致的 IT 组件漏洞所引发的处理操作而制定的。然而值得一提的是，IT 业界似乎对需要处理这些漏洞感到很惊讶。软件开发从开始就明显存在下面的问题：当程序开发者未能充分考虑程序的使用情况时，程序中就会出现一些故障和缺陷。换句话说，漏洞是由于未能全盘考虑产品在整个系统中的使用情形，未能充分考虑各种可能的输入和输出，未能充分考虑可能超出接受范围的操作参数而导致的。

2.2 漏洞产生过程

对外行人来说，可能会觉得漏洞管理中涉及的诸多细节过于烦琐。但如果你记得计算机科学基础入门（Computer Science 101）中的内容的话，也就是输入、输出、处理，所有的程序都不过如此。例如，一个电子邮件程序，发件人发送的消息即是输入，然后该程序对消息进行处理，将其按照语义分解为各个部分，最后，程序将邮件消息按照应有的格式显示给收件方。

这些流程看起来似乎很简单，下面来详细看看这些程序是如何开发出来的：

- 1) 概念和方案：企业或政府运作的某种功能需求。
- 2) 需求收集：开发者分析需求并向用户阐明需求。
- 3) 设计：针对需求进行开发。
- 4) 编码：程序员编写代码并进行测试。
- 5) 测试：如果是一次全面的程序开发，测试团队将确保代码符合需求。
- 6) 部署：企业宣布在生产系统中安装软件。或者，封装为软件套装销往海外。

这些流程听起来很耳熟，其实这就是所谓的系统开发生命周期（System Development Life Cycle, SDLC）。上面的每一环节中，用户都有可能会因为漏掉一些东西而导致问题的出现。因此还需要下面几步：

7) 漏洞检测：攻击者、研发者或最终用户都可能在输入、输出或在与其它系统功能的交互过程中发现安全漏洞。这些漏洞也可以被认为是需要完善的潜在功能。

8) 漏洞利用：测试者从各种不同的角度尝试利用该漏洞来突破软件原有的各种使用限制。

9) 漏洞修复：软件补丁被开发和安装部署，重新运行程序，于是新功能被自动启用。

这几步也是我们这些年在生活中经常遇到和熟知的。复杂性、连通性和互操作性，这三个因素使得步骤 7) 到步骤 9) 比以往更普遍、也更困难。

2.2.1 复杂性

程序运行的环境和系统都极为复杂。任何应用程序都是建立在一层又一层的软件组件之上的。即使最基础的计算机软件构件也异常复杂：由无数的模块和各种更加复杂的硬件选项执行重要的交互功能。以传统的“hello world”为例，这是所有计算机科学专业的学生入门时都会学到的程序。任务很简单，就是要写一个能输出文本字符串的程序。但在图形界面下，这种类型的程序可以变得相当大。首先，必须要运行解释器，以加载程序、读取指令及执行任务。此外，该程序还必须通过库与操作环境交互。库是一些简单的代码段，能读取简单的程序指令指令并将它们放在操作环境中执行。本例中的库为输入/输出库（即通用 I/O 库），能够在屏幕上创建一个窗口并将文本映射到其中。为了支持该库，操作系统还必须加载一些额外功能，以便与显卡及显示器的硬件驱动程序通信。从上述例子可以看到，这是一个非常复杂的关系链，每个环节都是安全世界中潜在的攻击的对象。

我们创建了一个假设程序来说明这种复杂性如何迅速地被系统中的另一个组件所利用。该程序的功能很简单：将文件作为输入，并将文件内容显示到屏幕上。当用户通过邮件接收到一个文件时，它会将该附件保存到本地文件系统中，然后运行我们创建的这个程序。选择该文件作为输入，程序的确如预期的那样执行。但这正是问题的开始。

这个文件中包含了一段不可显示、无法被该程序中使用的通用 I/O 库处理的字节序列，这使得库函数跳转到指向该文件剩余部分的内存地址处。该文件的剩余部分将遗留在库函数运行过的内存空间中。该文件的这部分代码其实是一段恶意获取计算机控制权的程序，但看上去好像是库函数成功完成了其自身的任务。

在本例中，我们写的程序完成了预期的工作。但在代码中并未顾及到底层组件的复杂性。于是，当考虑谁应该修复这个问题时，争议出现了。在处理数据之前，是否应该验证输入数据能正常显示？库的编写者是否也该这么做？这些争议虽然有趣，但已经超出了本章的讨论范围。然而在某些案例中则必须考虑这点，尤其是评估内部开发的软件和外部供应商提供的软件时。

2.2.2 连通性

网际协议是使程序能远程运行的基本协议。它允许复杂的软件以模块化的方式工作，在层次化结构的不同层面处理信息。这些层面通常通过网络连接进行区分由填充着基于 IP 消息的网络连接来区分。

但是 IP 最初是美国国防部为了在核攻击情况下也能有效进行通信而设计的。其思路是：如果网络中的一个结点被摧毁，那么信息流可以绕开它，由其他路线到达目的地。该功能的设计初衷是要应用于封闭的军事通信网络中的。然而，IP 现在已广泛应用于人人都能够访问的开放式网络中。这是一个技术误用的典型例子。IP 的使用远远超越了当初的设计初衷。

如果一个系统能够与另一个系统进行通信，那么就可以通过网络控制某个结点的输入和内部处理过程。这种能力跨越了物理屏障，并容易被滥用。远程访问是提高生产力的关键因素之一。但像其他所有为改善工作环境而做的事情一样，远程访问是一把双刃剑。网络的复杂性和工具软件的易得性进一步加重了该问题。

DNS 就是一个很好的例证。一个应用程序的组件可能遍布全球，记住成千上百万的 IP 地址一度是一件不可能的事情。有人发明了 DNS，使得大家能够方便地记住网络名称。但 DNS 系统的设计从未考虑过安全性，只考虑了效率。于是它被黑客广泛利用。甚至是 DNS 服务器上运行的底层软件也不断遭到攻击，并且，这种趋势很可能持续下去。

2.2.3 互操作性

互操作性成为当前所有硬件和软件生产商竞相争取的对象。如果不这么做，市场就会变小。想象一下，如果一个操作系统不使用常用的 SMB 文件共享协议，或者一台打印机仅支持独有的文字处理软件，这将是用户无法接受的。随着技术在企业和政府的运作中得到普遍应用，计算机制造业逐渐认识到了这个问题。例如，EBCDIC 和 ASCII，这两个数据编码方案都是为了使系统和程序之间的信息交换保持一致，以及编码信息的处理可重复（只存在很小的

差异)而提出的。

我不认为这些编码方法本身有任何内在漏洞,但正是系统中所有组件的兼容性,使得该编码方法具有可攻击性。某个处理 ASCII 字符集的程序可能存在于一个漏洞,如果其他几个程序都用到了这个 ASCII 码处理程序,那么该漏洞就会造成更为严重的影响。一个程序可以被用来利用另一个程序,一个程序的缓冲区溢出将使得整个系统被随意访问。如果计算机系统中一个被各种程序广泛使用的标准组件存在漏洞,那么所有使用该组件的程序都可能被攻击。

关于互操作性能够导致严重且广泛传播的漏洞的一个更为具体的例子是微软 Windows 的“GDI+”缓冲区溢出漏洞(buffer overflow)(源自 MITRE Corporation, CVE-2004-0200)。GDI+ 应用编程接口提供给程序开发者一套显示和打印信息的函数库。在这个例子中,处理 JPEG 图像的组件未能以安全模式对输入进行解析。当 JPEG 图像格式畸形时,程序失效并允许图像的剩余部分留在计算机内存中,这部分可能被当做程序执行。这是一个显而易见的问题。

由于 GDI+ 产品是一个 API。这意味世界上的许多程序开发者都在调用某个版本的 API 执行功能。这些程序的安装者,在一些情况下,希望确保他们拥有正确可用的 GDI+ API。于是,他们安装自己的副本。结果,一些计算机系统的目标文件系统的几个不同的部位,最终被安装上一些存在漏洞的 API 版本。为了解决这个问题,每个供应商都必须检测他们的产品并及时打上补丁。

2.3 创建漏洞: 一个例子

现在,让我们使用上文提到的所有项目设计一个场景,通过实际演练来了解善意程序造成的恶劣后果。本章开始提到的电子邮件程序就是个好例子。稍后,我们将按照 SDLC 的所有步骤,来看看是哪里会出现问题。

1) 概念和方案: XYZ 公司认为,如果他们使用比市面上任何其他程序都好的特定程序发送和管理电子邮件,是一件很不错的的事情。该特定程序将使用经特殊改写的格式,在消息中提供独一无二的功能。会出现什么问题呢?它将会使系统更加复杂。跟 GDI+ API 例子类似,更多的复杂性带来更多的故障。如果新功能对该产品的成功非常重要,那么从项目一开始,就应该重视系统中特定部分的安全性。至少,关键组件的可靠性和安全性应该是概念中的一项重要内容。

2) 需求收集: 从提出概念的人那里收集详细的功能需求。我们对所有将采用的功能设计

及其能够对业务模型起到的提升作用都感兴趣，因为它们可能会在将来的某个时刻起到重大作用。会出现什么问题呢？没有一项需求包含有关安全性和可靠性的内容。而提供安全性和可靠性应该是任何应用需求的一部分。来自需求收集过程中的每个想法都将导致某种程度的项目需求变更，这将引发功能性的设计缺陷。在需求阶段，限定需求的范围，除了能够使成本最小化且确保时间进度外，还能最小化漏洞产生的几率。

3) 设计：应用程序的细节，包括数据库、文件系统、操作系统、兼容性，都需要仔细设计。会出现什么问题呢？设计，像需求分析一样，只考虑功能，所有的注意力都集中在如何使用应用程序工作上。应该将一部分工作优先放在确保应用程序创建严格符合限定条件的数据流上，然后对输入/输出数据的有效性进行验证，这样就可以较好地保证输入的应用程序函数参数的合法性。进程间通信必须设有数据有效性检测和权限授予的代码段，以使进程能够验证来自另一个进程数据有效性和进程来源的合法性及操作权限。从这个阶段开始应该引入威胁建模分析。通过识别程序可能遭受攻击的高危部分，开发团队可以集中精力来改进各个组件，尤其是最易受到攻击的组件。

4) 编码：完成设计之后，开始编写代码。会出现什么问题呢？在漏洞世界里众所周知的是：程序开发人员并不都愿意遵照那些能够避免类似缓冲区溢出等漏洞的优化编码方案。但是，应该尽量使用最佳安全编码方案，例如动态调整的数据结构的边界检查（bound checking）和输入数据的有效性检查。

5) 测试：测试的频率很高。故障不断被发现并被迅速修复。测试员不断对所有未包括安全组件的功能需求项进行验证以确保它们的正常实现。会出现什么问题呢？软件测试一般承受着最大的时间压力，因为管理者希望尽快生产出成品，并且几乎不能容忍丝毫的延迟。开发人员常常将程序分解以期尽快通过测试阶段，因为每当这样做时，测试人员的注意力就会全部集中在功能上而跳过回归测试（regression test）。应该参照前文提到过的威胁模型分析来对产品进行某些渗透测试。

6) 部署：最终产品出厂了，经过包装被运送到世界各地毫无戒心的顾客手中，附带软件的安装使用说明书。会出现什么问题呢？说明书中没有对软件运行最合适、最安全的系统配置做任何说明，也没有指出在网络接口上哪些网络端口是除了服务器外对其他任何 IP 地址都不能开放的。开发团队应该制定一些策略来处理发现的漏洞。如此一来，就需要更多的测试人员来对软件进行渗透测试了。

7) 漏洞检测：一个对该产品感兴趣的用户可能会好奇，如果利用这个电子邮件软件发送

一封经过特殊设计的电子邮件会产生什么情况。这封特殊的邮件能够满足软件特定模块的处理要求，但是，其消息数据超出了该模块标准数据结构的要求。结果是程序崩溃了，并在操作系统的内存中留下一些消息碎片。

8) 漏洞利用：现在，该用户想知道是否能在在这封特殊邮件的消息内容中嵌入一个程序以使其能够留在内存中并运行。结果发现这确实可行。在实验中，一个小的“hello world”程序运行起来了。

9) 漏洞修复：此刻这个非常激动的用户开发了一个增强版本，在邮件消息中嵌入的程序能够生成邮件自身的多份副本，并自动将其发送给通讯录中的其他收件人。电子邮件程序强大的功能设计使得访问邮件通讯录变得更为容易了。就这样，该产品的首个电子邮件蠕虫诞生了。

建议：应该尽早解决能够被利用的漏洞。从长远来看，在达到相同安全等级的情况下，这样做能有效减少技术上和安全上的工作量，并能减少检测和防范工具的开销。

2.4 使用漏洞管理程序的理由

我们为何要开发漏洞管理程序呢？原因既有技术方面的，也有业务方面的，下面将一一列出，你总会找到与你的情况最相符的理由。

2.4.1 网络过度开放

在IT安全方面，企业通常有两种不同的目标：任务性目标（mission）和强制性目标（compulsory）。任务性目标的实现直接关系到企业创收与增加利润；强制性目标的实现则主要是出于谨慎和遵守规则的考虑。购置保险就是强制性目标的一个例子。公司购买责任保险以降低可能的损失。很多公司十分重视任务性目标的实现，将其置于很高的优先级，而不是强制性目标。网络安全是强制性业务目标的另一个例子。当现有的网络防御措施不足以抵抗精心设计的攻击时，一些公司会自然地选择进行风险分析（risk analysis），并继而以此来决定在网络安全方面的支出。有时，网络的规模和复杂性使得符合成本效益的防御实际上无法抵御风险。所以有的公司干脆选择了放任风险的存在。

也许你的公司的网络防御焦点是放在检测和防范攻击上的，但是，仅仅阻挡网络访问或

关闭边界是不够的。这些防御措施不可能是完全可靠的、全自动化的或是能全面应对各种潜在攻击的。任何从事安全行业的专业人员都知道内部威胁和外部威胁同样严重，然而大多数防御措施都不是面向内部威胁的。目前，在一个拥有 5 万结点的网络中，要在每一个端口都实现入侵防护、病毒防范、内容过滤、流量分析以及应用行为分析是不经济、不现实的。即使公司财力充足，能做到这点，也还会需要更多的防御措施。因为上述网络防御措施不可避免地会存在一些漏洞，而这些漏洞可能会被利用且直接用于攻击该公司。例如，众所周知的绕过入侵检测系统的方法中，加密就是其中较为简单的一种。采用加密的应用软件能很好地降低攻击的成功率。

解决这些漏洞的唯一出路就是采用基本的防范策略，该策略的做法就是移除每一个防御失效的结点。大多数网络安全策略依赖于边界防御（perimeter）和 / 或链接防御（bolt-on defense），但如果这些防御措施失效，就会导致主机向各种漏洞利用手段完全打开大门。这就是网络过度开放所导致的最糟糕的情况。用户觉得安全，但并不是真正的安全，需要增加额外的安全防御措施，其中一项就是主机必须加强防御，排除各种可能被利用的漏洞。

2.4.2 安全系统配置标准缺失

大公司通常会为联网系统制定一个或多个配置标准。这些标准包括台式机和服务器操作系统、网络设备，甚至打印机的配置标准。在这些标准中通常有内置的实践案例。如果设置的标准没有实施到位，那么漏洞可能会比标准缺失时出现得更多。其实，有成文的标准仅仅能说明人们确实关注到了设备的状态。

但是，即便存在标准，配置仍可能老化过时。标准一经建立就不会再轻易更改，因为很难同步所有的主机使其按修改的内容变更配置。即便有一个良好的补丁管理系统，也无法依靠补丁完全解决这些固有的配置问题。大多数情况下，补丁管理系统无法检测出每一个需要修复的问题，因此它并不能取代漏洞管理。

标准化的缺陷在于漏洞的普遍存在性。如果一项标准配置存在某一漏洞，而这一配置又被全面部署，那么该漏洞就会广泛地存在于各个部署结点中。如果不能对其进行及时的检测和修复，将会导致严重的安全问题。

2.4.3 重大经济损失风险

当遭受攻击的风险较高时，管理层自然地会将注意力转移到风险识别的作用上。随着政

府和维权体系法规的日益完善，由诉讼和 / 或民事处罚等造成经济损失的可能性显著增大。想象一下，如果因为没能及时补救一个危险的、已发布的漏洞而导致客户机密数据的丢失，那么客户可能会马上诉诸法律。

加利福尼亚州民事法（California Civil Code 1798.84）条规定，与加利福尼亚州居民有商务往来的公司如果未能在规定期限内将安全漏洞告知受害者，则将处以高额罚金。公开这些漏洞可能会造成巨大损失，而如果对这些漏洞放任不管，所导致的民事罚款也很高。该法律条款仅是一个例子：无论公司的实际地理位置在哪里，只要公司与当地居民进行商业活动交易，就要为数据漏洞承担责任，接受惩罚。

2.4.4 收益损失

任何商业活动最为关心的问题就是收益或潜在收益，因为它们的损失将是直接的损失。当一个客户流失时，企业不仅遭受了收益上的损失，名誉也会受到影响。挽回名誉损失比挽回其他的损失要困难十倍。对潜在客户来说，也是如此。通常来说，一起安全事故被大肆宣传后，若想在此逆境中赢得新客户，是非常困难的，并且要付出巨大的代价。

有一个非常奇怪的现象，普通客户比集团客户更容易抚慰。由于某种原因，普通客户常常忘记或忽略某个特定公司的安全危害，但集团客户则不同。但是，普通客户也会逐渐变得不愿与任何遭到过政府处罚的公司进行交易。鉴于此，企业更应该致力于漏洞的管理。

2.4.5 生产力损失

系统遭受损害后会有一段时间不可用，如果是核心系统，员工将无法正常工作，企业生产力也将显著下降。比起员工完全停止工作，我们更难评估以下这种情况造成的损失：员工仍能继续工作，但要花更长时间才能完成任务。有时，这种应急工作状态很难立刻启用；而同样的，当遭受损害的系统恢复服务状态后，应急状态也难以立刻终止。即使对事故进行总结分析后，应急工作状态下产生的数据记录可用，这些数据也必须与恢复后运行的系统重新进行同步。

并且在系统恢复服务前通常都有一些耗时的事情必须先行完成：分析故障原因、重建工程、打补丁、考虑附加的安全性，以及密切监视第二轮攻击。为了完成这些事，负责处理故障恢复系统的 IT 员工将无法从事那些能够更直接地增加收益、减少成本、提高生产力的工作。而启动应急工作状态并不能及时地满足市场需求，甚至还有可能会增加一些机会成本。而且

不可能让一些 IT 员工专门等着应对突发事件，所以我们只有在事件发生时，将现有的资源临时投入到应急响应措施当中。

2.5 漏洞管理程序故障

在继续讨论之前，必须要向初次接触漏洞管理的读者澄清一些漏洞管理中经常遇到的问题。漏洞管理不等同于渗透测试。漏洞管理仅是渗透测试的一个部分，它用来发现可能导致侵入内部计算机系统的漏洞。例如，一个漏洞扫描程序能够发现一台电子邮件服务器的某个漏洞，攻击者可以利用该漏洞将这台服务器当做跳板，以安全方式访问组织机构中的其他更关键的目标机器。渗透是利用漏洞获取其他系统访问权以及利用相关漏洞获取更高级别的未经授权的访问权的行为。

漏洞管理是一门学科。它需要有一致性、专业性，并需要不断地重新评估。虽然能够通过技术提高执行效率，但却无法回避下述事实：必须不断地对系统打补丁和调整系统配置来满足复杂而多变的商业需求。当公司启动一个准备不足的漏洞管理程序时，失败将如影随形。单纯的购买技术、运行软件，然后期待程序自身能完成一切是不可行的。下面，让我们来看看 Acme 公司的例子。

2.5.1 案例研究1：获得组织的支持

Acme 公司在世界范围内拥有 8 个办公地点，总计 15 000 台工作站，130 台服务器。其管理架构并不太复杂，从 CEO 开始，最多不超过 3 级。制造工厂位于中国，部分关键组件生产企业位于北美。公司增长稳健，大多数生产和销售人员都在不断增加。工程团队则是一个总部设在北美的封闭式机构。

该公司的 IT 运营高度分散，由各地的行政经理聘用 IT 经理进行管理。虽然各地行政经理（office manager）需要向全球运营总监（global operations director）汇报，但地方 IT 管理人员并不直接同全球 IT 运营总监联系。全球的 IT 预算虽不富裕但还算充足，一般都不包括员工培训费用，因为员工能力较强，能通过自学掌握大部分相关技术和处理流程。地方的 IT 预算是由当地行政经理结合本地 IT 经理的建议制定的。全球 IT 对网络基础设施，如 WAN 和 LAN 的配置等，负有总的责任。各地的 PC 机和服务器则由当地 IT 团队管理，但也需要遵循全球统一的标准。这种架构，使得全球 IT 处于为各地办公室提供网络服务和安全保障的地位，本

地 IT 只需为本地商务需求（主要是销售和结算功能）提供支撑。

去年，一位 IT 员工由于没有获得年终奖（而他的同事却获得了），一气之下决定实施报复。他知道公司的大部分机器都没有打补丁，而他的计算机也连接在同样的网络中，他于是想通过修改震荡波蠕虫病毒（Sasser worm）的载荷，对特定的主机发起攻击，并将自己不喜欢却还获得了年终奖的同事的工作站作为攻击的目标。

不幸的是，震荡波病毒危害巨大且持久，导致好几个系统陷入瘫痪，并渗入到几台关键的服务器上。后来，法庭调查人员找到了攻击的来源。Acme 公司花费了几个星期的时间来给这些机器打补丁以恢复服务。这场攻击严重影响了公司的生产，给公司造成了巨大的收入损失。这个员工因而被解雇了。

Acme 公司安装了一套复杂的网络监视系统。同时，还购买并安装了一套补丁管理系统。一名日常负责电子邮件管理的 IT 职员自愿管理该系统。

首先，让我们来看看图 2-1 中的 Acme 公司的组织结构。这个结构看上去与常规不太一样但多年来一直提供着良好的服务。随着新技术的应用和基于互联网的日益密切的商业联系，技术战略总监和独立的 IT 运营总监的重要性逐渐凸显出来。随着安全问题逐渐成为企业关注的焦点，Acme 成立了一个专门小组负责解决该问题。Ward 是安全风险经理（security and risk manager），负责管理商业风险，他还是一个技术发烧友。他把这视为横向的职业发展，并在总部建立了一套入侵检测系统（IPS），每天都沉浸其中。

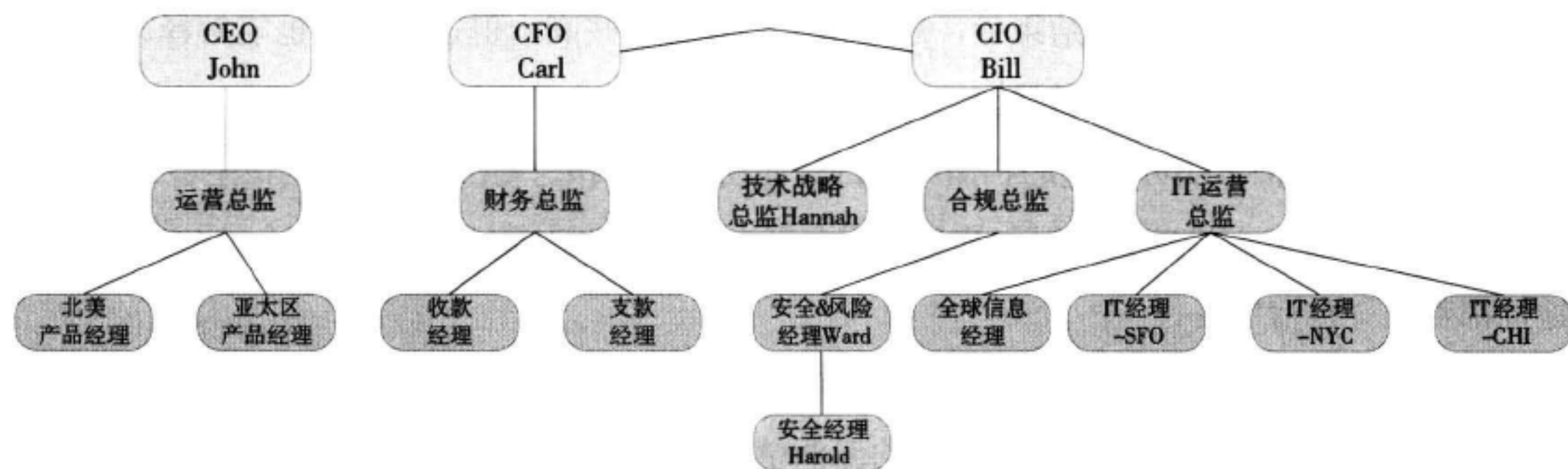


图 2-1 Acme 公司组织结构图

Harold 是一个长期以来一直受到信赖的员工，他领导实施一个风险管理项目来避免发生上述的安全问题。Harold 的直属主管是 Ward。Harold 彻底研究了市面上所有风险管理工具，通过与 PC 机管理员和服务器管理员进行讨论，最终选择了一个工具。Acme 的 8 个办公地点

都部署了相关设备，用于扫描漏洞。扫描从5月3日开始，下面是自5月3日起发生的事件的日志。

1. 事件

5月3日：Harold主持了首次漏洞扫描，地点是旧金山办公室。令Harold感到意外的是，当地只有300名员工，但扫描程序却报告了4094台漏洞宿主机。拨打供应商的技术支持电话后，供应商检查了扫描程序的配置文件，运行了一些诊断程序，并试验性地扫描了几台机器，结果发现一切正常。供应商建议Harold核查一下网络配置，称也许是路由器将扫描包发送到了其他办公区域。

5月4日：Harold怀疑San Francisco的扫描器有问题，但没有同支撑团队争辩。也许扫描器设置了不正确的路由配置。但这里的网络配置同其他地方的配置一样。该问题在产品评估时并没有出现过。

全球信息经理（global messaging manager）通知Harold，如果不能保证不影响公司业务前，就不要扫描电子邮件服务器。因为电子邮件服务器需要按服务等级协议（SLA）进行维护。Harold有些不满，因为有几台关键服务器都是邮件服务器，其中还有几台接到互联网上，面临着更高等级的威胁。他将情况汇报给他的直属上级领导Ward，Ward告诉他不要在这个问题上引发争议，因为Harold在这一职位中是位新人，还需要取得其他技术经理的信任。

5月7日：对其他办公室的扫描结果看上去正常。当地的IT经理收到了初始的漏洞报告。

5月12日：第一周扫描结束后，总的宿主机数量有些高，但这种情况也能解释。Harold和所有的经理进行了后期电话沟通。

5月15日：旧金山办公室的扫描问题持续存在，仍然报告检测到4094台宿主机。芝加哥也显示同样多的宿主机数量。对扫描结果进行长时间仔细地检查后发现每一个被扫描到的IP地址都对应一台宿主机。

5月16日：总的来说，宿主机的平均漏洞个数急剧下降，这是个好消息，但是单个主机的最高漏洞个数依然不变，而有些主机的漏洞情况甚至变得更加糟糕。Harold发电子邮件给纽约地区的IT经理以寻求漏洞补救方面的帮助。

5月20日：进一步的研究表明，在每个地区，扫描器都发现了更多的宿主机，这导致了宿主机平均漏洞个数的减少。每个地区分公司都有4094台主机，已经超过了扫描器的扫描能力。此外，纽约的IT经理还没有回复Harold的电子邮件，于是Harold决定给他打个电话。这位经理解释说他正在对纽约的网络架构进行一项大的部署，有可能会对其初始的架构设计

进行一些小的改动。他承诺一旦完成部署，就立刻看邮件。

5月31日：在技术支持下，Harold发现网络中有一些数据持续地发送到实际上并无主机对应的IP地址。解决该问题的应急方案就是手动输入所有活动的主机地址，但这是不切实际的，因为许多地址都是动态分配的。Harold需要找到到底是什么东西在响应设备发现探测器。

6月5日：所有报告的漏洞都未修复。Harold咨询了他的主管Ward，Ward建议召开一个本地IT经理参加的电话会议，但至少还要再等一周Ward才能挤出一个小时的时间。

6月12日：应有8个地区的IT经理参加电话会议，但只有5个参加了。IT经理们说他们没有资源用于修复漏洞，但如果工作负荷允许，他们将每周尽力处理一次最高优先级的主机。电话会议上，一个IT经理向Ward提出，应该一次只部署一项新技术，而不是并行部署，这样IT经理们能够在下一次部署前评估该技术对整个公司带来的影响。IT经理们还抱怨没有通知他们要部署该漏洞扫描系统，并担心扫描程序会影响他们的网络工作性能。Ward同意扫描只在晚上进行。亚太产品经理在电话会议中也抱怨扫描导致他的一台关键服务器结点失效。由于亚洲的白天是美国的晚上，他表示不希望进行扫描除非Harold能够保证扫描不会影响系统。

6月16日：两个办公区中一些情况最糟糕的机器已被修复。剩下的IT部门员工花了一个周末来清除由于某个用户插入被感染的U盘而引入的一个新病毒。即便大多数台式电脑都处于关机状态，扫描仍然显示许多网络存在4094台主机。

6月23日：Harold忙于跟踪扫描程序的问题并继续补救行动。他发现新部署在防火墙内的一个入侵检测系统（IPS）导致了扫描器在每个IP地址上都显示有主机。他还做了个测试，关闭防护功能后进行扫描，这一次结果尽善尽美。Ward知道Harold关闭了某个办公点新部署的IPS后，口头训诫了Harold，告诉Harold应该先跟自己商量一下。他让Harold重新开启IPS，并将扫描集中到有固定IP地址且系统所有者允许扫描的服务器上。Harold成功的向亚太区产品经理展示了扫描对服务器没有任何害处。于是亚太区产品经理同意扫描该区主机并同意一周内进行关键漏洞修复。

6月25日：漏洞扫描报告的结果并不乐观。由于IPS系统导致的IP对应的主机是实际存在的，因此漏洞评分情况并没有改善多少。走势图显示漏洞的数量仍呈上升趋势但并没有发现新的漏洞。Harold对此结果感到困惑，于是联系了产品线技术支持部门，报告了一个可能的缺陷，但技术人员说这是正常的行为表现。

6月30日：新的漏洞扫描系统扫描了公司范围内的25台活动主机。平均每台主机花费了

400 美元，大大超过了他的预期。大约 17 台主机得到了修复。

7 月 18 日：公司的表现让 Harold 感到非常沮丧，他辞了职，决定去一个能“严肃对待安全问题”的公司找一份新工作。

那么，对 Harold 和 Acme 公司来说，到底发生了什么呢？Acme 公司需要更多的人手吗？好像不是。Harold 选择了错误的产品吗？好像也不是。他起初是一个乐观主义者，打算周密细致地完成好工作，但是问题很快地出现，并带来了更多的工作量，而只有极少数问题得到了修复。在缺少内部支持的情况下，系统的有效性和 Harold 都受到了质疑。

9 月 5 日：致命一击。公司远程分部的一位员工被解雇了，他决定临走之时做出一些轰动的事情来。他知道公司的核心服务器系统是基于有缺陷的标准开发的，于是写了一个能够利用内部雇员常用的管理员密码的 perl 脚本发起攻击，最终对许多系统造成了损害。全球各地内的 IT 经理们都不知所措、尴尬无比。他们这才记起来应该有人负责实施漏洞管理才对，但为时已晚。

2. 分析

那么，成功的漏洞管理程序到底需要些什么？本案例从一开始就充满了错误。让我们来看看是哪里出现了问题。

- ❑ 过程失败：过程中有两个问题。首先，缺乏正式完整的项目计划以及上级管理层的支持。首先 Harold 应先努力调研技术产品。第二，Harold 缺乏对过程的重视。风险管理是一个过程，而不是一个技术解决方案。
- ❑ 准备：技术处理无疑是一个相当大的问题，好几件事都没有做好。Harold 居然不知道防火墙内部署了一个 IPS 系统。漏洞扫描系统出来的分析报告也难于理解。而发送给 IT 经理的那些报告显然被忽视了，Harold 很可能认为他们根本不关心，或是不理解这些报告。
- ❑ 相关人员：IT 经理们显然没有参与项目的很多策划活动，并且对项目投入甚少。查看 Acme 公司的组织结构图（图 2-1）会发现 IT 经理们只向全球 IT 运营总监汇报，不需要向同一级的组织（如 Harold 等）汇报，除非汇报到 CIO 的级别再向下下达。他们凭借对各自网络的理解，防护着自己的网络。当面对一项他们一无所知的新型网络防护技术时，他们感到对这些技术工具完全无法控制，于是他们对自己网络的保护就是将这些工具拒之门外。

□ 责任人：Harold 和 Ward 召开的与 IT 经理们交流的电话会议效果如何呢？在 Harold 看来，IT 经理们的低出席率反映出了他们感兴趣的程度。但是，对一个忙于部署新技术（在本例中是 IPS）并正在进行一项重要的网络设计变更的人来说，自然的反应是先把精力放在他所负责的事情上，回避那些会导致更多事情的工作。IPS 是防火墙软件的一部分，因此被当做是一项已有技术的配置变化。此外，漏洞扫描器生成的报告也需要管理员承担更多的工作，因而搁置该报告能暂时延缓对计算机环境造成的影响。

如果 Harold 采取如下不同的做事方法，可能会有一个更好的结果：

□ 责任人：初期就取得高层管理者的直接支持。确保管理者理解漏洞修复和监控的需求。如果没有迈出这重要的第一步，就没有明确的预期、对运营系统的影响也不可知。Harold 就没有得到领导的大力支持，无法引起各地 IT 经理的重视。在计划、设计和选择技术过程中，IT 经理们都没有参与，他们没有机会在自己的环境中去测试、验证该技术。同时，补救工作没有及时完成时，Harold 也没有其他渠道将问题上报。此外，高层管理者能够促进团队中关键参与者之间的沟通。理想的情况是，由级别类似总部 CIO、CSO（或 CISO）的高级管理者一起讨论初次进行漏洞管理的重要性。

在高级管理层的承诺中，明确指出：成功的修复是一项关键的工作成果指标。在前面的例子中，Harold 没能引起各地 IT 经理们的重视，他们几乎没有修复任何漏洞，因为这些提交的漏洞报告在他们看来就像是 Harold 在抱怨他们的系统多么不堪一击，而这将增加他们不认可的工作量。因此 IT 经理们根本就没有动力去修复漏洞。他们从未把 Harold 当做严峻的 IT 挑战的解决方案的提供者。而如果告知 IT 经理们将以修复漏洞的表现对他们进行考评，则他们将更愿意遵从。事实上，他们很可能会表现得更加积极主动。为了在大棒之外再加一根胡萝卜，设置最低平均漏洞分值的竞争可以让这一过程更加追求卓越。可以对取得最低平均漏洞分数或最大改进的 IT 经理实施额外的奖励。

□ 准备：测试！测试！测试！任何联网的、能被任意访问的，并有可能被入侵的网络系统，其全部组件都应该在实际环境中进行充分测试。最终，形成一份可接受扫描的系统标准。扫描给系统带来的不良影响，以及被鉴定为不需接受扫描的系统，都应当清楚地记录并存档。

□ 过程：任何管理上的变更都应该是深思熟虑后再实施的。在 Harold 的公司里，扫描检测开始前，内置了 IPS 的新防火墙系统已经进入实施阶段，当扫描检测开始时，旧金

山办公室已经完成了该防火墙的部署。该防火墙为分配到该区域内的所有 IP 地址的探测扫描提供应答（本书后面将对该现象做更多介绍）。当 Harold 持续进行扫描检测时，该防火墙持续应答。如果 Harold 参与了变更管理过程，他就会知道公司正在部署防火墙，那么他就能提前测试扫描效果，并将防火墙部署前后的扫描结果进行对比，找出差别。

2.5.2 案例研究2：技术集成的挑战

Abacus 公司是一家跨国的电子计算器制造商和经销商。他们使用了液晶显示技术和一种特殊的手势识别技术，使得他们的电子计算器较传统计算器执行得更快。总的来说，Abacus 公司就是一个拥有几项核心专利的快速发展的小型电子公司。

Abacus 的制造厂位于内布拉斯加州和阿拉巴马州。Syllog 是其在德国的一个商业伙伴，负责采购国外设备并进行配送。Syllog 是多种独特电子设备的分销商，主要面向亚洲消费者以及美国和英国的小型零售商。大多数设备都只由三个制造商提供，Syllog 公司在业务和基础设施上同这三家制造商保持着密切联系。Syllog 公司约 40% 的基础设施由 Abacus 公司与它联合建设，并直接为 Abacus 公司服务。

在 Abacus 公司，IT 基础设施建设非常成熟，拥有基于信息技术基础构架库（ITIL）的变更管理（change management）和事件管理（incident management）工具。其所有的商业伙伴都要遵守 Abacus 公司的政策和标准，或者采用更为严格的政策和标准。他们已经用 ITIL 框架中的基本模块实现了一个“服务台”（service desk）模型。该服务器是接收和管理事件、逐步升级处理的中心点。由于 Abacus 公司是一个制造企业，订单量难以预测，因此其生产都采用适时反应（just-in-time）策略。换句话说，他们只在接到订单后，根据订单的要求设计和生产产品。此外，公司承诺在接到订单的 10 个工作日内发货，因此决不允许任何原因导致的停工。

Abacus 公司中与销售相关的 IT 操作都由当地专门的市场定制应用软件进行控制。每个市场都有不同的语言、不同的文化和独特的商业关系，因此相应地也需要不同的硬件、软件和应用程序。通用操作系统和基本的工具软件由全球 IT 运营部门提供和管理。但是，各种应用软件和特殊软件均由当地 IT 运营部门负责。为了保持这种运营模式，Abacus 公司建立了专门的介于全球、地区 IT 运营团队之间的服务系统。

管理层决定下一步在整个公司实施一个全盘的漏洞管理试点计划。该计划涉及公司所有站点，还包括有意愿参加的商业伙伴们。Abacus 公司内，PC 主要使用微软操作系统，而服务

器则使用 Linux 操作系统，只有部分办公地点使用 Solaris 操作系统。

除了清除漏洞之外，Abacus 公司还希望通过这次的漏洞管理计划尽可能全面细致地对公司的应用软件进行合规性验证、补丁状态验证。他们拥有成熟的漏洞管理的内部流程，但其执行系统却没有有什么技术支撑资源。技术支持工程师与员工比例为 1 : 300。对 Abacus 公司而言，自动化是一项关键因素。例如，在 Linux 系统上，每周就有 20 个标准系统维护和数据采集 Shell 脚本在运行，因此采用自动化的漏洞扫描而非依赖于内部流程是非常重要的。自动检测的实施，由系统工程部负责人 Carl 负责。

1. 事件

11 月 2 日：在高层领导的全力支持下，Carl 组建了一个由系统支持组、PC 工程师和网络总监组成的小团队来共同挑选工具软件、修订已有的流程。该项目获得了 350 000 美元的预算。

11 月 23 日：在项目时限和预算范围内，该小组选择了两种工具，从运行结果上看，它们与各种系统配合得很好，包括 PC 和服务器代理上的网络漏洞扫描设备。最初的想法是让代理软件在所有系统中运行，但普通 PC 上已经安装了很多办公用的应用软件，再继续在其上安装代理软件并加以维护对 PC 的日常操作影响太大。考虑到代理软件对服务器操作影响较小，且服务器生成的报表还能提供集中的补丁修复信息列表等特殊信息，因此自动打补丁将能够减少工作量。合规性报告有助于管理者将精力集中在实现预期目标上。两种工具总计花费：330 000 美元。

11 月 30 日：工具软件采购完成，安装在了一台服务器上，硬件设备也已部署到所有需要部署的办公室。得到商业伙伴的许可后，在一些扫描广域网（WAN）连接时曾出现过问题的区域也安装了扫描器，但是当地的管理者抱怨说就为了几台漏洞宿主机，这个花费太高。针对这一情况，公司做出了一些调整，让一些服务器代理专门扫描没有安装代理的本地 PC。

12 月 6 日：由于假期来临，许多关键员工都将休假，于是暂停了所有非关键项目的变更控制。暂停时间为 30 天。休假前还没有完成所有网络扫描器的安装。

1 月 15 日：所有人休假结束，Carl 制定了一项交流计划，保持与所有漏洞扫描环节中的关键管理人员的沟通。他为 6 个关键办公地点的 IT 总监创建了初始用户名、密码和权限。他们将是漏洞管理系统的主要用户。流程中有一项一致同意的变更是，IT 总监将能够登录到系统中查看他们所在区域公司的漏洞状态。是否能访问漏洞信息主要是通过判断访问者的 IP 地址是否在允许的 IP 地址段内实现的。

2 月 2 日：初次扫描完成，系统自动给 IT 总监们通报了扫描结果。他们对报告的质量、

内容以及准确度都很满意。这似乎是 Abacus 公司又一项技术和流程的成功。Carl 把这套系统交给生产系统支撑团队，他们也是漏洞管理开发小组中的成员。有些 IT 经理抱怨他们不得不进入两套不同的系统中去查看主机状况。运行代理的系统给他们提交了几个服务器漏洞信息，网络扫描系统则提供了 PC 的漏洞信息。Carl 开始同供应商讨论解决这一问题的最佳方案。

2 月 10 日：内布拉斯加州的 IT 总监要求给予他的 PC 支持团队全部 9 名成员系统访问权限以便获取漏洞报告、执行漏洞修复。系统管理员于是提供了一份需要完整填写的信息表，用于创建用户。创建上述两套漏洞扫描系统的系统访问用户，需要经过新用户填表、表格返回、数据录入、密码校验 / 重置等操作，完成这些大约花费了两天时间。

2 月 12 日：商业伙伴 Syllog 公司抱怨得到了一份奇怪的扫描结果。该报告中，Syllog 公司网络中的 PC 名称不正确，显示的是 Abacus 公司的服务器群。总部 IT 管理人员经过核查后，发现 Syllog 的管理人员实际上拿到的是这两个网络共同的扫描结果。原因是他们使用了相同的 IP 地址。漏洞扫描器发送给该服务器的报告是正确的，只是因为两个网络的 IP 地址重复。

2 月 14 日：另一个办公区的 IT 管理人员也提出了同样的要求，他的 PC 支持团队成员共有 6 人。该 IT 管理员称他们也必须通过访问才能执行修补工作。用户增加后，必须制定详细的流程来支持两个分开运行又协同工作的系统。

2 月 18 日：针对 IP 重复的问题开发出了一套新的技术解决方案，该方案在报告中通过计算机名称而不是 IP 地址来区分主机。为适应这一方案，所有的网络定义和权限都需要升级。与此同时，开始将两个漏洞系统集成到一个报告系统中。这项工作由一位具有丰富 Perl 编程经验的系统工程师完成。

2 月 23 日：其他 6 个地区的 IT 总监也都要求赋予他们的支撑团队成员漏洞管理系统的访问权限。其中有两个总监希望将系统按照服务器和 PC 分类，并交由两个不同的支撑团队分别负责，因为这两种设备上的漏洞修复技术有所不同。上述变化导致系统用户总数增加到 32 位。

2 月 25 日：经过一位员工 3 天的工作，网络定义和权限许可的修改完成了。目前系统有 47 位用户。一些服务器管理员抱怨安装的补丁未经充分测试，并需要制定更加精确的运行时刻表，以免对生产作业产生影响。

3 月 1 日：漏洞管理小组接到通知，内布拉斯加州的一位用户两周前离职，应该撤销其访问权限。用户离职手续办妥后，该用户的访问权限就被撤销了。还没有找到接替的人选，因此，内布拉斯加州当前只有 IT 总监能接触这些报告。

3月4日：最初的报告显示，漏洞从记录到最终修复，周期较长。经过一天的调查，漏洞管理团队的领导成员发现，问题主要发生在阅读报告发现漏洞和将漏洞录入事件及变更管理系统两个操作之间。该流程是在实施之初就确定的，一直工作正常，但需要3周的时间才能实施修复。Carl会见了合规管理总监和信息安全总监，讨论新系统的当前性能以及补救缓慢的问题。合规管理总监指出修复关键漏洞能允许的最长时间为7天。但根据系统的报告，目前需要花费2到3周时间。

3月5日：基于微软的系统都配置了活动目录证书，通过它来进行深入的扫描。实际运行中，证书允许扫描器登录到系统中，检查关键补丁并配置项目。管理报告已经按照需求，通过Web接口从两个系统中合并到一份报告中。系统工程师开始按照详细修复报告进行工作。但是，两个系统中的许多数据因类型不同无法进行对照、匹配，因为一个使用CVE（常见漏洞和错误）编码来检测漏洞，而另一个使用Bugtraq来检测。

3月10日：Carl上周全部时间都在努力完善当前的报告和修复流程，但从运行结果来看仅仅将平均处理时间缩短了一天。看起来关键问题出在负责录入信息到事件和变更管理系统的工作人员还有其他的工作职责，例如要进行实际的修复工作。通过补丁管理系统发布补丁能够解决一些问题，但如果问题不能在一到两天之内得到解决，那么很快就需要发布新的补丁。然而，约有30%的漏洞需要人工修复，这意味着会很慢。另外，基于代理的漏洞管理系统设计为了使用自己独有的变更管理系统，不具备自动输出变更和导入变更发布状态的功能。

3月11日：尚未扫描基于UNIX的系统。为了使用新系统进行深入的扫描，需要手动为每个UNIX系统配置合适的证书。这一过程相当耗时，因为需要创建大量变更管理事件。系统中有些变更出于节省时间高效工作的考虑而被绑定到了一起，但只有具备相同配置的系统，换句话说，就是完全相同的系统，才能将这些事件绑定在一起，例如负载均衡配置（load-balanced configuration）下的一些系统。

3月12日：经过深思熟虑并同老板讨论，Carl决定最佳方案是让专人开发一个补丁—变更—事件管理系统和基于代理的漏洞管理系统之间的接口。但是，曾用Perl编写报告的系统工程师给Carl展示了他的时间表，表示没有时间进一步开发这个系统，并将系统整个的源代码交给了Carl。

3月20日：开发人员和供应商的技术支撑小组对经费支出的初步评估结果大约为152 000美元。这主要考虑了三个方面：首先，系统最初的设计只是在发现漏洞时产生SNMP陷阱。没有一个变更管理系统或事件管理系统与该协议兼容，除非在漏洞管理系统或内部系统中编

写一个自定义接口。其次，事件管理和变更管理工作流仅当手工输入事件时，才会自动生成变更事件。需要开发自动化接口，该接口能接入两套系统，并通过事件管理数据库中的信息将两者联系起来。此外，漏洞的处理过程同其他事件的处理流程也有所不同，在变更结束之前必须进行一次验证扫描。最后，漏洞管理系统使用 CVE 和 BugTraq 编号来判断系统中应当使用哪些补丁。而补丁管理系统使用专有编码集，这种编码更详细地表明了需要哪个补丁。还需要生成一些额外的数据，以便正确匹配这些编码，并识别不一致的地方。

3 月 23 日：功能全面的系统总预算造价目前为 482 000 美元（比初始预算多了 152 000 美元 152K）。高级管理层担心这会是个无底洞，于是将总造价限定在初始水平（350 000 美元），多出来的 20 000 美元可用于开发人员发送需要的 XML 格式的消息到事件管理系统，以便节省数据录入的时间。这样，Carl 的预算就花光了。

后记：剩下的一年里，漏洞管理系统的用户一方面对系统运行效果感到很满意，但另一方面又对不完善的修复流程感到困惑。大量的工作仍然放在漏洞修复和事件处理上。用户还得建立变更请求、手工运行验证扫描，然后结束变更。Carl 一直认为高级管理层小事聪明，大事糊涂。有人就奇怪为何公司不干脆使用一套系统，这样就能够避免所有的混乱了。

2. 分析

事情起初看上去开展得很顺利。很明显，该公司 IT 运营部门非常成熟，过程管理和支撑系统能力很强。项目管理看上去也很有条理。行动之初有良好的行政支撑，制定了预算并得到批准。项目管理者 Carl 联系到了所有相关的业务伙伴，并建立了互助性所有权的关系，以使得系统虽然出现明显缺陷但还能保持运行状态。

首先，让我们看看漏洞管理的流程，这有助于更好地理解漏洞管理是如何在 Abacus 公司生效的。虽然 Abcaus 的员工训练有素且高效，但仍然在这一流程中经常感到棘手。

在 ITIL 框架中，意外事件的定义是，一些打乱事件约定的服务级别或干扰基本商业活动的事件。服务级别是按照漏洞和修复的时间框架确定的，一个新发现的漏洞如果不能被及时纠正，则很有可能不符合 SLA（服务等级协议）规范。当 Abacus 的新漏洞管理系统发现一个漏洞时，将立刻在事件管理系统中产生一个事件进行追踪解决。由于 Abacus 公司没有丰厚的财力，因此服务台上的功能可以根据所要处理的事件的不同，由不同部门的 IT 员工进行操作。所以，服务台是一个动态实体，包含通达各方的基于软件的路由。请参见图 2-2 体会这一过程。

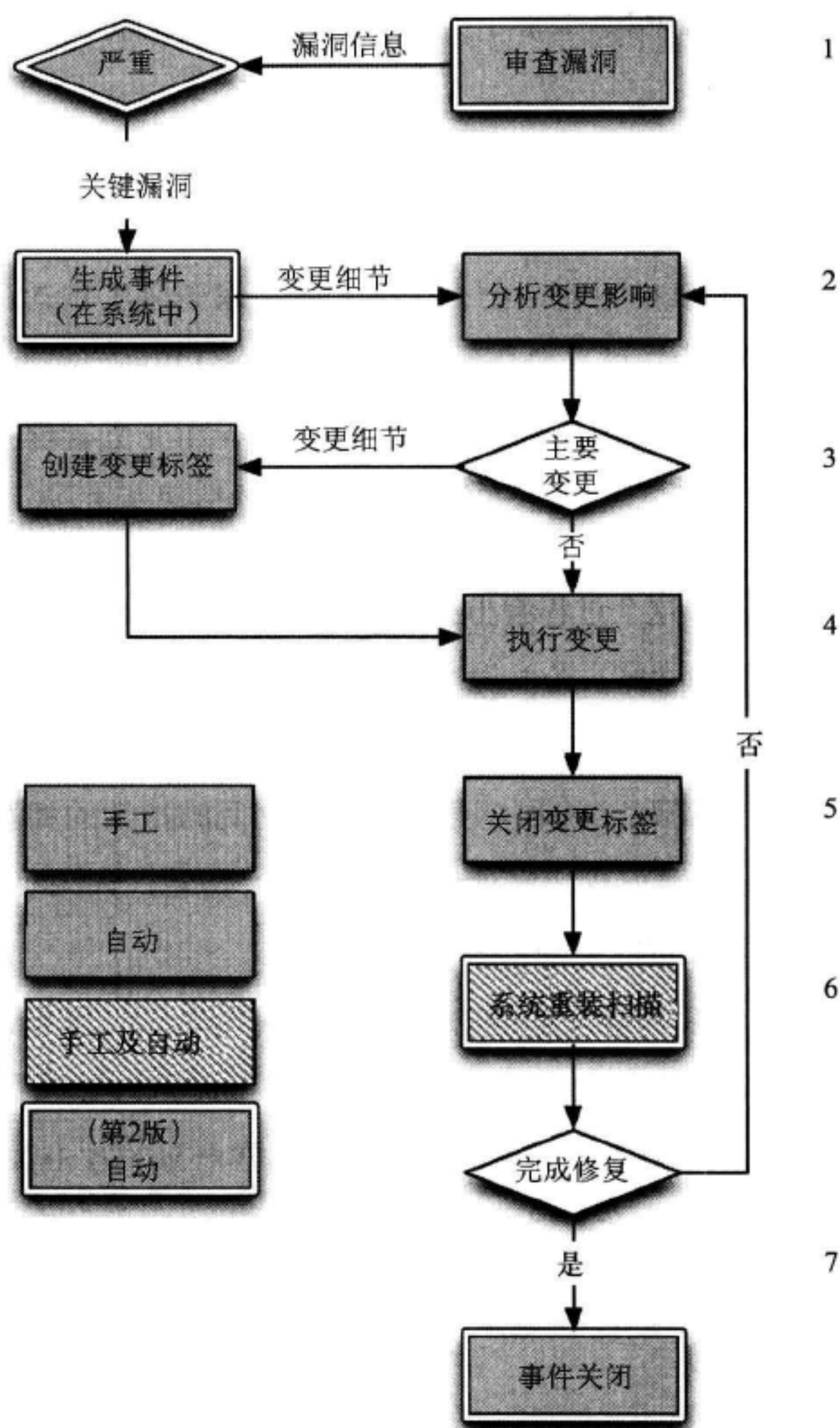


图 2-2 Abacus 公司漏洞服务台流程

- 根据网络和主机感染的不同类型，选择适当的人来评估漏洞信息。如果一个漏洞是高危漏洞，则需要创建事件并追踪其解决进度。如果是非高危漏洞，则将其放到非紧急变更流程工作表中。
- 当解决方案中的某一变更操作很复杂，并且已经确定实施这一变更会对其他功能造成影响时，变更管理系统则会创建一个变更并产生一个变更标签。这使得受影响的部门

能够通过相应的潜在影响通知来了解变更过程。它还能将目标机的技术性配置数据的变更通知给管理人员。

- 一旦为修复漏洞打上补丁或升级配置，变更标签就可以关闭了。变更标签关闭时还将发出警告：相同的漏洞在下次漏洞扫描时应当不会再出现。
- 要么是根据常规时间表的安排，要么是在系统请求下手工启动，漏洞扫描程序再次运行了。响应请求性质的扫描一般执行得很快，因为只扫描受变更影响的系统。
- 当变更标签关闭后，事件标签也将关闭。这一过程由变更管理系统和事件管理系统自动执行，但只在事件管理系统自动产生变更标签的情况下进行。

对于缺乏经验的人来说，这个过程看上去相当烦琐，但 Abacus 公司的员工有很强的纪律性且训练有素。该流程运行得非常好，为关键产品系统提供了典型的、不间断的、可预测的服务。在本例中，看上去整个过程都安排得非常妥当。但是，在系统集成过程中仍然存在几个缺陷。

一个明显的问题是事件管理系统和变更管理系统之间的持续的相互影响。对每个关键漏洞，负责服务台的 IT 员工都需要创建事件以便于追踪。接着，如果确定需要进行重大变更，还得创建一个变更标签来通知可能受到影响的其他人。

下面举一个利用该流程妥善管理漏洞的例子。如果微软 SQL 服务系统的某个普通账户的用户 ID 设置了一个弱密码，该密码的脆弱性将成为一个漏洞。如果很多人，甚至公司外的人都能访问数据库服务器，这个弱密码可能成为一个非常严重的攻击载体。为了修复该漏洞，必须更改该密码，但这么做可能会打乱几个依赖于该密码的应用程序的执行。于是，创建一个变更标签并通知所有应用程序的管理者。一旦应用程序的管理者根据系统的变动对程序做出相应的更改，该变更就完成了。

变更完成后变更标签将被关闭。在 Abacus 公司，仅仅是标签被关闭，而不是事件被关闭。在事件关闭之前必须先对该系统进行一次扫描，以确认漏洞修复成功。本例中，即是对密码强度进行测试。如果该漏洞确实不存在了，那么就手动关闭该事件。

到目前为止，我们有四项手工操作都可以自动实现。图 2-2 展示了该过程。图中用不同阴影的方框来标识人工执行和自动执行的步骤。图中基本都是自动执行的过程。图中的数字与如下步骤对应：

1：漏洞管理员审阅报告，识别主机中的关键漏洞。该项工作最适合由自动系统来完成。漏洞通常是众所周知的，由世界各地的专家评估。可以将这些评估信息内置到自动化系统中，

系统就能基于该信息进行漏洞鉴别。

2, 3: 如果发现了一个关键漏洞, 管理员就打开事件标签, 并将它分配给该系统的负责人。

4, 5: 修复完成后, 变更标签被关闭, 漏洞管理员重新扫描问题主机。

6, 7: 再次审查漏洞报告, 以确认漏洞是否还存在。这其实是重复步骤 1, 因此也易于自动执行。

创建事件标签是一个简单的可以由机器执行的利用接口的追踪活动。在步骤 3 中, 可以自动实现重新扫描的行为: 将变更管理系统和漏洞管理系统关联起来, 使得发出变更完成的通知后, 发起后续扫描触发扫描操作。但仍然保留重新扫描行为的手动执行功能, 使得可以根据所使用的流程选择执行方式。如果程序要求暂停等待, 直到下一次预定的扫描操作, 那么步骤 4 就可以省略了。如果在该过程中需调用手工扫描, 则也许需要重新扫描。重新扫描的一个细节是对目标进行限制或参数化使用。例如, 某台特殊主机或某个特定网络只允许在晚上扫描, 以免服务受到影响。这将避免在工作时间发生任何可能的服务中断。因为一旦修复完成, 重扫描只能在第二个晚上进行, 而不是立即进行。

现在, 让我们从自动化的角度来分析该流程。重新回到图 2-2。下面的流程非常类似于前面所描述的流程。但是, 该图中还用双层框线指示能够自动执行的部分(第二版 自动化)。在所有自动化步骤中, 过程都被加速, 只需几秒就能执行完毕。图中数字的含义如下:

1: 关键漏洞触发漏洞管理系统自动创建事件标签, 并将其分派到漏洞主机所属网络的负责人处。本例中, 责任人为事件管理员。漏洞管理系统将获取事件编号(ID)。

2: 事件管理员检查变更需求, 以确认是否需要大量的工作或是否会影响其他系统。

3: 事件管理员进入事件管理系统, 标记该事件需要变更, 该需要变更的标记在变更管理系统中通过两个系统间已有的接口进行实例化。

4: 变更将按计划由一位工程师执行, 大部分情况下由事件管理员执行。

5: 工程师完成工作后, 通过事件管理系统的用户接口, 关闭变更活动。这一步无法自动化执行, 但需要的工作量很小。事实上, 一些变更管理系统能够监听电子邮件消息对升级状态的回复。升级自动触发发往事件管理系统的通知, 以临时关闭该事件。

6: 事件管理系统按照事件编号, 向漏洞管理系统发送确认消息。利用事件编号进行索引, 创建另一个对该漏洞主机的扫描操作, 检查特定的漏洞及其他漏洞。

7: 如果重新扫描发现漏洞已不存在, 事件管理系统将收到一个确认消息, 随后关闭该事件。如果漏洞仍然存在, 将返回确认消息, 并发出另一个通知给事件管理员。然后再次重复步骤 3 到步骤 6。

Abacus 公司实现中的另一个问题是选择了两个不同的产品，而这两个产品在设计之初从未考虑过兼容工作。虽然市场上的一些产品都能进行基于代理和基于网络的漏洞评估，但基于代理的更多一些。经常会发生这样的情况：两个看起来都很不错的产品由于缺乏关键的兼容性而无法在漏洞管理中很好地配合使用。

结果是，用户一开始就被迫使用两套不同的系统。首先，基于代理的系统只扫描服务器，而基于主机的系统只扫描 PC。这种分别管理的逻辑与公司的地方及全球的责任分离管理模式相符。但是，如果在某些情况下，基于代理的 PC 扫描能执行，但基于网络的扫描在经济上无法承担时，这种责任分离和系统可用性就失效了。

一位知识丰富的系统工程师满腔热情地试图进行一些补救。但是，结果并不理想，仅仅得出了一份管理报告。由于不必进行系统交互，数据采集和规范性都很简单。但是系统中的数据结构和标准值都仍然存在问题。在任何其他的应用中都能发现类似的问题。后面我们可以看到，工业界正努力地避免这样的问题。

2 月 10 日，要求将 9 个用户 ID 加入系统中。此时，系统管理员肯定在心里祈祷希望这个任务不要太复杂，因为收集数据并录入到系统一向是个耗时的工作。由于 Abacus 公司大多使用微软产品，那么为何不用活动目录（Active Directory）来整合用户认证和授权呢？或者使用几乎无所不在的 RADIUS 协议。这种系统整合对今天的大多数使用标准授权机制的企业来说是非常重要的。我们看到在 2 月的 13 日、14 日、25 日，陆续有其他的用户加入到系统中。最后大约共有 50 名用户。当一个用户从公司离职后，Carl 的小组两周后才意识到应该解除该用户的访问权限。如果用活动目录组技术，则分配到活动目录组中的用户可能早就被删除或禁用了。

图 2-3 展示了目录结构与漏洞管理系统的结合情况。漏洞管理系统中有各种对象和行为。一个组或一个用户可能是一个成员、也可能是一个角色，包括对象和行为在一起的种种组合。例如，角色为“管理员”的人员要负责维护所有的系统。因此，必须授予他访问所有对象和行为的权限。但是，前面提到的工程师只能有访问特定办公室报表的权限。因此，该组被允许的行为是“报告”，被允许的对象是当地的“网络”。

可以分配给每个组或用户一个或多个角色、特定的网络和图 2-4 中所示的相应功能。这种分配的优势在于，当一个用户从一个组转到另一个组时，将被授予新组的角色，而不会依然是先前的角色。例如，如果一个工程师从加利福尼亚州的 IT 部门调到内布拉斯加州的 PC 团队，该工程师只能够按他新职位的权限访问相关漏洞信息。漏洞管理员也只能对分配给该组的网络进行扫描。

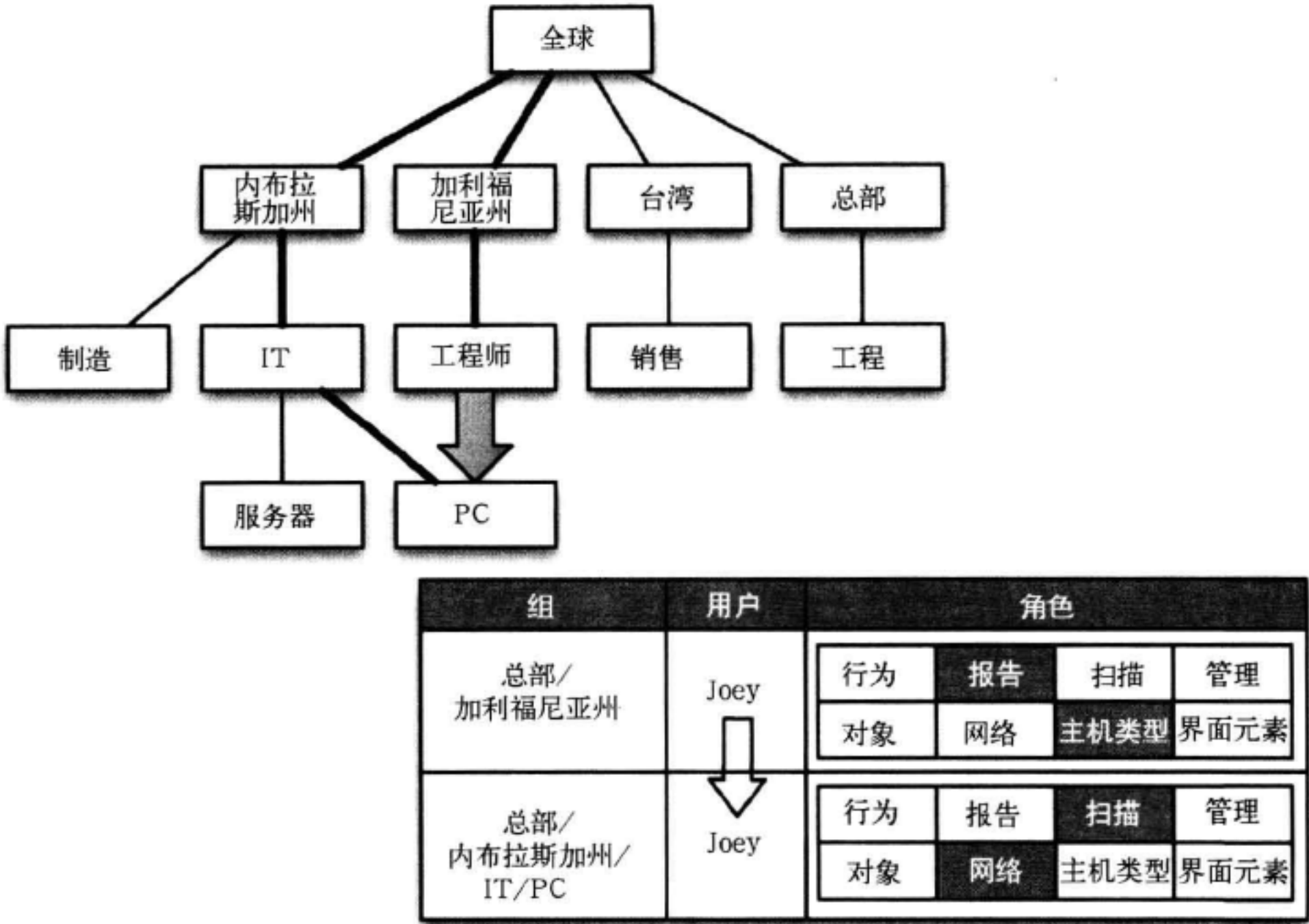


图 2-3 用目录结构分配漏洞系统角色

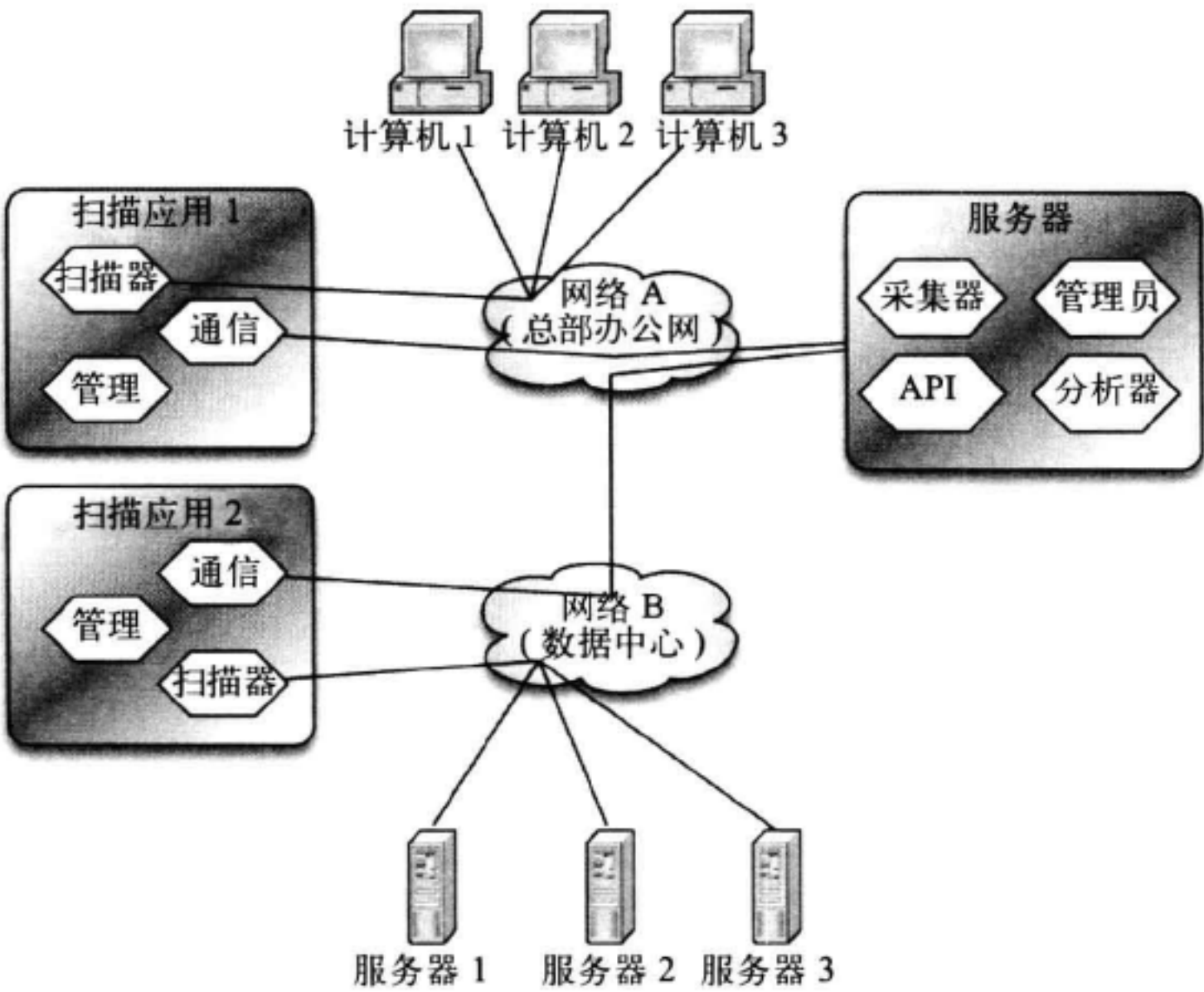
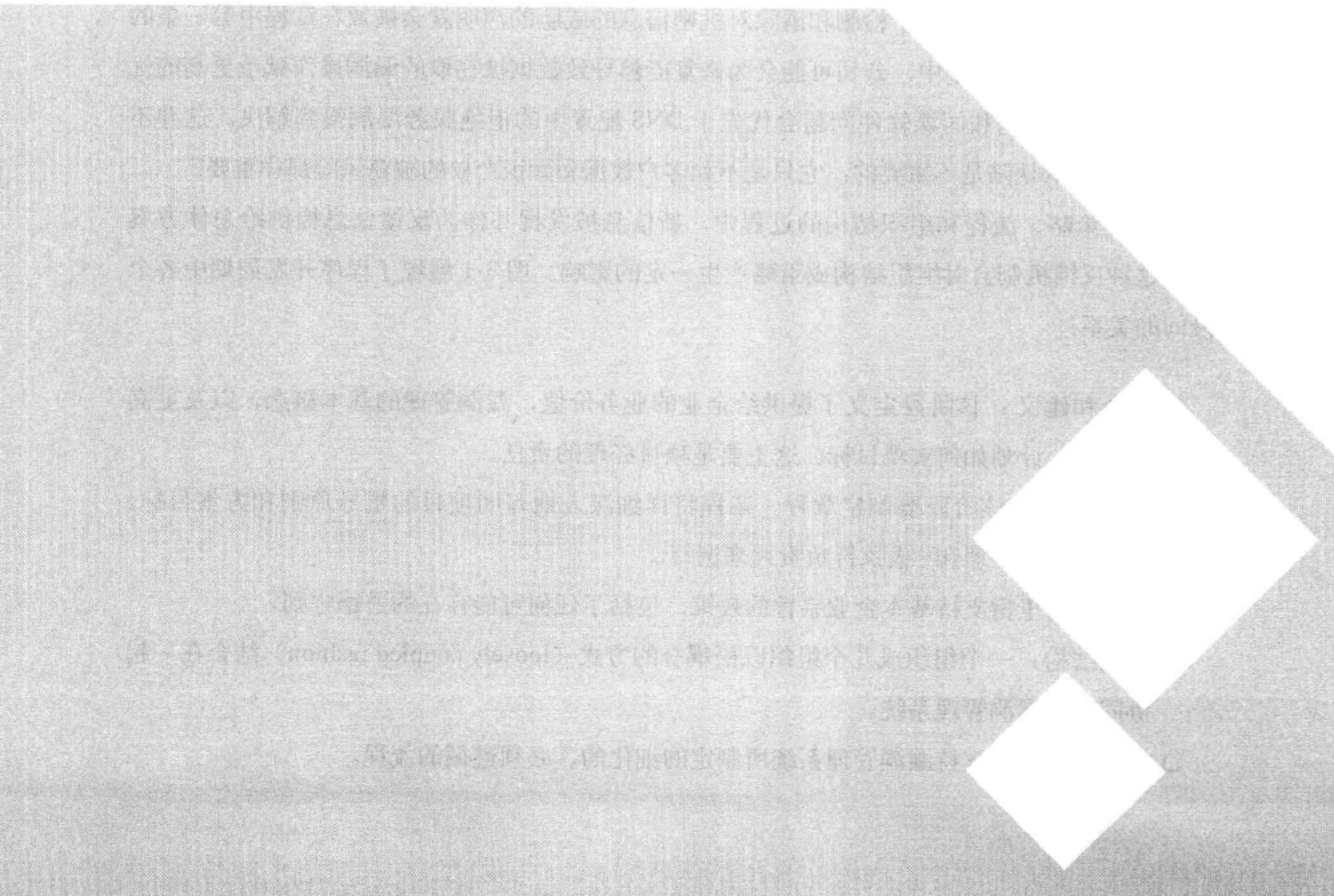


图 2-4 目录结构及漏洞系统角色

这种整合方式现在看上去是显而易见的，但当初，对 Carl 或漏洞管理小组的其他人而言，可能都难以想象会有这么多的用户。这也将我们带回前面的讨论：漏洞是如何产生的？本场景与另一个场景相似：原始系统的设计者没有考虑到系统所有可能的使用方式，这会带来包含有太多假设的需求采集活动。Carl 的设计与选择小组应该准确分析使用该系统的用户、用户的数量、这些用户将开展的主要活动。其中的情况很多：可能有人离职、有新人加入、分析员核查报告，此外还可能有人变更扫描参数及日程计划。

第3章

计划和组织



3.1 概述：计划结构

IT 组织或合规组织的架构和组合成会对漏洞管理的成效产生重大的影响。正确理解企业利益相关者和运营 IT 资产的管理者之间的关系非常重要。正是这种关系，反映了那句格言：IT 的存在是为了支撑业务。如果想要支撑业务，那么 IT 产业就必须支持漏洞管理项目并且遵守配套政策。简而言之，漏洞管理必须成为企业的优先之选。否则，它就不值得去做。

支持业务是漏洞管理项目的本质。它包含了所有用于规定、设计、部署和操作漏洞管理功能的活动、技术和人才。漏洞管理从一开始就是有专门的章程进行支持。该章程的功能就像漏洞管理计划的宪法一样，所有的活动、策略、流程和计划，都应遵循它。它规定了活动管理的原则。当对策略、流程或者组织产生疑问时，可以查阅该章程以确认所做出的决定是否与企业业务保持一致。

该章程不是烦琐冗长的文件，而是反映公司理念、目标和发展重点的一些精炼的语句，并将体现在公司的漏洞管理功能中。例如，如果一个公司因收益来源的缘故而特别重视运算服务的可用性，那么章程中就会包含一个禁止干扰生产电脑正常运作的声明。如果企业更关注机密信息的安全问题，那么关于检测和清除对机密信息的威胁的声明就会被放在章程中第一条的位置。在后面的一个例子中，公司可能会为修复能够导致数据被窃取的漏洞操作赋予更高的优先级。例如，弱密码和间谍软件问题会优先于 DNS 配置中的拒绝服务漏洞得到解决。这并不是说 DNS 配置的缺陷是不重要的；它只是不如客户数据和知识产权的被窃风险那样重要。

在制定策略、流程和组织结构的过程中，新信息被发掘并作为反馈信息提供给总体方案设计。这种反馈机制会对组织结构或策略产生一定的影响。图 3-1 展现了程序开发周期中各个阶段间的关系：

- 概念和建议：该阶段定义了提供给企业的业务价值，漏洞管理的基本概念，以及更高层面上，计划如何实现目标。这主要是项目经理的责任。
- 制定章程：该阶段要制定章程。章程将详细深入地写明项目的指导原则和方案目标。章程由项目经理和 / 或执行负责人来撰写。
- 策略：这里指支持基本企业宗旨的政策，包括了任何可能存在的道德守则。
- 组织结构：一个组织或几个组织以松耦合的方式（loosely coupled fashion）结合在一起协同支持漏洞管理系统。
- 流程：是为了支持漏洞管理系统所制定的细化的、必须遵循的流程。

一位有能力的项目经理能够控制每个组件的质量，并有效调节反馈流程，使得总体方案保持正常运转。值得注意的是，项目章程一般不会根据从其他阶段得到的反馈信息而轻易变更。项目章程只有在发现缺少做出某重大决定所需的解释性条款时才会进行修改。另外，在章程的制定过程中，如果公司的业务模型或道德规范发生了变化，那么项目章程也有可能修改。

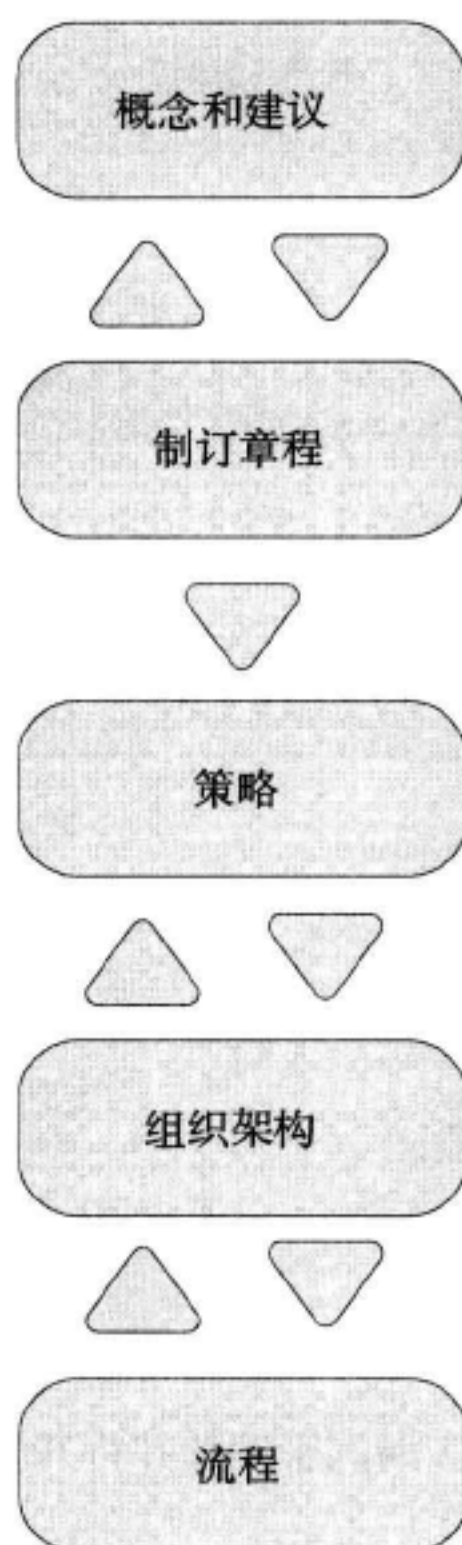


图 3-1 开发周期中的项目阶段

从图 3-1 中我们能够看到一个清晰的发展层次。但是，没有哪一个阶段是能一劳永逸的，它们都需要持续的维护。主要框架一旦完成一般不会再大肆改动，不过某些内容也会有一些轻微但必要的调整。正是这种动态性成就了项目的成功。每一阶段的灵活性都通过来自后一阶段的反馈得到改善。漏洞管理的一个特殊的阶段是，发现一个复杂且困难的变更管理流程并不能适应某个特定的政策。可以对该策略进行修改，使其仍然满足漏洞管理章程制定的目标，但是允许那些不合常规但能更好地兼容现有变更管理流程的流程。

3.2 漏洞管理计划和技术开发

漏洞管理项目中同样也包括主要的技术开发部分。这一开发过程同样可以套用前面提到的开发周期的各个阶段。如图 3-2 所示，在图 3-1 基础上加入技术开发流程，可以看到一组并列的流程活动。

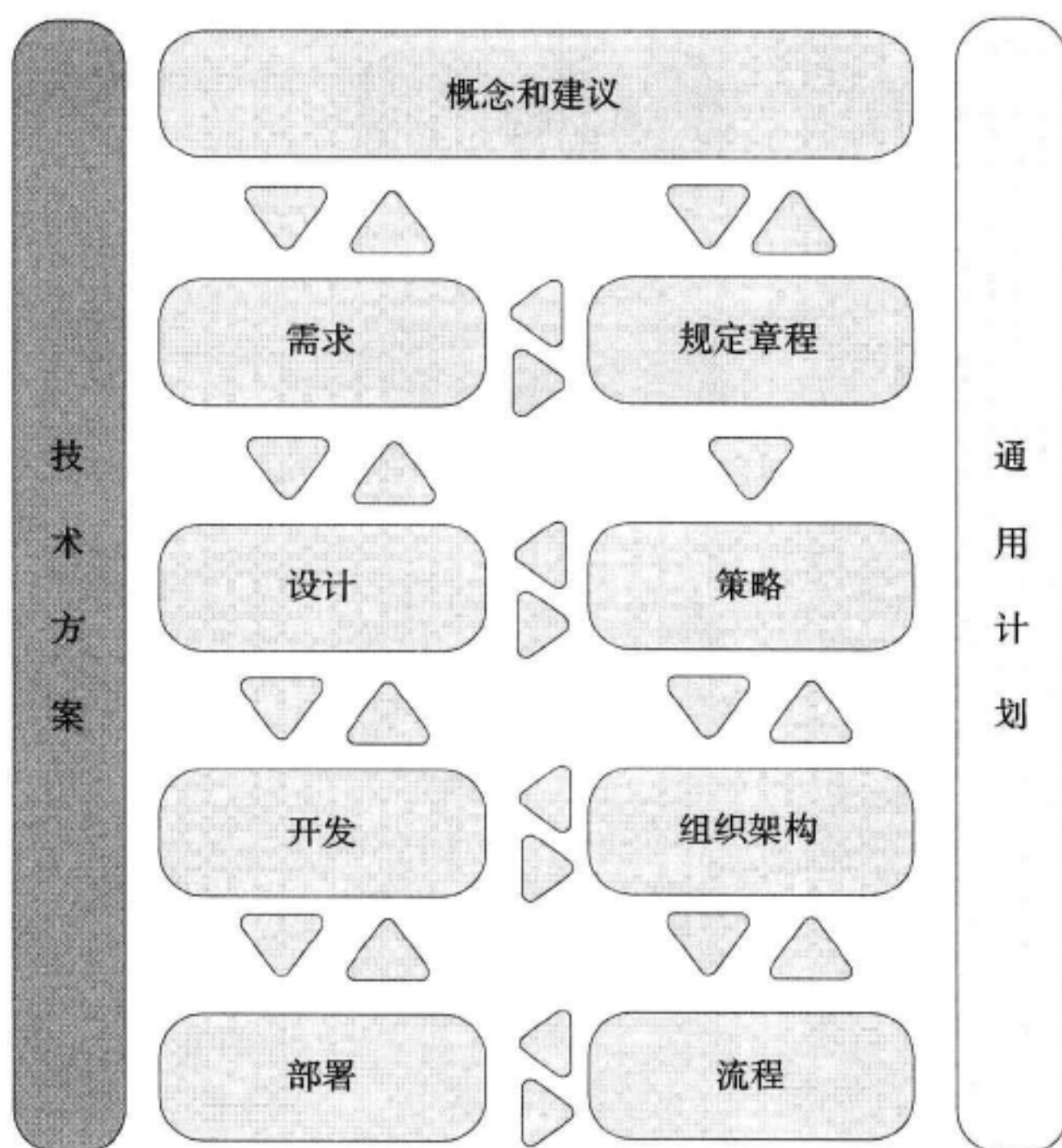


图 3-2 漏洞管理和并行开发流程

当技术开发与该项目的组织和计划上的各发展阶段并行开展时，反馈信息必须向上、向毗邻的左右以及向下传递。毗邻是指策略开发需要通知工程师该如何设计系统。或者，变更系统设计以提供简化流程的依据。在 2.5.2 节我们看到，系统集成极大简化了事件和变更管理流程。向下是指一个细微的策略变动可能因为去除了不必要的冗余的内部审计功能而使得系统编码变得更简单。例如，某审计功能可能为了检测漏洞而要求每次扫描都跟踪记录系统的每个操作。这一策略非常糟糕，因为这种频繁的记录行为带来的海量审计信息将压垮任何扫描软件、硬件及其支撑网络，这些审计信息的数量也许等于但更可能远远超过实际发现的漏洞信息的数量。如果仅把漏洞结果数据记录为审核信息，将更加高效。

向上越级反馈是很重要的，但经常被忽视。例如，漏洞管理计划的开发可能引起策略中

的一些不一致，可以将这些不一致的地方反馈到以前的策略开发阶段。例如，在漏洞管理计划开发过程中，可能会发现在已扫描过的类型的系统中，没有发现某个特定的漏洞。因此，一项要求对所有系统进行 UNIX shell 漏洞扫描的政策并不适用于只使用微软应用程序的商业系统中。这就需要修改策略，添加类似“在适当的地方”等说明语句。

技术开发程序的反馈会为并行开展的组织项目提供信息。例如，漏洞管理系统技术中所发现的某一特征可能会对入侵检测主动防御性能有所帮助。所以，需要增强入侵防范的策略、技术和相关的流程。在早期规划阶段就实现该功能，会自然地将漏洞管理系统集成到组织架构中的其他部分，同时也能分析需要将技术方案与入侵防御系统（IPS）集成的位置。但是，并非所有好主意都能在开发前想到。技术项目负责人的任务就是与小组成员一起，确定所发现的问题是否确实需要额外的工作来解决。

在图 3-2 中，你能看到项目的技术开发贯穿整个计划流程。在各个方向上都应当试试，思考一下一个阶段可能会影响另一个阶段的情况。

3.3 参与者

前面已经讨论了高级管理者的支持对漏洞管理计划自顶向下贯彻的重要性。但是，还有其他一些参与者也承担着非常重要的角色，不容忽视。对他们角色的清晰定义，有助于防止纷争、促使开发过程合理、便于技术部署、激励个人及团队的漏洞管理工作分派。

漏洞管理计划的参与者基本分为两类。一类是贡献角色，他们协助计划的启动和运行。这些参与者不直接参与执行漏洞评估，但没有他们的帮助，评估将无法执行。另一类是操作者角色。这些参与者是日复一日直接执行漏洞管理技术的人员。他们执行扫描、漏洞评估、优先级与其组别相匹配。他们还确保漏洞管理技术在一个动态变化的环境中能够保持功能的持续优化。

漏洞管理流程中的一些关键组包括资产拥有者、安全负责人、人力资源负责人、IT 负责人、漏洞管理员、事件管理员、变更管理员，以及合规管理员。这些角色要么直接被包含在漏洞管理流程中，要么至少受它的重要影响。

3.3.1 操作者角色

即将开始运转的漏洞管理计划所包括的其他角色，要么直接要么间接的参与，都为该计

划的有效性做出了巨大贡献。开发的早期阶段就需要确定这些角色，获得硬件和软件后，要进行更具体的修改。这是因为，选择的技术将影响人们工作的方式，与其他团队成员沟通的方式，以及他们之间的关系。如果一个自动化的流程实现了某个角色的关键活动，那么对该角色的需求将大大减少。

例如，起初按计划，应该有一位管理员来发现关键漏洞，并发布修复请求给相应的系统所有者或管理员。但是，随后选择的技术能自动进行该流程，于是最终只需要一个检查员就可以了。

1. 漏洞管理员

该角色负责确保正确的配置和技术操作，以及创建、监控及发布需要的报告。这绝不是一个简单的管理员角色。该角色必须能读懂系统产生的技术报告，并能解释漏洞产生的原因和修复方法，需要有操作系统、网络和安全实践方面的知识。这一角色将与系统管理员和网络管理员打交道，确保漏洞检测和修复过程达标。

2. 事件管理员

当出现高危漏洞时，必须有人承担修复责任。通常这个人就是受攻击的目标主机的所有者或管理员。这个人应该对目标主机的配置和操作有深入的了解，能对变更系统造成的影响进行评估。这个人，被称作事件管理员，将与漏洞管理员一起工作，共同完成必需的修复任务。事件管理员的责任是追踪分配的修复任务直至任务完成。有时候，该角色会与变更管理员的职能合并。例如，小公司可能派一个人同时完成工程师和管理员的工作。这个人需要负责接收事件、协调变更并分配修复工作。

3. 变更管理员

在一个复杂的修复情况中，多个系统或业务功能可能受到复杂变更的影响，变更管理员需要像项目经理一样监督整个变更过程。该管理员需要通知受影响的各方、协调行动、执行测试或确保完成了适当的测试，以及同漏洞管理员一起验证结果是否合规。

4. 合规管理员

该角色通常是一个接受者、一个漏洞管理系统的最终用户，或者是一个主要受益人。常见的合规检测活动是，合规管理者要确保公司使用的系统符合策略和标准。该管理员通常是漏洞管理系统报告的接受者或顾客。更重要的是，在一个动态的环境中，合规管理员将审核趋势报告，判断系统中是否重复出现导致不合规结果的持续或重复活动。这使得合规管理员

能够发现该组织流程中重复导致偏离策略的流程。

在一个采用服务品质协议 (Service Level Agreement, SLA) 而建立的服务等级环境中, 漏洞管理的计划管理员可能会为合规管理员创建一个 SLA, 以确保按所需的频率进行审计, 对每个目标持续进行适当的核查。通过漏洞扫描报告, 就能进行简单且容易的度量。

3.3.2 贡献者角色

最常见的对漏洞管理流程有贡献的角色群体是: 资产所有者、人力资源、IT 及安全部门。你可能有点奇怪, 认为安全部门应该是操作者角色, 而不是贡献者角色。虽然安全部门是系统主要的操作者, 但我们站在更高、更抽象层面可以将安全部门当做一个客户, 他们为漏洞管理流程的需求做出了贡献。

1. 资产所有者

资产所有者是那些最终买单并获得最大利益的人。他们掌管着财务的支付, 因此在该做什么上有相当重要的话语权。在很多公司, 资产所有者就是产品购买者, 常常发生账单拒付或直接购买的情况。这在中、高层管理者中体现得最为明显。

一般的 IT 工作人员很自然地把他们管理的系统当做是自己的。这种物主身份感不是实际可见的, 但却是一种确实存在的感情依恋。在一个大公司中, 当需要制订计划对资产的安全性进行评估时, 取得资产管理者的合作将能产生更好的成效。在进行资产关键的安全性评估时, 收起对资产的个人情感将有助于提高评估结果的客观性。资产所有者有两个非常重要的贡献: 资产分级 (asset classification) 和定价 (valuation) 职能, 这两件事不能也不应该由系统管理员来执行。关于这点, 将在介绍漏洞管理计划的规划和执行时展开讨论。

2. 安全部门

安全部门工作人员通常是直接与漏洞管理打交道的人。但是, 对于特别关注服务管理 (即 ITIL 服务管理框架中描述的服务管理) 的公司, 可能只是将漏洞管理作为服务管理的一个部分。不论哪种情况, 安全部门和 IT 部门都应当保持密切合作的关系。这种关系将使漏洞管理实现起来更容易并能获得更好的内部支持。

由于安全是漏洞管理系统的终极目标, 因此安全部门自然是漏洞管理计划关键的参与者, 并很有可能是漏洞管理程序完全的所有者和操作者。但是, 根据业务的类型, 也可能是别的团队 (如合规团队) 承担这个角色。例如, 特别倚重支付卡行业 (PCI) 标准合规的公司可能

希望合规部门获得漏洞管理的所有权，而将安全部门作为顾客和关键成员，与之密切合作。

3. 人力资源部门

人力资源部门（HR）是最容易被忽视的部门之一。漏洞管理系统常常发现关键的合规问题，这些问题有时甚至可能成为员工导致的安全事故的证据。HR 是报告流程中的很重要的角色，并在安全策略中起着“指挥棒”的作用。最终，HR 帮助处理员工的所作所为给公司造成的风险。当需要提起诉讼而不仅仅是进行打补丁和配置管理时，开发出来的任何报告流程都要考虑同 HR 的关系。

HR 也被包括在创建和维护绩效管理计划的工作中。经过仔细的规划，可以将漏洞修复的绩效同员工的绩效目标挂钩。为了实现这一点，可能需要让 HR 对漏洞管理程序和支撑系统的工作模式有一个清晰的理解。然后 HR 就能与漏洞管理程序管理者协同工作，一起决定在管理员工时可能遇到的冲突调解过程中，各自应当承担什么样的角色。

4. IT 部门

很显然，信息技术同技术和流程密切相关。如果你作为安全管理或合规小组的成员单独工作，建议找一位 IT 项目经理做搭档，来实现技术部署。一个高层 IT 管理者对系统和网络修复也能提供很大帮助。漏洞管理程序管理者应该与高层 IT 管理者合作，共同开发流程、任命监管工作的负责人。你很可能需要从管理者那里得到一些最初的指示，然后提出流程方案。切记提供一张流程图。IT 人员能很好地在图表配合下工作，一般都很擅长分析已有设计。

3.4 策略和信息流

任何合规或安全倡议都必须有策略支撑。这并不是有关策略的效用或创建过程的问题，而是策略的定义。我们特别关注为了完成从业务目标到漏洞修复的一系列文档链而需要的各类策略。

3.4.1 现行策略

现有的安全策略应当充分覆盖实际操作中存在的软件和可接受使用未授权的现象。漏洞管理系统很可能会发现这种类似的违规现象将不止一次的出现。如果安全策略规定了违规检测必须采用的方法，就有必要对这些策略进行一些修改。类似于前一章节漏洞管理计划和组织中的例子，操作经验将以一种自适应的、循环的方式反馈给策略。如果“可接收使用”具

有优先权，则需要确保有一个支撑过程来应对新发现。

此外，漏洞管理系统自身也可被看做是一个策略的违反。请确认漏洞管理系统的使用符合授权的 IT 或安全系统的任何策略规定。例如，考虑这样一种情况，漏洞管理系统发现某个员工严重违反了公司策略，于是解聘了这个员工。而该员工的律师紧接着发现，漏洞管理系统自身就违反了公司的策略，那么公司将可能会在司法审判中被判处严重的责任。

3.4.2 新策略

有时为了强制实施修复活动，漏洞管理合规策略是必需的。在公司里，指导 IT 管理者规范化的提升漏洞修复优先级的政策是很有用的。策略应当提供如下信息：

- 漏洞优先级：发现的漏洞需要标注优先级。在很多情况下，一段时间内所发现的漏洞要比能被修复的漏洞多。因此需指定哪些漏洞应优先被修复。甚至需要制定一条策略，声明在哪种情况下，系统管理员应该立刻终止手头正在做的所有事情，转而去修复或关闭有问题的系统。
- 资产评估：每个系统都是公司的一项资产，应当被估价，以作为标注优先级的依据。我们将在第 6 章中对这一点展开更多的讨论。
- 时间限制：根据漏洞严重性和漏洞类型，需设置漏洞修复的时间限制。这实际上就是一个公司的 SLA。需要基于多种标准来考虑组织的风险和威胁。这些标准可能是一些支撑标准。

1. 使用策略

另一项重要的策略类型是关于风险管理系统自身使用的。该策略应该将关键性操作的一些限制条件作为重点内容。需要限制的方面如下：

- 免于扫描的系统类型：包括已知会受到扫描带来不利影响的其他安全设备或关键网络设备。
- 获得扫描许可的操作要求：必须取得系统所有者和 / 或管理员的同意才能执行扫描操作。
- SLA 参数：这项要求具体说明了在对特定网络或群体目标进行扫描时需要指定的扫描参数。包括时间、带宽限制、扫描影响评估、扫描终止请求响应时间等。这些参数对

于维持漏洞扫描者与系统所有者之间良好的关系具有重要的作用。如果扫描干扰了系统及其操作环境，系统所有者即使之前允许进行扫描，在执行中也可能会要求停止继续扫描。

2. 所有权和责任

一旦证明系统对于自身在管理企业关键业务的管理具有重大作用，接下来就应该考虑“谁来负责安排扫描时间表”、“谁来决定扫描对象、扫描时间”等问题。由于这些攻击性行为都有可能给系统带来危害，这些问题的提出都是合理的。避免在此类问题上产生争执的最好方法就是事先做好决定。但要注意清晰性，含糊不清是流程最大的敌人。

建立清晰的所有权描述的第一步是将其写入策略中。流程中的关键职能应当在名称上就清晰指定。至少需要如下职能：

- 扫描参数定义：扫描使用的业务和技术参数必须被定义且小心控制。其他人可能会参与扫描过程，而不经意的参数改动可能导致主机或整个网络陷入瘫痪。
- 扫描日程安排：制定扫描日程要经过周密的思考。谨慎对待扫描日程。日程和参数一样，一个小小的改动就可能对业务运行造成重大冲击。
- 报告发布：漏洞报告是机密数据。在心术不正的人手里，这些报告可能带来巨大危害。黑客、别有用心的人或心怀不满的员工，都可能利用漏洞报告制造麻烦。
- 本地主机修复：当一台主机不能通过企业主机管理工具打补丁或修复时，需要通过当地管理员或其他合适的人来修复。
- 远程修复：与本地主机活动相反，远程修复工具通过网络来修复主机。一项修复工作由一个或多个组织负责。例如，PC 小组可能负责为通用主机打补丁，安全小组可能要负责反病毒软件和加密程序的升级。所有这些组织都必须事先明确为 VM 流程开发的积极参与者和贡献者。

3.4.3 合规和统辖

为进行良好的管理，需要有人来监控修复的过程。在一个拥有将近 10 000 台主机的组织中，修复过程的监控者由负责扫描的人来担任就可以了。但是，在一个更大的组织里，最好有一个合规团队来执行这项工作。此外，合规团队应该定期监控系统配置和操作。

图 3-3 展示了合规管理组织如何使用现有的操作文档、过程文档，以及扫描结果，来验证

修复过程是否合规。这三份数据之间存在重要的关系。操作文档告诉合规团队漏洞管理小组所采取的行动。合规小组应当验证漏洞管理活动的执行是否符合策略要求。

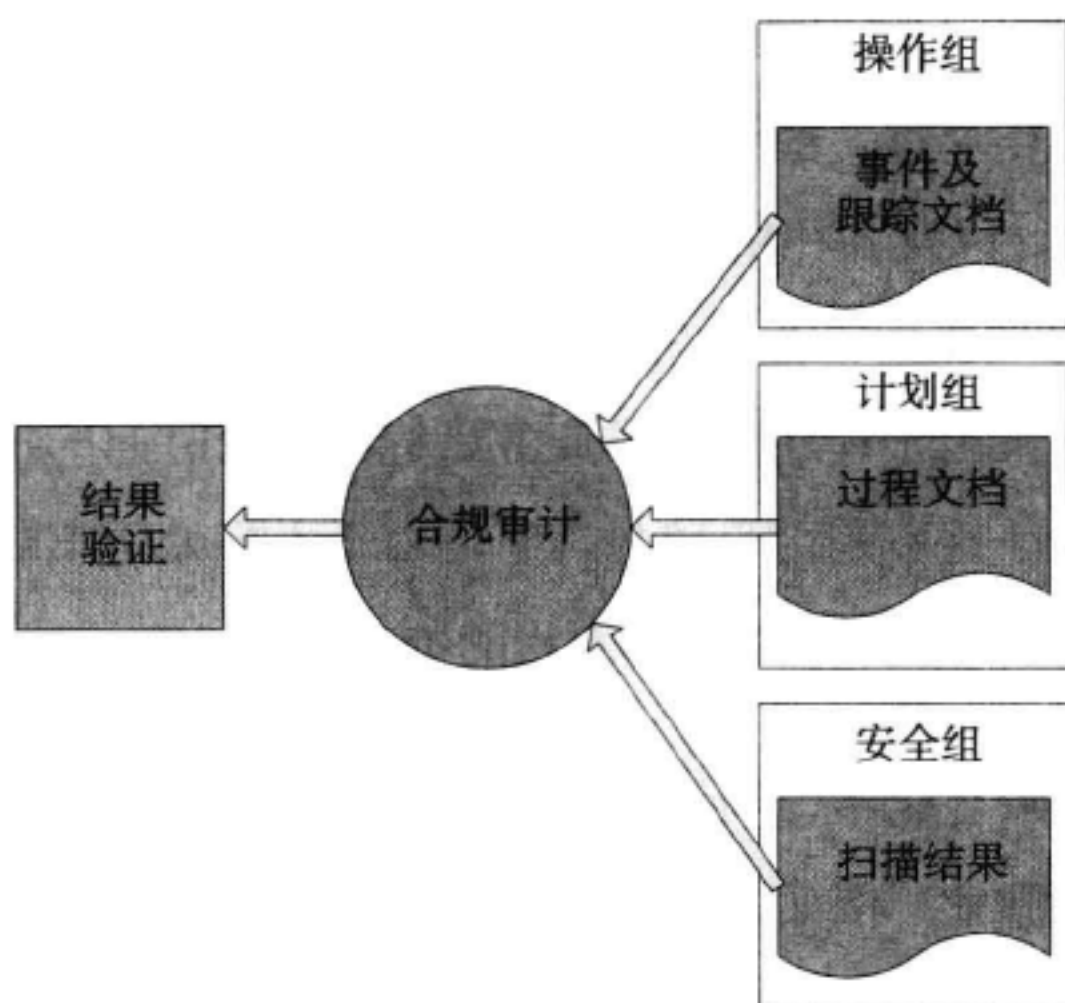


图 3-3 合规数据流

过程文档清晰地定义了执行漏洞管理功能的细节。正是通过该文档，合规功能可以验证操作行为和支持输出文档。过程文档自身也应该进行核查，看是否违反政策，以确保合规。某些合规检测中，该步骤不是必须的，因为合规已成为创建漏洞管理过程的一部分。在此情况下，外部审计有时候能够起到保障作用。

最终，在漏洞扫描报告中产生详尽的扫描结果数据。这些报告反映了每个负责修复的 IT 团队所达到的合规等级。稍后，我们将详细讨论这些报告中的内容以及它们在成熟的、成功的漏洞管理系统中的相关性。

系统审计

漏洞管理过程中，另一个关键的步骤是审计。在安全机制的年度审核中，最好由第三方来核查系统的配置和操作是有利的。任何审计都应该包括如下要素：

- 流程：审计员应当核查扫描、修复、验证这个过程中没有关键错误。审计员应该提供改进建议。
- 范围：理解现有网络分段的结构和应用后，审计员必须验证完整、合适的目标集合已经被扫描。根据程序章程和策略，目标机列表中可以包括供应商和商业伙伴。除了现

有目标之外，很重要的是认识到组织、系统和网络是动态的。环境的改变将导致扫描目标的范围变化。扫描器的流程和配置应当充分适应这一变化的环境。

- 培训操作人员：操作人员必须非常了解系统的技术细节。他们不仅要理解操作过程，还要了解漏洞的工作原理，漏洞带来的威胁，以及威胁可能造成的风险。如果有操作系统、网络、协议，以及各种相关应用的知识则更好。
- 策略调整：漏洞管理操作符合现在的策略吗？如我们在前面讨论过的，漏洞管理流程来源于策略，策略来源于程序章程或业务目标。随着时间的推移，策略可能会变得不再满足程序的需求。这不是由于疏忽，而是由于人们处于不断调整以适应环境变化的自然趋势中，没能全面照顾到对程序章程的影响。

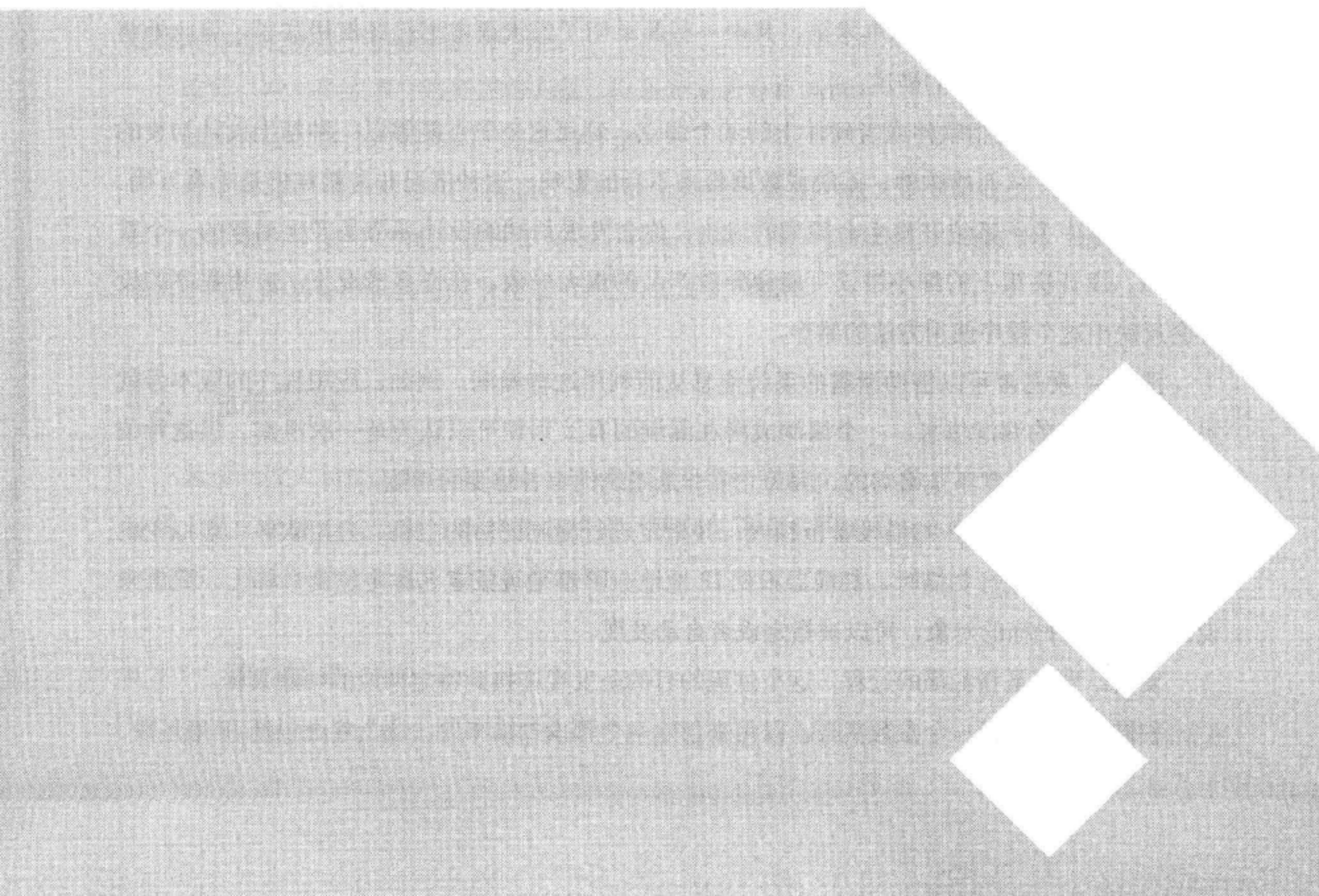
在响应业务环境的不断变化中，网络环境和系统环境也在逐渐地发生变化，于是原先的策略变得不再能反应业务需求。例如，以往产品销售是面对面进行的，因此，没有关于合理加密或处理顾客财务数据的策略。后来，人们开拓了网络市场。现行的策略规定电子支付数据必须通过银行实现转账而不能由公司系统处理。但是，在最近被采纳的在线销售模型中，顾客提供的支付信息是由公司计算机系统处理的。现在，大量漏洞和合规问题出现在加密方面，网络设计和系统配置中。由于策略一直未被修正，难以发现和修复这些系统中的合规问题。所讨论的系统可能超出了漏洞管理的范围。

3.5 小结

在一个庞大且复杂的组织中，漏洞管理程序可能会面临很多挑战。必须克服策略、恐惧以及人们拒绝变更的自然倾向。成功的基本要素是包容性、行政支持、结构、策略和流程。本章概述了很多基本挑战，以及能应对这些挑战的组织结构和流程。要将所有这些因素综合起来，必须进行缜密地规划并获得自顶向下和自底向上的支持。完成这一任务依靠的是四分之三的科学规划和四分之一的说服艺术及激励艺术。倾听客户所关心的内容将有助于在完成所有工作的过程中避免过大的阻力。这是组织策略和实力外柔内刚的一种巧妙结合。

第4章

漏洞管理技术



4.1 简介

针对不同的部署策略形成了多种特色的漏洞评估技术。这些策略主要根据商业运作模式所限定的要求制订。很多因素都会对技术的选择产生重要的影响，例如设备的分布和基础设施的组织。成本和功效则是在做任何决策中必须考虑的因素。技术选择的标准将会在下一章中讨论。本章将讨论各种漏洞管理技术以及它们的特点和存在的问题。

主要包括了行业内普遍接受的五种漏洞管理技术：

- 主动扫描。
- 被动评估。
- 代理。
- 混合方法。
- 推理扫描。

针对不同的商业结构，这些方法在灵活性方面有着细微的差别。在开始深入讨论这些架构之前，我们先回顾一些基本术语，其中一些简单明了的术语本书已经使用过了，但此处将会进行更加精确和完整的描述：

漏洞：漏洞是指软件或者硬件上的某个弱点，利用这个弱点能够以一种超出设计初衷的方式使用产品，从而给软件、系统或数据带来不利的影响。这种情况非常糟糕但是千真万确。查阅第2章中关于系统开发生命周期的讨论，你会发现前期的设计环节是产生漏洞的一个重要来源。设计决策上的细小错误可能会导致产品的重大失败。在许多情况下，应用程序的设计会反映出这个程序使用方法的雏形。

泄露：攻击者可以借助泄露的系统信息从而利用这些漏洞。例如，应用程序的版本号就可能泄露潜在的漏洞信息。一个漏洞或潜在漏洞的有关知识可以认为是一次泄露。但这种信息对于用户来说通常不是必需的，因为它并非是系统所有者想要的功能。

目标：目标是指在扫描设备范围内，并指定进行漏洞评估的主机、应用软件、虚拟主机或者主机集群。进行扫描时，扫描器根据IP地址、IP群地址或者名称定位到目标上。简单来说，目标就是扫描的对象，可以被指定或者自动发现。

发现：发现是指扫描的过程，这个过程的目的是发现连接到特定网段的一些目标。

扫描配置：是指一个参数集合，以用来创建一个预设扫描事件。这个集合包括IP地址段、

端口、日期和时间、带宽限制、证书、执行的特殊检查等。

漏洞检查：如同字面意思一样，是一组事项的特定集合，用于检查目标以发现可能存在的漏洞。

合规性检查：类似于漏洞检查，但不是发现漏洞，而是发现与已经确立的公司策略不一致的行为。例如，如果要求密码至少包含 10 个字符，合规性检查就是要记录下不符合该规则的情况，即少于 10 个字符的情况。

安全态势：基于各种因素影响主机、网络或组织安全状态的网络。

攻击向量：类似于漏洞，攻击向量是攻击者危害系统的一个基本策略。包括利用漏洞或设计缺陷。

公共漏洞与泄露（Common Vulnerabilities and Exposures, CVE）：一个由 MITRE 组织（cve.mitre.org）发布的公开的漏洞通用名称列表。在分析和修复漏洞时作为参考的主流漏洞列表。它不是唯一可用的漏洞列表，而且也并不完备。但它确实为漏洞识别带来了一定程度的标准化。

扫描器：扫描器以软件或硬件的形式存在，用以执行自带或用户定义的检查，扫描配置有时用来指定让哪个扫描器执行检查。

审计：对一个或多个目标进行扫描，以确定是否存在漏洞。

漏洞评估：审计的同义词，通常指评估。

希望读者已经对网络和主流操作系统非常熟悉。而且，在大部分的论述中，我们将会假定目标环境主要使用 IPv4。虽然 IPv6 是一个新兴的标准，但还没有广泛应用。此外，当今市场上还没有漏洞管理工具可以支持 IPv6 主机的发现和扫描。

4.2 总体架构

本节将深入讨论三种不同架构中的组件以及组件间的交互：独立硬件、代理和被动网络分析。因为软件架构与独立硬件十分相似，所以这里我们将不会再讨论软件架构。它们之间主要的区别在于销售者并不供应和维护硬件。

4.2.1 硬件模式

硬件模式具体是指利用内嵌了软件的硬件设备进行漏洞扫描。这些硬件设备通常部署在

整个网络中，并向中心服务器返回扫描报告。扫描器通常是一台完整但相对简单的计算机系统。其典型结构包括一个通用操作系统、支撑软件模块以及完成扫描和通信任务的特定代码。有些供应商会使用开源工具，而有些供应商会使用商业操作系统和组件。

基于硬件的系统的主要优点在于供应商在主机配置方面非常专业。供应商会负责维护和确保这些配置的稳定性。软件如果有问题导致不能像宣传中所说的那样执行，这些问题都应在客户与供应商的合作关系中声明。

在部署中，硬件设备也有不利的地方，即不得不运输到安置地点，并且可能由并不具备相关资格的人员安装。然而，在大部分情况下，部署并不是如此复杂的。如果当地的技术人员能够配置一台典型的主机，那么他就有能力配置一台漏洞扫描器。如果你对当地的有关人员的能力没有信心的话，那么稳妥起见，建议你预先配置好设备并提供简单明了的安装说明。

在大部分设计中，每一个扫描器都会回传报告给中心服务器。这样收集到的漏洞与合规性信息将被传回服务器以供分析和报告。设备也会从网络上接收评估指令。这些指令可能会通过投票、即时连接，或者逆向投票进行传输。这些策略的影响极小，但是非常重要，这取决于用户的网络安全架构。

投票是对与中心服务器相连的扫描器进行投票。每一个扫描器往往通过 TCP 端口用专用认证方法来保证整个会话是加密的。进行投票的设备可能是那些服务器准备让其进行某项工作或者正在处理某项工作的设备。服务器检查其状态来确认是否有数据可用或者是否准备好了接受一项工作。这种方法可能有些麻烦，但是有一点好处就是只需要从服务器发起一个连接。在某些情况下，并非所有的扫描器都需要投票，除非有一项预定的工作直到它要执行的时候才能得知某个扫描器的状态。大部分投票者会选择所有扫描器。图 4-1 就展示了一个简单投票方法。

逆向投票是每台扫描器与服务器的定期联系的过程。如果为扫描器预定了一项工作，那么就会使用逆向投票，并且会使用相同强度的认证和保密手段。扫描器将会把扫描结果在扫描过程中或结束时发送回中心服务器，这取决于软件设计者的设定。这种方法有一个额外的好处就是即使在与服务器连接丢失的情况下也允许扫描器完成本地工作。扫描的结果会被缓存起来直到与服务器重新建立连接。

逆向投票被部署在安全区域时优势明显，在这些区域会将拨入的与扫描器通信的连接视为不良连接以限制可能来自外部的连接。不过当扫描器被部署在组织界限之外时，这一点则将成为其劣势，因为必须对安全基础设施进行一些调整才能允许连接到扫描器。

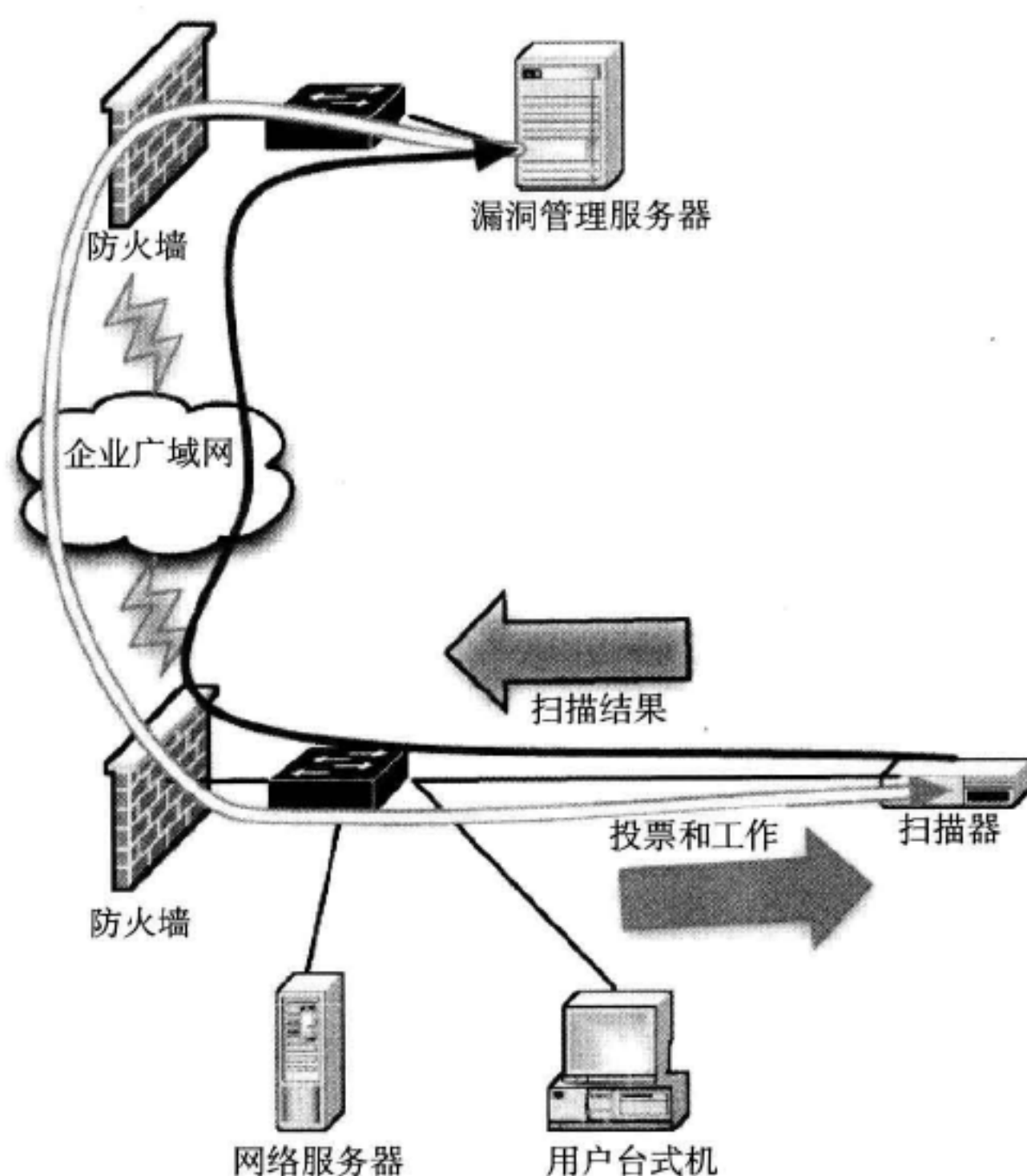


图 4-1 简单投票方法

4.2.2 用户提供的硬件和虚拟化

这很类似于硬件模式，区别就在于需要提供自己的硬件去安装漏洞工具。这在开源产品（如 Nessus）中是很常见的。大部分的供应商不喜欢提供纯软件解决方案，因为客户群很大，而他们的硬件平台也千差万别，要使软件解决方案适应所有这些差异将非常困难。一个控制良好的硬件平台将能提供更好的稳定性和可预测的行为，使供应商能够通过仅提供产品所必需的性能来满足客户的需求。这一点在后面论述漏洞评估工具的操作细节时会更加明显。

对供应商来说，更好的解决方案就是使用虚拟机版本的产品。虚拟机中的硬件系统更加可预测和可管理，因为它是抽象出来的，并且由底层的管理程序控制。可以有效地为产品制定内存大小、CPU、网络连接和产品所用的其他硬件设备的精确规范。而且，操作系统的版本和配置也是由供应商控制，这样可以排除在配置过程中的错误，提供更多的可预测的和可支持的行为。

虚拟机方法还有其他一些主要优点，即在虚拟环境中易于部署、节约能源、没有发货成本，并能更有效地使用已有硬件。

4.3 代理

代理技术从 20 世纪 90 年代初到现在已经取得了很大的进步。几年来，“代理”成为了一个系统工程师心目中的贬义词，因为它们被认为缺乏稳定性并会影响系统的可靠性。自从操作系统扩大了内存和提高了处理能力，代理也随之提高了系统的性能并且不需要人工的直接参与。代理和实物的关键区别在于代理能够接受一组指令或参数，并且据此持续地自动地去实现结果。代理不仅仅是一个工具，还是一个助手。对于系统设计师们的预想来说，漏洞代理的能力还相对简单。

4.3.1 代理架构

代理通常会被赋予充分的系统特权，在系统后台执行服务。除了执行重要任务之外，这些服务通常消耗很少的 CPU 资源。通常，至少有两项服务在任何时间都和其他活动服务一起运行，这取决于产品的架构。漏洞评估代理不可避免地会与目标的审计联系到一起，然而工具可以被用于不止一种审计方法中。

如图 4-2 所示，一个服务在监听网络上的控制服务器发出的配置和评估指令。这个服务或另外的服务有可能用来向服务器发回评估结果。第二个服务是对本地主机的实际漏洞评估，而某些情况下是对网络上临近的主机执行漏洞评估。

基本的代理类型包括：

- ❑ 自治型：这种类型不需要其他系统或是人工进行持续地输入和操作。
- ❑ 自适应：这种类型会根据一些特定的规则对环境的变化作出响应。一些代理会比其他代理更具适应性，这取决于复杂度等级。
- ❑ 分布式：代理并不局限于单一的系统或者网络中。
- ❑ 自升级：有些人认为这并不是代理独有的特征。对漏洞管理来说，这是一个非常重要的功能。代理必须有能力收集最新的漏洞和应用审计功能。

漏洞管理代理是一个软件系统，和主机的内部工作紧密联系，能够识别环境中有可能构成

漏洞的变更，并对其作出响应。漏洞管理代理主要有两个基本功能：第一，它们监控系统软件状态和配置漏洞；第二，代表管理员对邻近系统进行漏洞评估。按照定义，代理以半自动方式运行。代理通过外界的参数设置采取相应的行为，并且一旦开始执行就不再需要进一步的指令。代理并不需要每次都提醒去评估当前机器的状态，甚至没有必要指示代理去审计邻近系统。

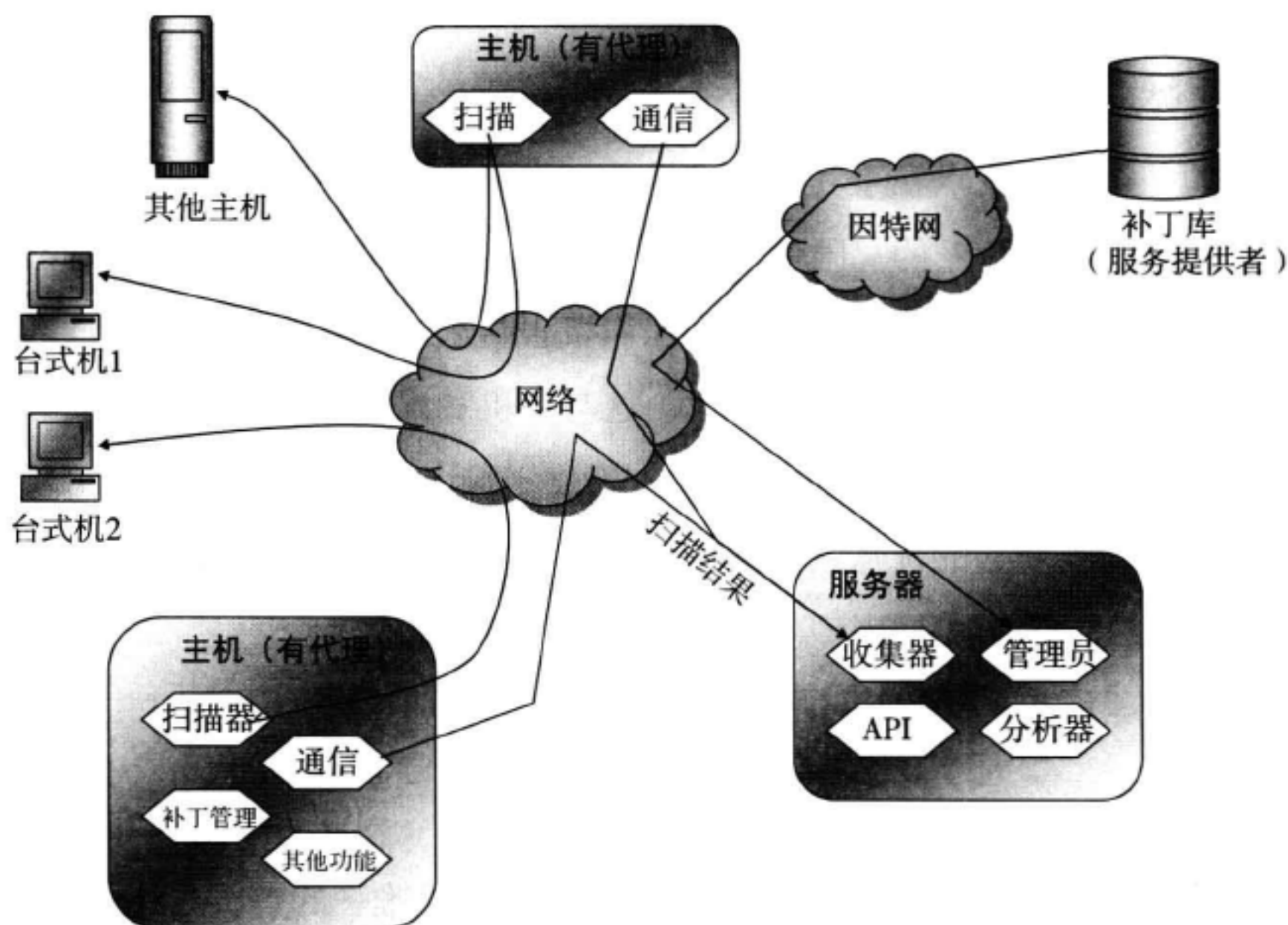


图 4-2 代理架构

与代理不同的是，基于网络的漏洞扫描器则需要向其提供关于实施审计的时机和方式的详细说明。每当发起一个审计时都需要向扫描器提供详细的指示。在设计上，代理对所有的漏洞管理系统都是松耦合的，所以它们可以对单一服务器上的负载及对其的依赖性降到最小。

实现方法包含一个或多个系统的服务，以及一些即时响应的程序完成相应功能，而不需要一个连续的基础。例如，代理需要主机有连续的监督和通信能力以使它可以接受指令，发送结果，并且执行所需要的审计。这些功能只消耗极少的内存和处理器的时间。

在需要的时候调用专业程序以执行 CPU 消耗大的活动，例如本地或者远程的网络审计。这些程序实际上在网络漏洞扫描中执行大部分功能。一旦完成，收集到的信息就会传输给监督服务，然后回传给中心报告和管理服务器。

本地主机漏洞的检测有时是在特定的时间窗口内进行，以一个单一的、已设定好的过程对目标主机所有配置项目进行审计。另一个替代方法就是持续监控当前机器的配置状态。当

出现一个变化时，漏洞评估软件就会对这一变化是否构成漏洞进行评估，并立刻向管理服务器报告变化。这种功能在当今快速发展的终端安全市场中是错综复杂的。配置变化的探测和新增的应用安全策略的功能模糊了终端保护、统一配置和漏洞审计之间的关系。这个组合最后将会带来更加紧密和敏感的安全保证。

4.3.2 优点与缺点

使用代理的一个显著优点在于其分布式特性所带来的可扩展性。由于所能部署的代理数量仅受限于可兼容的主机数量的局限性和许可成本，所以理论上讲，可以对每台机器执行审计而不产生任何网络活动，除非有配置代理和报告结果。虽然审计并不通过网络执行，但是代理和服务器的通信量通常不能最小化，这与主机和漏洞的复杂性，以及可能产生的相当大的报告通信量有关。然而，扫描并不发生在网络链接上。

使用代理的一些很明显的优点就是几乎不涉及部署附加的硬件，并且也几乎不用考虑是否有足够的带宽和扫描器资源可用。

然而，使用代理可能遇到以下问题：

- ❑ 它们会和目标机上运行的其他应用软件发生冲突。对于如今运行在复杂计算机系统上的所有软件来说，这是个非常常见的问题。测试是唯一的解决方法。
- ❑ 它们在本地安全策略上可能没有足够的安全权限来审计每一个配置项目。
- ❑ 可能会有一些错误导致中断，而故障通知没有及时反馈给管理服务器，而在这段时间内，可能会错失审计。
- ❑ 对于操作系统生产者和正在使用的版本来说，代理可能无法使用。几乎所有制造商都生产能够适用于微软 Windows 的代理，但是很少能支持 Linux、FreeBSD，或者 Solaris。
- ❑ 嵌入式系统（例如取款机和其他销售终端设备）是个封闭系统，很难和代理相集成。然而，支付卡行业安全标准则要求对这些系统的文件完整性进行监测。
- ❑ 如果代理只有有限的大小、空间和运行能力，它就不能拥有几千余种可能的漏洞。
- ❑ 在虚拟机上，可以有多个代理同时运行，但这样会对基础硬件和主机操作系统的性能造成不利的影响。
- ❑ 代理自身也可能会因为漏洞而成为攻击者的目标。由于代理要从网络上监听来自服务器的指令信息，因此其开放的端口就有可能被利用。

漏洞审计代理与其他方法相比有很多优点：

- 它能发现所有的漏洞，其中某些漏洞在网络上是无法发现的，除非扫描器经过了认证。
- 代理甚至可以在系统没有联网的时候运行。
- 在它检测漏洞时，代理并不主动要求安装在系统上的其他软件参与，因此最大程度降低了对检测漏洞的干扰。
- 因为它不通过网络进行操作，所以既不会引起入侵防御系统的注意，也不会产生过多的网络通信量。事实上，代理产生的总通信负载比典型的 Web 上网活动产生的要小得多。
- 就像本地运行的软件一样，它可以把功能拓展成更活跃的终端安全功能。

4.3.3 检测方法

代理可以全面观察到主机内部的工作。系统一发生改变，它们能立即探知。虽然并不总是用这样的方法实现，但是这样做却能更好地分享终端安全代理的功能。

为发现漏洞，文件校验和、注册表内容和配置文件都要进行分析。既然主机类型都是代理所熟知的，那么就能事先知道必须要检测哪些特定漏洞。代理像系统进程一样运行，它对所有文件甚至于必要的内存空间都有访问权限，从而能在变化发生时立刻做出准确的评估。需要继续精确探测时只要从中心服务器发送更新给代理即可。网络扫描方法可能需要更多的时间来探测到变化，因为它们不能持续不断地对单个主机进行探测。

某些代理也有能力对于网上其他目标主机执行基于网络的主动扫描检测。大部分配置计划只允许扫描同一物理网络上相邻的系统。

4.4 被动网络分析

被动网络分析指在网络交换机上安装设备来监听和分析通信流量副本以发现漏洞。这很类似于入侵检测系统（Intrusion Detection System，IDS）或者嗅探器的功能。用一台有网络端口的硬件设备连接到网络交换机上来检查其通信流量。网络交换机将通信流量发送到此网络端口以进行分析，也可以用网络分流器来检查一个物理网络连接的流量。这个物理网络连接可能会包含大量来自多个网络的数据流量。

分析人员可以检查一些可以暴露漏洞的东西，如 IP 地址、网络、应用程序协议以及通用通信模式等，以发现异常或者暴露可利用缺陷的一些属性。

表 4-1 比较了应用网络分流器或者端口镜像的被动漏洞扫描与漏洞扫描器的不同。

我们可以注意到主动扫描器可访问没有在网络上发现的信息，然而被动扫描器有权访问那些主动扫描器没有扫描的信息。

表 4-1 主动和被动扫描的比较

网络通信流量类型	主动扫描器	被动分析器
ARP	来自于单个 VLAN	来自于多重 VLAN，包括远程结点
目标的 TCP/IP	来自于主动扫描目标	来自于多重目标，任何会话都在被监控的 VLAN
VLAN 标签	来自于连接的 VLAN	来自于多重 VLAN
可观测的协议	只有在参数中指定扫描的	主机所使用的任意和所有协议
应用程序发现	扫描器所知道要查找的，包括不适用网络的应用程序	任意使用网络连接的应用程序

端口镜像，思科公司有时也称其为交换端口分析器（Switched Port Analyzer，SPAN），是在现代网络交换机中普遍使用而有效的技术。图 4-3 描述了 SPAN 的工作原理。这是一个基本的 SPAN 配置，几个成对的 VLAN 的内容被复制到交换机的物理端口上。网络管理员可以在入口通信量、出口通信量或者入口和出口通信量三者之间进行选择；通常，两种通信量都是分析者想要的，因为这样就可以看到两端的会话情况。SPAN 的功能根据安装在交换机上的模式、品牌和特征的不同，其复杂性和局限性也会有所不同。一些简单的交换机只能复制通过物理端口进入的通信量，不必关闭交换机的背板。某些交换机可以看到单一的 VLAN 的流量，而其他可以看到骨干 VLAN。

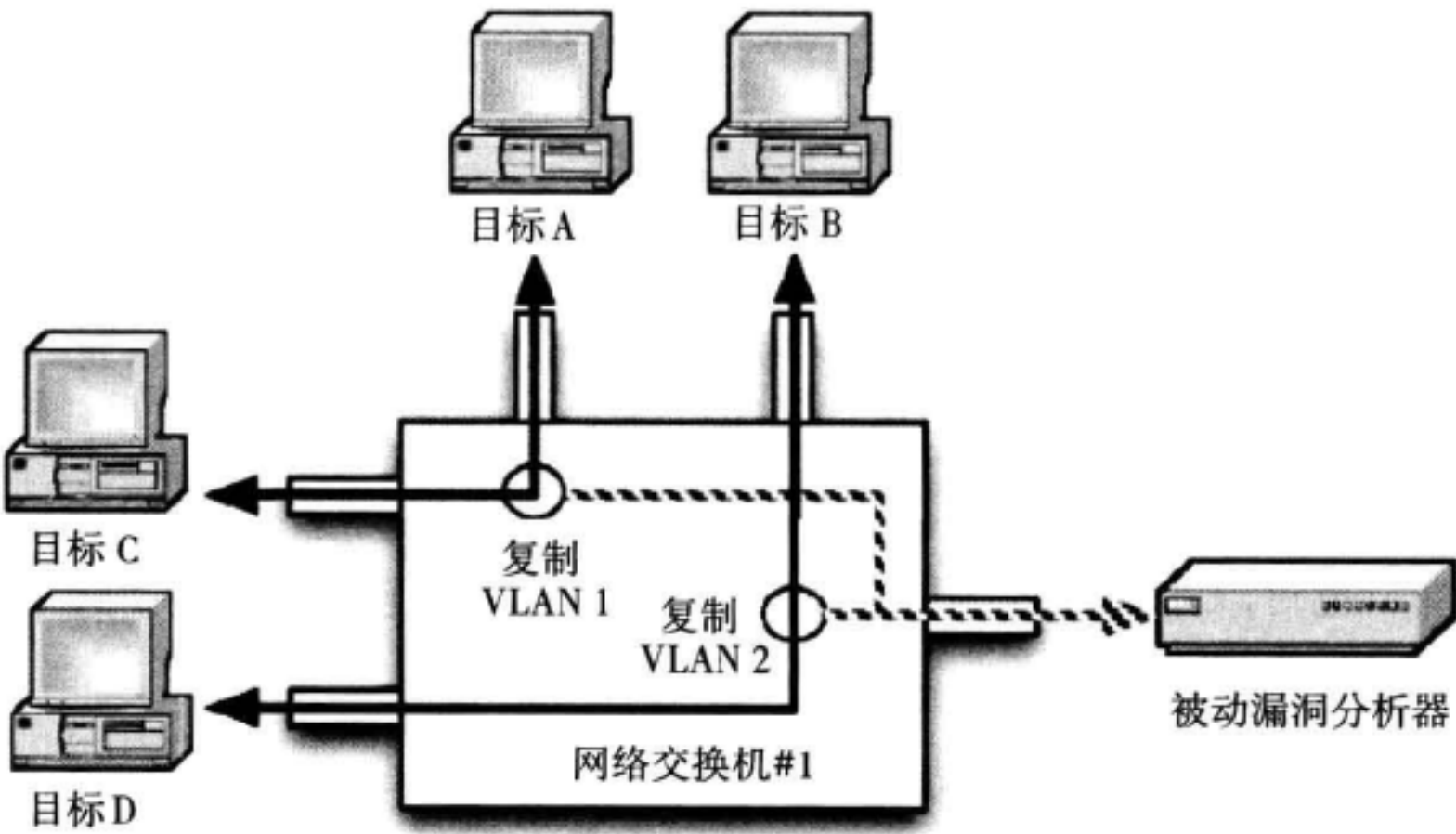


图 4-3 基本 SPAN 配置：将一对 VLAN 的内容复制到交换机上的物理端口

很有趣的是，分析器似乎必须连接到物理交换机上，此交换机传送需要分析的数据流量。不过，有一种 SPAN 的改进方法，可以在一定程度上解决这一问题。有些交换可设定远程 SPAN（RSPAN）模式，允许远程交换机上传来的 SPAN 结果被发送到另一台分析者能够连接到交换机上。此时，有些 SPAN 的能力可能显得异常。网络管理员需要认真评估需求，选择最有效的方式为分析器提供适当的信息。

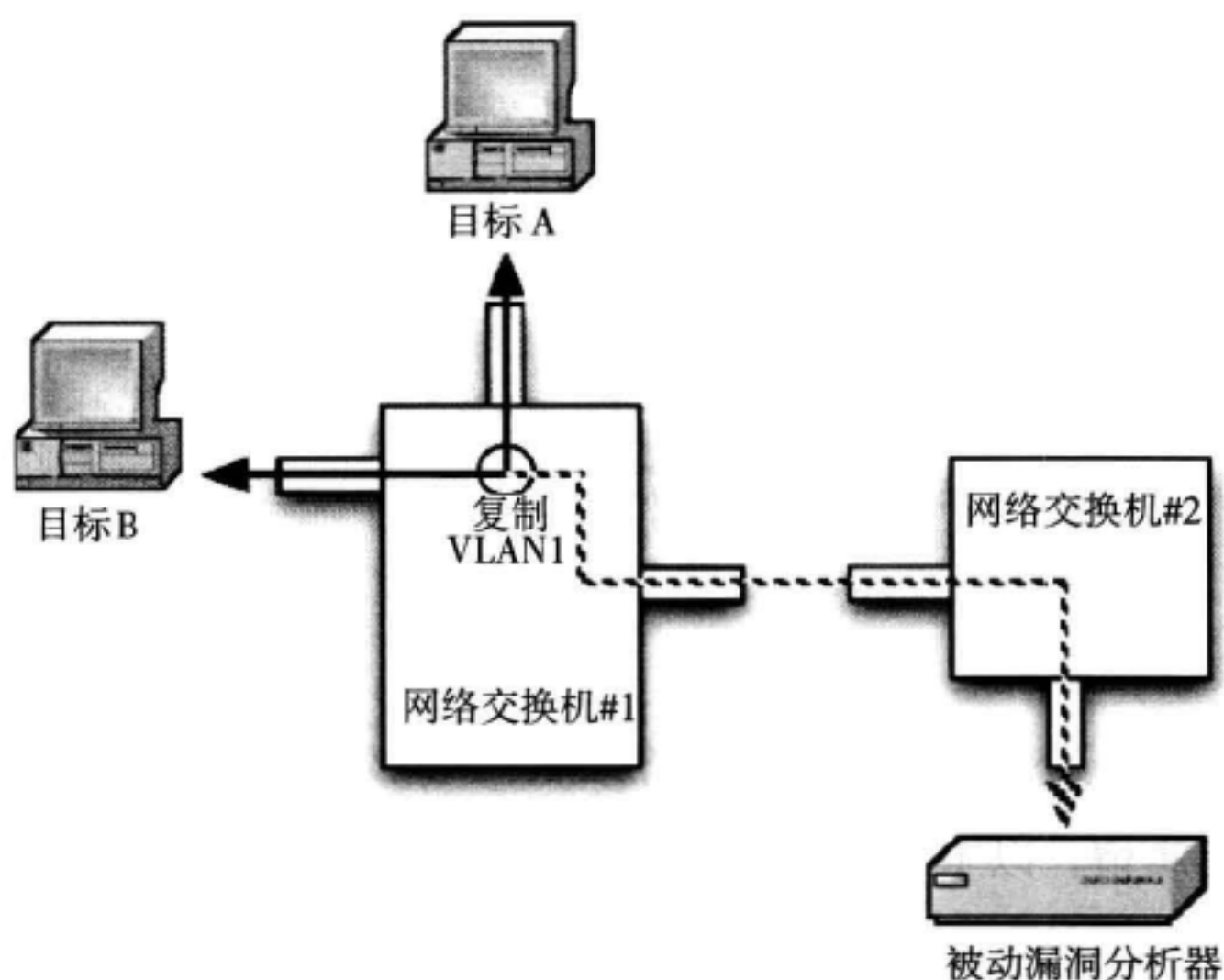


图 4-4 RSPAN 实现目标 A 和目标 B 在远程交换机上的监控

通常来说，被复制的流量称作是“洪泛”（flooded），被传送到两台或多台交换机之间共享的特殊 VLAN 上。在思科的产品中，这种方法催生了一种创新的 RSPAN VLAN。这是一种特殊的 VLAN，其交换机功能专门为远程监控设计。有了这种技术，就可以用多个地方的多个设备进行漏洞评估。

同时，也可以将该 RSPAN VLAN 加入到其远程交换机在 100 英里之外的 WAN 配置中。这是一种非常规的配置，伴随有带宽风险。这会导致漏洞扫描被动方法的不利情况。如果你使用 SPAN 技术来对远程目标进行漏洞评估会很不合算。被动漏洞分析是非常昂贵的。要对远程的 20 ~ 30 个在 100Mbps 甚至 1000Mbps 的带宽上通信的目标进行监控是非常困难和昂贵的，因为你必须要提供足够的硬件来分析这么庞大的通信量。因为不太可能有一个 1Gbps 的 WAN 连接专门用于监听，而且采购一整套设备来安装在本地是不现实的，所以被动设备的使用并不总是很理想。

SPAN 和 RSPAN 都可能会出现问题，这些问题必须由有资质的网络管理员评估。分析器所连接的监控端口可能会超负荷。也就是说，可能会有比这个端口所能承受的更多的流量到来。大部分流量会保存在与被监控网络共享的缓冲区里。如果这个缓冲区满了，与 SPAN 操作所涉及的端口的通信速度将会变慢。这在下面的情形中会体现的非常明显：如果分析器连接到一个百兆端口，而且监控着其他 4 个物理网络的端口，每个端口的带宽都超过 40Mbps，那么总监控量将达到 160Mbps 以上。这就意味着交换机必须保留 60Mbps 的附加流量直到发送给被动分析器端口。为了避免这种情况发生，有必要对每个目标和监控端口的峰值流量进行评估。如果既有的 IDS/IPS 实现此功能，那么就可以共享这些 SPAN 端口来节约成本。

SPAN 的一个替代方法就是分流器。其功能就和它的名称一样，一个安装在网络连接上的物理设备，它允许被动分析器查看通信量。分流器在以太网中是用电式的，在光纤通行中是光学式的。以太网分流器稍稍复杂，因为它需要供电支持。有些分流器甚至会有内置电池来保证分流器在电源不能供应时继续保持正常运行。光学分流器并不需要任何电力，但是需要一个如同分光镜一样的棱镜。

分流器在双工管理上处于劣势。因为现今多数网络都是同时发送和接收数据的，分析器必须要能够做同样的事情。以 100Mbps 的以太网为例，一根连接到分析器的单一电缆要么只能监听监控目标之间发送的流量，要么只能监听接收的流量。在两个目标之间，发送和接收流量最高都可以达到 100Mbps。所以，总量将会达到 200Mbps，这超过了单一分析器端口连接的能力。这个问题只能靠分流器把对话的发出与接收的信息分别用两根单独的电缆连接到分析器来解决。分析器再把这两边的会话重组来精确地加以分析。

4.4.1 优点与缺点

被动分析方法有以下优点：

- ❑ 分析器不需要与网络交互就能发现主机以及与相关漏洞。只有用户使用软件取得报告的接口是活动的。
- ❑ 几乎不需要任何测试来确认是否对网络或者主机有负面影响。因为技术上是完全被动的，所以几乎不需要任何验证。甚至于当设备发生物理故障，也不需要联网检查。
- ❑ 有时，设备可以与一个已经存在的入侵检测系统一起串行安装。这样不用对网络交换机做任何改变，大大简化了它的实现。

- 整个发现程序是持续进行的。新主机一旦被连接上网络并开始通信就会被发现。与主动扫描和代理相反，只有新一轮的扫描之前方能发现新的漏洞。
- 不监听网络上主动探测开关的隐匿主机可以被发现。反之，这些主机只通过网络上发起对话时进行在通信，因此可以被动检测到。
- 因为路由协议和其他网络信息对通信量分析器来说也是可见的，因此它可以得知网络的拓扑信息，并利用这些信息来了解更为复杂的网络上的攻击情况。这种信息同样可以被已授权的主动扫描器获得，以及通过向专业化工具提供配置数据获得。

被动分析技术也有一些缺点：

- 设备必须被安装在交换机上，交换机传送被监控的数据流量。远程网络监控对于一个繁忙的 WAN 连接来说往往很不实用。这样将会限制能够扫描的地点数。如果要监控一个很大规模的网络，那么被动分析技术不是正确的选择。
- 把交换机流量复制到物理设备的机制可能会给交换机 CPU 带来额外的负载。这些额外的负载将会降低路由选择、访问控制或其他 CPU 操作的性能。
- 被动分析对于漏洞的检测是非常有限的。大部分漏洞可以被主机代理或者经过认证的主动网络扫描器检测到，但却无法通过网络扫描分析网络流量来探测到。

总的来说，被动分析可能难以发现所有的系统漏洞，但是他们却能每天 24 小时运行并提供通过其他方式难以获得的网络拓扑信息。如果那些漏洞使用过网络，那么网络 and 主机环境的改变将会在第一时间被被动分析方法探测到。

4.4.2 检测方法

被动分析探测漏洞完全依赖于对 OSI 模型每一层通信内容进行剖析和解释的能力。

4.4.3 物理层

在物理网络层上基本上不使用被动技术来检测漏洞。物理连接的终端在网络接口适配器的硬件平台上，该硬件平台部署有被动分析软件。这些硬件常常只能提供非常有限的关于物理连接状态的信息。

4.4.4 数据链路层

这一层仅仅在漏洞扫描器以非混杂模式连接到网络的时候才会进行测试。这意味着在动态环境中扫描器可以与这一层网络交互而获得一个 IP 地址。探测能力大体上局限于设备所连接的交换机。有关连接到交换机的其他主机的信息将被收集，例如速度和双工之类的交换机基本配置信息，以及交换机如何响应冲突感应的变化和检测协议（如 IEEE 802.3 规格说明中的 CSMA/CD）的变化。总的来说，被动漏洞分析器将会找寻实际通信与 IEEE 标准之间的偏差。

4.4.5 网络层

网络层会受很多变量的影响。IP 地址、标志位、路由信息和选项参数可以组合起来确认一个主机和漏洞，其中的一些组合将在 4.5 节介绍，这里不再赘述。但这足以说明任意网络连接漏洞评估技术都可以从网络层中获取大量的信息。

4.4.6 4至7层

剩下的这些层都可以提供大量关于检测目标的信息。被动分析器不但可以解析这些层以搜索系统交互的行为模式，而且会搜索单个数据包中的特定内容。这是一个包含很多分析方法的复杂过程，而且在设计上更类似于一个入侵检测系统。

4.5 主动扫描技术

主动扫描应用软件生成网络数据包，主动接触目标，以探测目标的存在和相关漏洞。这种方法虽然复杂，但却是现今最流行的且具有高可扩展的方法。扫描器如同其他任意主机一样连接到网络。扫描器相对于目标的位置决定了是否能取得最佳结果。稍后将详细讨论。

从本质上来说主动扫描和黑客发现目标基本是类似的，但有一个很关键的差别：黑客使用工具和技术旨在隐藏他们的活动，而正当的主动扫描工具不会这样做。扫描器也会执行一些漏洞利用行为来判断敏感性。漏洞利用的程度取决于扫描配置中的选项。大部分产品都避免使用这种行为，因为当管理员没有指定扫描选项时这种行为可能会给目标带来不利影响。此外，我们应该明白大部分商业工具的设计目的是用来探测漏洞而不是利用漏洞。虽然它们

也可以部分用于渗透测试，但有一些更合适的工具来完成这样的测试。

4.5.1 优点与缺点

主动扫描的关键优势如下：

- 高可扩展性。根据安全架构师的选择，即可以进行集中扫描也可以进行分布式扫描，而且都不需要为目标机安装软件。
- 这项技术使得漏洞管理员可以从一个黑客的视角审视网络和目标，从而可以真实地了解在产品环境中存在的风险。
- 有支持任意网络设备的潜力，也就是说，并不只局限于与代理兼容的平台；
- 可以提供不受限于平台支持的增量信息（例如，开放端口、识别协议或者应用程序），甚至在漏洞管理系统还没有预先发现设备的时候一样可以提供此功能。

主动扫描的一些缺点：

- 目标没有连接到网络时不会被扫描。代理可以检测到漏洞的发生，并在主机下一次连接到网络的时候报告结果。
- 当所有的扫描都被这样执行的时候，对网络公共基础设施会有潜在影响。然而，一些基本的计划可以避免这样的不利影响。
- 扫描在低速网络连接中会变得更慢。尤其是在一些网络信号较弱的办公室中。现今，在南美、非洲和部分亚洲地区，都存在这种情况。

4.5.2 检测方法

评估漏洞的过程有三个阶段：发现、勘测和白盒测试。发现过程必须要识别出连接到扫描配置中指定的特殊网络或 IP 段的所有主机。通过勘测可以获取主机的类型、版本和可被动检测的漏洞等相关信息。这通过协议的创造性使用和应用程序行为的分析来完成。一旦一个主机被识别，白盒测试将会使用特权访问以收集更多信息。

1. 发现

如同字面意思一样，发现阶段主动使用网络关键协议探测目标的存在。因为并不是所有的协议都是可靠的，所以只有一部分被使用。原因很明显，有时候需要使用多重协议来核实目标的存在。

(1) ICMP

一般来说，ICMP 回送请求经常用于识别一台主机。这不是一个完美的解决方案，因为在某些情况下，网络和安全设备可能会干扰或阻塞 ICMP 类型或者协议。如果主机返回了一个回送应答，系统就会注意到存在这样一个活动的主机，并在勘测阶段将其添加到目标列表内。有些工具会使用其他 ICMP 类型，这取决于环境和设计师的方法。

(2) 控制 TCP

有些扫描器并不认为 ICMP 是可靠的，因为有些主机被设置为不回应 ICMP 请求。所以，有时会发现各种各样的网络设备会使用一些常用端口发送 TCP SYN 数据包。表 4-2 列出了一些被用来扫描是否存在主机的常用端口。

其他端口也会被扫描，这取决于供应商和这个阶段的配置。一旦这样的 SYN 数据包被发送给一台主机，就期望会收到一个 SYN-ACK 的回复。如果这个回复没有按时到达，扫描器就会认为这个端口是无响应的，并且主机不存在，除非主机在其他端口上作出回应。发现这些端口并不一定是一个连续的过程。扫描器可能同时“喷洒式”发送 TCP SYN 数据包给许多主机上的很多端口。这样既节约时间又能更有效地利用带宽。某些性能的问题我们稍后讨论。

TCP 发现方法的一个负面影响是会潜在地留下开放或者半开放的套接字，这可能会给系统带来不利的影响，这取决于监听的应用程序和操作系统协议栈的完整性。半开套接字会在扫描器没有发送 ACK 数据包完成连接配置的时候产生。其造成的影响包括消耗内存和导致产品系统的拒绝服务（Denial of Service，DoS）。通常，在每个 TCP 端口只可能会发起一个主动连接，所以这并不是一个问题。但是，一次错误配置的扫描可能会改变这种情况。这很类似于穿透防火墙的扫描。很多防火墙也都作为代理执行着连接主机的任务。当成百上千的主机被同时扫描时，将会给防火墙带来沉重负荷。某些路由器同样也会受影响。

表 4-2 常用扫描端口

端口	协议
20	FTP
21	FTP
22	SSH
23	Telnet
80	HTTP
443	HTTPs

(续)

端口	协议
137	NETBIOS 名称服务
138	NETBIOS 数据报服务
139	NETBIOS 会议服务

如果收到 ACK 包完成了握手连接的消息, 该入口就会保存在主机和 / 或防火墙的连接表中, 其结果是更多的资源消耗, 直至连接超时或者重置。由于这个原因, 测试扫描器的行为、确定是否发送 TCP 重置请求给主机、确定该行为在大规模的情况下会对你的网络造成怎样的影响等都是非常重要的。如果大量的扫描需要穿透防火墙执行, 那么就需要测试这个方案。建立和断开到防火墙的连接是两项非常占用 CPU 的任务, 有可能会降低性能。然而, 这不是绝对的。在扫描配置中适当地进行计划安排、设置合理的带宽和连接限制可以帮助避免这些问题。图 4-5 展现了在多重并发扫描时, 防火墙可能受到显著影响的情况。你可以看到, 每秒的连接总数可能会迅速增长。与常规的网络通信流量相比, 扫描时的通信流量会非常大, 因为被集中到了很短的时间片内。

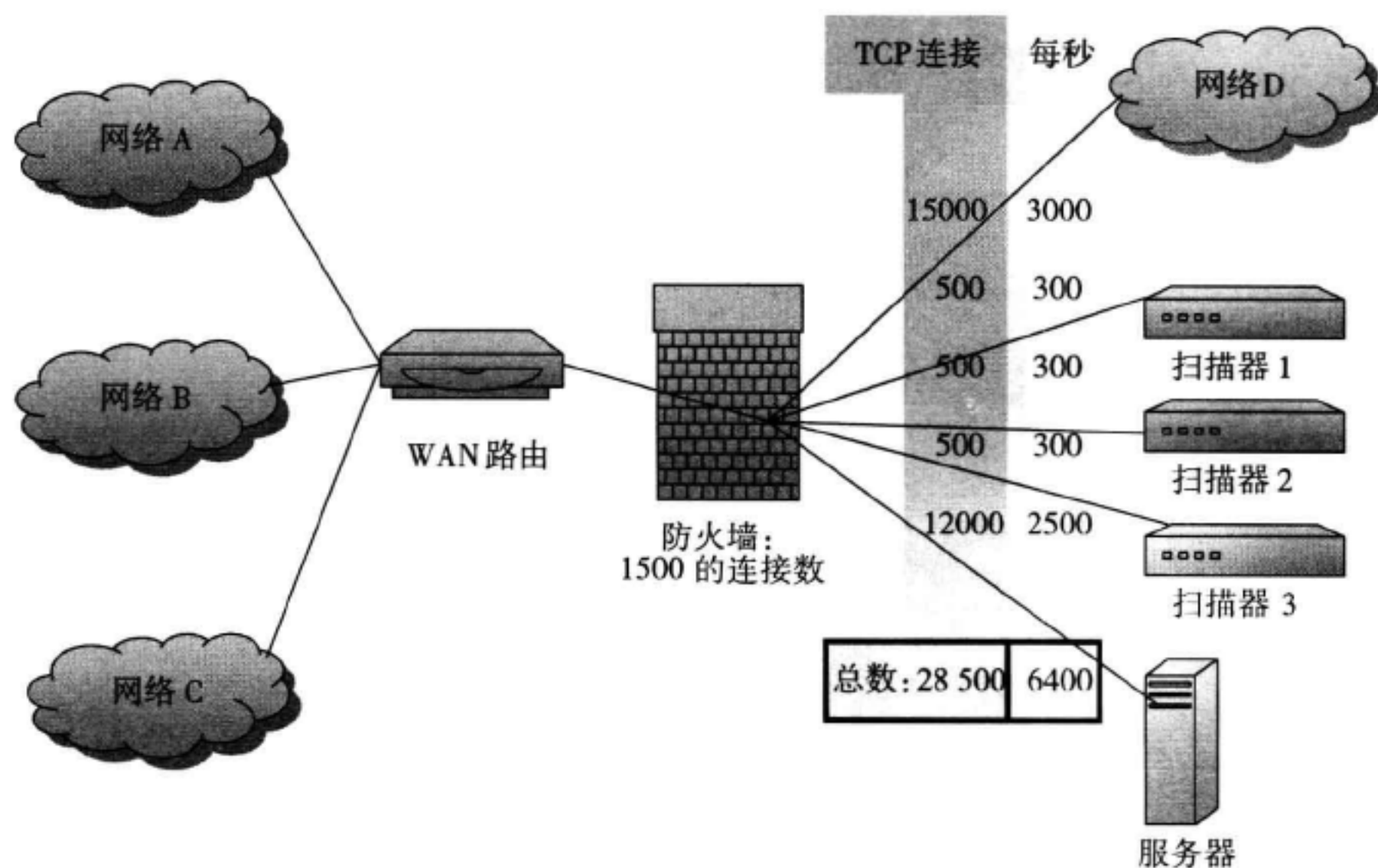


图 4-5 防火墙被多重并发扫描显著影响的情况

相反, 一个销售商添加了诸多附加安全特征的防火墙会干扰扫描。例如, 有些供应商给

防火墙安装了入侵防御功能和 SYN 代理。这些特征会给漏洞扫描器提供错误的信息，让它以为有些地方好像存在一台主机，而事实上并没有。这是因为很多扫描器把一个已经关闭了的端口的回应认为是主机存在的证明，在扫描器看来，如果不是存在一个主机怎么会有回应呢？

这些安全特征同样也可以做逆向工作，并且可以使主机及其开放端口模糊化。我希望防火墙销售商能够给产品嵌入根据源 IP 执行例外行为的能力。把正确的扫描器 IP 配置进例外名单的时候，扫描就可以无误地运行。

然而，并不是所有的防火墙在建造时都是一样的。由于这个原因，我们来讨论防火墙的数据包处理过程，因为它和漏洞管理有关。图 4-6 展示了防火墙处理通信流量的基础结构。请注意这个栈架构，这样构造的原因是首先让数据流量满足对最基本错误的处理，而不需投入更多处理时间。举例来说，如果 TCP 标志位是一个无效的组合，那么这段通信量将被丢弃。很多这样的过程可以在物理层中进行以避免给防火墙的 CPU 带来更多的负担。另一方面，如果流量通过一些规则来处理，就需要更多的 CPU 时间。

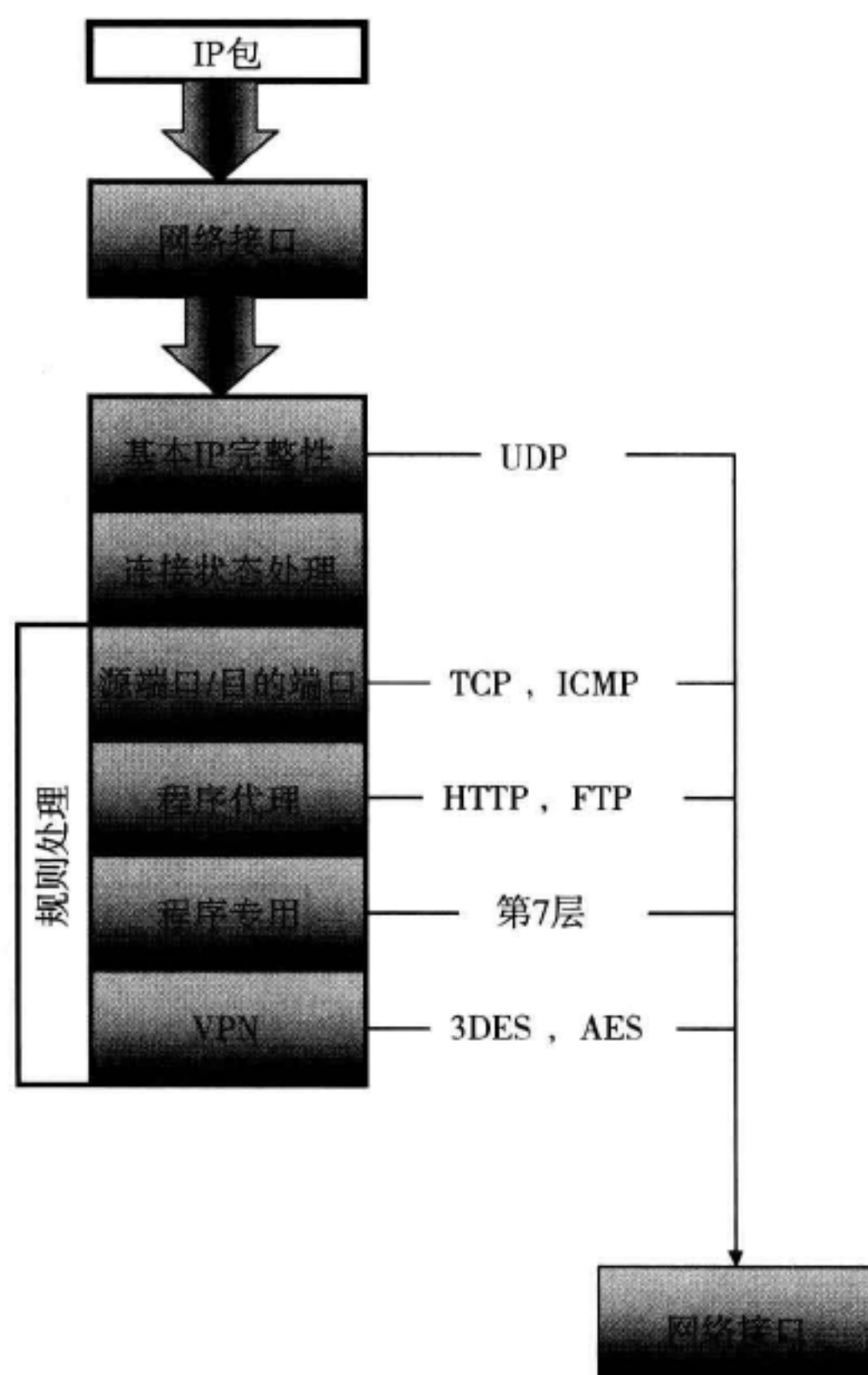


图 4-6 防火墙处理通信的基础架构

数据包处理的下一步就是保存和监控连接状态。如果从漏洞扫描器接收到一个 SYN 数据包，就会对比连接表中的词目。如果这个连接已经存在，这就可能是一个重复的连接，数据包就会被丢弃。如果连接并不存在，就会在表中建立一个词目。当接收到 SYN-ACK 数据包时，会和前面一样与连接表中的词目进行对照以追踪连接状态。所有的相关数据包都是如此。如果收到一个 RST 数据包，那么就从表中删除这个连接。如果每秒钟有成千上万连接并且还有其他防火墙活动也在进行，这个过程可能会消耗大量的 CPU 资源。连接表数据结构和处理数据包的相关程序代码将是最容易受发现扫描影响的关键部分。如果状态表的数据结构过小，那么 SYN/SYN-ACK 发现过程将会迅速填满这个表。如果测试中有这些约束，你将需要确保在发现过程中开放连接数在限制范围之内。

从另一方面讲，与这个表相比，如果处理代码是低效的，这将会对防火墙的吞吐量造成很大的影响。在这种情况下，保持对连接率的限制是至关重要的。

大多数防火墙的处理规则都是有序的。从队列中按序抽取数据包与规则相对比。一旦一条规则与通信匹配，检查过程就停止。然后这个数据包将转发给网络接口，然后从队列中抽取下一个数据包。

另一种避免对防火墙产生重大影响的方法是把扫描器重新安置在它们周围。这很大程度上取决于你的网络设计。应该优先考虑的是谨慎地放置扫描器，因为这会对扫描效果产生很大的影响，相关细节会在本章后面讨论。

（3）性能问题

前面已经使用较长的篇幅论述了识别端口和处理 TCP 连接的过程。所有这些因素都需要在扫描的过程中考虑到；然而，扫描器不能长时间的等待。在某一时刻，尝试将会“超时”。这种现象可以被称作“发现容忍”。各种各样的供应商都有不同等级的发现容忍。所容忍的总量并非总是与发现的精确度不完全成比例，这种发现的精确度会使成功鉴别率迅速下降。幸运的是，从经验中我们可以得知超过一定时间的等待回复是毫无意义的。在任何编程中要决定这样一个时间点都是需要技巧的。要完全又精确地实现目标，就存在一个收益递减的问题。两个关键的计时器影响着发现过程的速度：连接建立计时器和重发计时器。

对于多数 TCP 实现来说，连接建立计时器（TCP_KEEPIINIT 参数）需等待 75 秒才能做出回应。对于单个端口 200 台主机的简单扫描，如果没有任何主机回应的话，需要 4 个小时才能完成。这需要调整来缩短时间。一个有效的方法就是最大化 ICMP 回送应答交换的循环时间（Round Trip Time, RTT），为交换再加上 2 秒。这就为应用程序为回应端口请求提供了足够的时间，而且很可能远小于默认的 75 秒。

使用 TCP 连接，当存在附加数据包需要与目标交换时，发现时间也可能会改变重发计时器。在普通的通信中，计时器以 1.5 秒开始。如果没有收到回应，就把这个值替换成 2 倍，3 秒。如果还是没有接受到 SYN-ACK 包，再把值增大成 2 倍，即等待 6 秒。这个过程会一直重复直到达到上限 64 秒。这个过程称为指数式退避（Exponential Backoff, EB）。理论上，这应该是和指数概率平行的，最终将会收到响应。然而，对于以发现主机为目的的漏洞扫描来说，这通常是不现实的。一个典型操作系统可能需要数分钟等待连接超时。

一个更实用的方法就是在整个时间段内以目标 IP 段综合某些因素所得的很小的值持续地增加重发计时器。举例来说，假定要发现一个重发计时器上限为 30 秒的网络 A (192.168.10.0/24)：如果开始 16 台主机需要平均 10 秒应答，上下差异是 5 秒，设定重传计时器起始值为 5 秒，以 5 秒为单位一直增加到上限 20 秒（2 倍平均值）。这是这样就可以避免普通 IP 栈值为单一连接等待长达几分钟的时间。要记住目标是发现开放端口和活动主机，并不是向另一台主机可靠地传输数据。

还有一个方法，它并不是一个准确的计时器但是能在一定程度加速发现过程：

实现所有的计时器，TCP 只需要周期性的调用这样两个函数：1) 快的计时器以 200 毫秒为周期；2) 慢的计时器以 500 毫秒为周期。TCP 使用这两个周期性“嘀嗒声”来安排和检查所有提到的计时器……并衡量循环时间。

基本上，操作系统的内核必须每 200ms 检查一次发出的 ACK 是否被收到。在现代网络和操作系统中，这是一个非常长的时间^①。

通过逐渐缩短这一周期，发现过程可以在一个更短的时间内认识到之前发出的 SYN 探测数据包已收到，然后继续下一次探测。如果从 SYN 到 SYN-ACK 的循环时间 RTT 是 10ms，那么在通常情况下，发现过程要等待 190ms 再继续处理下一个动作。如果有数百个这样的主机和几十个这样的端口，那么这一数值还要乘上几百，浪费的时间将是巨大的。

修改 TCP 计时器需要注意的是某些程序回复过慢。这是在探测开放端口方面最好的方法，而对于应用程序却是不必要的。扫描性能还有很大的优化空间。本节简单地阐述了一些设计者在进行优化扫描过程和最小化对网络的影响方面可能面临的挑战。

2. 黑盒测试

一旦证实一个主机的存在并且被记录在扫描器的内存中，将会执行一系列测试或者“检

^① Tweaking TCP's Timers, CUED/F-INFENG/TR.487, Kieran Mansley, Laboratory for Communication Engineering, Cambridge University Engineering Department, July 6, 2004, 3.

查”以发现漏洞。检查的类型由主机类型和扫描器的配置决定。大体上说，会执行两种检查：基于网络的检查或者表面检查，包括探测和分析那些在机器上明显受限或者无权限的端口，而不是那些相同网络上存在于扫描器和目标之间的其他端口。这是一个未认证检查。其他类型的检查，如内部检查或者白盒测试是已认证的。当扫描器取得特殊信息和访问主机细节的权限证书时，会执行这样的检查。

表面和内部检查的不同是很明显的，不仅指他们获得信息的方式，而且也蕴含了信息的价值和质量。很明显，登录一台主机详细考察其配置可以获得更多的详细数据。虽然这样可以告诉我们主机的很多信息，但这并不是攻击者对陌生主机执行勘察时通常能看到的。虽然从一个分析者的角度来看是很有意义的，但是从外面来看，有些攻击是探测主机发生的；所以用同样的方式收集的信息经常更有价值。概括来说，从主机外部发现的可利用的漏洞要比经受信任的或内部的检查发现和利用的漏洞暴露性更强。

普遍认为已认证的检查要比远程检查更加准确，但事实却常常并非如此。Windows 的注册表通常被用作已验证的检查，但是却经常出错。不是所有的已认证检查都是一样创建的，并且远程检查是一个确认验证信息的非常好的方法。

黑盒测试过程涉及一些直接的网络测试和对 IP 以及其他协议的创造性应用。通常，简单测试是无害的、有效的。对于 IP 协议过多的非正常操作会给被扫描的主机带来很多问题，因为这些主机所安装的应用程序可能没有充足的准备来处理很多变化。这是它自身的一个漏洞。主机的 IP 栈通常有能力处理几乎任何种类的流量，但是上层的应用程序有时做不到。另一方面，还需要大量的测试来保证不会给产品系统带来负面影响。大部分供应商能够提供一个在交互中会产生负面影响的应用程序列表。

下面列出一些探测的常用方法包括：

- ❑ 通过给主机发送畸形包并分析响应信息来确认主机是否存在漏洞。这很类似于发现过程，并且偶尔会为了提高效率合并到一个阶段中。发送给目标的信息可以在任意层或者是 OSI 模型中的 3 ~ 7 层的组合层。
- ❑ 普通包被发送给一个已知的应用程序来收集可以暴露漏洞的信息。在 http 协议中常常用这种方法来收集关于网络服务器、应用服务器或者后端数据库的信息。
- ❑ 向目标发送有效信息来收集有效数据包报头响应数据，此数据可能会暴露回复服务请求的软件的版本。这就是所谓的标识检查。很多软件应用程序会把这种信息混淆在简单配置改变中，所以，这并不是一种最可靠的方法。

这些方法可以用伪代码形式概括为：

发送 X 给目标
监听响应 Y
将 Y 与可能的应答列表做比较
如果 Y 在列表中，则标注为漏洞
如果 Y 不在列表中，则忽略该响应
获取下一条检测消息，循环

(1) 用 TCP/IP 进行指纹识别

一种简单的指纹识别的方法就是应用熟知的 ICMP。ICMP 数据包用来监控主机的接口或者报告访问一个连接设备的状态。有 9 种消息格式可用：4 种用来查询、另外 5 种用来报告错误。每一种类型都由一个数字定义，如表 4-3 所示。PING 是一个非常流行的可以发送 ICMP 类型 8 的消息。类型 8 是回送请求，而类型 0 是一个回送应答。

表 4-3 ICMP 类型

ICMP 代码	类 型	
0	回送请求	
1 ~ 2	未赋值的	
3	目标不可达	
	代 码	意 义
	0	网络不可达
	1	主机不可达
	2	协议不可达
	3	端口不可达
	4	需要分段而没有设置分段
	5	源路由失败
	6	目标网络不可知
	7	目标主机不可知
	8	源主机孤立
	9	与目的网络通信被强制禁止
	10	与目的主机通信被强制禁止
	11	由于服务类型问题，目标网络不可达
	12	由于服务类型问题，目标主机不可达
	13	通信被强制禁止
	14	主机越权
	15	优先终止生效

(续)

ICMP 代码	类 型	
4	源抑制	
5	重定向	
6	修改主机地址	
7	未赋值	
8	回送	
9	路由器通告	
10	路由器选择	
11	超时	
	代 码	意 义
	0	传送超时
	16	分段重装超时
12	参数问题	
	代 码	意 义
	0	指针指示错误
	1	缺少所需选项
	2	长度不正确
13	时间戳	
14	时间戳应答	
15	信息请求	
16	信息应答	
17	地址掩码请求	
18	地址掩码应答	
19 ~ 29	预留	
30	路由跟踪	
31	数据报转换错误	
32	移动主机重定向	
33	IPv6 你在哪 (Where-Are-You)	
34	IPv6 我在这 (I-Am-Here)	
35	移动注册请求	
36	移动注册应答	
39	跳过	
40 ~ 254	N/A	

除了 ICMP 类型之外, 还有一种代码用于报告更多的错误信息。通过将这些代码设置为无效值, 可以捕获目标的回应或者故障回应。这本质上可以告诉我们一些与操作系统相关的内容。有些系统并不着眼于回送请求的代码域, 而其他的会注意到该代码域并回应一个错误。

当扫描器发现这样的值被发出而时间戳选项没有使用时，操作系统的选择余地将会变得相当有限。

另一个需要评估的现象就是时间戳的增量。在首先决定扫描器和目标之间循环时间 (RTT) 时，就可以知道在 TCP 段之间中需要消耗多少时间。远程的操作系统将会在每个分段上按照一定量给时间戳增值。目标上时间戳的增加方式能够反映出操作系统的类型。

例如，操作系统 XYZ 每 500ms 给时间戳增值。在目标和扫描器之间的平均循环时间 RTT 是 100ms，在每一个方向上是 50ms，如图 4-7 所示。收到第一个分段带有时间戳 100 (TS1)，随后报告收到这个分段并开始一个计时器。第二个分段带有时间戳 102 (TS2) 到达，于是停止计时。在分段 1 和分段 2 之间消耗的时间是 1100ms。时间片的传输时间是 100ms。所以用这个时钟值 1100，减去循环时间 100，得到在主机上两个分段之间时间是 1000ms。而 TS2 和 TS1 之间的差别是 2。这就意味着，在 1000ms 内，时间戳的值增加了 2，是每个时间戳在 500ms 内的增值。查看一下表中时间戳的值随时间变化的情况，可以知道目标操作系统 XYZ 每过 500ms 给时间戳增值，于是判定这个操作系统就是 XYZ。该技术结合其他指纹识别方法，能从根本上缩小操作系统的定位范围。这一定位对于决定未来阶段的漏洞扫描是很重要的。

无效标志位混合是另一个方法。常设定的是 SYN、SYN-ACK 和 ACK 的正常组合，是期望的有效值。但是各种主机操作系统在普通握手中不会出现的组合（诸如 FIN+URG+PSH 等）会有奇特的反应。这种情况被称为 Christmas（圣诞）扫描，因为它把 TCP 标志位都点亮了，看上去就像棵圣诞树。另外一个可以利用指纹识别的操作系统的组合是 SYN+FIN。除了发现主机之外，这些类型的扫描还可以确定在一台没有建立 TCP 连接或半开放连接的主机上是否有端口打开。那是因为依赖于 RFC 的 IP 栈在端口打开的时候会回应一个 RST 数据包。如果关闭，那将不会有回应信息来自于这台主机。

这些标志位的使用也可以变得更复杂。如果一个端口已经使用无害的 TCP-SYN 扫描打开，那么这个端口也可以用 FIN-ACK 组合来探测。结果表明使用柏克莱标准分配 (BSD) IP 协议栈的系统将不会按照 RFC 标准回复一个 RST 数据包。这将给目标操作系统的类型提供更多的依据。

通过组合这些和许多其他类型的探测，将可以大体猜测出系统的类型。这项工作已经被 NMAP (www.nmap.org) 的创建者建立了。他们在不断地探索新的扫描方式和将目标对应到网络中的方式，并将这些技术嵌入到他们的开源工具中。这方面的阅读和实验都是很有价值的。

然而，操作系统指纹识别和 IP 栈指纹识别方面可能需要技巧，并且不太可靠，容易混

淆。有些操作系统会共享相同的 IP 栈代码，但操作系统的版本却不同。例如，各种不同的 Linux 发行版本都会使用相同的栈，这并不一定能暴露操作系统的特点。虚拟机技术会让这个问题变得更加混乱，因为下层管理操作系统可能会回复网络通信并代理连接到实际主机的操作系统。指纹识别就很可能出现出乎意料的结果。防火墙和虚拟机都会进行网络地址转换 (Network Address Translation, NAT)，这样的话就会隐瞒真实目标的操作系统。

(2) 应用程序指纹识别：标识

扫描的一个重要目的是确定需要让网络知道哪些应用程序存在。类似于操作系统的指纹识别和 IP 栈的指纹识别，一个漏洞扫描器可以尝试连接到已知或未知的端口上各种可能的应用程序上，这一过程就叫做指纹识别。标识检查是指纹识别的一种方法。

每当试图建立连接时，系统上一些通用的应用程序就会产生一个标识。这个标识的内容可以提供确定操作系统版本和主机上运行软件的价值信息。这种指纹识别常常通过使用一个可以兼容任意操作系统平台的简单程序：Netcat 来执行。

我们以网络服务器来说明。图 4-8 展示了一个典型的 Netcat 会话。当使用 Netcat 时，可以指定想要连接的 TCP 端口。所以，在命令行输入 “nc 10.1.1.10 80.” 将会运行远程登录程序，并向服务器建立一个监听 80 端口的连接。

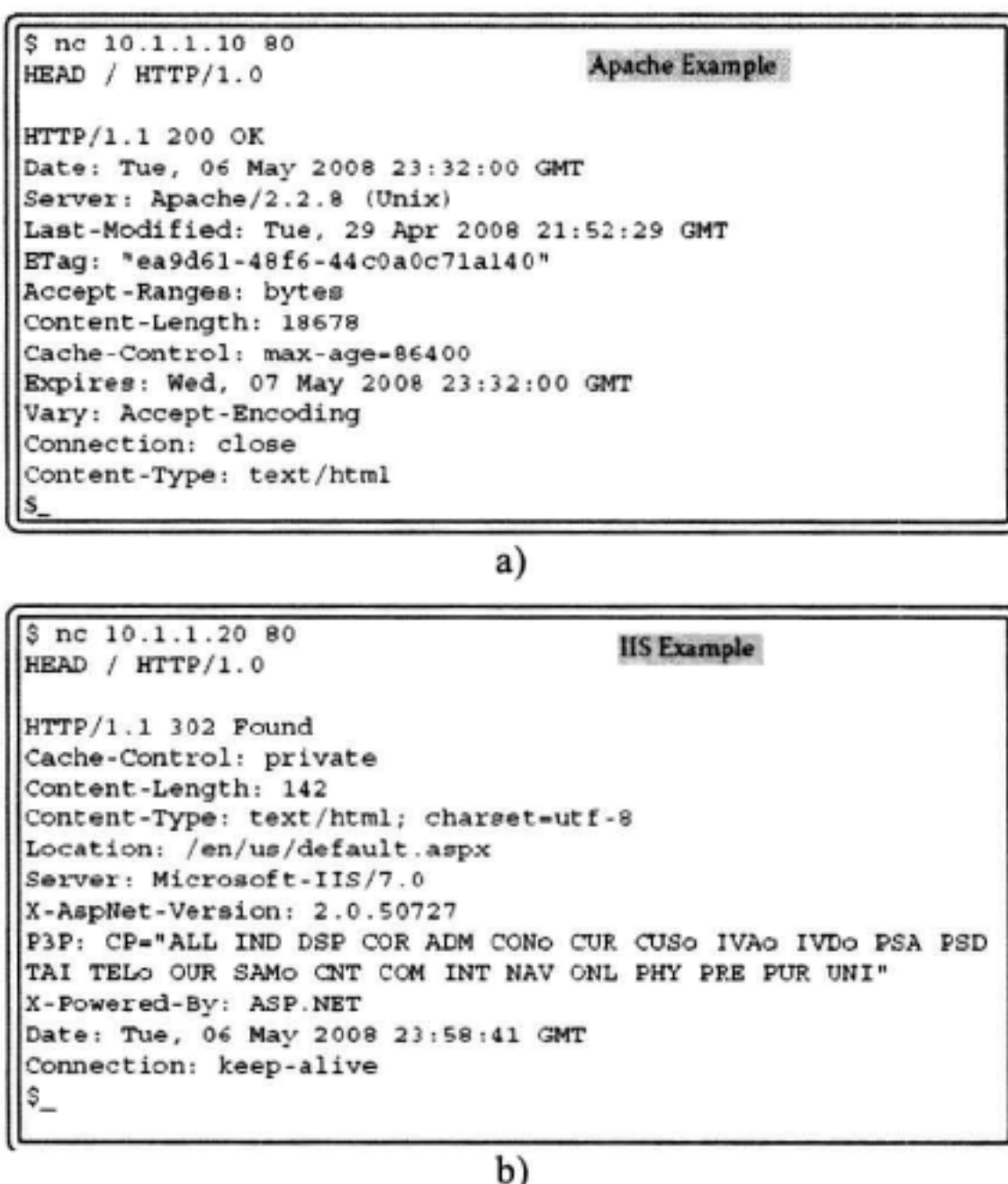


图 4-8 典型 Netcat 会话

由此，我们可以发出一条命令给远程服务器。因为端口是 80，看起来像一台网络服务器；所以我们发送命令：“HEAD/HTTP/1.0”（见图 4-8a）。服务器回应了一些详细的头部信息，包括网络服务器软件和操作系统的类型。本例中，是运行在 UNIX 系统上的 Apache。这排除了是 Windows 的任意版本系统的可能性，也让搜索漏洞变得更加容易了。

然而，因为安全原因服务器管理员应该隐藏这些头部信息，尤其是在“服务器”部分的信息。但是，这并不能阻止一个好的漏洞扫描器工作。扫描器可以通过制造一个无效请求来检查服务器类型。通过在头命令中使用无效的版本类型，可以从各种网络服务器那儿得到不同的回应。请注意 Apache 网络服务器的结果是“400 Bad Request”消息（见图 4-8b）。这个连接被关闭；但是，在 IIS 7.0 上，连接并没有被关闭却收到了同样的“400 Bad Request”消息。但这里，你也会注意到还有很多在有效请求中没有发现的更多的服务器信息。

在早期版本的 IIS 中，你可以将其与对“200 OK”消息的回应区分开来。类似的方法用在有效和无效回应的捕获和分析中。有些情况下，这些回应被称为漏洞或者简单信息泄露^①。

（3）其他指纹识别方法

其他种类的指纹识别方法也使用协议执行，例如 SNMP，用来监控和管理网络设备的协议框架。这种能力经常被拓展到防火墙、IPS，甚至普通主机。为简单起见，在状态报告涉及网络上所有设备的集中管理方面，SNMP 几乎都有涉及。

网络管理软件使用 SNMP 向设备发送质询。质询消息的结构格式如同图 4-9 所示。

版本号（version number）指定了使用中的协议版本。版本号很重要，因为依靠它目标才能知道如何去处理这条消息。

通信（community）是一个用来鉴定请求发送者的一个口令。在接下来的例子里，我们将会伪造这个值来搜索基本漏洞并确认设备。

请求 ID（request ID）是一个类似于用在 TCP 中的序列号的数字。这个数字将包含在回复请求的消息中，这样的话请求者就知道回答的是哪个请求。它只是一个具有参考作用的索引。

错误状态和错误索引和这里的论述的关系并不密切。这些值在目标产生错误消息并将其报告给网络管理服务服务器的时候使用。这些消息使用 UDP 在端口 161 和 162 上发送。虽然并不太可靠，但 UDP 是很有效的，也很少被安全人员发现的，因为在网络中使用这个协议进行通话的情况较多，难以观察到不恰当的行为。因为它没有像 TCP 中一样的状态，所以这个协议

① 如果应用程序防火墙部署在目标服务器和扫描器之间，此类指纹可能会失败，因为它将会检查并忽略无效的 HTTP 请求。——译者注

更难被滥用。从另一方面讲，UDP 数据包很容易欺骗，也经常能穿越网络安全设备，因为网络工程师为了收集重要的网络统计资料通常会采用较为简单的过滤方式。

版本	通信	请求 ID	错误状态	错误索引	变量
----	----	-------	------	------	----

图 4-9 网络管理软件用 SNMP 协议给设备发送问题

指纹识别过程的一部分就是鉴定设备是否在运行 SNMP。这可以通过发出一个“GetRequest”（使用默认值“public”）消息来实现。这就像是给一个传统主机设置口令一样。有些扫描器也有能力循环产生用在各种设备上的默认值。它们偶尔也会尝试由其他用户指定的值，可以把这认为是一种针对 SNMP 配置的暴力攻击方法。

使用一个被称为管理信息库（Management Information Base, MIB）的分层结构，可以请求有关于目标的信息。所有的信息都可以请求，有一些对供应商来说是独一无二的。很多设备都默认安装了 SNMP，而且因此会接受请求和暴露信息。更常见的值是“public”、“private”和“secret”。也有很多通用的字符串，通常被指定的供应商使用，并且没有改变过。

类似于 SNMP 通信串的方法，Telnet 和 SSH 协议也可以用于指纹识别。有时，服务器名称或者操作系统类型在登录之前就提交给交互用户了。所提交的这些串被扫描器解析为关键字，如“Windows”或者“UNIX”等，查看这些信息并不需要登录。通过 SSH，如果一个连接可以建立，那么我们就能确认正在运行的服务，然后就可以尝试去发现相关的漏洞。

还有其他一些可用于指纹识别主机的技术，其中一个很著名的工具是 NMAP。这个工具可以在 www.insecure.org 上找到。这个命令行工具可以使用各种 ICMP 和 TCP 协议对一段 IP 地址进行扫描。某些已使用的方法在之前已经论述过了。需要记住的是，商业性的漏洞扫描工具都极力做到不干扰其他设备，但扫描同样可以被安全设备发现。它从来不会将偷偷扫描作为目标，仅仅是为了健康有效地工作。

指纹识别过程可用来辨别扫描器，因为输出可以用来确定全体目标漏洞评估检查的类型和精确度。基于前面的介绍，你可以领会指纹识别工作的复杂性。了解操作系统版本可以帮助我们确认需要检查的通用应用程序。作为另一种选择，先发现一些应用程序也可以帮助我们更加准确地识别主机。协调这些活动的艺术虽然是很精细烦琐的，但是对于精确度来说却非常重要。

3. 白盒测试

白盒测试是一种通过提供认证、特权或者“打开”系统设置的访问权限以发现漏洞的方法，

因此通常不明显的细节也可以被观察到。这种测试既被用于基于代理的技术，也被用于主动扫描。

(1) 证书

要在主动扫描技术中执行这种类型的测试，扫描程序需要特定的证书来远程访问目标。证书的类型依目标类型的不同而不同。通常使用的访问方法有 SSH、WMI、Telnet、SNMP、SMB 和 Remote Registry。Windows 的机器通常需要 Remote Registry 或者 WMI 来访问，然而 *nix 的机器需要使用 SSH 访问。

一个常用的认证渠道就是 SNMP。这是一种在网络设备和打印机上非常通用的协议，甚至也用于普通的计算机，例如台式机和服务器。这个协议模式包括管理站或者工作站与一些代理，其中每一个都运行在被管理的设备上。SNMP 通常有三种版本：SNMPv1 (RFC 1157)、SNMPv2 和 SNMPv3。版本 1 和 2 使用通信串，这些通信串作为单一口令从被管理的设备上执行命令或者收集信息。既没有用户 ID 也没有办法最终确认谁执行了什么样的命令。此外，通信串是明文传输，并且通常认为是不安全的。

SNMP 的简单性部分是因为使用了 VDP。对于不熟悉这种协议的人来说，它允许无连接或者无担保地通过网络传输数据包。该简单性也是 SNMP 的一个弱点，使得该协议可能会在黑盒测试中被滥用。虽然如此，使用 SNMP 的只读通信串和命令从目标收集漏洞信息通常是很必要的。

SNMPv3 提供了相当多的安全性。它用于加密消息内容、认证发出请求的用户，以及认证消息来源以避免欺骗和损害等多种安全元素。事实上，它提供了一个用户 ID 和口令（密钥）。这个密钥用来加密消息并具有很强的防篡改能力。

Telnet、SNMPv3、SMB、WMI 以及 Remote Registry 都应用了用户 ID 和口令。SSH 可以使用用户 ID 和口令或者证书。最好目标都不要要求明文证书和 Telnet 访问。这种方法经常出现在很多较老的产品中，并且这些产品也没有别的方法可用。通常可以在不断电设备和老式的网络设备的控制接口上看到它。新的设备则提供 SSH 作为首选。

在商业漏洞管理系统中，这些证书通常被加密存储直到它们被使用。真正的挑战并不是证书的应用。为目标设备管理和配置这些证书才是最需要关心的。我们将会在第 5 章中更加详细地论述。

大概每个环境都有主要的管理目标。例如，在庞大的微软 Windows 环境中，活动目录（Active Directory）经常被用作认证、识别、政策管理和基础软件配置。活动目录可以用来创

造和部署一个单一证书以提供给组织单元或者整个企业。这些证书在管理下很可能被安装在95%的机器上。与此不一致的是，有些原因可能会致使发现一些本地事例的存在。有些机器可能不是活动目录的一部分，所以很明显不会对其管理。而有些机器则为了防止组策略的应用而进行了一些配置。这可能会导致运行在目标主机上的应用程序被活动目录中的活动中断。不过本地管理员也有可能简单地手动安装这些证书。

在创建证书过程中，大部分工作主要集中于也应该集中于使不一致的目标受控于中央管理设施或者手工安装证书方面。如果不积极解决这些问题将会导致一些经常引起问题的主机的存在，而这些主机则会一直处于未配置状态。这将使它们不受认证扫描的管理，导致得分一直偏低而造成误导，难以获得漏洞管理者的注意力。

通过已经建立的证书，白盒测试可以简单而快速地进行。协议中任何可操作的东西都可以被执行，而准确来说并不是任何东西，例如那些不可操作的东西。事实上最好的情况是，这些证书都应该持有完成任务所必要的最小权限。这通常意味着只读权限。原则上扫描不应该要求具有改变目标上任何东西的能力。其他所有东西都应该被怀疑。那可能是一段粗心的代码或者软件设计，也可能是寻找漏洞的侵略性的途径。在一些特有的情况下，可能目标的限制条件为证书只有“写”权限时允许访问关键的信息块。这通常是软件开发者对目标过度精简化的权限所导致的。当对系统访问权进行审计时，扫描专用证书对于追踪目标的用法和划清责任也是很重要的。

（2）扫描

当针对一个目标检查多种漏洞时，这种行为称为扫描。把扫描想象为执行检查的过程。在检查另一个相同的目标之前，扫描需要包含大量的数据收集、组织和分析等工作。这个复杂等级需要一个分层的方法来收集、存储和处理结果。并不是简单地为每一种类型的主机设置一个检查项集合，然后据此对目标机进行相应的检查就够了。这会导致处理器、带宽的浪费，更严重的是，可能会带来目标的“崩溃”。作为一个医生，作为一个好人，我们“不做有害的事情”。以下的情况恐怕是难以容忍的：当你去看医生，而医生完全不问你的症状和对各种情况的反应，而直接简单地为你做一连串测试，其中一些可能是有害的而且和你的情况无关的。

一个更合理的途径就是开发一个目标的属性资料，之后我们在属性资料里寻找漏洞。例如，如果某个操作系统上安装的是 Microsoft Office，那么探索与 WordPerfect 相关的漏洞则是毫无意义的。不但会有影响机器和消耗网络带宽的风险，还会浪费本来可以用来扫描其他主机的宝贵时间，而这个时间可以用来完成其他主机的扫描。通常使用一个预先设定工作量的

特殊的扫描器，这样可以使得要求的扫描在一个典型的报告时间内及时完成。

扫描主机时的另一个子过程就是生成一份类似于软件或配置清单的文件。这份清单，一旦完成就会与一份可能问题的检查列表进行比较以便于发现已知的漏洞。

这一功能中的数据结构类似于表 4-5。每个过程都执行一个检查来查找“标识符”列以匹配过程名称、协议和结果。这个匹配将会暴露漏洞是否存在、是否需要再做一些额外的检查、已有描述，以及所属类别等信息。结合其他的数据库表和配置参数以及主机细节，一次扫描将可以给出一个数值作为结果。

表 4-5 漏洞数据库结构

标识符	V 指针	C 指针	种 类	描 述
HKLM/Software/Mozilla/Current-Version/20.0.0.14	9574839	9037593	操作系统	Mozilla 2.0
HKLM/Software/Office/MSWord/CurrentVersoin/5	7752955	879	生产力	MS Word 5.0
SSH:Netoper14	773845	948346	实用	Secure Copy (SCP)
TELNET:IOS 11	99175	86849	网络	Cisco IOS
BGP: TCP179 ACK	3980	8756	协议	未过滤的 BGP Peer
TCP50:AXFR	122	7112343	域名解析	未授权的区域 允许传递

例如，在表 4-5 中，如果一个 Telnet 端口在发现过程中生成一个回应，包含“Cisco IOS 11*”，然后它就会到“标志符”对照到表中去找找到“C 指针”86849。这个 C 指针导向扫描器加载和执行发现索引的检查。另一方面，如果发现或扫描阶段在表中找到“BGP:TCP179 ACK”，那么这显然就是一个漏洞，于是就不需要更深层的探索了。即使如此，还是有一个 C 指针指示扫描器必须执行另一个检查，因为扫描结果同时显示了这是一个运行着 BGP 协议的路由，而且有一个漏洞。所以，必须另外再执行路由检查来记录已存在的漏洞。

图 4-10 展示了基础扫描过程的两个部分。图的左边表明了整个扫描过程。在这个过程中，结果被读取并与漏洞表进行匹配。一旦匹配，就会创建一个目标漏洞对象的列表并且将该目标漏洞添加到列表中，从而建立一个匹配记录。一旦一个特定目标的所有漏洞都被收集了，右边的过程将会实例化。“实例化”是因为不止一个这样的过程可以同时运行。然后，再重复左边的过程。

这个目标扫描进程一直持续到扫描完所有的主机或者达到最大的并发目标扫描数量。当达到最大并发目标扫描数量时，这个过程将会等待直到有空间运行另一个实例为止。当到达

目标数据的尾部（EOD）时，过程终止。然后，后续的数据收集和报告活动开始。这些后续的过程会一直等待直到所有扫描过程结束之后（达到 0）报告所有结果。

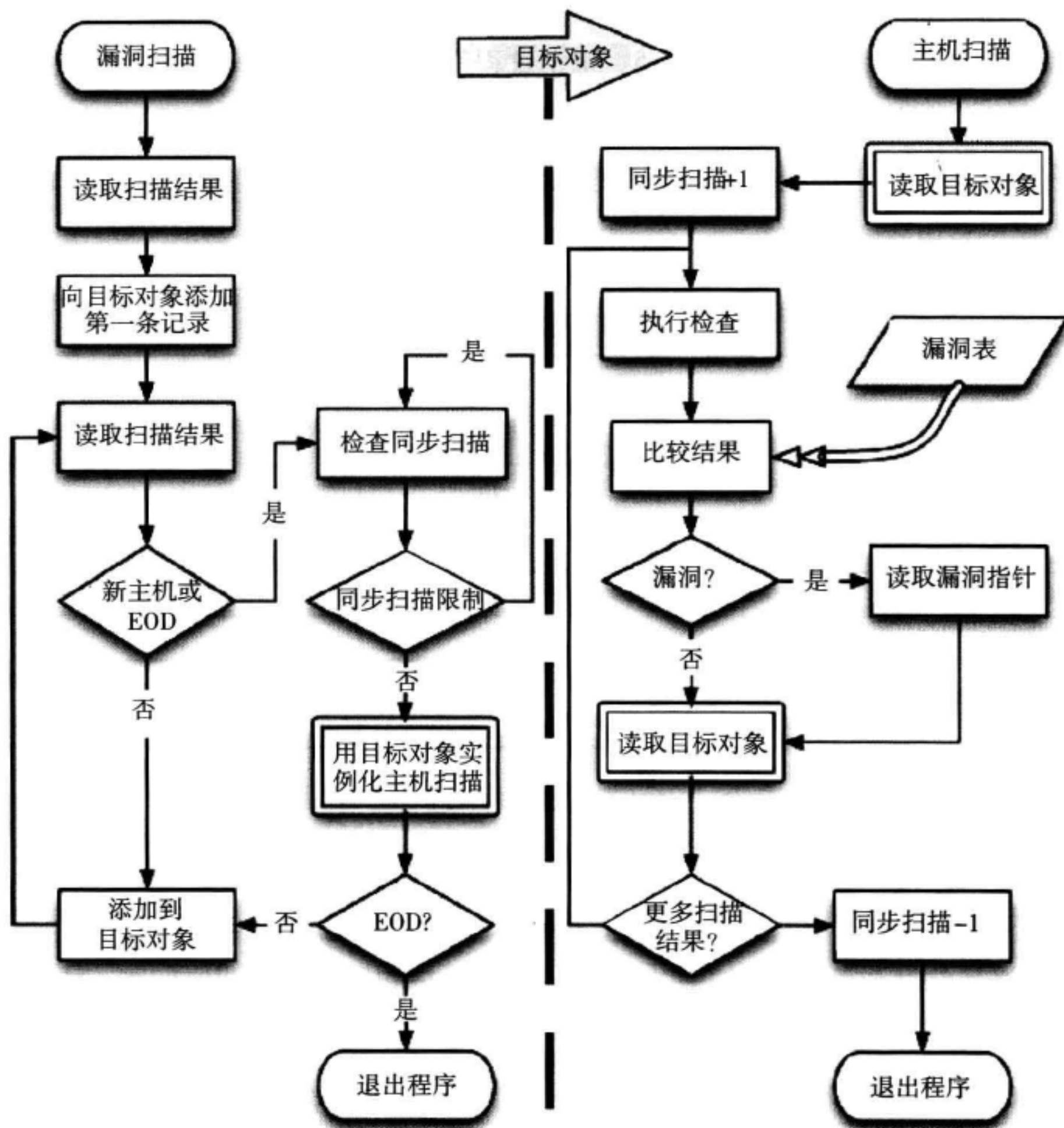


图 4-10 基础扫描的两个部分

右侧的过程相当直截了当。漏洞列表中每一个要检查 C 指针的项，都会被读取并执行相应的检查。一旦所有的检查都被执行，过程将会退出。请注意一个成功的过程的退出将会使扫描并发数计数器减 1，留出空间来给左侧实例化另一个过程。基于前面所描述的原因，利用处理器的多任务运行的能力是非常重要的，序列化这个过程只会延长扫描。

如同之前所建议的，最后将会为发现的漏洞赋值。然而，这个值有很多种计算方法，取

决于特定扫描软件的设计策略。在一项优秀的安全实践和代码实践中，这个活动不会在扫描过程中执行。所有资源都应该致力于执行一个精确和及时的扫描而没有其他开销。这同样涉及优秀代码的实践。在扫描过程中计算漏洞的值是没有任何价值的。但是在随后的报告阶段确实会生成这样的值。此外，通过在扫描阶段引入一个赋值计算方法，程序设计员也可能因为疏忽而在代码中产生一个可以被利用的漏洞或者缺陷。这是在该关键阶段可以被排除的又一点复杂性。

（3）性能事项

在扫描中，扫描器的目标是进行尽量完整和精确地扫描。然而，扫描的性能和行为对漏洞管理来说也是很重的。理想状态下，我们希望在一个分派的时间窗口内扫描得尽量多并得到完整的结果。然而，也要避免影响产品的操作。首先，来看一下对产品的潜在的不利影响和避免它们的方法，然后我们再关注优化扫描的方法。

在大部分情况下，有四种方法可能使扫描对产品环境产生负面影响：

- 消耗带宽，妨碍其他应用程序的服务水平。
- 在一个已经任务繁忙的目标机器中消耗目标 CPU 资源。同样，这将会导致无法达到服务水平。
- 打断目标程序或操作系统，导致拒绝服务攻击（DoS）并需要目标被修复。
- 破坏辅助扫描的组件而不是目标。各种网络组件将会受到扫描过程的不良影响，而使它们并不服务于扫描。

网络活动消耗带宽。在扫描过程中由漏洞管理器提供的参数往往会用于衡量网络足迹。并不只是带宽一个因素，并发连接的数量也可能影响媒介设备。因为 TCP 被普遍地使用，所以每个目标都会建立连接。在某些情况下，一个连接可能仅仅是尝试性的，于是留下了潜在的半开放连接。追踪连接状态的设备，例如防火墙、IPS 或路由器，都会被这些连接影响。并发连接的总量，以及这些连接建立的速度，都可能产生影响。限制这两项对维护与网络管理职人员的良好关系具有很大帮助。

在扫描器中，最能有效加强这些限制的就是 IP 协议栈和接口驱动程序。带宽的限制在接口驱动程序上能够最好地执行，然而连接数的限制更容易在保持传出连接表的数据包创建阶段实行。免责条款应该适应于扫描器的关键命令和控制功能。因此，扫描器管理系统的位置应该免于这些限制。

在扫描器对其连接本地段的外部进行扫描时带宽消耗对网络的影响是最大的。对 WAN 链接的影响尤为显著。应用当今成本效益最好的技术，大多数扫描器都不能在一次典型扫描中产生 10Mbps 以上的带宽消耗。然而，当 T1 仅仅从远程的办公室连接到公司扫描器所在的公司 WAN 所在地，则很容易因为一次小型扫描就让链路饱和。在大多数公司，常常需要在工作时间执行这样的扫描，在这个时间里，台式机和笔记本电脑都打开着，并且连接着网络。所以，对商业操作的影响是很显著的。

为了确认这个方法给目标网络带来的带宽消耗的总量，我们推荐以下策略：

1) 确认超过 2 个月时间的 WAN 链接利用的峰值与你计划运行扫描时间的商业周期的峰值相同。另一种选择是，计划运行扫描的时间内、商业周期的带宽消耗很低，但是目标仍然是可扫描的。

2) 与当地的商业经理和网络操作管理员在你希望使用的剩余带宽量上达成一致意见。在协商讨论时设定好一个预期的结果。

3) 对目标网络的五台或更多台目标主机执行测试扫描来测量所需的带宽。这个数值需要按比例放大以得到更加精确的值。这里有一个警告：在这种评估中，测试连接越接近最大连接数，准确度越高。有时，限制连接数会减少最大带宽的消耗，但其他情况下不会。这都取决于各个目标的配置情况。由于扫描是高度动态化的，测试的强度应当与环境的复杂性和多样性以及业务操作中 WAN 链接的临界值相一致。这些既是艺术也是科学。

4) 时刻让扫描活动处于关键安全功能的位置，这个位置能够给这个场所的 IT 和网络管理员提供最终的报告和分析。他们会希望你提供尽可能多的信息。在关键主机的位置，很值得购买少量的额外带宽用于执行扫描。

另一方面，并发连接对管理来说是相当直接的。就像之前提到的，这个参数会影响带宽。我们的主要目标不是掩盖目标或介入网络和安全基础设施。即使在低带宽情况下，小数据包的大小和半开放连接都可以产生一个相当大数量的并发连接。由于防火墙维护着一个包含每个连接状态的表，因此它会需要多消耗一些 CPU。面对公共网络站点的大型商业防火墙一般都会有充足的资源来处理极大数目的连接。但是这并不是所有的情况。防火墙是复杂的设备，它运行着很多并发程序。此外，它们大部分都是单线程应用程序，这种情况限制了其功能的可扩展性。正如针对路由器的 DoS 攻击是利用了对于众所周知的有限 CPU 资源某些特性的信赖一样扫描器也可以对防火墙做同样的事情。大部分情形可以按照以下的指导方针处理：

- 测试防火墙在多个目标负载下受到的影响。不同的目标可以产生不同的连接率。密切监控防火墙 CPU 反应。这个反应并非线性的。在某些情况下，可能产生数据包转发延迟。在另一些情况下，这仅仅是花更长的时间建立和中断连接。厂商的性能说明并不适用于漏洞扫描器产生的通信量。
- 针对使用对抗分布式拒绝服务攻击（DoS）和有入侵防御能力的防火墙的目标进行扫描测试。因为防火墙会和 OSI 的第 3 层和以上的层次交互来提供这些服务，防火墙可能会把扫描认作一个威胁。它可能被阻塞或者扫描的结果不理想。这是因为防火墙的特征对于指定的应用程序可以扮演代理的角色。TCP 连接尝试将被防火墙终止，仅在与净化过的 OSI 上层结果完成握手协议之后才从防火墙重新与目标建立连接。任何没有被扫描器完成的通过 SYN-SYN/ACK-ACK 执行的探测都可能会造成一种假象：原本没有活动主机的地方好像存在活动主机。一个针对这种情况的解决方案就是配置防火墙允许扫描器的源流量绕开所有的防火墙检查过滤器。
- 路由器同样可能被影响。某些路由器用它们自己的有限的处理器功能来处理无效的 TCP 标识组合。这是漏洞扫描器针对操作系统或者应用软件进行指纹识别时使用的较为普遍的探测和发现技术。这也是针对路由器进行 DoS 攻击的一种方法。虽然不见得扫描器可以产生足够的畸形流量而对网络设备产生很大的影响，但你应该注意这种可能性。
- 在实验室中的 WAN 回路上做扫描时模拟典型的延迟。这将会帮助测量扫描性能和对产品系统的影响。有些销售商提供的工具可以从网段捕获数据包和一份性能简介，然后使用扫描通信和任何注入的应用程序重建体验。举例来说，记录系统可以被放置在这样的设备的一侧，同时扫描器在另一侧。这个设备用预先准备的资料伪造了在高峰时间特定办公室的通信状况。
- 限制在发现和扫描阶段的被扫描的 TCP 和 UDP 端口数量。应尽可能全面地这样做，因为你不能确定会发现什么。在最初扫描有限的数量的主机的时候这可能是有效的。但是，稍后，你就应该选定一个可以接受的端口数来最小化对环境的影响并且最大化扫描的效果。扫描的端口数呈对数等级递减。你也会发现减少扫描的端口数量实际上减轻了带宽的负载。这是因为这时数据包可能是大而多的。

4. Web 应用程序测试

面对如此多的竞争压力，各厂家必然使用某种方式来区分它们的产品。所以在原有的核

心功能上，产生了数以百万级的定制程序来完成用户服务和应用服务以增加附加值。黑客们自然也会想办法利用这些应用程序，特别是因为通过这些程序可以接触到有价值的信息。由于这些应用程序独立于标准技术和基础设施，所以更容易被利用。这不是一种批评，而是一个事实。例如，今天大多数数据库使用 SQL 语言，许多 Web 应用程序使用 JavaScript，基于它们的具体实现方法，这两项技术都可以被利用。因此根本没有任何一种固有安全机制能在应用级别避免黑客利用。

更令人担心的是对开源 PHP 语言的利用。PHP 一般用来开发 Web 应用，但在代码级别上它也有很多可怕的漏洞。因为这个功能强大的脚本语言与 Web 页面及用户交互如此密切，并且它还有非常多高灵活性的命令，这就让漏洞利用成为可能。

自然地，Web 应用也带来了一个新的领域——漏洞测试。随着用户应用成为了危险黑客们的重点关注目标，这些检测也就变得更为重要。现在有很多简单的方法可以利用这些漏洞，例如通过简单的在 Web 浏览器中构造 URL 的内容就可以实现，也可很容易地通过操纵屏幕上的输入框输入来攻击。所以，漏洞检查就必须结合使用上述相同的技术来重复构造可能的攻击形式。下面一些最常见的攻击形式：

- ❑ 输入框的使用：修改屏幕上输入框的输入，但不使用程序正常要求的形式。很多程序员可能忘记验证这些输入的长度或边界。黑客就可以利用这一点输入异常的字符串，这样会导致程序出现不该有的处理过程。
- ❑ SQL 注入：这是一个非常流行的利用后端数据库程序查询语言攻击的方法，它能够暴露敏感信息甚至能修改数据库中的数据。可以这样构造，在输入框中输入部分 SQL 串（‘ or 1=1—），但去掉括号，这将会触发一个潜在的 SQL 状态去检查是否存在没有过滤的字段以及实际可用和可访问的 SQL 语句。这种方式之所以有效是因为单引号“'”终止了 SQL 语句想要的信息，而加上了一个恒等式逻辑判断“or 1=1”。剩下的部分告诉 SQL 服务器忽略掉剩下的查询语句。这是一种无害的通过修改 SQL 查询的方式来判断是否存在 SQL 注入的可能。漏洞扫描器会使用更深入的渗透方法来暴露更多程序在输入方面的缺陷，包括数据库配置方面的缺陷。
- ❑ 跨站脚本攻击（XSS）：这是对输入框的扩展利用，它注入 JavaScript 到 Web 站点，这些站点会出现在其他用户的浏览器中，这样就可以攻击他们的系统了。这种行为包括但不局限于将用户的输入导向其他站点上，捕获用户数据，以及展示错误信息。这些

恶意脚本可以和 SQL 注入结合使用，把恶意脚本代码存储到目标系统的数据库中。当这些信息被某个用户使用 Web 应用程序检索出来的时候，这些脚本就会加载到用户的 Web 浏览器中，并被执行来完成攻击目的。

所以，对 Web 应用程序的检查会出现在很多的漏洞管理系统中，这样就可以检测 Web 站点代码中是否有这些漏洞。当然也有其他的产品能分析编程代码，但是它们不能替代模拟黑客从外部攻击的方法。从本质上讲，这些检测是暴力的但通常没有破坏性。程序会对每个输入、每个超链接、每个可能 Web 站点的 URL 进行检查。下面是其他类型的检测：

- 边界检查：就是对程序允许的输入值范围做检查。使用小于、大于或者在允许访问内的值测试每个出现的输入域，当然也有程序无法预料的无关数据。例如，一个 8 位的 ASCII 值使用 16 位的 Unicode 数据代替，这通常被称为压力测试。
- 分支测试：用于检测程序接受的所有的链接或可能的路径。目标是达到 100% 覆盖分支，也就是程序的每个可能的路径都要测试一遍。这也是一种发现过程或者部分漏洞识别过程。在此区域中，有一些漏洞就是因为分支代码过于模糊而极少被访问所造成的。正因为如此，有些代码很可能是存在漏洞的，因为它们从未被测试过。
- 暴力破解：这种类型的检查主要是针对用户 ID 和密码。与边界检查不同，它对能够访问的范围有一个非常广泛的限制。如果一个密码域由 8 个字母与数字长度组成，就有 36^8 种可能的组合。
- 缓冲区溢出：这种类型的检查是针对目标软件范围因素的测试，即针对那些范围可能会影响内存分配空间的输入。两种类型的范围因素需要被测试：缓冲区定义和缓冲区使用。例如，一个程序被设计为接受一个特定长度的输入并将其存储在内存缓冲中，一个稍大的输入就可能造成缓冲区溢出。这是一个缓冲区使用范围因素的例子，当输入的参数直接或间接地决定着分配的缓存空间的时候就会出现。如果输入的参数的大小大于缓冲区，那么就会出现缓冲区溢出。
- 代码注入：在许多脚本语言中，进程之间可能接受某段代码作为参数。程序员使用这种技术来达到高效的代码复用，或者将高级的参数从一个程序传递到另一个程序。程序的某个错误可能无意中引入接受终止指令的字符到代码里。它与 SQL 注入攻击很相似。
- 会话劫持：HTTP 本是一个无状态的协议。也就是说它在网络中接受业务请求，但是它并不知道这个业务是属于某个特定用户的会话。为了跟踪这种会话或者“状态”，使用

了一种称之为会话 cookie 的形式来实现。当然存在多种方式检查某个会话 cookie 是否被修改成别的用户，从而获取该用户的访问权限。

当然，所有的这些测试都可以单独自动运行。这里有一个关键因素来区分漏洞测试和渗透测试。用户使用应用程序漏洞扫描工具做检查测试，并结合各种结果形成更强的攻击。例如 SQL 注入攻击可用于插入恶意代码来获取其他用户的会话 cookie，进一步讲，不管会话 cookie 加密与否，都可以用来获取用户的信息或者更改他的密码来获取相应的权限。

附加功能

漏洞管理产业在不断地更新换代，技术体现的价值越大，产品就越有竞争力。漏洞管理工具也可以收集漏洞之外的信息。当核查一个主机的时候，为什么不检查一下其他的东西呢？可以枚举配置信息来核对与标准是否一致。只需要一点额外的努力就可以提供配置管理功能，这可是一个非常不错的机会。

既然能够检查与标准的一致性，为什么不根据美国国防信息系统局（DISA）提供的标准和建议来进行检查呢，这些标准和建议已通过美国国防部信息保障认证和认证过程（DIACAP）、美国国家标准和技术协会（NIST）的核准。PCI 是一系列商业安全标准。互联网安全中心（CIS）为保证系统安全提供了一套标准和各种配置项。CIS 是个例外，因为它的标准是广泛共识的，但在定制客户策略发展方面它们是很好的资源。

总之，附加的漏洞管理功能已经存在于定位在对每个可能的主机进行主动扫描的角色中了。被动扫描器的定义与网络嗅探器、性能检测器和一般的网络映射机相似。促进这些附加功能的关键是促进厂家提供这些功能。关于选择策略将在第 5 章中介绍。

4.6 混合方法

结合来自不同供应商的漏洞管理多种解决方案将有利于更加快速和完全地应对出现的漏洞。然而，标准化输出可能有困难。如果你能够使用同一厂家多种不同类型的技术，或许一个统一的控制台就可以消除这些问题。

一种可选的方法是按照组织或网络来分配评估资源。例如，在公共 DMZ 上使用被动漏洞扫描以获取 24 小时内主机的安全状态是有益的。这些最近产生的评估信息将自动填入到安全事件管理系统（SEIM）中。它有一个很大的优点，当新的利用漏洞的事件发生时，新发布的

漏洞能够更快地予以考虑。主动漏洞扫描可以获取对后端系统和工作站更深入的分析，而它们可能不需要快速地响应。

在内部网络中结合使用 DMZ 代理和主动扫描是一个不错的选择。代理被部署到 DMZ 主机上，这样就没必要主动扫描整个网络安全系统，相反的，它需要一个更加复杂的配置。另外，常规的 DMZ 审计和渗透测试应该规范化，代理形式可以作为主动扫描提供的常规检测的替代品。

4.7 推理扫描

最后一个扫描方法是推理扫描，它很少单独用于漏洞检测。该方法涉及对数据的分析，这些数据是为了另一个目的而获取的——检测漏洞的存在。例如，一个配置管理系统可能收集了一个组织目标机器上详细的配置数据，推理扫描过程使用无损方式从资源数据库中读取配置信息细节并分析它们的漏洞。一个简单的例子就是需要谨慎处理的配置项，例如 SNMP 群字符串或者是漏洞应用程序版本。

既然推理扫描是基于通过正常的搜集配置数据过程而获取的真实信息的，那么识别的漏洞就有很大的可靠性。当然，因为漏洞的检测过程不是通过主动探测网络中的主机执行的，它将不会对目标产生影响。当推理扫描被严格执行时，它就不是可靠和完整的了，因为它不能通过其他途径来验证。但是，它却可以补充前面提过的扫描过程或者作为一个附加的特性添加到配置管理产品中。更进一步讲，推理技术在框架上使漏洞扫描变得更有效率。例如，一个主动漏洞扫描可以收集所有可能的漏洞信息并记录下来以便分析。然后，推理引擎就分析主机上的数据来查找漏洞。在接下来的阶段中，在确定某些漏洞的类型前，这些漏洞将会被标记以便通过其他方式验证。总体而言，推理扫描是一个有价值的工具，但仅靠其自身来产生全面可靠的结果是不够的。

4.8 CVE

一旦收集了漏洞信息，必须对其分类和评估。评估和分类的方法对于制造商来说是千差万别的。但众多的产品总使用一个关键点来凸显自身的特色。当发现一个漏洞时，就根据必要的检测类型或者被授予的访问等级为该漏洞分配一个唯一的分类类型。为了讨论的方便，我们将避免使用某特定厂家的方案，而是使用 MITRE 的 CVE 方法。根据 MITRE 网站介绍：

CVE 是信息安全漏洞和暴露列表，旨在为公开的漏洞提供统一的名称。CVE 的目标是使用统一的列举方式跨越不同漏洞区分方法（工具、库和服务）来共享数据。

4.8.1 结构

MITRE 是一个非盈利组织，它多年来为漏洞管理做出很多有价值的贡献。它也提供一个开放、标准的平台来分享漏洞信息。当某人发现一种新漏洞时，他通常（但并非总是）向 MITRE 报告这个发现和详细的信息，MITRE 将快速地发布这些信息。不幸的是，这些标准仍然很难被不同产品所接受。在第 5 章中将会更加详细地讨论选择技术。

每个 CVE 都会被分配一个标识。从效果上讲，这个标识可以使来自不同制造商的不同工具使用同一种语言。一个 CVE 为所有的厂商提供相同的描述和对附加信息源使用相同的引用。例如，“CVE-2001-0010”代表在 BIND 8 中事务签名（Transaction Signature, TSIG）缓冲区溢出，它允许通过远程攻击来获取管理权限。这对于每个人来说都是被唯一理解的，不会在不同厂商间产生混乱。

在 CVE 中的引用最终将向 NIST 运营的美国家漏洞数据库（NVD）提交一个漏洞管理报告。前面提到的 CVE-2001-0010 在 NVD 中相关的信息，参见图 4-11：

- ❑ 综述：与 CVE 相似的关于漏洞的概述信息。
- ❑ 影响：在影响部分为漏洞评一个分数。关于“影响”的更多细节将在讨论 CVSS（通用漏洞评级系统）时提及。
- ❑ 关于漏洞报告、解决方案、工具的引用：这些是典型的网络引用，它们可提供关于漏洞的更多的详细信息、检测和补救方法。在这个实例中不同厂商提供了相关补丁的信息。
- ❑ 漏洞软件及其版本：拥有该漏洞的软件版本号列表，这能进一步帮助检测。
- ❑ 技术细节：关于漏洞的明确本质信息。例如，当利用该漏洞时软件会怎样反应，为什么这样的漏洞不好。同时，该项通常提供链接网站，在那里研究人员会发布一些关于他们发现的信息。

注意： CVE 是一些标识符而非实际的技术细节。CVE 的主要目的是对漏洞标识信息提供一个跨平台标准。为了保证标识机制的质量，每个漏洞都会经过一个审查过程。首先，给予漏洞一个候选状态，该状态意味着漏洞信息发布但是还未授权 CVE 状态。CVE 编辑委员会讨论候选者的价值并对漏洞是否应该成为 CVE 条目进行投票。

漏洞名称: CVE-2001-0010
 原始发布日期: 2001 年 2 月 12 日
 最近修正: 2005 年 5 月 2 日
 来源: US-CERT/NIST
 概述:
 BIND 8 中事务签名缓冲区溢出, 它允许通过远程攻击来获取管理权限。
 影响:
 CVSS 严重度 (版本 2.0 不完全近似)
 CVSS V2 基础得分: 10.0 (高) (AV:N/AC:L/Au:N/C:C/I:C/A:C) (图例)
 影响子得分: 10.0
 可利用子得分: 10.0
 访问媒介: 网络可利用
 访问复杂度: 低
 认证: 不需要
 影响类型: 提供管理员访问; 允许完全违反保密性、完整性、可用性; 允许未经授权信息的暴露; 允许中断服务。
 关于漏洞报告、解决方案、工具的引用:
 CERT/CC 报告: CA-2001-02
 名称: CA-2001-02
 类型: 报告、补丁信息
 超链接: <http://www.cert.org/advisories/CA-2001-02.html>
 外部源: Security Focus (免责)
 名称: bid 2302
 类型: 报告、补丁信息
 超链接: <http://www.securityfocus.com/bid/2302>
 外部源: PGP Security (免责)
 名称: Vulnerabilities in BIND 4 and 8
 类型: 报告、补丁信息
 超链接: <http://www.pgp.com/research/covert/advisories/047.asp>
 外部源: REDHAT (免责)
 名称: RHSA-2001:007
 超链接: <http://www.redhat.com/support/errata/RHSA-2001-007.html>
 外部源: NAI (免责)
 名称: 20010129 Vulnerabilities in BIND 4 and 8
 超链接: <http://www.nai.com/research/covert/advisories/047.asp>
 外部源: DEBIAN (免责)
 名称: DSA-026
 超链接: <http://www.debian.org/security/2001/dsa-026>
 漏洞软件和版本
 配置 1
 -ISC,BIND, 8.2.2 P7
 -ISC,BIND, 8.2.2 P6
 -ISC,BIND, 8.2.2 P5
 -ISC,BIND, 8.2.2 P4
 -ISC,BIND, 8.2.2 P3
 -ISC,BIND, 8.2.2 P2
 -ISC,BIND, 8.2.2 P1
 -ISC,BIND, 8.2.2
 -ISC,BIND, 8.2.1
 -ISC,BIND, 8.2
 技术细节
 漏洞类型, 没有可用漏洞类型匹配。
 CVE 标准漏洞条目
<http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2001-0010>
 通用平台列表
<http://nvd.nist.gov/cpe.cfm?cvename=CVE-2001-0010>

图 4-11 CVE-2001-0010

有几个关于 CVE 数据库的解释，第一，它不是一个漏洞数据库，而是一个漏洞索引数据库；第二，它不会包揽所有的漏洞，它只包含那些公开的漏洞。所以，存在一种可能，一个漏洞被厂商和研究人员关注，但是不会出现在 CVE 列表中。在某种情况下，这是因为研究人员与厂商达成一致，在公开的补丁发布之前不会公布漏洞。自然地，研究人员对该发现要求保证承诺。

继续我们的 CVE 讨论，CVE 有两个状态：候选或者进入。编辑委员会必须在授权进入状态前审查提交的漏洞。在此之前，漏洞拥有候选状态。当你要了解漏洞细节时，在 CVE 列表中会提供漏洞细节信息。在查看一个 CVE 时，检查一下状态并查看相关引用对提供的信息的信用度和精确度形成自己的认识。

4.8.2 CVE的局限

CVE 有一定的局限性并且绝对不可能会是所有漏洞管理标准问题的唯一答案。正如前面提到的，CVE 没有一个完整的、已存在的所有漏洞的列表。一些厂商能够识别 CVE 未记录的漏洞。并且，它没有必要包含所有所需的元数据来使漏洞系统具有所有的功能，而这正是厂商希望拥有的一种技术。自然地，这是不可能的，因为它试图提供一个对所有人都有用的通用命名信息。

CVE 并不总是及时更新，一些漏洞几年的时间仍保持着“CAN”状态，即候选状态。人们也许会好奇如果这些漏洞信息非常精确，它们将来是否会被更新。可能其中某些漏洞是配置最佳实践，但没有必要被认为是漏洞。相反，CVE 不包含所有产品的最佳实践配置漏洞，因为它们太多了以致没有办法去审查，并且现在全世界有成千上万种产品在使用。

4.9 漏洞测试数据标准

一般来说，某个漏洞扫描器或分析器的供应商将所需要的漏洞信息存储在一个数据库中，这个数据库与软件在工作时进行无缝连接。因为相对于技术发展，标准通常落后几年，所以没有任何数据库是一样的。不同的产品使用的漏洞信息和识别方法差异很大，也就是说，根本就没有统一标准存在。

然而，MITRE 有一个想法，就是为这些类型的数据结构建立一个标准，称为开放漏洞评估语言，简称 OVAL。正如名称所示，它是一种使用 XML 去记录具有某种漏洞的（目标）机器状态细节的结构化语言。它使用状态机方法标识漏洞，高度结构化地指示漏洞状态和非漏洞状态。因此，让我们看一个 OVAL 过程来了解它的优势，参见图 4-12。

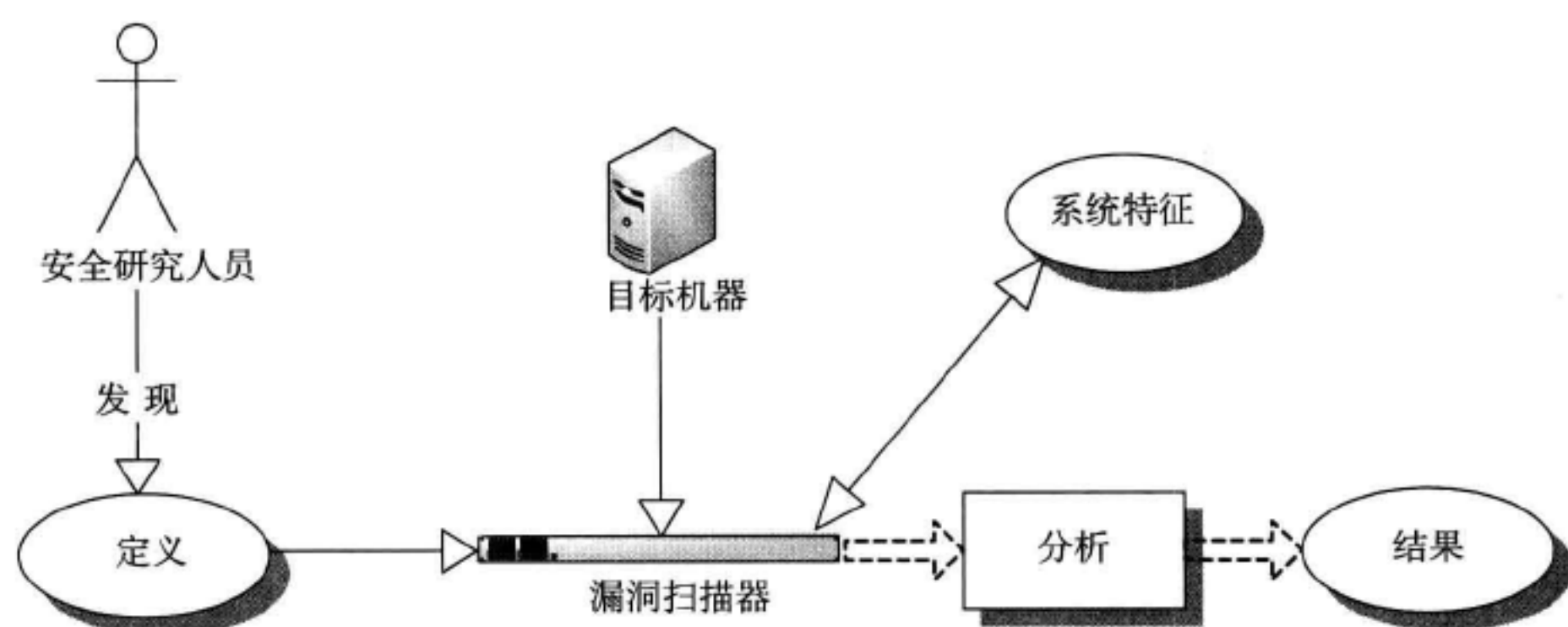


图 4-12 OVAL 过程

我们从安全研究人员在产品 X 中发现一个新漏洞开始，在了解了漏洞的第一手资料和怎样在一个易受攻击目标中识别漏洞后，研究人员如何使用一种 OVAL 标准化 XML 文档来编码漏洞详细信息。这个文档精确描述了如何通过提供易受攻击机器中特定项目的状态信息识别目标机器上的漏洞，这个漏洞识别方法被称为测试。

4.9.1 架构定义

从高层次上讲，OVAL 包含三种被称为架构的文档类型，这些定义的架构用来编码计算机系统必须存在的不同部分的状态，以便构建符合某个漏洞要求的机器。例如，一个目标系统必须是拥有服务包 2（SP2）、IIS 和匿名 Web 访问配置的 Windows 2003 Server，它对漏洞 XYZ 没有防御作用。这就提供了编码上述条件的框架。

其他的定义会捕获另外的重要信息来完善漏洞管理过程。补丁定义记录了某个符合要求的补丁所需要的条件，这主要是为了避免向系统不适当地任意打补丁。所以，提供的补丁是特定的，例如，Windows 2003 SP 2 只能用于安装了 SP1 的 Window 2003 的目标机漏洞修复。补丁定义的功能可能更深一些，不过上述解释对本书的范畴来讲已经足够了。

OVAL 定义正如名字所表示的那样，是对定义某个特定详单元素的描述。如果你想指定 SNMPv3 服务被安装在路由器上，必须检查某个元素是否存在，详单定义提供 XML 模式的实现正是如此。既然漏洞扫描和自动配置发现是如此紧密相连，那么 OVAL 也应该具备该能力。

合规管理是漏洞管理业界非常关注的另一个领域。MITRE 小组也意识到了之一点，并且在 OVAL 中制定了合规定义的架构。与其他架构相似，它定义了符合某个特定策略的条件，

而这是对漏洞管理工具和过程的扩展。

4.9.2 系统特征架构

该架构提供目标系统特征的标准定义，这些系统特征一旦被收集起来就会用来分析、识别漏洞。当所有的目标系统特征收集完毕，漏洞分析器就会将它们和定义的数据细节来比较以发现漏洞。系统特征架构的设计与架构定义大体相似。尽管该架构聚焦于漏洞，但它也计入到配置管理数据库的配置项目中。

4.9.3 结果架构

第三个，也是最后一个高级别架构用来提供记录漏洞评估结果的标准结构。它首要的优点是一旦发现某个漏洞，就以能被其他安全工具识别的形式捕捉相关细节，这样就可以及时给系统打补丁、更新配置、重新初始化变更进程，或采取其他必要措施。结果模式指明了在某个特定目标机上特定漏洞的细节。

4.9.4 测试描述[⊖]

在 OVAL 说明中，测试按 OVAL 的定义格式化并使用 <definitions> 标签定义。测试用三个关键 XML 项记录：<objects>，被测试的元素；<states>，测试对象的属性值；<tests>，使用针对 <states> 定义的对象进行的测试。每个定义、对象、状态和测试都有一个 ID，测试利用类似解析反向 DNS 项的符号标识。现在来看一个 OVAL 定义的部分实例，它检查是否安装了 Windows XP。这是一个简化的版本：

```
-----
1  <definitions>
2    <definition id="oval:org.mitre.oval:def:105"
      version="3" class="inventory">
3      <metadata>
4        <title>Microsoft Windows XP is installed<
          title>
5        <reference source="CPE" ref_id="cpe:/
          o:microsoft:windows_xp"/>
6        <description> The operating system installed
          on the system is Microsoft Windows XP. </
          description>
```

⊖ 假设你了解 HTML 和 XML 的基本知识，特别是需要知道 XML 标签是如何构建的，这也有助于理解面向对象设计中对象的概念。


```

7     </metadata>
8     <criteria operator="AND">
9         <criteria comment="the installed operating
            system is part of the Microsoft Windows
            family" test_ref="oval:org.mitre.
            oval:tst:99"/>
10        <criteria comment="a version of Microsoft
            Windows XP is installed" test_ref="oval:org.
            mitre.oval:tst:3"/>
11    </criteria>
12 </definition>
13 </definitions>
14 <tests>
15     <family_test id="oval:org.mitre.oval:tst:99"
            version="1" comment="the installed operating
            system is part of the Microsoft Windows family"
            check_existence="at_least_one_exists" check="only
            one">
16         <object object_ref="oval:org.mitre.
            oval:obj:99"/>
17         <state state_ref="oval:org.mitre.oval:ste:99"/>
18     </family_test>
19     <registry_test id="oval:org.mitre.oval:tst:3"
            version="1" comment="a version of Microsoft
            Windows XP is installed" check_existence="at_
            least_one_exists" check="at least one">
20         <object object_ref="oval:org.mitre.
            oval:obj:123"/>
21         <state state_ref="oval:org.mitre.oval:ste:3"/>
22     </registry_test>
23 </tests>
24 <objects>
25     <family_object id="oval:org.mitre.oval:obj:99"
            version="1" comment="This is the default family
            object. Only one family object should exist."/>
26     <registry_object id="oval:org.mitre.oval:obj:123"
            version="1" comment="Registry key that hold the
            current windows os version">
27         <hive>HKEY_LOCAL_MACHINE</hive>
28         <key>SOFTWARE\Microsoft\Windows NT\Current
            Version</key>
29         <name>CurrentVersion</name>
30     </registry_object>
31 </objects>
32 <states>
33     <family_state id="oval:org.mitre.oval:ste:99"
            version="1" comment="Microsoft Windows family">
34         <family>windows</family>
35     </family_state>
36     <registry_state id="oval:org.mitre.oval:ste:3"
            version="1" comment="The registry key value is
            5.1">
37         <value>5.1</value>
38     </registry_state>
39 </states>
-----

```

1. <definitions>

该例中有几个基本的组成部分：definitions、test、objects 和 states，它们是漏洞检查组件的高层容器。definitions 又分为四个不同的类型：vulnerability、patch、inventory 和 compliance。例子中，第 2 行中的 class 项是一个 inventory 定义，第 3 行是元数据标签。开始部分的描述语段提供给漏洞评估软件的最终用户，它们不需要了解测试如何进行。在此，标签 title 和 description 不是最重要的条目。注意到在第 5 行的元数据是引用源，它是一个 CPE 名称（将在后面解释），但这些已经足以说明一个系统的情况了。

这个实例最终的目的是判断 Windows XP 是否安装在目标系统中。通过对 tests、objects 和 states 的比较给出“是”或者“否”的答案。

第 8 行中，criteria 标签表明“operator”的值设置成了“AND”。这是应用漏洞检测的基本逻辑方法。如果 X AND Y，那么结果就是 true，换句话说，所有测试的结果都将使用逻辑与（AND）操作。接下来的 9、10 行是应用操作的两种 criterion 类型语句。

每个 criterion 或“criteriontype”都有注释和“test_ref”引用。这个引用指向将进行测试的细节。如果测试被标准否定，那么另一个测试项加到指定的标准语句中。这就是标准中的“negate=‘true’”语句。如果在某个测试类型上终止，那么测试被认为是“false”或没被排除。

正如该实例中的另一个准则：第一个准则说明“oval:org.mitre.oval:tst:99”测试应该执行。这是一个反向 DNS 模式的标识符，表明测试来源于 oval.mitre.org，编号是 99。所有的标识都以“oval”开始，紧跟着 org.mitre.oval。因为这是一个测试，字母“tst”随后，最后是一个冒号和至少有一个数字的整数。这些组合在一起构成一个唯一的定位符，被称为“testIDpattern”。

2. <tests>

当所有的测试都按定义的标准指定后，将开始一个新的部分，用标签 <tests> 指示。在该部分中，第一个也是最常见的元素是第 15 行的“family_test”，紧跟其后的是“id=oval:org.mitre.oval:tst:99”，这是前面 criterion 引用指向的元素。随后是需要比较的 objects 和 states。如果比较的结果是 true，就需要通过前面的定义评估结果是否符合预期目标。在元素 family_test 中有“check_existence=”和“check=.”，对应的值分别是“at_least_one_exists”和“only_one”。这表明有且只有一次检测。对比检测第 16 行的 object 99 和第 17 行的 state 99。

接着进行另一个测试。该例中，19 行是一个注册表测试。第 20、21 行是需要检测的 object 和 state。Object 编号是 123，state 编号是 3。

3. <objects>

在 objects 部分定义了需要测试的对象，本例中第 25 行的 “oval:org.mitre.oval:obj:99” 是一个 “family_object” 类型的引用。意味着这个对象定义了一个特定的系统，它只会被第 16 行的 “family_test” 所引用，它可以判断一个目标系统是 Windows、MAC、OS X、Unix 等。

下一个对象是 “registry_object。” 该对象类型只用于微软的注册表系统。它由三个项组成：<hive>、<key>、<name>。如果你查看过微软注册表，就明白其中的细节，Windows 注册表的结构和功能超出了本书的范围，在此不做介绍。这足以说明第 27、28 和 29 行的值是用来识别 Windows 注册表中用来识别特定项的值。

4. <states>

最后，解释一下 states 标签，它包含需要测试对象的值。第一项是 “family_state”，定义了计算机系统的状态。本例中第 33 行，“family_state” 标签告诉我们处理的是编号 99 的簇状态。你可能会想起来第 17 行引用该状态的 “family_test。” 回到第 34 行，“family” 的一般值被设置为 “Windows”。

定义的最后一个是前面提到的第 26 行对象的注册表，也叫做对象 123。为了分类，测试编号 123 的对象（注册表项）拥有状态 3（值为 5.1）。第 36 行，我们用第 37 行找到的值定义 “register_state” 元素。

下面用更加自然的语言来解释本例中 definitions、objects、states、tests 的用途：

```
-----
Definition: Inventory (line 2)
  This is a Windows OS (test 99, line 9)
  AND (line 8)
  Windows XP is installed (test 3, line 10)
Test 3: Test the Windows family (line 15)
  Which has object 99 (line 16) and state 99 (line 17)
Test 99: Test Windows XP
  Which has object 123 (line 20) and state 3 (line 21)
Object 99: family_object (line 25)
Object 123: registry_object (line 26)
  Which is "HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\
  Windows NT\CurrentVersion\CurrentVersion" (lines 27,
  28, and 29)
State: family_state (line 33)
  Which is Windows (line 34)
State: registry_state (line 36)
  Which has the value 5.1 (line 37)
-----
```

因为一些测试不能只使用 <state> 值，所以存在另一种类型的 <state>，称为 <variable>，

它允许最终用户选择某些值来构建规则。

MITRE 的研究者意识到他们不能为所有人定义所有的情况，所以制定了在 <metadata> 部分使用特殊 XML 标签的扩展 OVAL。

漏洞扫描器利用一个集成的 XML 文档来识别特定目标上的漏洞。在做比较之前，扫描器会评估目标机器上的状态，并在类似格式化的 XML 文档中记录这些状态信息，然后使用这些漏洞状态信息和目标机器状态信息做出分析。系统状态信息使用一种称之为 OVAL 系统特征架构的形式集成。这个 XML 文档有目标系统上的配置数据。使用标准化格式能规范化这些信息，方便与其他系统交流。例如，系统特征可能对配置管理工具和 SEIM 有所帮助。

如果漏洞扫描是为了识别漏洞，那么事实和相关的细节将被记录下来以便进一步处理。完成了分析 / 结果步骤后将能建立真正的厂商区分特征。但是首先，识别漏洞的过程被标准化了，以产生一致的结果。结果往往通过一种 XML Schema 规格说明的形式给出，这种 XML Schema 类似于一种被称作 OVAL Results Schema 的 OVAL 定义模式。需要注意的是，OVAL 只支持经过授权接入的目标的认证测试。这是一个必要的限制，因为有很多创新的方法可能会执行未经身份验证的检查。

4.10 漏洞危害程度评价标准

评估漏洞的一个重要部分是知道它对组织机构的影响或者风险。许多供应商都有他们自己的评估方法。但是，必须有一些被所有软件制造商和漏洞研究人员都认同的危害程度评价标准。通用漏洞评分系统（CVSS）就是开发出来提供评估漏洞影响和其基本特征的标准框架。尽管没有提供其内容和方法的全部细节，但是它能帮助我们在事件响应和安全小组论坛（FIRST）评估漏洞的风险之前做一些相关工作。FIRST 是一个由不同厂家、研究人员和致力于加强安全事件响应的其他志愿者组成的非盈利组织。

CVSS 提供相关漏洞评测因子让用户能查看和判断是否需要采取进一步措施来应对风险。这些标准分为三组：基本、临时和环境。对于每个组将计算一个分值。每个组有一些测量因子综合计算出该组的分值。图 4-13 展示了因子组、因子和等式之间的关系。在接下来的讨论中我们会不时地提到这张图。需要澄清的是，使用虚线标识的元素是子等式或者子组，它们只提供中间值或者逻辑因子组。

基本因子组包含一些常量，它们不怎么变动，包括：有访问载体（AV）、访问复杂

度 (AC)、认证 (AU)、保密性影响 (C)、完整性影响 (I) 和可用性影响 (A)。正如每个 CISSP 人员应该知道的, 保密性、完整性、可用性 (CIA) 是安全的三要素。

基本因子组中的每个因子都有一个值来表明严重性和影响。例如, AV 表明攻击者要利用该漏洞需要哪种类型的访问权限。如果漏洞需要攻击者亲临现场并且使用键盘 (例如, 本地访问), 那么该因子的值为 0.395。如果漏洞可以通过网络 (远程访问) 来利用, 那么该因子的值为 1.0。这一过程在测量应用于计算评估漏洞的所有因子中反复进行。CIA 因子合起来就是影响因子, 并组合在一起计算出最终的影响度, 它是公式中的基本分值。完成所有测试后, 使用公式计算出该组的总分数。

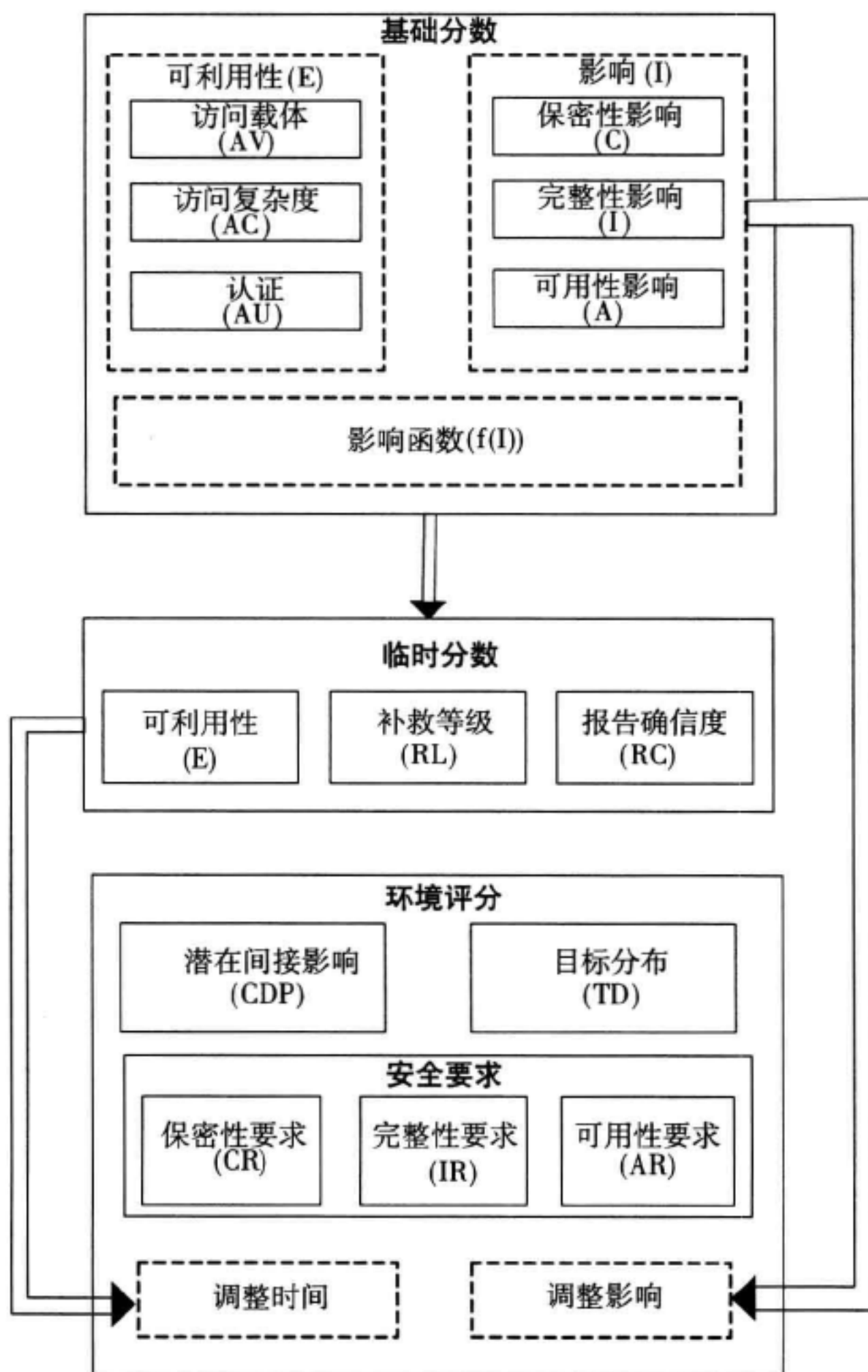


图 4-13 在 CVSS 中的因子关系

因子使用这些特殊的值是为了便于理解一个漏洞利用的有效性以及该因子在计算整体组得分中的重要性。影响越大，其值也就越大。但是，并不是所有的因子都是平等的。对于一个漏洞整体严重度，AV 可能比漏洞利用的复杂性更重要。低复杂度（0.35）和高复杂度（0.71）相差 0.36。但是需要本地访问（0.395）和网络可访问（1.0）相差 0.605。AV、AC 和 AU 是决定漏洞最终可利用性的三个基本因子。可利用性的公式是： $E=20*AV*AC*AU$ 。公式和因子的值已经都为你计算出来了，这样可以节省不少时间。

临时因子是可选的，其值也是随着时间的变化而改变的。基本因子被当做输入用在“临时”计算中，产生一个分值，它在范围 0 ~ 10 之间，可以更精确地反映风险。例如，一个漏洞可能处在提交证据阶段，意味着它具有较小的威胁，因此会赋一个 0.9 的值。随着时间的流逝，一个自动的脚本可能变得很流行，它使这个漏洞的利用更方便，甚至一个小孩子都可以做到，那么对于该因子将被赋值为 1.0。临时公式使用分类函数，它能够通过乘以先前提及的因子，从基本公式调节有影响的计算。表 4-6 详细表述了 CVSS 因子和它们的值。注意，每个值还有一个数字分值并未列到表中。这些分值是随着公式而变化的，随后会详细介绍。

环境因子组是另一个可选的，但是又很有用的一个组。这些因子独立于其他因子组，但是又对其他因子组是一个补充。如果不使用该组，也不会影响其他因子组的评价。是由你（一个 CVSS 用户）来决定是否需要它。然而，为了本书内容的系统化，所以同样需要解释它。环境因子组包括潜在间接影响（CDP）、目标分布（TD）和安全要求，安全要求包括保密性（CR）、完整性（IR）和可用性（AR）。它还包括一个从基本因子和可调整的临时分数而得到的最终调整后的影响分值。

表 4-6 CVSS 因子

因子类型	CVSS 因子	描述	值
基本	访问载体	需要本地访问	0.395
		相邻网络访问	0.646
		互联网访问	1.0
	访问复杂度	高	0.35
		中	0.61
		低	0.71
	认证	需要多种形式认证	0.45
		需要单一形式认证	0.56
		不需要认证	0.704

(续)

因子类型	CVSS 因子	描述	值
基本	保密性影响	无	0.0
		部分	0.275
		完全	0.660
	完整性影响	无	0.0
		部分	0.275
		完全	0.660
	可用性影响	无	0.0
		部分	0.275
		完全	0.660
临时	可利用性	未证实	0.85
		提供证据阶段	0.9
		功能	0.95
		高	1.0
		未定义	1.00
	补救等级	官方修正	0.87
		临时修正	0.90
		替代方案	0.95
		没有	1.00
		未定义	1.00
	报告可信度	未验证	0.90
		未确证	0.95
		确证	1.00
		未定义	1.00
环境	潜在间接影响	无	0
		低	0.1
		中低	0.3
		中高	0.4
		高	0.5
		未定义	0.0
	目标分布	无	0
		低	0.25
		中	0.75
		高	1.0
		未定义	1.0

(续)

因子类型	CVSS 因子	描述	值
环境	保密要求	低	0.5
		中	1.0
		高	1.51
		未定义	1.0
	完整要求	低	0.5
		中	1.0
		高	1.51
		未定义	1.0
	可用要求	低	0.5
		中	1.0
		高	1.51
		未定义	1.0

CDP 是一个衡量潜在经济损失和伤亡代价的经典风险管理方法。在风险管理里，它被称作单一损失期望 (SLE)。对于那些没有受过正规训练的安全专家，在风险管理中，SLE 就是当一个错误发生一次时产生的期望的风险代价。尽管它是一个衡量潜在损失的方法，并且 CDP 在 0 ~ 0.5 的范围内，但它不代表实际的金钱值。

TD 是衡量机构受到攻击比例的方法，它能帮助评估环境中受到威胁的范围。如果 50% 的目标主机都有这个漏洞，那么这个因子会被设置成中等。当 TD 使用公式来计算时，High=1.0，medium=0.75，low=0.25，none=0.0，有趣的是，not defined=1.0。之所以觉得有趣是因为不清楚漏洞的 TD，就假设它是 “high”，这样允许进一步评估它潜在的破坏性。我们建议如果根据一个漏洞评测结果知道组织里存在漏洞的主机分布，那么需要将这个百分比换算成十进制数。这个方法超出了 CVSS 的指导意见，但是它比 high/medium/low 方法更精确。

环境安全要求因子是独特的，这些因子给每个 CIA 基本因子一个权重。例如，如果你的特定环境注重数据的可信性，那么这个值会增大。如果这个值是中等，C 等级比较合适。在基础分数中安全要求是用来重新衡量在基础分数中的影响因子 (I)，需要根据组织内环境的不同要求重新修改基本因子组的评估分数。然而，如果在基本组中一个影响因子是 0（即不是一个影响因素），最终修改的影响分数将不受影响。这是因为应用于修改的影响因子的是安全要求和基本组的影响值相乘得到的：

$$\text{调整后的影响} = \left(10.41 \times \left(1 - \min \left(\begin{aligned} &(1 - \text{保密性影响} \times \text{保密性要求}) \times \\ &(1 - \text{完整性影响} \times \text{完整性要求}) \times \\ &(1 - \text{可用性影响} \times \text{可用性要求}) \times \end{aligned} \right) \right) \right)$$

其中调整后的影响 = min (10, 10.41 × (1 - (1 - 保密性影响 × 保密性要求) × (1 - 完整性影响 × 完整性要求) × (1 - 可用性影响 × 可用性要求)))。

对于每个因子组，根据一系列数学规则设计了一个用来计算分值的等式。该等式是在根据度量类型而定的推理的基础上构造出来的。该等式是根据度量类型而定的推理的基础上构造出来的。这些公式中每个因子都是经过广泛分析和讨论过，在这里就没有必要讨论了。更多讨论 CVSS 的解释细节参见 <http://www.first.org/cvss/cvss-guide.html>。

如果想计算自己系统的 CVSS 分值，可以使用一个 Web 计算器，一个非常流行的计算器可以在 <http://nvd.nist.gov/cvss.cfm?calculator&adv&version=2> 找到。可以输入任何值然后计算出 CVSS 评分。要了解某个特定因子在整体评分中的比重，可以只改变该值然后重新计算。你将会意识到什么分值是好的，什么分值是糟糕的。如果去除环境因子，你就会更加了解 NVD 上的 CVSS 分数。我们将在 4.11 节讨论这些问题。

CVSS的命名

一个简略的语言可以使 CVSS 更易转述漏洞因子细节。前面章节中提供的字符串将使用冒号 (:) 来连接字母和对应的值，用斜线 (/) 用来区分不同的因子，例如，一个基本组可能表示成如下形式：

10.0 (High) (AV:N/AC:L/AU:N/C:C/I:C/A:C)

基本组的分值为 10.0，AV 的值是“N”（代表网络），下一个因子使用斜线分割，AC 的值是“L”（代表低），这些基本因子的值随后，直到所有基本组的因子都在括号中囊括。在本例中有三个因子，保密性影响、完整性影响和可用性影响，都用“C”表示会产生混淆。

正如前面描述的，这些项将出现在 CVE 标识符中，并且通常包含影响性子分数和可用性子分数，以便了解某个漏洞能造成的潜在危害。如果能够真正地理解，那么这个缩略系统能让你快速获取一些有用的信息。CVE 标识符为所有组织和产品广泛共享漏洞信息提供了重要帮助。使用 CVE 让你清楚地了解某个产品中有哪个漏洞。在管理漏洞方面需要整合各种不同

的信息源来最大化这些信息源的可用性。CVE 数据、OVAL 检测和 CVSS 评分，连同互操作的标准协议都可以在 NVD 中找到。

4.11 美国国家漏洞库

美国国家漏洞数据库（NVD）是一个由 NIST 运营的在线数据库，可以在 <http://nvd.nist.gov/nvd.cfm> 中找到。NVD 使用安全内容自动控制协议（SCAP），这个协议包含一系列支持漏洞管理自动化、合规管理和其他安全功能的标准。我们已经讨论了其中的一些标准，例如 OVAL、CVE 和 CVSS，还有三个标准没有讨论：CCE、CPE（涉及目标条例）和 XCCDF（为目标评估提供检查列表和为报告提供标准格式）。

CCE 是通用配置枚举标识符，是用于关联系统文档配置管理检查和提供相关信息工具的标识符。我们不对 CCE 标识符进行深入讨论，如需要可查看 Mitre 公司提供的 CCE 列表。

4.11.1 CPE

通用平台枚举标识符为技术型系统和组件提供一个标准的命名框架，在实际漏洞管理概念下，CPE 标识符用来指明什么样的系统或者组件有某种类型的漏洞。当一个新的漏洞发布时，“哪个系统易受攻击”是第一个被提及的问题。CPE 清晰的文档化系统平台，让漏洞的发布很容易被自动或者人为的方法确定。

某个计算机系统会被分配一个 CPE 名称，它列举了系统平台所安装软件的详细列表，包括硬件、操作系统和应用程序，但它不包括软件详细的配置选项，例如某个特定开关或者安全策略的状态。所以，你会遇到的第一件事情可能就是有百万个组合需要枚举。确实是这样，但是 CPE 有一个基本的要求来解决这个问题。如果一个漏洞是用“`cpe:/o:microsoft:windows_xp::sp2`”发布的，那么使用 CPE 名称如“`cpe:/a:microsoft:office: 2003::standard`”的系统也有该漏洞，这是一种用于枚举漏洞系统主题的“分组”方法，在 CPE 用法中称之为前缀。既然一个平台可以有多个部分组成，那么对平台的枚举就需要多个 CPE 名称。

1. 编码

CPE 名称的编码在逻辑上被结构化为类似于之前章节描述的硬件、操作系统和应用软件等。虽然不会被官方 IANA（管理因特网，分配经常见的 URL 的机构）所认定，但是编码是遵从 URI 格式的。这种格式对于既定惯例的利用非常方便，这些惯例能够很好地命名互联网上

的资源。下面是一个 CPE 名称的基本结构：

```
cpe:/ {part} : {vendor} : {product} : {version} : {update} : {edition} :  
{language}
```

所以，该格式由七部分组成：

- Part：该部分定义了它是一个硬件“h”，操作系统“o”或应用程序“a”。MITRE 已经预留了一些其他元素，例如磁盘“d”。
- Vendor：使用厂家的域名的一部分来表示，例如 Mozilla.Org 代表厂家 Mozilla。严格地讲，它是一个厂家的最高的域名。如果多个组织公用一个 DNS 域名，那么就需要使用完整的 DNS 域名。
- Product：CPE 使用厂商提供的软件的简写形式，用户通常使用 IE 来表示 Internet Explorer。所以，会使用简写形式。
- Version：产品的版本号，例如，“5.0”。
- Update：厂商可能对特定版本应用了更新，这些更新可能是不定时的，依赖于厂商是怎样发布更新版本的。有些厂商习惯使用小版本号来发布更新。
- Edition：edition 域用来区分同一产品的不同配置版本。所以，Windows Vista 有多个版本，例如家庭基本版、家庭高级版、商业版和旗舰版。
- Language：这自然是目标 CPE 名称使用的语言。这使得我们可以根据自己已安装的语言包来识别有漏洞隐患的系统。

2. CPE 命名范例

显而易见，`cpe:/a:adobe:flash_player:1.1` 代表 Adobe Reader 版本 8.1.1。它并没有指明操作系统，这说明所有 8.1.1 版本的 Adobe Reader 在所有的系统都包含该漏洞。如果想修改这个标识符，使其仅适用于 Window XP 系统，那么可以使用下面的名称：`cpe:/a:adobe:flash_player:9.0.20.0::windows_xp`。如果指明了特定的操作系统（例如，Windows XP，所有版本），那么下面的名称将被使用：`cpe:/o:microsoft:windows_xp`。然而，如果我们需要更详细的信息，如 Windows XP SP1 Pro，那么可以使用：`cpe:/o:microsoft:windows_xp::sp1:professional`。

硬件信息也使用相同的方式：`cpe:/h:cisco:ip_phone_7960`，其代表思科（Cisco）iPhone 机型 7960。注意，机型号不是版本号，它是嵌入到产品中的，这是因为思科使用不同的机型号

来代表不同的产品型号。当标识硬件时，可能不仅需要标识计算机系统，同时还要标识特定的主板型号，例如 Intel D845WN (`cpe:/h:intel:d845wn_motherboard`)。

4.11.2 XCCDF

前面讨论了使用 OVAL 标准化漏洞测试方法的必要性。当然，我们讨论了数据结构怎样依赖于漏洞扫描器。同样的，可扩展配置检查列表描述格式 (XCCDF) 是基于 XML 的一系列文档，这些文档是详细描述了针对各种类型的目标系统使之安全合规的检测列表。XCCDF 也描述报告的合规性和评分的标准格式，这简化了各种安全系统的交互操作。它不是 OVAL 的替代品，但是它支持扩展的 OVAL，并加强所有权技术之间的互操作性。

XCCDF 有一个基本用途就是定义合规性检查、机器状态合规性和结果报告。该语言允许定义反应在详细配置信息中的安全基准，形成检查信息并提交给 NIST 检查列表程序来复审。如果接受了，那么这个检查信息就会被支持 XCCDF 的产品所使用。但是很少有厂家使用，XCCDF 基本上只被在 NIST 的 800-53 和 FIPS199 支持下的美国政府安全部门所使用。该语言最大的好处就是它提供了一个标准的机制来验证和报告在不同厂家和安全系统之间在检查列表和规则上的安全一致性。XCCDF 支持把漏洞管理看做配置管理的整体的看法。

首先，你或许会想到任何一个这样的语言在遇到定义检查和评分时必定会有所限制。然而 XCCDF 是可定制的，可应用于完成不同的系统之间一致性结果的检查。例如，某个部分是可选的。一个特定的 XCCDF 文档可能包含一系列描述目标状态一致性的规则，这些规则是可选的，可以打开也可以关闭，这依赖于详细审查下的目标机器。同样的，参数可以被替换来形成灵活的配置规则。例如，在数据不重要的情况下，一个 VPN 配置的加密密钥是 256 位，在传送敏感信息时使用 4096 位。

XCCDF 有四种类型对象：

- ❑ **benchmark**：是文档的主容器，类似于 OVAL 的 `defininitions`。
- ❑ **item**：类似于 OVAL 的 `object` 和 `test`，它包括描述和标识符。它有三种不同类型的 `item`：`group`（可包含其他的 `items`），`rule`（包含检测、评分、权重和补救信息），`value`（提供前面提到的替代能力）。
- ❑ **profile**：提供对 `item` 对象的引用，它包含构建一个系统特定框架所需要的值。它能根据规则所使用的值来恰当评定资源分类。

- test result：包含测试的结果，在这个对象里面最重要是“rule-result”和“target-facts”元素。该对象包含实际的测试结果和报告以及修复过程中产生的有用信息。Target-facts信息可以是敏感信息，这些敏感信息与对目标机器测试合规性或非合规性得到的实际结果相关。

XCCDF 中一个有趣的属性是它会在规则中参考 OVAL 的内容。规则中的一个叫“check”的子元素允许文档作者引用从中获得某个系统，而检测正来自于该系统。例如，‘<cdf:check system=http://www.mitre.org/XMLSchema/oval>’，紧接着是对源系统的引用：“<cdf:check-content-refhref=” ovaldefs.xml” name=” OVAL99” >”。它只是对 OVAL 规则的引用，但不是事实上的规则。执行检查的软件将适当地解析并执行相应的引用。OVAL 的使用对所有系统不是必须的，除非检查的目标是 SCAP 合规性，我们接下来将讨论这个问题。

总之，XCCDF 是一个伟大的观念，它能将不同厂商和组织之间的配置合规性检查标准化。然而，它的使用大部分局限在美国国防部。如果作为制造商，并且想与政府合作，那么可以从 Neal Ziring 和 Stephen D. Quinn 撰写的文档《Specification for the Extensible Configuration Checklist Description Format (XCCDF) Version 1.1.3 (Draft)》中获取更多细节。

4.12 SCAP

安全内容自动控制协议（SCAP，发音为“ess-cap”）是前面提到的标准的集合，包括 CVE、CVSS、CPE、XCCDF 和 OVAL。NIST 维护 SCAP 的内容，其中定义了所有的这些协议怎样通过自动的方式相互配合。它也包含 NVD 中的所有标准内容。

SCAP 也有一个验证程序来帮助评估不同开放标准下产品的兼容性。NIST 提供对验证方面的详细描述。在这里给出一个验证方面的简略描述：

- 联邦桌面核心配置（FDCC）扫描器：一个能够核查和评估目标系统是否和 FDCC 要求一致的产品，这些 FDCC 要求是美国政府 OMB 备忘录 M-07-18 的结果。这个备忘录陈述了信息技术提供商应该保证应用能够具备完整功能，并且能按照要求使用 FDCC 在系统上正确运行。
- 经认证的配置扫描器：使用目标系统登录权限的能够审核和评估目标系统是否符合配置需求的产品。

- 经认证的漏洞和补丁扫描器：具有目标系统登录权限的，能够扫描目标系统并定位和识别已知软件的缺陷及评估软件补丁的状态来决定目标系统是否与已定义的补丁策略相一致的产品。
- 未认证的漏洞扫描器：能够通过评估在网络上的目标系统来判断是否存在已知的软件缺陷的产品。
- 入侵检测和防护系统：监控系统或者网络是否存在未授权的或恶意的行为的产品。IPS 主动保护目标系统或网络不受这些行为的攻击。
- 补丁补救：在目标机器上安装补丁以与已定义的补丁策略保持一致。
- 错误配置补救：对目标机器上配置作出调整，来让系统与已定义的配置补救措施相一致。
- 资产管理：主动发现、审查和评估资产特性，包括已安装的和已注册的软件产品的能力；在世界、网络或者企业中的位置；资产所有权；其他与 IT 资产（例如，工作站、服务器和路由器）相关的信息。
- 资产数据库：被动存储和报告资产特性，包括已安装的和注册的软件产品的能力；在世界，网络或者企业中的位置；其他与 IT 资产（例如，工作站、服务器和路由器）相关的信息。
- 漏洞数据库：包含 CVE 标记的安全相关的软件缺陷目录的产品。该数据库通过检索为用户提供访问，其中包含软件缺陷的描述，对附加信息的引用（例如，到补丁或者漏洞信息的链接）和影响值。
- 错误配置数据库：包含有 CVE 标记的安全相关的配置问题目录的产品。
- 恶意软件检测工具：能够识别并报告目标系统中的病毒、木马、间谍软件或其他恶意软件。

当一个产品被评估和验证时，需要从多个方面考察。产品的验证状态将被发布到 NIST 的公共 Web 站点上。经过了验证并不能保证产品的质量 and 可靠性，只是它满足了 SCAP 程序所定义的标准才能保证。

4.13 Nessus

不介绍 Nessus 就等于没有完整地介绍漏洞管理技术。一些公司使用开源版本的 Nessus 来降低成本并最大限度地控制漏洞。现在也有很多介绍 Nessus 使用方法的书籍，所以我们不会深

入探讨 Nessus。如果你打算使用开源版本，建议购买一本相关的书。Tenable Network Security 有偿提供对 Nessus 的更新和维护。下面将简单介绍这款产品。

4.13.1 优点与缺点

Nessus 是一个常用的免费开源扫描器，对于不打算在这类产品上投资的公司来说是一个很好的选择。该软件有优于其他许多产品的特性，当然也有一些缺点，如表 4-7 所示。

表 4-7 Nessus 的优点与缺点

项 目	优 点	缺 点
单一服务器平台扫描，并捕捉结果到数据库	高性能地数据捕捉，最小化报告结果对网络的影响	强制要求使用集中服务架构，所有的扫描在一个服务器上进行
开源产品	低成本使用，可通过专门知识由高级用户定制使用	不交额外费用不提供支持。需要很强的安装和运行产品的知识
用户自己编译代码	能在多个平台上运行：OS/CPU	需要很强的关于目标系统和开源软件的知识
推荐使用 Nessus 的优化版本扫描 Windows XP SP2 平台来避免错误否定问题	测量问题：如果你的组织使用了多平台（如 Linux 和 Windows），那么最好两个版本都要使用，如果不能使用，则最好使用 Windows 版本	
专业版本提供即时更新	能收到最新的漏洞更新	你必须花钱买这些更新，但是价格可能与同类产品相同或者低于
家庭版本提供免费漏洞更新	这是获取工具评估开端的一种好的方式	不能商用
插件	Nessus 的这些允许可扩展和可定制的元素在其他产品中通常不提供	增长的复杂性要求具备一定的知识和经验才能部署
NASL ^①	它允许用户使用脚本来运行特定的漏洞检查。这些检查能提供大多数产品没有的特性	需要了解 NASL 和怎么使用命令行

① Nessus 攻击脚本语言。

4.13.2 扫描模型

Nessus 提供三种类型的检查和扫描模式：

- ❑ 发现模式：这个过程使用基本探测协议，例如 ICMP 应答质询 / 响应和 TCP 扫描来识别网络中的活动主机。现在某些产品在没有进行全面的核查的情况下是不能进行一次简单的快速的漏洞发现的。而拥有这项能力则表示这个工具在商业产品中能够在定义

网络时缩减未知网络的范围。当然，也可以使用命令行形式的 NMAP 产品来实现类似的功能。

- ❑ 操作系统指纹：可以通过本书中讨论的很多方法来实现该操作。畸形数据包和合法的 SNMP 请求被分别用于收集被动信息或者主动信息。这些或者其他方法并不一定总有效，主要依赖于目标系统。其他的产品可能使用其他的方式来收集这些信息。
- ❑ 完全扫描：该类型的扫描执行发现和操作系统指纹识别操作，还会进行多种漏洞检查，包括暴力密码攻击。它也有前面章节中提到的相同的限制和问题。

一个值得一提的 Nessus 的特性就是对 Web 应用程序的检测。这个特性许多厂商是不会免费提供的。大多数情况下，需要另行购买该特性。Nessus 毫无保留地提供了这一功能。不过需要进行配置来与目标进行匹配，但这肯定是值得的。用户可以自己测试跨站脚本攻击 (XSS)、SQL 注入攻击和通用网关接口漏洞等。

与其他产品相比，在高层控制方面 Nessus 是一个很不错的产品。然而它缺少其他商业软件所提供的可扩展性，因为它是使用集中扫描方式。所有的扫描都发生在一个中心位置，而不是具有一个中心数据汇集点的多物理分布采集系统。然而，这对许多商业 /IT 架构来说已经足够，并且价格也适中。

4.13.3 使用Nessus

本书不提供安装的步骤，因为其他的书和 Nessus 文档都有全面的介绍，本节假定用户已经安装了 Nessus 客户端和服务端。Nessus 在用户界面上有两部分：扫描和报告。图 4-14 显示的是扫描目标对话框。它允许用户输入一个主机的 IP 地址、DNS 域名、IP 地址范围，子网说明或者是有主机或 IP 地址的输入列表文件。每个这样的目标说明都会保存下来以便将来重复使用。

接下来，用户需要通过创建扫描策略来定义扫描的参数。图 4-15 中显示了该策略包含的内容：

- ❑ 扫描的基本选项（攻击、捕包、端口扫描类型、登录，等等）。
- ❑ 证书（如果使用白盒测试）。



图 4-14 目标扫描对话框

- ❑ 插件选择，使用最新的和最相关的检测。Nessus 的插件库是可扩展的。高效扫描的关键是只选择目标网络所需的插件。
- ❑ 网络拥塞控制配置（并发连接、超时和目标脱离规则）。
- ❑ 针对不同协议和检测的高级配置参数。

一旦所有的参数都设置好了，就可以进行扫描了。完成后就会有一个层次化的适于阅读的报告。这个报告可以导出为 HTML 负面形式，且很容易看懂，如图 4-16 所示。这个实例显示了用于获取信息的端口和协议，Nessus (tcp/1241) 使用的是 SSL (协议)。接着，报告显示支持的 SSL 加密插件被用于获取漏洞信息。在该实例下它并不是漏洞而是一个列表。然后，报告中剩余的部分都是一目了然很容易看懂的内容了。Nessus 一个非常好的特性就是能打印插件的输出信息，在该实例中它打印了 SSL 服务所显示的扩展信息。更好的特性是，对于一个倾向于技术型的分析员，Nessus ID 将能够通过 Tenable 网站中的链接来了解更多有关插件的信息。这些插件有相应的源代码供你参考，这样可以让你更全面地了解检测方法。

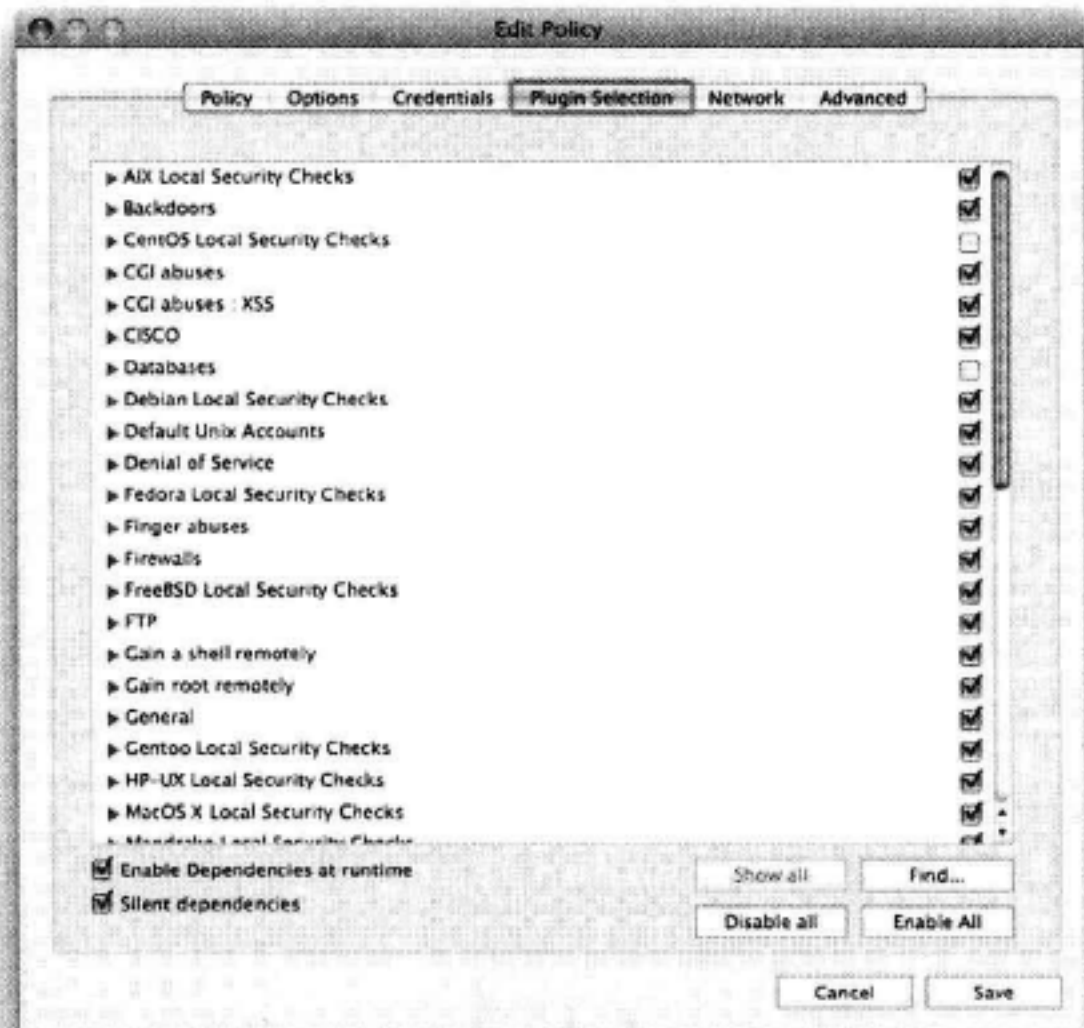


图 4-15 Nessus 漏洞扫描插件选项

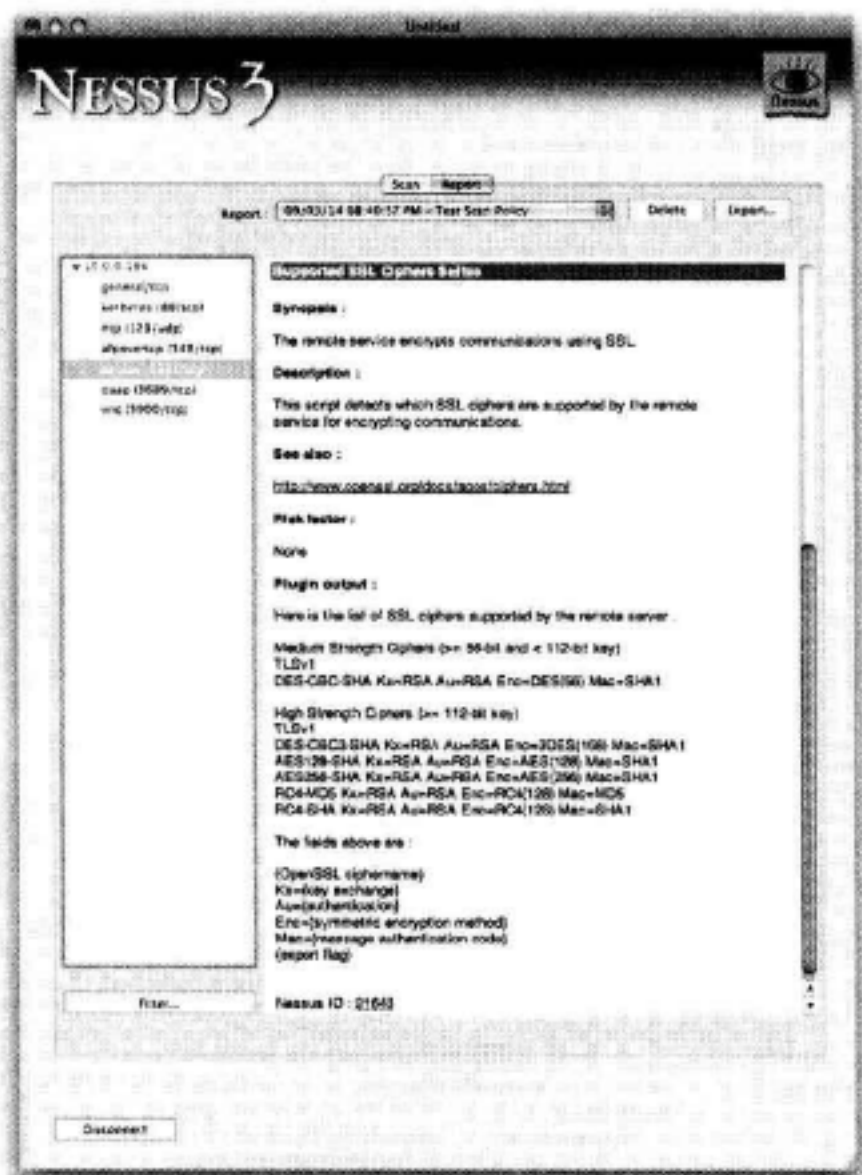
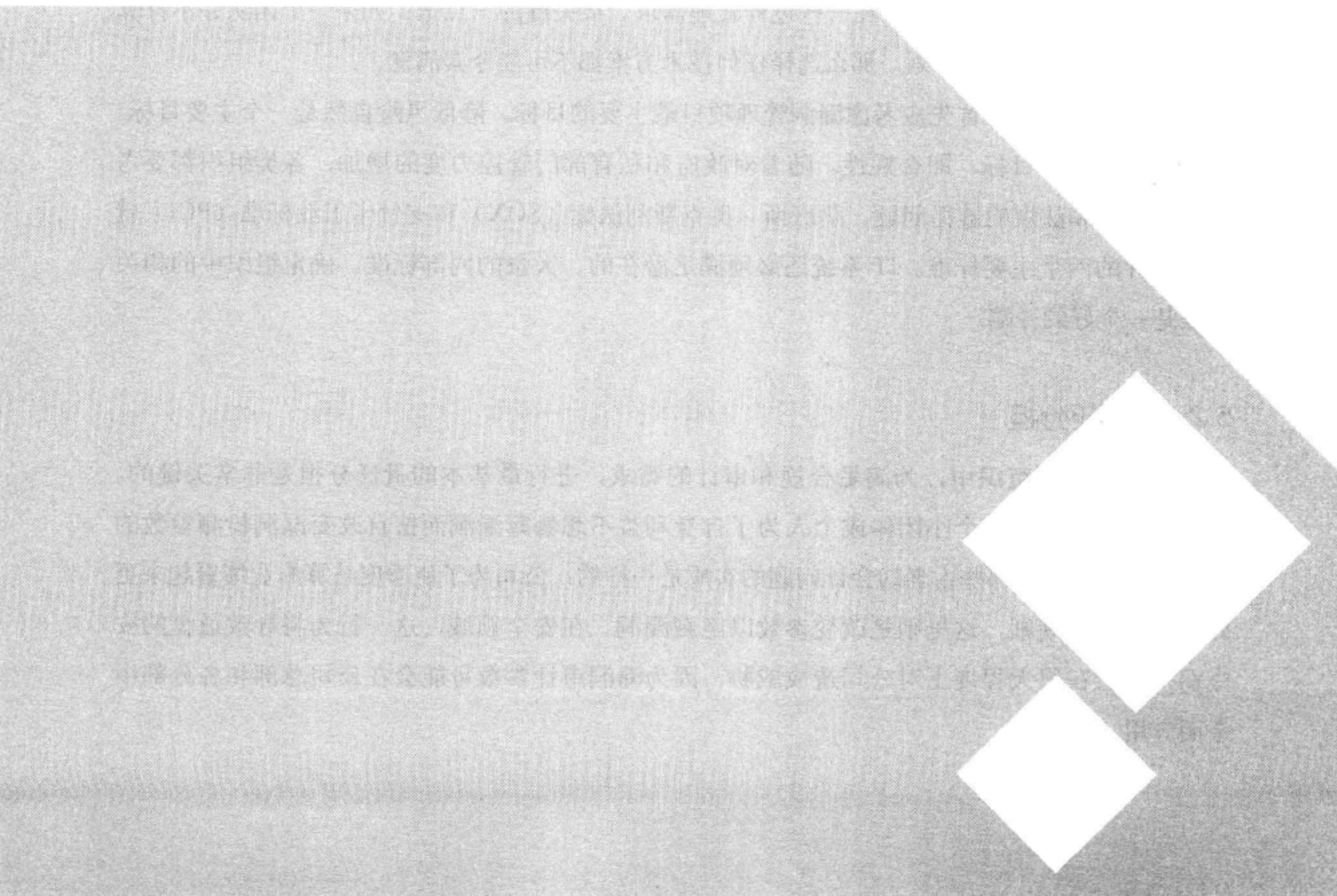


图 4-16 Nessus 漏洞细节报告

第5章

选择技术



5.1 概述

选择漏洞管理产品是一个复杂的过程。由于销售商的可靠性在不断增强，漏洞从发现到修复的时间在逐渐减少，因此在选择产品时不能只依据这两个因素，必须引入其他衡量标准。产品选择的第一步也是最重要的一步就是理解需求。开始时，人们可能意识不到需求的多样性，而只是将用户界面设计作为选择标准，但事实并非如此。本章将重点阐述那些最为重要的需求，以及关乎整个产业未来发展的一些事项。

5.2 总体需求

在进行技术选择时，最重要的是要先明确总体要求。这些需求与公司在漏洞管理项目中的主要目标紧密相关。常见的需求有：准确地监测漏洞、跟踪修补状态和汇报处理进展。其他一些需求并不常见，例如，是否由不同的团队执行漏洞管理的不同方面？在一些情况下，漏洞检测参数通常由一个合规小组来定义和管理，而安全和网络管理团队则负责计划安排和带宽设置。除技术需求外，还存在一些过程管理需求。很关键的一点是：如果一个团队并不打算进行企业级的漏洞管理计划，那么选择任何技术方案都不可能令其满意。

为了确定需求，首先应考虑漏洞管理项目最主要的目标。降低风险自然是一个主要目标。但还有另外一个目标，即合规性。随着对政府和私有部门管控力度的增加，各类组织都要考虑许多标准和法规的遵循问题。萨班斯-奥克斯利法案（SOX）和支付卡工业标准（PCI）就是需遵守的两个主要标准。IT 系统还必须满足潜在的、大量的内部标准。确定组织中的相关标准是一个好的开端。

5.2.1 责任分担

在一个大型组织中，为满足合规和审计的需求，进行最基本的责任分担是非常关键的。主要应考虑规避因一个小团体或个人为了自身利益不想暴露漏洞而擅自改变漏洞检测参数的情况。这种行为和那些古老的会计问题的实质是一样的：公司为了使季度结算的业绩看起来更好而改变记账规则，这里则是改变参数以隐藏漏洞。在安全领域，这一行为将导致通盘的安全弱点，会在很大程度上对公司造成威胁，因为漏洞审计参数可能会在公司总部和各分部中全面应用。

责任分担不仅有利于合规管理，而且使得操作更为简单有效。将 IT 服务传递给业务部门，不同组织都负有责任，并且依赖于其他组织。在向公司提供 IT 服务的过程中，各小组都负有责任并依赖于其他小组。漏洞管理在提供 IT 服务的过程中，就扮演着这样一个责任小组的角色。例如，网络部门可能需要负责维护服务级别协议（SLA），因为 SLA 可能会受漏洞扫描影响。这些服务级别需要根据内部客户的需求而进行调整，并可能包含许多复杂的因素。内部的服务提供者，可能是一个网络组或服务平台组，将需要对漏洞评估过程中服务资源的消耗情况进行控制。网络小组可能比较关心在一次主动扫描过程中，某个 WAN 链接占用了多少带宽。

而一个服务器小组则可能比较关心在不同结点上运行代理时所需的系统资源。有可能在对主机进行主动评估时，代理会占用大量 CPU 资源，或者与某个关键业务应用服务相冲突。因此，对审计过程进行限制是非常必要的，质量控制方面应从安全性和 / 或合规性方面进行限定。

一个虚拟机环境会放大代理的影响，因为多个客户操作系统实例的创建将会要求多个漏洞评估代理的运行。而且，运行虚拟机管理程序的主操作系统可能自身也在运行一个代理，这会进一步增加同时运行多个操作系统实体的风险。且不说资源管理小组要负责为评估活动制定时间安排这些问题，事实上，当一些特定活动的需要测试和管理需求已经给定时，如果有经验丰富的人员愿意承担这部分责任的时候，任何一个独立的安全小组都不会愿意承担这种责任。

总之，这里要表达的意思是在一个 IT 机构中的每个小组都要为其所提供的服务负责。引入一个新的 IT 职能都会对其他小组产生影响。将受影响小组的代表纳入到漏洞管理项目中，对识别漏洞管理产品的影响和明确其需求很有帮助。过程依赖图（process dependency diagram）是用于识别这些需求的一个很好的工具。图 5-1 展示了一个实例图，其中的漏洞管理服务是一个聚焦点。不过，该图也可以进一步扩展以引入更多的服务和服务之间的依赖关系，并且这些服务的需求应该是可变动的。

根据图 5-1 所示，我们可以根据漏洞管理系统快速地识别需求，而不仅是识别漏洞，而且也会提高受影响系统的服务级别。在该图中，网络服务提出了影响现有服务级别的需求：

- 工作期间的审计安排。
- 各站点或其他组需要的带宽消耗（例如源网络之类）。
- 同步网络连接以最小化对路由器和防火墙的影响。

然而，业务应用服务小组的需求不仅会影响应用程序性能，而且还要求能够执行特定类型的漏洞检测。这通常是为反映系统安全性而巩固 SLA 所造成的结果。应用服务小组的 SLA 可能明确指出对每一个发布的程序版本都要进行跨站脚本和 SQL 注入漏洞的测试。于是，漏洞管理系统不仅对于特定 IT 服务产生了服务传递任务，而且通过部署 OLA（运用级别协议）加固了现有的 SLA。

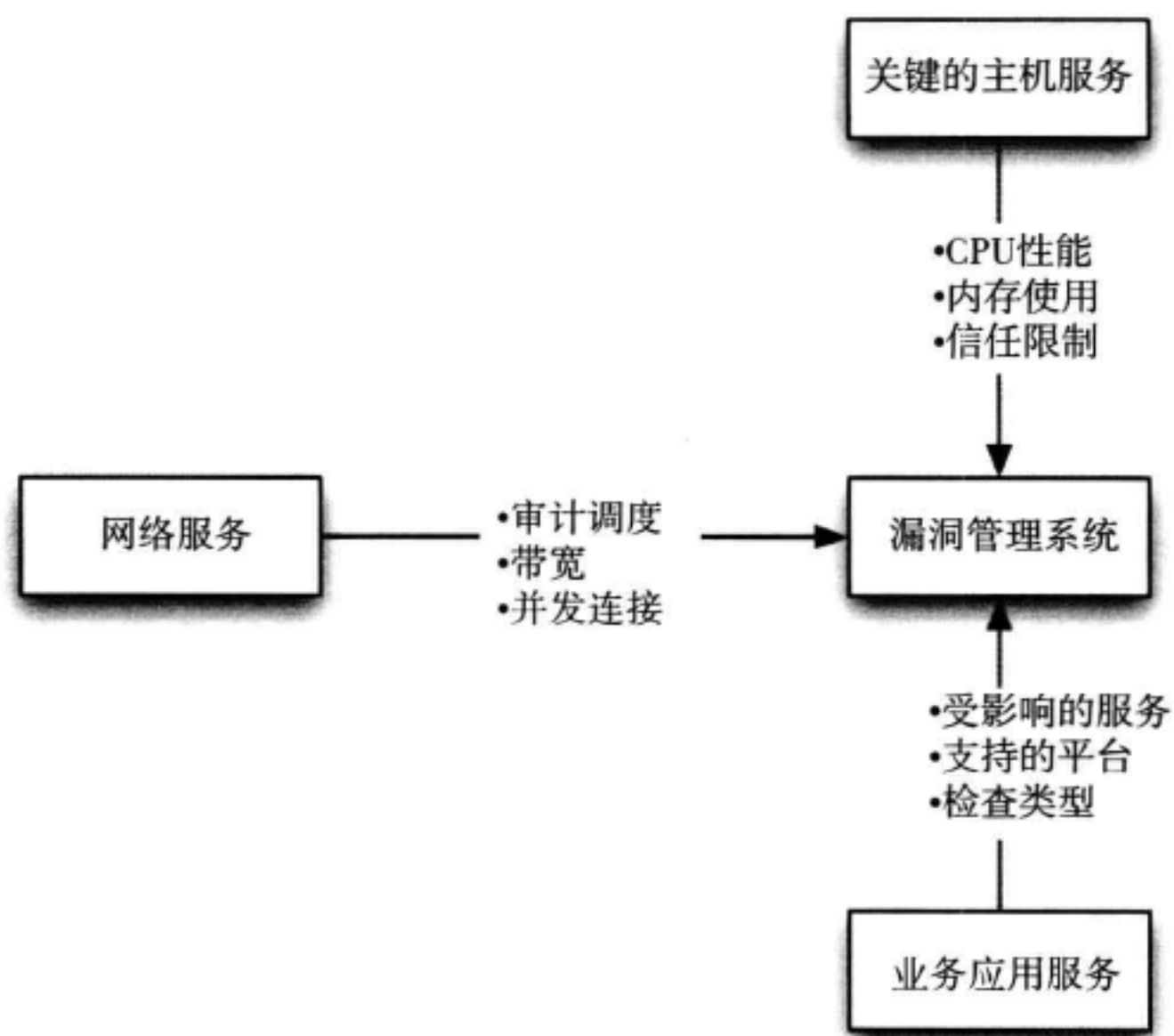


图 5-1 识别漏洞管理系统需求

5.2.2 时间表

在 5.2.1 节我们讨论了制定审计时间表的需求问题，也就是评估过程中所需要的控制类型的规范。不同厂商提供的扫描工具或代理的定时和资源控制功能可能会有明显的差异。例如，对于一个主动扫描，能设置在某天或某周的特定时间实现某种功能是很正常的。但是，公司的商业运转可能要求每周在相同的时间点执行评估，并且如果由于技术原因，原计划的时间错过了，那么评估活动只能在下一个执行日的相同时间继续执行。这种需求在一个繁忙的、合规性不断受到监控的商业环境中并不少见。下面列出的是其他时间安排需求中需要考虑的事项。

- 一天的某时刻，一周的某天：选择一周的某一天或一天的某个时刻是一种非常普通的、典型的时间安排方式。一般来说，这种设置应该能针对特定的时间区间进行调整。
- 一个月的某天（第一天、最后一天、指定的某天）：这种时间安排对下面这种业务运营情况是最有用的，评估目标的日期必须定在工作日，但又不能具体确定到底是哪一天。但可以确定为一周的哪一天，或者一个月的哪一天。因此，时间参数就可以是一个月的第一个工作日，或最后一个工作日，甚至是最后一个周一。某些业务周期要求评估只在每月或者每周的特定某天进行。例如，一个零售商会在每周一早晨收集关键销售数据。在这种情况下，他可能会希望漏洞评估在每周一的中午 12:00 之后执行，或者在每周的最后一个工作日执行，或者在每月的最后一个周五执行。
- 只在当地工作时间内或工作时间之后：对桌面电脑的评估通常需要在工作时间进行，因为只有这段时间电脑是开机状态。由于公司越来越关注节能问题，台式机的评估时间控制变得越来越重要。
- 在一个时间 / 日期窗口中可以开始 / 暂停 / 继续：在一个时间窗口内，当评估需要的时长超过可得的时长时，这种安排很有用。例如，如果一个扫描必须在工作时间执行，但需要超过 12 个小时才能完成，这个评估就应该可以在夜间暂停，然后在接下来的工作日继续。这个过程应该继续到评估完成。如果一个评估的日程与下一个评估有重叠部分，则下一个评估应该推迟一个周期开始。这种需求可能听起来很复杂，但并不少见，而且现有产品都还没有能很好地解决。
- 对所有评估的时间安排，日期和时间参数都应该调整为当地时区。有各种不同的实现方法，然而，结果应该是这样的：如果评估从纽约的周一 10:00 开始，相同的参数应用在旧金山，就应该是太平洋时间 10:00 或者是东部时间下午 1:00。这对于大型的全球分布的公司就更为重要。我见过很多系统执行在制定时间安排时全部采用世界时 (UTC)，以此来应对各地时间调整功能不足的问题。假设某网络全部处于同一时区中，一个很好的办法是为目标网络或主机标明该时区。然后，所采用的时间安排可以用一个参数来表明是否要根据目标的时区进行调整。
- 如果一个网段的全部计算机都不在一个相同的时区，那么谨慎的做法是在漏洞管理系统中，根据时区从逻辑上对网络进行划分。这将允许系统采用优化的时间安排扫描网段中的每一部分。如果网络划分不可行，那么就有必要把扫描时窗限制在一个所有主机都可以访问的共有的时间段中。

5.2.3 标准

每个厂商用不同的方法来检测漏洞。早期的漏洞管理产品主要就是通过这些专有的方法来相互区别。性能上能够准确而快速是必不可少的。然而，这导致了产品供应商之间严重缺乏信息共享，而这种情况对于客户而言从整体上威胁着安全的有效性和一致性。不同厂商进行漏洞评估的结果可能有所不同，那么如何确定应根据哪个评估结果来采取补救行动呢？如果一个产品通过检测注册表项找到了一个漏洞，另一个产品因为采用的是文件检测方法而没有发现该漏洞，既然没有任何独立的、可靠的最佳实践检测方法作为根据，那么安全经理如何选择应该信赖哪个产品呢？如果使用多个漏洞管理系统会产生结果差异，那么需要额外的工作来解决这些问题，所以，这也不是一个有效的解决办法。

反过来，当组织中使用的软件分发方法产生的结果与标准的本地安装产生的结果不一致时，也会出现类似于上面的选择难题。在某些情况下，微软 Windows 平台上的应用程序缺少某些注册表项也可以进行安装。当漏洞扫描器查找不到这些注册表项时，得出的漏洞状态可能会与实际不符。

一个更为普遍的问题是缺乏标准的漏洞识别码。例如，如果漏洞管理系统使用其专有的漏洞识别码，而补丁管理系统却使用 Bugtraq 上的 ID 编号，这两者如何能相协调呢？在某些情况下，漏洞管理系统厂商发现的某一个漏洞可能不存在可用的标识符，而只有使用专有的索引编号。需要推行一些标准，以最大程度发挥整个漏洞管理过程的效能，并保证全球一致的识别。

解决方案是有的。如果厂商和 MITRE 公司等能够将它们发现的漏洞集中起来一起探讨共同标识和描述，那么补丁管理系统就可以依靠这些标识符。

另外，一些厂商已经集成了诸如补丁管理等功能，解决了漏洞识别和相对应的补救措施之间的协调问题。但这种做法限制了客户对产品和功能的选择，客户只能依靠某些厂商提供的集成漏洞管理系统来实现这两项功能。然而截止到写本书开始写作时为止，还没有厂商提供对所有操作系统和所有漏洞补丁的全面支持，并且将来他们也不可能这么做。在本书完成时，已有超过 33 000 个常见漏洞和 CVE 项。即使顶尖的漏洞管理厂商也只覆盖了其中的不足 75%，但有如此多的消费者能容忍这种明显的不足？如果一个厂商告诉你他的产品只有 75% 的效果，你能接受吗？消费者需要认识到这种关键的不足并提出更多的需求。

当认识到自己不能完成所有事情，厂商会内置与关键的管理和跟踪系统的接口能力，如变更管理和异常管理。兼容的厂商数量可能有限。在这种情况下，厂商开发了 API，以允许创

建一个在漏洞管理和其他系统之间的定制连接器。

我们前面讨论了与标准有关的技术和支持，诸如通用漏洞评分系统（CVSS）、CVE 和 OVAL。对任何技术采用一个标准都非常有好处。但是，由于安全技术的迅速发展和相互间的激烈竞争，标准的采用应该成为一个更关键的需求。最初，对厂商来说，“赠送”功能给另一个厂商似乎很愚蠢。但是，通过提供选择产品的灵活性，厂商会提高他们被选择的机会。这就产生了一个“鸡”和“蛋”的问题，其中厂商必须支持开放的标准而消费者必须需要这个标准。OVAL 是需求列表中应该认真考虑的且更为明确、重要的标准之一。

5.2.4 报告

正如将在后面的章节中看到的，报告是漏洞管理以及管理任何可度量的过程的一个重要组成部分。在对任何供应商进行评估前，负责漏洞管理的关键人员都应该对如何执行漏洞识别和修复的过程有一个清晰的认识。然后，就经理、工程师、系统管理员、审计员和风险管理人員在漏洞管理过程中执行各自的职责时需要知道的事项进行询问，确保他们都牢固掌握了相关知识，以便能最好地利用系统。为此，需明确报告的内容和样式。可以考虑向利益相关者分发报告样本，以确保报告是可以接受的。这些报告具有以下高层次需求。

- 时间安排（即产生报告的频率）：在某些情况下，报告需要每周生成以修补最新的漏洞，而在其他情况下，只需每月评估组织的安全态势。在另外一些情况下，每当发生一个特定的事件时即需要产生报告，如一个关键目标的综合风险达到了某一阈值。了解所有报告的产生时间和频度有助于发现在自动化报告生成时需要的时间安排的灵活性。
- 目标：了解报告的确切目的往往能帮助识别系统是否已生成了适当的报告，还是尚待生成。一个系统产生的某些报告的标题可能不合常规，但也能满足目标需求。而有的系统则可能正好相反。一个报告可能标题为“审计核查”，看上去好像是供审计员使用的；然而，仔细检查之后发现这个报告并不满足审计员角色的需求。一个建议是，在你详细确认报告的内容之前，不要急于勾选需求检查项。
- 格式：大多数报告都不是严格定制的。列的顺序、分类和过滤选项、页眉和页脚的内容可能不适用于你的组织。页眉和页脚的要求可能是个大问题。如果一份漏洞报告的页脚中没有写明公司要求的文档分类说明，页眉中也没有注明公司的 Logo 或部门名称，在审计人员审核文档处理的合规性时可能就会出现一些问题。而且，报告的效果

因其排序和过滤能力的不同而有很大的差异。如果一份扫描结果报告只能按 IP 地址进行排序，而不能按严重性排序，那么据此报告可能就无法有效地对漏洞按优先级标注处理。

- 定制：在产品预先内置的报告无法使用时，应允许一定权限内的数据库和数据结构访问，以生成核心产品未能提供的报告。也可以附加一个报告生成工具来实现此功能。任何此类附加的工具也同样需要根据需求进行审核。此外，必须注意的是任何附加的产品或咨询服务都会提高系统的整体开支。
- 类型：报告中都会有的关键内容不外乎补救措施、安全态势和过程监控。你所需要的报告类型取决于你的工作过程。补救报告应该能翔实、具体地记录实施修复的方法和过程。这听起来理所应当，但实则不然。例如，一个公司在各个城市或者办公网络中可能会有一个不同的小组负责漏洞修复。那么，报告就应该是按地区分类的。
- 或者，你的公司可能会根据设备类型提供不同类型的修复。例如，服务器管理员可能会组建一个全球服务器修复小组。有一个中央小组会对服务器漏洞进行评估、制定优先级和分配修复任务。虽然系统对服务器的审计近乎于对整个网络进行审计，但是报告还是会作为服务器类报告。
- 触发器：某些报告需要在若干条件满足的情况下才会生成。这些报告通常会以邮件形式发送或者复制到某个特定的地点作为备份，以方便以后系统恢复。触发事件可以是漏洞评分或者特定指标达到了某个水平，或者一个关键主机出现了某个严重的漏洞。这些报告就会自动地给系统的授权使用人员发送邮件。
- 角色：任何报告都应该指定能由特定角色查看或修改。这些角色可以是系统管理员、网络管理员、网络组管理员或系统（目标）管理员。指定哪些角色或角色组合能够访问或修改报告有两个目的。
- 责任分离：这确保不会发生当责任方未能有效修复系统时，为避免罚款而篡改报告结果的情况。
- 数据保密：每份报告都包含了对于黑客而言非常有价值的信息，这些信息无疑可以用来评估目标和攻击目标。

设置角色最灵活的方法是让漏洞管理系统管理员来创建角色，并为角色分配用户和用户组。更多角色的定义将在后面论述。

□ 控制：报告应该能够基于网络、IP 范围、系统类型、扫描器、应用或网络的所有者等属性进行过滤。你很快就会发现，报告的数据筛选性能越好，管理员和工程师就越愿意使用。由于具有对本地网络结构的深入了解，网络管理人员可能会需要特定子网的报告或指定 IP 网段的报告，即使审计过程发生在所有能够发现的 IP 地址上。事实上，漏洞审计时所扫描到的 IP 地址范围会大于网络所有者需求的漏洞审计报告中的 IP 地址范围，即前者是后者的超集。

应用程序的所有者可能要求报告展示应用程序的支持系统的漏洞状态。这可能包括路由器、防火墙、数据库服务器、应用服务器和 Web 服务器的组合。这可能对萨班斯法案（SOX）合规性或受限 PCI 审计尤其重要。受限 PCI 曝光的大型组织将需要评估与付款处理系统任何直接或间接相关的系统的漏洞。能够轻松地识别和审计那些不会产生模糊的、或含有其他不相关的系统漏洞数据报告的系统，能够节省审计时间和成本。同样的原则也适用于非 PCI 金融系统。SOX 合规性不应该被看做是接受或拒绝漏洞管理系统功能的理由，但它对评估有关金融体系的控制很有帮助。上市公司应根据 2007 SEC 指南认真考虑对 SOX 的真正需求。如果要提供直接与 SOX 合规性及其产品相关的可靠的建议和根据，厂商将承受巨大的压力。

5.2.5 高级报告

除了目前讨论过的报告和相关的定制报告，厂商还需还应提供更多的高级报告功能。我们将在本书中用单独一节来讨论，因为对于报告处理过程还不成熟、完备的组织而言，对这些高级功能的需要还不是特别迫切。然而，如果你认为你的组织为了支持过程优化对报告有很强需求，那么请考虑下面的基本需求：

- 产品应该可以允许基于访问原始审计记录的定制报告。
- 在一个开放的数据库结构（模式）中应该能获得审计一览表。
- 附加的功能应该允许对漏洞数据进行一些统计分析。
- 除了数据分析，信息应该能够或者容易与网络拓扑信息相关联，以便通过攻击向量评估风险。

这些需求具体到人员配置和流程会更复杂，也有更严格的要求。这些报告将直接为风险管理和安全事故管理功能提供信息。已上市的产品可以把这些信息与主要的漏洞管理厂商提

供的信息相结合，也可以与防火墙、入侵预防系统和网络设备相结合来映射和识别威胁。有些厂商甚至可以允许用户评估高级别风险，然后深入到主机、网络和安全设施上，通过修改特定的配置项来解除即将发生的威胁。

评估自己的候选供应商与这些先进的产品的兼容性，为将来留下进一步提高的空间。在撰写本书时，这些最先进的风险管理产品还很少，而且很昂贵，但为技术风险管理人员提供了强大的功能。

5.3 自动化

漏洞管理过程要尽可能多地实现自动化以获得组织的青睐。自动化应该以透明的方式与现有系统整合。整合方式应使用广大接受的标准而不是劳动密集型编程，需要大多数即装即用的集成。如 XML 和 RPC 都是非常有用的可选标准。

5.3.1 标签生成

创建变更标签或事件标签是自动化的一个关键领域。漏洞管理系统与标签系统的接口应满足：所有需求的字段都已经提供了或综合了，并且能够向标签接受者提供最相关的信息。列出标签系统所需的事项条目，并验证所有这些事项在标签接口中都可以得到，或者可以很容易地生成。例如，某些标签系统要求创建标签的人要有一个用户 ID。如果标签系统用户可以接受该标签，那么就可以创建一个漏洞系统或其他系统的用户 ID，使得来自漏洞系统或其他系统的人员可以进入标签系统创建标签。或者，也许漏洞管理系统的接口可以提供目标所有者名称或网络所有者的名称作为用户 ID。像这些细节之处很容易扰乱你的计划，除非对其全部做出清晰明确的规定。

完善漏洞管理流程是对漏洞修复而言非常重要。一旦标签被接受，并且修复工作完成后，该标签就应该暂时关闭了。这时，事件应自动通知漏洞管理系统修复工作已经完成，该执行核查操作了。于是一个追查性扫描就可以立即执行，或在下次排定的时间执行。如果目标确实已经不存在漏洞了，那么标签系统就可以自动地将标签完全关闭，或由漏洞管理系统向标签系统发送一条消息来完全关闭标签。如果该目标漏洞没有得到完全的修复，那就可以根据内部程序的要求，要么重新打开该标签，要么生成一个新标签。

在产品选择过程中将标签系统所有者包含进来是进行整合时需要考虑的一个重要因素。

应当精确评估整合的工作量，因为其开销可能会很大，这取决于系统接口的灵活性和易用性。此外，永远不要期望，整合工作能在没有系统所有者的支持下进行。

5.3.2 流程整合

任何漏洞管理系统应该具有一些与现有 IT 流程整合的灵活性。虽然有些 IT 流程在公司添置某些硬件或软件之前甚至都不存在，但是漏洞管理流程是必需的。漏洞被发现、分析和修复，尽管每一步可能都不完善，也没有完好地存档，但每一步都是不可或缺的，并且应当尽可能地按需求加以记录，以备系统功能验证的进一步需求。在许多情况下，重整漏洞管理流程使之更加符合标准规范，可能会更容易。流程和技术的整合会严重影响补丁管理系统实施的修复工作，因此该补救操作也要求有更大的灵活性。

5.3.3 流程和系统的灵活性

一些漏洞管理系统是围绕一个特定的、相对刻板的流程建立的。该程序的供应商在销售过程中，常常通过展示其产品如何提供了一致的结果来吹嘘其产品的卓越性能。然而，除非这些产品的买家能非常灵活、有效、适应性地改变其关键的和早已确定固化的流程，否则该工具往往无法提供良好的结果。

这会对漏洞管理系统的有效性产生限制，而没有起到改善公司漏洞状态的作用。围绕一个严格的流程来实施漏洞管理系统，能帮助产品开发人员简化设计和编码，同时也简化了最终用户的使用操作。但是，对于需求易于进行文化和技术整合的组织，如果漏洞管理工具能够符合组织的需求，那么组织能获得更多的益处。以下是在技术方面要求的各种灵活性：

- 提供在格式上适合环境和文化的报告。一些公司不愿意登录到系统选择参数，以获得一份报告。这通常是因为用户都很忙，无法在其日常例行活动中关注一个流程和系统变化。一种轻松地让重要数据呈现在他们面前的方法是将重要数据发送给合适的管理员。发送电子邮件是很常见的一种方式，如今已经非常普及。
- 此外，一些 IT 经理有用于处理日常公务的门户网站，将一些关于网络的关键信息放在他们的个人主页上就非常有用。例如，一份显示他们网络中评分最低的“前 10 名的主机”的报告将帮助他们快速评估和分配修复任务。对于更高级的管理人员，一个“前 10 名网络”或“漏洞趋势”的报告将有可能使管理员花 10 秒钟浏览一下便能快速发现问题所在。

- 允许个别 IT 经理在经一致通过的标准的审计计划之外，对其所负责的网络区域安排和执行额外的审计。在主动扫描中，这就是所谓的扫描点播。这是一种有效修复检验措施，能最大限度地减少在修复中的错误。工程师有时会修复错误的漏洞，或因理解错误而没能正确地修复漏洞。另一个常见的错误是安装某些补丁后没有重启系统。由系统管理员或 IT 经理负责实施的验证过程，可以提高修复成果的质量。

5.3.4 补丁管理支持

我们已经讨论了补丁和变更管理接口。如果你的组织存在相关的流程，那么应当满足一些基本的数据需求。而漏洞管理系统也应该提供一个能灵活应对流程变化的接口。例如，内部流程可能不会在本次漏洞审计中对漏洞修复的结果进行验证，而要等到下次审计。虽然变更流程可能已经完成，并且在完成时会自动通知漏洞管理系统，但是验证扫描还是应该能够灵活地在下一次定期审计中作为常规计划审查自动执行。此外，很多事情可以证明在漏洞管理系统的工作流程中还存在不够灵活的地方。

进行定制通常都需要专业服务。我们需小心应对这个挑战，并深入考察流程适应性的实现需求。如果产品的供应商声称定制唯一可靠或最可靠的办法是使用专业服务，那么部署成本不会太低，如果价格很低通常都是不可信的。定制成本中包括对特定流程整合需求的传递费用。

5.4 体系结构

漏洞管理系统的体系结构极大地影响着它在企业物理结构中的作用，也会极大地影响系统成本，如地理位置、办公室规模、网络设备、安全体系结构、广域网带宽以及政府规定等因素，都将极大地影响你对供应商、产品类型和成本的决定。

最后一项（成本）是我们选择供应商的一个首要考察因素。首先应对漏洞管理市场进行调查，了解各产品和售价。确定在你的组织中每个活动 IP 地址的大致费用。估计一下 IP 地址数量，偏差能保持在 10% 以内就足够了。用 IP 地址的数量，乘以每个 IP 的平均市价，然后加上每年 20% 的维护费用。如果有国外的办事处，再加上 10% 的航运和海关费用，如果有超过 15 000 个 IP，再另加 10% 的咨询服务费。例如， $20\,000 \text{ 个 IP} \times \$8/\text{IP} = \$160\,000$ ，然后添加 10% 的航运费和 10% 的咨询费 = \$32 000。全部固定费用：192 000 美元。然后，计算经常性

的维护费用：每年 \$32 000 ($160\,000 \times 0.20$)。在第一年的系统的总费用是 224 000 美元。

这只是一个成本估算，会受到前面提到的体系结构因素的显著影响。在主动扫描系统中，必须购买专门的扫描设备，并安装在网络中的最合适的位置。在一个极端的例子中，如果有 100 个办公室，而有利的扫描位置分布在各办公室，并且没有强大的广域网连接，那么就不得不购买 100 个设备。这些设备可能很昂贵，当然这也取决于所选的产品。也有较便宜的解决方案，包括采用产品的虚拟机版本，或使用主机代理以避免每台设备都要收费。然而，对于自己提供硬件的购买者来说，还是有一些硬件成本。如果在虚拟机中还有 CPU 周期可用，性价比会更高。

5.4.1 被动的体系结构

监视网络流量的被动扫描仪，可能会更加受限于这种被动的体系结构，因为被监视的网络交换机上的流量都需要复制到扫描仪上。对于大型的复杂的中央办事机构，被动扫描仪是一个非常可行的解决方案，其连接通常非常快，并能承受由远程交换端口分析器（RSPAN）功能施加的负荷，RSPAN 在第 4 章讨论过。对于广域网带宽充裕的，拥有几个大型办公室的组织，主动扫描器可能是一个很好的解决方案。由于带宽越来越便宜，而且在世界上的偏远地区也变得更充足，因此通过认真规划和安排审计日程，使用更少的扫描设备，减少运输和海关开支，就能够使扫描更便宜。

漏洞管理功能集中化程度越高，所节约的费用就越可观。

5.4.2 基于代理的体系结构

使用代理模式，有必要在每个要被评估的主机上安装软件，除非有些代理对邻近的系统也能执行基于网络的审计。在某些情况下，服务器代理的售价要比台式机代理的售价高。其实这两种代理没有明显的功能差异，采取这种定价模式，只是一个销售数量和成本回收的问题。一些漏洞管理体系结构策略认为代理不会影响其他软件而只会影响网络连接，因此只在服务器上部署代理，以尽量减少对网络连接的影响。供应商往往能觉察和利用这一点，于是会将服务器代理的售价设置得较高。

在使用虚拟机的情况下代理的使用会更加复杂。代理可能会被安装在多个客户操作系统上，而这些客户操作系统却部署在同一个物理服务器下。于是对硬件的影响就成倍增加了。人们应该从供应商那里寻求一些指导：关于代理程序会如何显著影响虚拟机的 CPU 和内存资

源，以及底层主机的操作系统和硬件。供应商很可能会按照每套操作系统而非每个 CPU 内核来收取费用。

由于代理程序必须在要被评估的每个主机上进行部署和维护，这种解决方案不再会受限于网络问题，而是会受限于每台主机上软件的运行问题。这种解决方案会使安装和部署复杂化，但也消除了运输成本。拥有大规模的、流动的销售队伍的组织将大大受益于基于代理的系统，因为广域网的连接速度是不可预知的，并且在地区公司使用不多。

5.4.3 主动扫描的体系结构

我们已经讨论了主动扫描的基本架构和关于带宽消耗的考虑。另外一个关键考虑是对一些硬件设备的管理。虽然这对于有数百甚至数千台服务器的组织来说好像微不足道，但通常漏洞管理的维护人员是有限的，而且需要接受特殊的培训。他们依靠其他地区和部门的资源来维护他们不熟悉的设备。这些设备中许多都有命令行、串行端口和一些网络要求，而这些不熟悉的人可能无法完全理解。虽然很多管理责任可以集中化和自动化，但还是难免出现设备或环境配置的故障，需要在现场修正。

例如，在管理服务器和设备本身之间的某些点会失去网络连接。尽管使用了所有可用的工具，可能仍然无法确定失败的原因。当地工程师必须检查物理连接、交换机配置、设备电源和逻辑网络配置。这可能包括插上串行电缆，配置一个终端，使用本地管理员权限登录，并执行命令行功能。在多数情况下，从设备部署以后，工程师已经有 6 ~ 8 个月没有这样做了。漏洞管理运营人员将不得不提供书面或口头的操作指导，并接收反馈信息作出指导。使用一个网络、键盘、视频、鼠标之类的设备很有帮助，但并不完美。物理环境可能仍然需要监测。如果无法连接到该站点的网络，那么通过远程几乎做不了什么。

此外，替换无法工作的设备可能很困难，但这并不以技术专家的意志为转移。有些国家有进口关税和技术限制，这会使设备的替换周期延长数月。如果有产品的虚拟机版本，可以通过电子方式传送并连夜投入使用。

有了对这些问题的理解，在做计划时就应该认真考虑设备的数量、每个部署地点、当地人员的技能、语言情况、网络可靠性、可用带宽、功率要求、环境条件、海关手续，以及供应商的当地情况和库存情况。有些明智的办法能够克服这些条件中的缺陷。根据你面临的具体挑战，这些办法中的策略将有所帮助。

1. 确定设备的数量

确定所需扫描器的数量和一个增长因子，其中数量更难处理。首先确定网络的数量，网络连接的强度和目標数量。根据这些信息，就可以为扫描的总负荷进行估计：

扫描负荷 = 带宽 / 目标 + 10% × 间接费用 + 15% × 增值率

然后，估计使用候选厂商的产品扫描这些目标所需的时间。特定条件下的扫描测试可以提供扫描每台主机的平均时间。推荐进行全面的测试，因为每个环境都是独特的，它们会对不同厂商的各种方法有不同的反应。

例如，一家公司有分布在不同的物理地点的多个网络，如表 5-1 所示。此表显示了带宽的供应值、使用值和可用值。为了精确估计审计一个网络所需要的时间，对这些站点进行了两次测试，以确定扫描一个主机所需的平均时间。通过一些简单的计算得到了收集和传递结果给报告服务器所用时间的一个近似估计值。这两次测试都要求采用最大的实际目标样本量。关键是要确保两个样本之间的差别足够大，这样计算出来的结果才有意义，其差值就是扫描一个目标所需的时间。以下是确定所需的扫描器数量的全过程。

表 5-1 扫描时间估算表

地点	目标数量	带宽	可用性	可得性	审计时间			20 个目标 估计用时
					10 个目标	20 个目标	时间 / 目标	
总数	450	N/A	N/A	N/A	9	11.5	0.25	119.0
达拉斯	260	2000	72	560	13	15	0.2	63.0
香港	241	4000	45	2200	12	16	0.4	104.4
圣地亚哥	75	384	80	76.8	11	15	0.4	37.0
墨西哥城	245	4000	52	1920	8	13	0.5	125.5
芝加哥	325	4000	62	1520	11	14.5	0.35	121.3
亚特兰大	175	1544	70	463.2	13	16	0.3	62.5
旧金山	310	4000	55	1800	9	12	0.3	99.0
所需总小时数：12.2								

- 1) 审查网络，并选择具有代表性的站点、广域网连接类型、带宽、主机类型的样本。广域网连接类型和带宽将影响到扫描活动的响应时间，并影响大规模扫描的总时间。如帧中继连接方式会比高速异步传输模式电路或专门的专用线路的响应时间更长。带宽的不同也会产生影响，不过其影响只在一定限度内。扫描活动在设备硬件和协议连接方面也会受到限制。
- 2) 在有代表性的网络中按样本量（S1）抽取目标。此样本量至少应为主机总数的 10%，

而且不能少于 10 个。第二个样本量 ($S2$) 应比第一次多出 50%，并至少不低于 20。这样的抽样数量可以确保测试结果能产生有意义的差别。

3) 计算并完成表 5-1 的填写。在表 5-1 中，用扫描第二个样本的时间 ($T2$) 减去扫描第一个样本的时间 ($T1$)，并除以样本的主机数量差额，计算出每个目标所需的扫描时间 (ST)。这就是扫描一个主机需要的总时间 (TT)，并且不包括扫描管理的结果收集、格式化和传输的处理时间。

4) 计算前面提到过的进行扫描间接费用所需的时间量 (OH)。样本 $S2$ 的扫描时间为审计 $S2$ 所需的时间减去扫描所有 $S2$ 的目标的时间： $S2 - (S2 - \text{hosts} * ST)$ 。问题是，扫描目标所用的时间和完成全部审计活动所用的时间之间，存在着一个巨大的时间差。

5) 推断和估计扫描整个网络的时间。用 TT 乘以网络中所有主机的数量，并加上管理时间消耗： $X = TT * OH$ 。表 5-1 的最后一栏就显示了这个数值。这些数值的总和显示了如果只使用一台扫描器，一个地点接一个地点地审计整个网络所需要的小时数。一般公司网络在地理位置上分布得很广，这种只使用一台扫描器的情况几乎不可能发生，但它确实能为设备所需的扫描时间量级提供一个直观的参考。

6) 需确定对组织中的每个目标进行审计的频率。这将指出如果使用一个单独的扫描器对目标进行扫描所要的时间总量，从而最终确定需要多个扫描器。一个典型的网络应该至少每周进行审核，以确保系统对报告中新发现或报告的漏洞正在采取及时的修复行动。

7) 创建一个审计时间表提案。由于许多目标在扫描期间都在使用中，应该参考上一步中的信息来创建一个时间表。通过这个时间表，可以确定使用一台扫描器在一天或者一周里可以审计多少目标。请注意，在编制这个时间表时，必须考虑当地的时间、办公习惯和目标的可用性。图 5-2 展示了一个时间表样例。查看各个扫描时间段，可以知道在给定的时间段里有多少个必须执行的审计工作。此示例将每天划分为 4 个时间段，每段代表跨越格林威治时间 (GMT) 的 6 小时。

8) 从供应商那里获取关于一台扫描器可以同时执行多少个审计，以及对扫描性能又有多大影响方面的建议。例如，当单独扫描网络 A 时可能需要花费 45 分钟，而与另一个相同规格的网络的扫描同时进行，扫描时间可能平均要延长 30%，还是在假设没有任何其他网络设备引入更多延时的情况下。

XYZ 公司-审计时间表																												
	周一				周二				周三				周四				周五				周六				周日			
格林威治标准时间 (GMT)	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3	0	1	2	3
总部																												
总部-服务器																												
总部-内联网																												
总部非军事区 (DMZ)																												
总部-VoIP																												
达拉斯																												
达拉斯-服务器																												
香港																												
香港-服务器																												
圣地亚哥																												
圣地亚哥-VoIP																												
墨西哥城																												
墨西哥-服务器																												
墨西哥-VoIP																												
芝加哥																												
芝加哥-生产部																												
亚特兰大																												
亚特兰大-服务器																												
旧金山																												

图 5-2 GMT 审计时间表

9) 估计所需扫描设备的数量。再参考一下表 5-1, 合计需要并发执行的审计操作的推荐数量, 并除以推荐的数值, 由此你可以评估需要配置多少扫描器。在某些情况下, 可以通过重新安排审计时间表来减少扫描器的配备数量。附注: 永远要为错误的增长和设备的增加保留余地。这项安排告诉我们, 如果一个网络审计是从一个中央位置进行的, 那么有可能只使用一台扫描器即可, 当然前提是该扫描器可以同时实施两个审计并不会出现显著的性能下降。

如果由于操作上的限制, 认定对“Chicago-Mfg”的审计只能在星期三进行, 那么就可能在执行时段上产生争议。综合各站点的审计需求, 并协商一个折中方案是解决此类问题的一个很好的方法, 并可以进一步减少所需的扫描器数量。还需注意的是, 服务器网络通常可以在周末和深夜进行审计, 因此可以在一定程度上减轻审计用户工作站网络在白天的工作压力。

每个网络都是独特的, 因此最后的扫描器数量结果可能会有所不同。测试是一种有效的手段, 但并不是能适应网络和系统特质的完美手段。在需求中留出一些灵活性对于调整是必

要的。一些先期规划和测试可以节省很多采购费用，也能增加系统和操作员在使用群体中的可信度。一旦部署了扫描器，任何高端的漏洞管理系统都应该提供一种报告扫描器资源的利用率的方式，以便系统管理人员在不断变化的环境中可以根据实际使用情况进行调整和重新分配。

2. 仔细选择地点

地点选择应该根据满足以下几点。

- ☐ 本地扫描的目标数量。
- ☐ 其他相邻网络的可用带宽。
- ☐ 支持人员的技能和技术水平。
- ☐ 在 IT 问题上的监管限制问题，例如隐私和联合工作规则。

当然，其他因素，如运费、税费和其他重要的因素也应考虑。此外，基本的部署（如电源供应、机架空间，以及网络的物理布局）都是要考虑的，但通常是次要因素。

5.4.4 保证平台安全

需求中也应强调安全设计的重要性。最不理想的情况是意图改善安全却反而带来了更大的威胁。引入到运行环境中的任何硬件或软件都应该接受安全审查。厂商不会故意提供一个脆弱的系统，因为这不会给他们带来任何利益，但即便如此，想要保证任何的产品的安全性，仍然需要付诸实施，不是想想就能实现的。人们当然希望出厂的所有硬件设备都很精良，开发出的软件都很安全。然而，如果一个漏洞管理项目经理连按照推荐检查项来验证系统的安全状态都没有做，那就太粗心大意了。

此外，漏洞结果可能会存储在设备或其他外部存储设备上。我们建议你检查所有设备的安全设计。如果某设备上有本地存储器，如硬盘，那就确定扫描结果是否存储在该硬盘上，并且是如何存储的。找出该硬盘上的数据存储了多久以及如何删除它们。如果在硬盘被退回或在运输途中丢失，这是否就相当于把机密泄露给了黑客，将公司置于危险之中了呢？该设备跟其他系统一样，应遵循统一的数据保存、备份和安全存储的政策和标准，只做少许改动便可以适应产品的局限性。

5.4.5 系统整合

公司通常只购买和安装技术产品，而事先很少考虑它对 IT 基础设施到底能起到哪些积极作用。这样做的结果往往是所安装的技术产品只能提供极为有限的功能，而公司则不得不采取更难实现的措施来弥补这一缺陷，如与其他系统集成或改动主要流程等。

漏洞管理系统能够以多种方式提供数据，最大化系统安全和运营的收益。这些系统中包括变更管理和事件管理系统。这些系统常见于那些致力于从 IT 服务中充分、持续地获得更高性能的中型或大型组织。

1. 变更管理

变更管理是修复流程中的关键环节。如前所述，当发现一个漏洞时，有关详情可自动发送到一个变更管理系统，以启动一个变更过程。

漏洞管理系统与变更管理系统之间所要创立的接口并不像厂商描述的那样简单。几乎总是需要一些定制开发。这种类型的开发往往要对数据类型、格式和通信方法进行转变。厂商的产品可能为新发现的漏洞提供 SNMP 陷阱，而变更系统会要求使用 XML 或电子邮件监听进程。于是有人将不可避免地要为这两种完全不同的技术编制接口。以下是无论采取什么接口方法都需要进行交换的通用的数据元素：

- ☐ 向变更系统发送漏洞的详细资料。
- ☐ 向变更系统发送漏洞事件标识符。
- ☐ 一旦修复完成，向漏洞系统发送变更状态更新。
- ☐ 如果安全漏洞仍然存在，重新打开该变更或创建新变更。

2. 事件管理

类似于变更管理的接口，事件管理可能是一个组织中的操作支持系统的一部分。接口一般都是类似的，只是流程有所不同。在某些情况下，组织倾向于在变更系统中处理变更，但事件会保存为一个特殊序列的环境。

另一方面，为追踪漏洞和修复过程而生成一个事件，当变更发生时，使用变更管理系统只跟踪对系统、资源、流程的影响的情况并不少见。

无论是哪种情况，正如前面提到的，一个变更的完成，或者引发事件的关闭，或者引发一个发送给漏洞管理系统关于完成的立即通知。漏洞管理系统将重新评估该目标，以确认是

否成功完成。

3. 入侵防御

有些厂商试图将自己的产品与 IPS 进行整合，结果却发现 IPS 厂商在与漏洞管理厂商竞争较量，这导致厂商的一些失败的尝试，但如果成功，则可以给消费者带来很大的好处。如果有一个 IPS 能够与某个选定的漏洞管理厂商的漏洞数据兼容，那么一定要全面严格地对二者的兼容性进行审查，因为这其中一定会存在很多障碍。

标准格式兼容是一个严重障碍。虽然本书中用了较多篇幅讨论标准，但在 IPS 领域、漏洞领域，兼容性问题上很少有厂商能完全支持它们。目前，这两个行业在互操作性上还有很大不足，唯有客户群的要求能够促使其发生改变。无论如何，一个基本的设想是，如果在一个目标系统中发现了一个新的漏洞，就通知前端相应的 IPS 来激活保护资产的相关标记，直到该漏洞得到妥善修复。

这种类型的整合有两个主要的好处。首先，一个漏洞能够得到保护，直到完全修复措施完成，从而降低了环境的整体动态漏洞的水平。其次，IPS 的性能得到优化，因为只需根据标准的策略实施激活必要的规则。这一点非常重要，尤其是当一个非常昂贵的 IPS 放在一个繁忙的 DMZ 区中，而硬件升级又不能带来足够的成本效益时。

4. SEIM

安全事件和事件管理（SEIM）的集成通常比较容易实现。SEIM 厂商将与众多数据源的兼容性当成一个重要的卖点。数据收集是他们最强的组件，并且该组件对来自漏洞管理系统的数据几乎不用修改就能接受。

如果你的组织拥有一个 SEIM 程序，那么不接受这些重要的数据反馈就是失职。在 IPS 不能整合的地方，SEIM 程序至少可以利用这些数据来确定一个事件的严重性并对它进行相应的升级。如果供应商不能支持漏洞管理供应商的产品，那么他们很可能很少被选择。

5.5 定制与整合

虽然经常被忽视或作为后续检查的事项，定制开发是确定需求中的一个重要部分。很多时候，那些在最初实施漏洞管理的人很晚才会发现，定制工作对于最大限度地提高其安全系统的潜力是多么重要。这可以发生在有变更管理系统的情况下，这些变更管理系统通常是定制开发的，并且围绕系统构建了根深蒂固的内部流程。因为不能采购一套新的变更管理系统，

所以能够高效地从漏洞管理系统中提取信息，并创建一个变更和 / 或事件的处理流程是至关重要的。

在高层次上，如前面所述，当一个接口到变更管理系统或事件管理系统时，接口需要管理下列条目。

- 从漏洞管理系统中提取漏洞数据。
- 在内部系统中创建一个变更或事件标签，同时保留由漏洞管理系统提供的参考信息。
- 对标签进行临时关闭，并更新漏洞管理系统。这将可选地启动一个验证性评估。
- 来自漏洞管理系统的一个事件会触发对标签的关闭，或者对标签的重启，这取决于验证的结果。

根据企业里的流程和系统，这些步骤在接口编码上可能会对数据结构和数据处理有不同的要求。在采购之前，对候选的漏洞管理系统的接口能力有一个清楚的了解是很重要的。一旦弄清楚以后，就要评估各候选产品对编码的要求。这将有助于发现大大降低整体效益的任何潜在缺陷。例如，预计的内部漏洞管理程序可能要求从漏洞管理系统向变更管理系统传输详细的修复建议。如果在标签生成程序中没有提供这项的关键数据元素，那么这可能会导致在实现 IT 安全服务目标上的严重缺陷。

5.6 评分方法

因为关系到安全运营，所以对漏洞管理系统采用的评分方法进行仔细审查是必要的。以下是对评分系统的基本要求的列表。

- 要考虑到资产的价值。组织中更有价值的资产应当得到比低价值资产更高的处理优先权。无论在评分方法或告警能力的这些方面，都应该采用数字或分类的方式来评估资产的价值。资产的价值如果不能从现有的资产管理系统中直接获得的话，对其进行估值将会是个非常耗时的过程。
- 严重程度可以从逻辑上或直觉上通过分数判定。如果评分看起来很随意或始终设置为

零，则很难判断 300 分到底表示“严重”、“中等”，还是仅仅“只具有提示”。在某些时候，无论是根据以往的系统使用经验还是根据对评分方法的了解，都应该对严重程度划分类别。

- ❑ 结合最新的漏洞利用代码知识对严重程度划分标准进行调整。如果公布了一个新的易用的脚本漏洞，供应商就必须修订自己的评分方法。评分应该是动态变化的，要能与动态的威胁环境保持同步。
- ❑ 统一使用一个基于类似 CVSS 标准的初级或二级评分标准，以便与公开的分数含义进行对照。这样能够避免由于某个分数是根据私有方案得出而造成释义混淆的情况。
- ❑ 可选地，一个评分应该有一个基准。这意味着评分会根据所选的评估基准不同而有所变化。如果一个漏洞是从公共互联网上检测到的，那么它应该有一个比从本地网络中发现的漏洞更高的评分。

为一个漏洞确定合适的分值，一定程度上是数学问题，一定程度上是需求问题，这与风险的计算没有什么不同：

$$\text{风险} = p(x) \times \varepsilon \times \rho$$

其中， $p(x)$ 是发生概率， ε 是从一个事件的损失预期，而 ρ 是每年的发生率。一个漏洞的评分计算的例子可以基于下列变量（见表 5-2）：

表 5-2 漏洞评分基于变量示例

损害严重性 (α)	系统远程控制 =100
	远程访问系统 =75
	远程监视系统 =50
	本地控制系统 =50
	本地访问 =40
	本地监视 =30
攻击易度 (β)	简单（脚本的）=1
	中等 =0.75
	难 =0.5
是否存在公开的漏洞利用代码 (\times)	存在 =1
	概念上存在 =0.5
	不存在 =0.25

所以，这个计算是一个简单的乘法公式：

$$\text{评分} = \alpha \times \beta \times \chi$$

这个评分方法的分值范围为 3.75 ~ 100。也就是说，如果损害严重性 (α) 为“本地监视” (30)，攻击易度 (β) 为“难” (0.5)，是否存在公开的漏洞利用代码 (χ) 为“不存在” (0.25)，那么：

$$\text{最后评分} = 30 \times 0.5 \times 0.25 = 3.75$$

如果损害严重性 (α) 变成“远程控制系统” (这是非常糟糕的情况)，同时其他所有因素保持不变，则得分上升到了 12.5。依此计算，最坏的可能分值是 100 分，此时 β 和 χ 因子等于 1，而“远程控制系统”对应的值就是最终得分。

许多评分方法更为复杂，考虑的因素更多，如漏洞存在的时间长度。另外，其他评分方法不会局限于 0 ~ 100 之间的一个简单刻度范围，而更可能是不设上限。例如涉及类似漏洞年龄之类无法限定的数值时，这种情况就会很自然地出现。此外，本例中的取值范围较为有限，因为它只使用了简单的乘法，但其他方法更倾向于明显地区分特别糟糕的和风险相对较低的情况。平方和阶乘运算能够使计算结果迅速增大从而能更明显地区分风险等级。但是这种分数可能更难解释。无论偏爱什么方法，都要确保能将其转换成一种适合风险评估的输入格式。

5.7 访问控制

访问控制是任何应用程序的一个基本特征。根据基础设施的标准，坚持一个外部的认证机制是有价值的。在漏洞管理系统中定义角色和访问权限是必需的，要求外部认证也是合理的。

5.7.1 活动目录

活动目录是 Windows 系统最常使用的认证机制之一。后期的版本的目录加载支持轻量级目录访问协议 (LDAP) 和 SSL 上的 LDAP。Kerberos 和 NTLM 是认证中常用的选择。由于活动目录功能在公司环境中很常用，并且也提供与其他系统对应的接口标准，因此使用活动目录是一个好的选择。任何 LDAP 目录服务都应该有效工作。有两种集成活动目录的通用方法。

一种方法是定期同步活动目录信息，检查增加和删除的内容。在漏洞管理数据库中会保存一份目录条目的备份，以便快速查询访问权限，这是使用 LDAP 的最通用、最合适的方法。通常情况下，需要为登录系统目录和获取用户基本信息的操作创建专门的证书。使用 LDAP 也提供了该系统移植到其他目录服务平台的选择。

然后，当一个用户试图登录漏洞管理系统时，用户证书被送到使用 NTLM 或者 Kerberos 的认证系统。一旦证书通过审核，漏洞管理系统将为该用户申请存储在漏洞管理数据库中的相应权限。

另一种方法是使用 Windows .Net Server 2003 自带的应用模式活动目录（AD/AM）功能集成活动目录。这使漏洞管理应用程序能够拥有自己的带有扩展模式和内置属性的目录服务实例，但仍然隶属于活动目录域下的安全结构。但支持这种功能的服务必须运行在基于微软技术的服务器上。这为基于微软目录服务的组织提供了一个紧密集成的目录产品。这种方法的显著优点是活动目录组能够用于漏洞管理系统中的授权，而不用于创建内部的角色和用户组。缺点是要使用活动目录平台。

5.7.2 RADIUS和TACACS+

在一个更侧重于网络的环境中，RADIUS 是一个非常普通的协议选项。它是典型的应用在拨入系统中的古老方法。然而，该协议对网络设备也很有用。在 RADIUS 和 TACACS+ 协议中，用户 ID 必须被输入到漏洞管理系统中，因为没有提供目录服务。这个用户 ID 必须与认证服务器存储的值准确匹配。RADIUS 和 TACACS+ 最显著的区别是他们分别使用 UDP 和 TCP。因为 TCP 在通信过程中有握手过程，所以它在与认证源进行通信时也具有安全优势。

一些网络认证产品的销售商甚至可以同时支持 RADIUS、LDAP、Kerberos、TACACS+ 以及其他认证方法。这些认证产品能有效地将多种授权和协议“结合”在一起，允许使用多种方法。

将认证方法从一种变更为另一种可能会比较困难，当然也取决于实现方法。为了避免其他情况出现，最好一直选用一个方法。

5.7.3 授权

授权功能应该满足计划中多种角色的操作需要。这是在开始 RFP 过程之前必须要定义需求的一个关键原因。需求中有一项是定义由谁来执行什么功能，需要什么能力。基于角色的访问控制是一种应该在选择过程中不断核实的机制。应当考虑以下功能。

- 通过网络进行分离：只允许一些用户在特定的网络中操作。例如，在孟买的 IT 工作人员不能检查芝加哥的任何漏洞报告。除了漏洞管理的管理员之外，其他人不能修改扫描参数。

- 行动：与定义一个角色相似，要执行一批特定的活动，如定义一个网络或者管理层 IP 的范围。
- 生成报告：许多人可以执行这个功能。能够生成报告是一项基本的功能。
- 实施扫描：大多数人不能实施扫描。主动扫描功能对业务操作是一种潜在的干扰，所以应该将执行权限赋给那些有资格评估扫描对网络造成的影响以及需要当前审计结果的人员。
- 维护整个系统和可用的审计参数的稳定：审计参数一般不能修改，除非需要做扩展测试。增加 TCP 端口或者额外的检查可能影响整个扫描时间进度表。应该使得只有少部分人能够修改审计参数。这些人可能有一个安全与合规方面的角色组合。

一个记录授权需求的好方法是做一个许可权限表格。这个表格的表头表示各种权限，最左边一列表示角色，见表 5-3。“Y”或者“N”表示该角色拥有或者不拥有执行的权限。如果指定了具体的访问类型，“R”代表“读”，“W”代表“写”。

表 5-3 许可权限表格举例

角色 / 能力	报表	扫描	定制日程	维护	参数
系统管理员	N	N	Y	Y	R
本地 IT 人员	Y	N	N	N	N
区域安全经理	Y	N	N	N	N
全球安全	Y	Y	Y	N	RW
全球合规	Y	N	N	N	R

5.8 部署方法

部署一个系统有很多种方法，而其运行环境的需求将很可能影响到部署方案的选择。在有些方法中，如本章前面所讨论的，选择方案与体系结构有关。除此之外，有关如何部署一个系统，还需要考虑很多别的因素。本节将讨论影响部署的主要问题。

部署方法的目标是在最短时间内以最少的投资获得最大效益。以下是达到这一目标的基本的部署策略。

- 1) 建立能够最大程度访问目标系统的据点。
- 2) 小规模测试并不断优化流程。
- 3) 通过不断增加目标机器来扩展系统直到据点中 75% 的目标机器都被部署。

4) 与第 3 条同时进行, 在所有水平上创建并优化管理报告。

5) 在其他区域同样采取上述步骤。

我们将针对主动物理扫描器、主动虚拟扫描器、被动分析器和代理分别就上述每一个步骤进行审查。

5.8.1 主动扫描器部署: 物理部署

实施物理扫描器部署策略的第一步是, 选择一个所有扫描器用来上报结果的中心位置。当如果存在多个中心上报服务器时, 选择一个在本地网络中拥有最多主机的位置, 最好是一个可以不受限制地访问其他主要站点的位置。确认有足够的带宽来支持前面讨论的方法制定的结构化的时间进度表。然后, 遵循下面这一重复性过程。

1) 在安装了报告服务器的本地网络中, 或者报告服务器安装在不那么关键位置的本地网络中执行测试扫描和修复报告。一个 24 位的无类域间路由 (CIDR) 块就足够了。

2) 审查报告结果和任何对环境的非期望影响。

3) 调校系统和扫描参数来进行弥补。要确保一旦开始部署, 你所做的任何调校对于数百个乃至数千个网络都是可测量的。

4) 在本地办公区增加尽可能多的相似网络域, 一次增加一个。

5) 不断重复上面的步骤以使报告有效, 并对环境产生影响, 可以进行相应的调整。

在等待本区域执行几轮上述步骤的同时, 规划和协调下一步对于扫描策略中规定的办公区域的 WAN 扫描。这些办公区不会安装扫描设备, 但是能够通过 WAN 连接接受扫描, 并且不影响本区域的正常工作运行。然后, 重复评估和优化过程并相应地更新扫描标准文档。

当扫描扩展到本地办公区之外时, 就开始生成和优化管理报告。在这个阶段, 管理层将非常关注新系统产生的结果。要确保这些报告之间没有矛盾, 并且其内容中反映的问题是可以解决的。如果采购前进行了适当的测试, 可能不需要太多供应商的帮助就能很容易地完成上述工作。如果有一个执行负责人能够在报告被送给更多的读者阅读之前预览这些报告, 则有助于提前发现不一致和有问题的地方。

到此为止, 已经部署了初始扫描区中 75% 的主机, 并且已经开始稳定、循环地执行这些工作流程。接着, 就应该开始计划部署下一个扫描区, 可以通过选择另一个与其他目标办公区有尽可能多的良好连接的主要办公区来实现。然后, 重复之前讨论过的步骤中的渐进的扩张过程。

5.8.2 虚拟扫描器

使用虚拟机技术创建的主动扫描器通常遵循与物理扫描器相同的策略，它们之间只存有一些很小的差异。虚拟机在很多地点可以更快速地部署而且几乎不需要依赖 WAN 链接。那些选择不安装虚拟机的地点，一方面是因为没有足够的硬件、人员和许可软件来支持。另一方面，由于每一个虚拟机实例都需要得到许可证、维护和技术支持，因此其数量应保持在最少水平。

对运行虚拟机的硬件要求给出一个明确的定义也很重要。许多供应商都是将运行在物理设备上的软件移植到了虚拟机上。这样做的结果是，当运行虚拟机软件的宿主操作系统将扫描器与硬件分隔开时，可能会产生让人意想不到的结果。针对一个硬件产品的虚拟化版本做一次完全分离的测试。

有时，同时使用物理设备和虚拟设备更实用，而不是单独使用其中一种。尤其是当无法将设备从一个地方运送到另一个地方时。更进一步来说，如果扫描设备只用于在工作时间审计台式机，或者在夜里审计服务器，那么存储空间和功耗都是需要考虑的问题。由于对节电越来越重视，虚拟设备的前景对高级管理层来说也变得更有吸引力。

5.8.3 被动分析器的部署

被动分析器是典型的硬件设施，如第 4 章所述。它们对安装环境有独特的要求，因此在地点的选择上会具有一定的局限性。所有的主动扫描器表现为主机连接到交换机，与此同时，被动设备由于其实施方式而完全不可见，依赖于它们的实现。目标安装网络必须有能力复制其想要的信息给扫描器的网络端口。并不是所有的交换机都具有类似于镜像或者交换端口分析器（SPAN）的能力，所以应将这种能力作为选择站点的要求之一来予以评估。

如果使用网络侦听器代替端口镜像功能，则被动分析器仅能看到有限数量的业务流量。这是因为并非所有的设备都与其网段之外的设备进行数据包交换。当在交换机上使用端口镜像功能时，来自特定网络段内的所有成员的所有数据包都将被复制给分析器。在单一网络线缆上使用网络侦听器可以观察到所有通过该电缆的数据包，但是无法侦听到同一网络的两台设备间的对话。

通过一些创新的网络配置有可能在一定限度内克服这个问题。当要完全检查所有通信流量时，镜像方案是有局限的，因为所有业务流量加起来的总带宽不能超过连接到漏洞分析器的链路带宽。例如，如果一个 DMZ 有 20 台主机，每台平均消耗 50Mbps 的带宽，则连接到

分析器的链路带宽至少需要 1Gbps 或者更大。更进一步来说，分析器的硬件和软件必须得有足够的能力去处理全部的业务流量。否则，结果就会出错，或是潜在丢失信息。如果用两台设备作为 IDS，则对性能的需求将更加苛刻。

适合于安装被动流量分析器的位置是那些主要与部分自有网络外部的宿主进行通信的位置。互联网 DMZ 区及内部服务器集群区是优良的候选位置。可以找到的带有端口镜像功能的网络网关或者高级网络交换机的任何位置都是非常有帮助的。此外，此类位置通常不能为太多的设备进行端口镜像。所以，目标网络最好不要使用这种技术来实现其他技术，如 IDS。任何这种类型的部署计划都需要同网络管理员以及交换机供应商密切协调，以确定该地点对于某特定实施方案来说是否是一个好的候选位置。

5.8.4 代理部署

代理程序是一种对网络影响最小的，也很有效的漏洞测试手段。它们能持续地监控本地系统的状态，并且常常可以对周边设备执行预定的审核。以下是一些常见的策略，可以根据不同供应商所提供产品的功能加以选择。

- ❑ 把代理程序安装在那些经常活动的工作站或服务器上，以便使用最少的代理程序执行扫描。缺点是，如果依赖于一个特定的工作站，那么就会受到该工作站用户的制约，因为他们可能会关闭工作站。如果将代理程序安装在服务器上，这个问题可以得到一定缓解。
- ❑ 在网络中的所有设备上安装代理，以避免在网络产生过多的审计流量。其缺点是会使代理程序因升级漏洞信息而产生更多的流量。但是，如果升级是每周进行，那么这种信息量通常很小。
- ❑ 在每个办公室里的域控制器上安装代理，它们总是活动的。然而，这会影响域控制器的性能或者可靠性。可选择性的部署代理程序的好处是它们可以提供从网络角度对漏洞的观察。也就是说，可以通过网络和基本安全架构组件对这些漏洞进行评估，从而更好地了解该漏洞所带来的安全威胁。请注意，为了使这一策略生效，很多规划工作是必需的，全面地测试也至关重要。

不管选用什么方式来部署代理程序，请注意，由于兼容性和管理的问题，代理程序正在逐渐失去企业的青睐。对部署的规划可能会很复杂，而且安装代理对基础架构所造成的影响

与其他方法相比更难预测。

5.9 小结

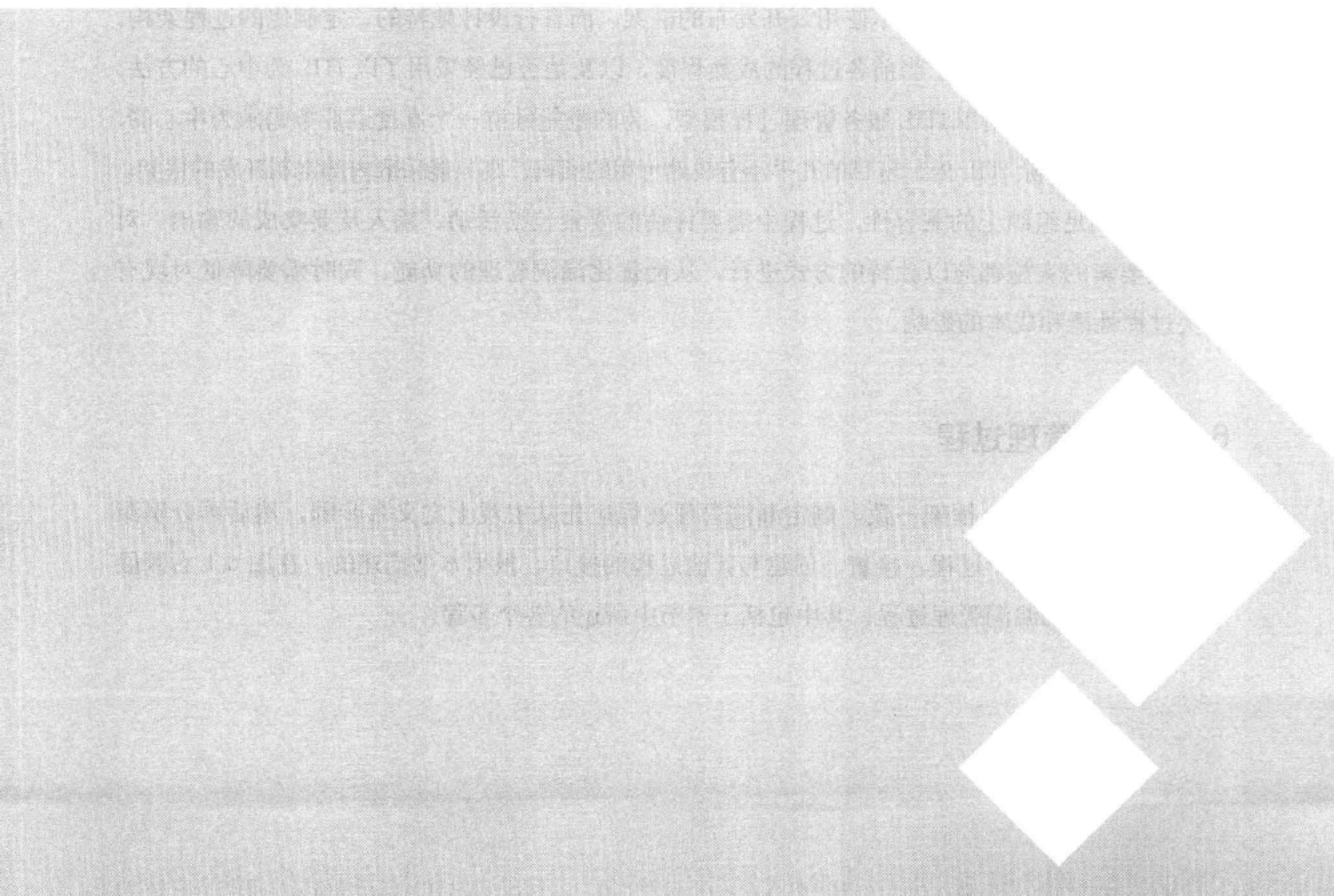
像往常一样，我们再次看到，流程和规划对 VM 技术的选择是多么重要。通过开发和协商技术选择过程中所要遵循的流程，关键的参与者将产生一种主人翁意识。为最完善和最期望的实施方案进行规划也非常重要。很多时候，公司会直接去评估产品，或者去查看厂商的评价，而不会考虑与环境相关的具体用例。一方面应努力保持实施方案与基础支撑架构具有兼容性，能够被用户组接受；另一方面应牢记，风险管理和安全的需求是最高需求。以下是指导技术选择活动的基本原则。

- 1) 将所有会受到系统影响的小组包含进来。
- 2) 共同设想工作的进展。
- 3) 通过将第 2) 步中的设想深入细化来获得需求。
- 4) 制定一种评价方法。
- 5) 保证结果公开透明。

通常，计划都会重点围绕下述内容展开：找出能够生成最有效的漏洞结果的产品，并使它能服务于已知的安全需求。然而，考虑到部署工作的潜在复杂性和组织围绕这样一个系统所能建立起来的从属系统，从运营流程角度和技术的角度考虑未来的使用情况是很重要的。如果在确定系统未来的应用和系统实施问题时考虑得过于简单，则会产生令人不满的结果。

第6章

过 程



6.1 介绍

过程是一个成功的安全方案至关重要的组成部分，它通常集成在组织中以支持策略或者项目章程。过程可以用来指导技术的使用，而并非技术的附属，认识到这一点很重要，因为太多时候，由于设计者忽视了这一点，过程通常单纯地设计为操控技术，而不是用来提供能够支持组织目标的结果。这样做的后果就是过程在部署后需要不断地对其进行修正。

过程的制定并不一定需要很长的时间。利用一个基本的过程架构，90%的工作可以迅速完成。本章将讨论漏洞管理过程的各个步骤，以及每个步骤与组织中的其他过程是如何相互影响的。

如前所述，漏洞管理是一个过程而非一项技术。在制定过程时，不断追求卓越，努力完善以期获得最优良的性能，这样做将使我们有信心能够把最严重的威胁识别出来并加以修复。制定过程的第一步是要确定哪些要求适用于组织。一个必须回答的问题是：如果要最小化漏洞管理项目结果造成的影响而最大化其带来的效益，那么在过程中应对组织中的哪些特点和行为进行调整？依据组织现行的标准，可以选择一种方式来制定过程。可以选择使用以 ITIL 为中心的架构进行设计，或者不使用公开发布的框架，而自行设计独特的、定制化的过程架构。选择何种方式主要取决于当前各过程的成熟程度，以及是否已经采用了以 ITIL 为中心的方法。很多大型组织选择使用 ITIL 服务管理过程模型，为的是能维持一个高度以业务需求为中心的、可衡量的结果。除 ITIL 外，若目前几乎没有成熟可用的架构，则只能采取内部定制开发的框架。

为了满足组织上的兼容性，过程中需要评估的要素包括活动、输入及要集成的输出。对每一项要素的实施都应以独特的方式进行，从而优化漏洞管理的功能，同时需要降低对现有相关过程性能和成本的影响。

6.2 漏洞管理过程

与“生命周期”框架一致，制定漏洞管理过程应先从宏观上定义各步骤，然后再分别深入细化直至完成整个过程。接着，创建与其他过程的接口。根据本书描述的，在图 6-1 右侧展示了一个典型的漏洞管理过程。其中包括了本节中所述的各个步骤。

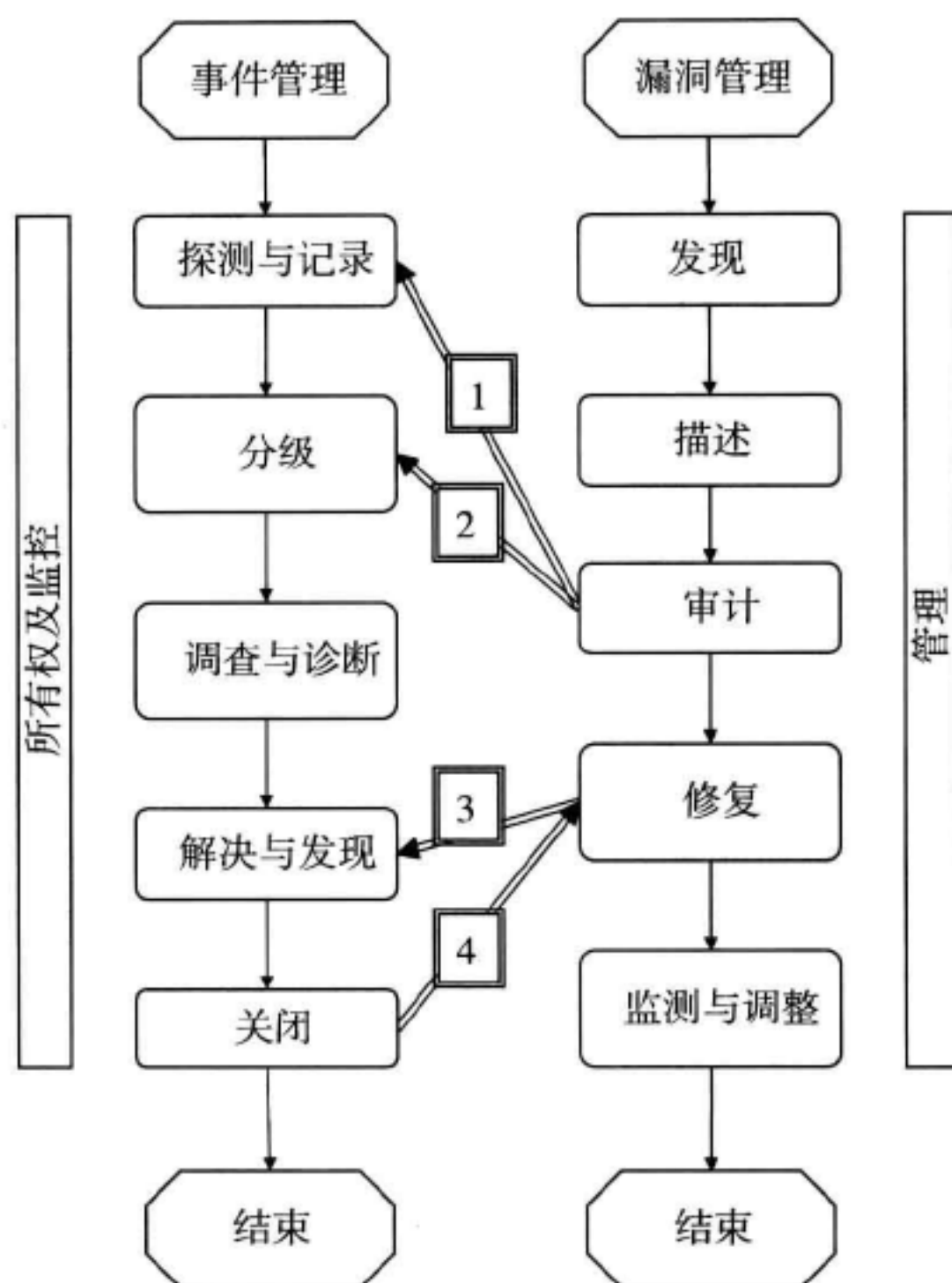


图 6-1 事件 / 漏洞管理过程互动

6.2.1 准备

制定过程前应先了解待审计的目标。为使审计有效，必须先确定 IP 范围和验证机制。

- 审计范围：依据目标技术和审计所需时间确定每次扫描主机的大致数量和扫描类型。
- 总体规则：确认目标，决定首先审计哪些网络。根据 VLAN、IP 范围或目标类型设想一种架构。同时，回顾一下目前已知的主动扫描系统的各种副作用，以避免因存在相似配置而产生问题。
- 确定修复管理员：确定谁来负责修复，并且应对谁负责。
- 制定扫描时间表：与网络和系统管理员确定每周扫描的日期和每次扫描的具体时间，这主要取决于主机的类型和业务运行模式。在工作时间实施主动审计可能会对业务产生潜在影响，但对台式机系统可能非常必要，因为只有在这时它们才处于开机状态。
- 环境调查：审查一下环境中主机的类型，以确定是否有特殊的异常情况存在。例如，

有些应用可能不允许主动扫描漏洞，即使这些扫描经过了认证。应确定是否有这种情况，如可能，将其从审计名单中删除。

❑ 名称和资产价值：每个网络和 / 或每台主机可能都有一个名称和一个指定的资产估值。

主动扫描一般能够提供登录目标主机（网络）的凭证，以实施更为全面彻底的审查。

❑ 基本架构：必须事先确定审计系统各组件的位置和数量，有关架构的内容在本书的各相关章节都有更为深入的描述。

6.2.2 发现

通过结合技术和人工手段鉴定工作环境中的所有目标后，详细清单可以自动列出，但资产所有者需确认每个目标的业务功能和相对价值。

6.2.3 轮廓

需对环境的复杂程度和基本的安全态势有所了解，应该知道 DMZ（隔离区）在什么地方，也应知道每个目标配备了哪些安全措施。了解这些将有助于更好地安排漏洞扫描和修复。漏洞管理过程的这一阶段包括两项活动：估值和场景化。

1. 估值

了解目标或资产的相对价值对后续步骤非常重要。根据业务环境的成熟度和实际的安全流程需要，可以选择将资产量化为具体金额，也可以选择进行简单的价值排序。估计目标的价值金额时，应计入目标暂时无法使用时对业务造成的损失以及修复目标时所需的成本。数值价值只需简单地按目标功能对业务的重要性进行排列即可。例如，将重要性等级设定为 1 ~ 10 可能就足够了。无论采取哪种方式，这些估值结果最终都将有助于确定资源有限时目标修复的先后顺序，也有助于确定扫描漏洞的频率和优先顺序。

2. 场景化

了解漏洞产生的环境条件对于避免浪费资源去扫描不太可能出现的漏洞是非常有用的。如果某个目标很容易受到 X 攻击，但在当前环境下要么 X 攻击发生的可能性极小，要么该目标的防护非常严密使得 X 攻击发生的可能性极小，那么这种情况下最好将资源用于防御其他漏洞。此外，漏洞很可能是为使某应用程序正常运作而必须使用的一种配置项，以下即是这样的范例。

- 当前安全防护：如前所述，围绕漏洞控制采取安全防护措施可能足以消除风险，使该漏洞不再显著。
- 应用要求：有些应用程序的配置要求可能被认为是一种漏洞。常见的，如允许任何人从未经验证的目录读取信息，而这可以看做是一个漏洞，因为它允许查看可访问数据的密级。
- 变更管理操作：有些漏洞可能会由于其他变更活动的实施而得到了修复。例如，如果一个有漏洞的系统要升级为新版本或被整体替换，那么这个漏洞可能在这些变更活动中被消除了。这里最需考虑的是漏洞的危险性和时间问题，如果变更活动6个月后才实施，而漏洞风险又很高，那么等待变更活动而进行修复就很不明智了。

6.2.4 审计

这是检测漏洞的扫描实施环节。这个环节可以为前两步提供很多有用的信息，以优化这一动态业务流程。审计一般比较频繁，而为了达到最好的效果，需要对目标拥有访问特权。

6.2.5 修复

在这一环节中对审计阶段发现的问题进行修复。有时，需要对修复活动进行缜密的规划和测试，以避免干扰业务活动，但更多的时候，也就是自动安装开发商提供的补丁。打补丁通常被认为是顺其自然的事，但其实不然。因为人们也许正在考虑是否改变当前的机器配置，或者更改默认密码，而这些改动经常会影响到其他依赖于当前这种特定的目标配置的系统。

6.2.6 监控和调整

通过检查审计结果，可以发现存在的问题，然后通过修改参数来提高精确度和避免潜在的服务崩溃。网络、目标配置及组织都是动态的，今天有效，但明天不一定有效。因此，为确保业务流程的顺利，有必要对结果和性能进行一定的监控。经常会出现网络管理员添加了VLAN或重新映射了主机IP地址，而安全和审计人员却毫无觉察的情况。但这类变动应该被快速地发现和得到妥善处理，否则漏洞管理系统的精确性和覆盖率就会下降。

另一种情况是，所使用的协议和服务的类型所造成的系统精确性和覆盖率的下降。目前，组织中大部分主机都会使用Web TCP端口80和443，而这些端口已被监控，但新的应用架构

可能导致使用 SQL Server 服务器之类的附加端口或者针对 Web 服务的更加隐蔽的端口。如此一来，漏洞管理操作员在设置审计参数时就应充分考虑这种风险并做出相应的调整。主动扫描极易受这类变更的影响。

6.2.7 管理

中长期趋势分析对于衡量漏洞管理项目的总体性能至关重要。如果至少完成了前 5 个阶段，那么你已经走在实现卓越的漏洞管理项目的道路上行进了 70%。而良好的管理能够提供持续稳定的效果和更佳的可可靠性，如果不加管理，那么漏洞管理项目的结果和效能就可能会减退。管理员要像维护机动车一样维护漏洞管理过程才能使之发挥最佳性能。

我们所谈论的整个过程会随着业务周期而不断重复，只是每轮循环之间的间隔有所不同。尤其是审计和修复环节，应在基础设施承受能力范围内，在不影响其他服务性能的情况下，尽可能多地重复。该频率不应低于每两周一次，如果可能最好一周一次，这是为了发现接入网络的新目标机器，并对其实施软件行业不断发布的新的漏洞检测。

以下是管理阶段应实施的一些有用的活动。

- 定期检阅报告：就检测到的所有漏洞生成总结性报告，并与此前周期的报告进行对比，这将有助于掌握安全态势的变化。
- 评选最差表现者：列举漏洞风险最严重的 10 个网络能有效地向 IT 管理人员施加压力，督促其采取措施。
- 鉴定最佳表现者：这些表现最佳的网络一定有值得借鉴的做法，对其进行总结并作为学习案例让所有 IT 团队共同探讨学习。将这些最佳表现网络置于所有人的关注之下，也有助于其严格要求自身不断完善其他各个方面。
- 监控趋势：如果整个组织或某个具体的网络漏洞数量和评分有上升趋势，应对其进行升级以避免其进入前 10 名。需要注意的是：一定要确保造成这种趋势的原因是性能低下而不是新发现的漏洞。第 7 章会更多地讨论这个问题。

6.3 基准

在任何连续的漏洞管理过程中，对于组织中基于风险需求的评估结果的细化将会优化整

个过程。基准是一种众所周知的、广为接受的安全配置。对特定的平台和目标组合而言，什么是合理的配置和良好的补丁状态？基准水平反映了当前对此问题的一种理解。普遍使用的标准的台式机计算平台就是很好的例子。该平台是一系列运行于指定范围硬件组件上的配置好的软件组件，平台的配置是根据在计算环境定义中组织的安全要求设定的。基准配置包括以下几个属性。

- 用户组：定义了本地和远程系统用户。对于台式机而言，在用户组定义中，本地用户包括了使用诸如 Word 和 E-mail 等办公软件的普通用户；远程用户则通常为台式机管理员，或帮助台式机用户使用远程访问应用和各种管理工具的服务台。
- 用例：是用户组的延伸，用例明确定义了如何使用平台。例如，普遍的 Web 服务器基准可能包含执行面向公众的 Web 服务器软件的应用例，其中该 Web 服务器软件含有与后端数据源相连的 Java 服务器应用。该用例会明确应用应包括哪些功能，排除哪些功能。这其中可能涉及财务应用、基础设施控制系统或组建安全体系。
- 环境：计算平台都运行在特定的环境中，这些环境对其所支持的应用程序的潜在风险具有重大影响。一般的 Web 服务器都运行在有不同风险等级的公共 DMZ 中，其各自的风险等级差异来自于数据库服务器，此服务器内部数据库层保存着关键客户数据。对物理环境较高的生产网络，可为其安装各种工业控制系统。
- 硬件支持：每台计算机都会有一些要求安装支撑软件和驱动程序的硬件组件，而这些支撑软件可能存在漏洞，需要加以处理。如果硬件平台不在预期的范围内，那么对于这些漏洞的管理可能会更加复杂。这并不是说我们只能选择前者，但如果可以通过已有的安全配置解决问题，就没有必要使用那些未经验证的所谓新硬件来增加风险。

上述所有属性的配置情况的不同，会导致风险、影响和修复成本的不同。通过预先设定，并尽可能将制定的基准限制在至少 95% 的组织都要求的内容中，漏洞的发现和修复过程是较易控制的。下列是设立基准水平的主要好处。

- 可以避免因未在基准平台上运行应用程序而不得不接受更高风险的事。
- 使用基准能使分析和打补丁流程中的变动降到最低，如果修复方案对于组织的绝大部分都是适用的，那么这种修复对应用程序和硬件的影响就更容易评估。
- 本地的 IT 工作人员在咨询系统工程师需要做哪些工作时，基准的使用也将使其对解决

方案的支持更容易。

□ 基准可以简化整体风险分析，并使安全管理标准的升级更为迅速、可靠。

基准应该根据风险环境以一定的时间频率进行合理升级，当新的基准经测试确定可以在环境中执行后，技术支持人员就能够对原有基准要求进行确认和升级。

6.4 ITIL-ITSM流程

以下文字来自 <http://www.itlibrary.org>:

信息技术基础架构库 (ITIL) 定义了信息技术公司的组织结构和技术要求，及一系列用于 IT 管理和相关基础设施管理的标准化操作管理流程和实践。操作流程及实践都是独立提供的，并适用于 IT 基础设施内的各个方面。

本节将探讨与漏洞管理相关的 ITIL 流程，即通常所说的 IT 服务管理 (ITSM)。全面讨论 ITSM 的服务管理超出了本书的范围，如果你对这方面的内容感兴趣，建议找一本相关的书阅读一下。这里所讨论的内容基于目前接受程度最高的第 2 版 ITIL，虽然 ITIL 框架目前已经有了第 3 版，但我们并不打算在此介绍，以免引起混淆。服务支持和服务交付是与漏洞管理相关的两个很广泛的领域，下面将详细介绍。

6.4.1 服务支持

服务支持是 ITSM 中对于安全最为活跃的一个领域，因为它是漏洞管理对当前 IT 服务提供支持的主要目标。服务支持包括以下活动：

- 事件管理；
- 问题管理；
- 配置管理；
- 变更管理；
- 发布管理。

像在之前的案例研究中提到的，事件管理和变更管理甚至在最基本的以 ITIL 为中心的操

作中也会经常用到，但全面讨论以上各个部分仍是有必要的。

事件管理应该集中研究如何尽快地恢复正常的运行，通常，这个正常运行是指要恢复到服务水平协议（SLA）中所述的正常的服务水平。对于那些不太熟悉 SLA 的人士来说，可以将 SLA 理解为业务主管与 IT 服务团队之间拟定的业务要求，其中包括明确的服务目标规定和符合 IT 服务交付要求的衡量方法。

服务团队需定期提供业务报告，这样业务主管就可以看到所要求的服务水平是否已经达到，这是一种能够为客户提供合规性服务，为 IT 团队提供连续可度量操作目标的负责任的做法。SLA 最大的战略意义在于直接绑定 IT 执行目标与公司业务目标，从而形成一个联系紧密的价值链。

漏洞管理服务也有同样的服务水平要求，虽然很多组织选择不为漏洞管理设置 SLA，但将 SLA 作为一项服务应用于整个业务的各个环节以得出可衡量的结果还是大有益处的。在第 3 章中业务案例的描述和一个漏洞管理项目的章程中已经讨论了设置可衡量结果的价值。

漏洞管理作为一种服务，对其进行衡量的基本内容如下。

- 列举出风险最高的业务单元。
- 每月和每季修复的所有漏洞的百分比。
- 每季估算的接受审计的目标的数量百分比。
- 每月和每季所修复的漏洞与所发现的关键漏洞的比例。
- 关键漏洞的平均修复时间。

以下是这些内容的一些基本特点。

- 可衡量性：如果要向公司报告上述要衡量的内容，这一点很重要。
- 相关性：可以显示 IT 安全组织的绩效。
- 现实性：多数漏洞系统应可以提供以上内容，但目标数量需由其他系统提供。

在很多组织中，单个漏洞在危害达到某种级别时会作为事件来处理，如果 SLA 要求的不是整体的安全态势，而是在一个细粒度的层次上，那么这非常适合。漏洞审计可以衡量 SLA 执行情况。修复和配置目标的团队负责 SLA 的执行。

经过相当长的一段时间，直到可以对漏洞数量和严重性进行预测时，就可以选择向其他 IT 服务甚至是整个公司业务提供操作级别协议（OLA），或者直接将其作为业务中 IT 风险评估的一部分。当出现不符合 SLA 的情况时，就会创建一个事件跟踪其解决方案，因此，此处

所说的事件是高级别的事件，显示了与 SLA 不符合的程度。

如果事件显示在服务层面有异常，就意味着安全态势超出了 IT 服务的预期。如果事件显示有需要修复的漏洞，那安全态势就像是存在一些需要扑灭的小火苗。这种区别看似很小，但重要的教训是避免将 IT 团队无法控制的新漏洞看做未能满足 SLA。

如果所采用的 SLA 规定需要在指定的时间内通过某种手段修复达到某种危险等级的漏洞，那每个达到这种等级的漏洞都将成为一个“事件”。

6.4.2 服务台

服务台是负责管理事件的一个联系点。从后面章节的讨论中可以了解到，服务台的主要目的是负责事件和相关通信的创建、处理，它就像一个中央通信集线器，负责维护提出解决方案的各参与方之间的联系。

6.4.3 事件管理

如果 SLA 限定了特定目标上的漏洞危害程度不能超过某种水平，该水平可能是用一个数值分值表示的，那么只要当目标的漏洞危害程度超过该分值就可以创建一个事件。

该方法的好处在于能将每个违反了 SLA 的漏洞都要进行正式的确认，但要对这么多的事件进行追踪和修复，工作量会很大，而这对目标的安全态势可能没那么重要。另一方面，这种方法也可能会引发对安全态势影响不太大的漏洞创建的事件过少，从而导致系统在存在一个或多个此类漏洞被利用而系统没有发现的情况。

事件管理可以很好地嵌入到漏洞管理过程中，因为它提供直接的解决方案。大量通过自动化方式发现的漏洞会被当成事件来处理，因为漏洞的触发原因和解决方案都是很明确的。图 6-1 将事件管理基本流程与漏洞管理过程并列展示，两个流程间的连线显示了信息交流和过程交互的位置。

- 漏洞审计是一种生成事件的检测和汇报机制。
- 部分审计数据也包含有要输入到分级流程中的关键信息，这些信息通常是对漏洞和危害程度的定义。
- 事件报告中会附带修复措施的详细说明，通常是来自漏洞系统供应商的建议。
- 在事件被成功修复后，系统近期状态或修复后的状态会被传送到漏洞管理过程并在监

控流程中进行验证，这一步有可能是自动的。

1. 问题管理

之所以要将漏洞管理过程与 ITIL/ITSM 相结合，是因为如果事件不断重复甚至引起了服务的中断，那么事件就变成了问题。这种情况中最常见的是配置错误引起的服务中断，SQL 服务器数据库的系统管理员密码设置为空就是一个很好的例子，该例子很明显是一个高危漏洞，如果在某些流程或标准的制定中出现了错误，这种情况就可能会大范围扩散，最终成为一个问题。问题管理主要针对那些风险巨大，但原因不明的事件。

在以上例子中，可能会出现一些服务器未按标准部署或由于恶意行为未按标准部署的情况，原因和解决方法都是未知的。但事实上，上述问题是由漏洞扫描器生成的事件造成的，解决的方法是根据要求设置系统管理员密码。如果该问题是由 IT 管理人员的错误导致的，那这就是一个已知错误，如果这是由某种恶意行为导致的，那该问题就是一起严重的安全事件。

多数情况下，由漏洞管理系统和过程来执行问题管理功能，以确定引发问题的根本原因，因此输出结果一般会显示为事件，而不是问题。

2. 配置管理

配置管理功能得到漏洞管理系统的极大支持，因为漏洞管理系统会为配置管理数据库（Configuration Management Database, CMDB）提供大量配置数据。对于那些不太了解 CMDB 的人来说，CMDB 就是一个包含了 IT 基础设施所有资产配置细节的数据库，这些细节就是所说的配置项（Configuration Item, CI）。变更、发布及事件管理过程都会用到该数据库，以验证变更和事件所造成的影响。

CMDB 也会存储来自上述问题管理功能的信息。在上述系统管理员密码置空的例子中，如果问题最后证明是由已知的、非恶意的原因引起的，系统就会将其列为“已知错误”一类，所发现的漏洞的结果通常是以这种形式记录在 CMDB 中。具体来说，系统管理员密码配置项就会标注为已知错误。将来，依据 CMDB 中所列的配置项标准就可以将这种错误变更降低到最小限度。

3. 变更管理

与配置管理和问题管理相似，变更管理允许对环境进行改动，以最大限度地降低对 IT 服务造成的负面影响。具体到漏洞管理，就是要在执行变更时列举出漏洞的实例。变更管理还提供了漏洞修复的标准化流程，包括：确定优先级、监测及报告完成情况。

变更管理的最大益处在于使大部分的修复过程更加流畅。ITSM 流程中的变更管理部分是对一个变更执行案例的拓展，而这可能会由于组织文化及业务优先级的不同产生偏差。漏洞修复的业务理由大部分是对漏洞成功利用所带来的潜在影响的推测。

当然，还存在一个费用问题。给桌面系统的网络浏览器打补丁的成本很低，尤其是当一个补丁管理系统自动执行时。尽管如此，一次可以中断已存在应用软件功能的 Java 虚拟机升级的成本非常大。因为该过程可能要更改大量底层应用，从而引发大量的回归测试，时间和精力投入会使成本大大增加，结果可能导致人们去寻求性价比更高的修复和预防措施。

6.4.4 服务交付

服务交付的首要任务就是走在业务的前面，确保服务正常运行，甚至是提供高服务水平。确保服务水平、服务能力及服务可用性机制全部就绪并处于正常运行状态，能够保证 IT 服务运行顺畅，并且即使业务发生变化也不会受到影响。

1. 服务水平管理

对于漏洞管理服务来说，服务水平管理像其他的 IT 服务一样适用。既然漏洞管理的目标是发现和修复漏洞以降低风险，那么设置 SLA 会要求一定的服务水平管理内容。如果漏洞管理项目采用的是 ITSM 架构，则会包含确定的 SLA，以及 OLA 和支持合同（UC）的内容。

简言之，OLA 是漏洞管理项目与组织中支持 SLA 的其他业务服务之间的协议，UC 是漏洞管理项目与供应商签署的提供支持性服务的合同。例如，网络团队可能会分配一定量的带宽用于目标的主动扫描，或分配一个带有某个 VLAN 通信量副本的专用交换机端口用于被动分析。这些情况会在提供的 SLA 中详细列出。

此外，漏洞管理系统供应商可能会为漏洞管理项目签署一个合同来确定由漏洞管理项目提供的 SLA，以提供每周一次的漏洞检查更新，这些更新包括微软一周内发布的最新漏洞。漏洞管理提供商一般不会提供这种严格限定的 SLA，因为很难确定处理一个随机报告的危害程度不详的漏洞需要付出多少时间和精力。因此，更为实用的 SLA 能够提供的承诺是：30 天内解决误报或 24 小时内及服务台接受事件标签。

2. 能力管理

当涉及对漏洞管理系统的能力的管理时，必须了解系统能够持续监控多少个 IP 地址执行漏洞分析及需要多少时间执行这些分析，这两项在主动扫描与被动分析中尤为重要。主动扫

描的性能会受到很多因素的限制，如带宽、业务时间、网络响应时间等，这些因素大多可以通过 OLA 解决，但有时随着公司规模扩大，要实现既定的服务水平需要更多的审计资源。

例如，公司可能即将新增一些办公网络，这就相应地需要更多的主动扫描设备来对这些位置进行扫描。识别业务中的这些变化并作出相应的处理，以保持公司业务的安全水平的稳定是漏洞管理团队的任务。当漏洞管理服务管理员未能及时了解其负责设置 SLA 的业务组增加了一个新办公区（扩大规模）时，就很容易在 SLA 中将其漏掉。

针对被动评估的能力管理的一项重要内容是在一定数据流量下持续，准确地捕获到漏洞。这种流量可能以每秒数兆计，或以每秒连接数计，无论哪种方式，在约定流量范围内，漏洞审计都应有效。此外，组织也可能会要求在更高数据流量水平下只检查某些关键漏洞。SLA 中这些约定的履行受制于流量分析程序的性能和网络中的所有限制。

3. 可用性管理

可用性管理是 ITSM 中应用于支持漏洞管理服务的一项内容，不过漏洞管理反过来对可用性管理也大有益处。可用性管理的一个方面是安全，主要是服务可用性安全，能够提供保证系统持续运行的信息。检测那些能够引起服务中断的漏洞是可用性管理的工作重点，这些信息必须定时提供给系统管理员以维持服务的可用性。

因此，负责可用性管理的人员可能就会收到与其所负责的服务的可用性有关的漏洞报告。如果某个网络管理人员负责监测网络和为网络服务可用性提供支持，那就应该向该网络管理人员提供这些网络设备的审计结果以督促其尽到责任。

6.4.5 其他方面

由于与漏洞管理关系不大，ITSM 中的以下方面在此未予详细讨论：

- ☐ 发布管理；
- ☐ IT 服务的财务管理；
- ☐ IT 服务连续性管理。

6.5 IAVA流程

与 ITSM 分别开来，作为私营部门漏洞管理过程的替代者，美国国防部（DoD）采用了信

息保障漏洞预警 (Information Assurance Vulnerability Alert, IAVA) 流程, 该流程是以数据库为中心的, 在美国劳工部监察长办公室留有备忘录, 国防部 D-2001-013 号报告中有如下表述:

本策略备忘录要求国防信息系统局开发及维护一个能够保证系统管理人员接收、知晓并遵从系统漏洞预警公告的积极控制机制的 IAVA 数据库系统。

IAVA 是一个基本过程, 当发现软件有缺陷时, 即发布以下三个公告中的一个:

- ☐ 信息保障漏洞预警 (IAVA);
- ☐ 信息保障漏洞公告 (IAVB);
- ☐ 信息保障漏洞技术顾问 (IATA)。

发布哪个公告取决于漏洞的危害程度或影响, 这种方法类似于供应商和 SANS 之类的互联网风暴组织等安全组织发布的“威胁水平”公告。

IAVA 过程与典型的漏洞管理过程几乎完全一致, 唯一的区别只在于 IAVA 是由国防部发布的。但 IAVA 的长处在于, 它会从已知补丁状态和配置状况良好的系统或目标入手处理要求修复的软件缺陷, 这称为“处于符合标准的状态”。此后的漏洞评估工作是要验证系统是否仍处于一种合规状态, 这要比试图去发现一切问题, 并通过简单地修复降低风险更为有效; 很多公司采用这种方法, 直到组织系统达到一个完全符合标准的状态。在私人领域, 这种“合规状态”等同于设置一个基准水平配置。

IAVA 过程包括以下几个基本步骤:

- 1) 建立一套 IAVA 合规系统;
- 2) 评估其是否符合标准;
- 3) 如符合, 回到第 1) 步;
- 4) 如不符合, 修复;
- 5) 回到第 1) 步。

如果发现目标存在相关漏洞, 那么系统就未能达到 IAVA 合规标准。达到 IAVA 合规标准的系统也可能存在其他漏洞, 只不过这些漏洞带来的风险是 IAVA 合规标准可接受的。

管理IAVA过程

国防信息系统局（DISA）负责维护 IAVA 过程。该过程使用的主要工具是 IAVA 数据库，国防部所属系统均在此数据库中有注册。库中的资料使得 DISA 可以跟踪各系统是否处于合规状态，并能生成报告。IAVA 数据库是一个能够支持前面过程中讨论过的跟踪、报告以及合规监测的工具。

当安全研究员发现新漏洞时，这称为 IAVA 请求，IAVA 请求会用来与数据库中所存内容进行对比，以确定所有注册系统是否处于合规状态。如果某系统不处于合规状态，那么根据所指定的流程，系统管理员就会接到通知，然后系统管理员将根据通知进行处理，使系统恢复到合规状态。

但 IAVA 预警、公告或技术建议的发布并不那么简单，而且也不一定是全自动的。以下情况须由计算机应急响应中心（CERT）来确定：

- ☐ 受影响的操作系统的类型；
- ☐ 受影响应用程序存在的漏洞；
- ☐ 系统访问的容易程度；
- ☐ 威胁的类型；
- ☐ 基础设施是否会受到影响；
- ☐ 该漏洞是否已被利用。

此后如果需要，CERT 会发布前面讲过的公告。如果是发布了一个预警，则目标系统管理员须在特定时间段内确认收到预警，这与 SLA 在私人部门的应用情况是一样的。

在收到 IAVA 信息后，DoD 组织需要报告以下情况：

- ☐ 注册系统数量；
- ☐ 合规系统数量；
- ☐ 免补系统的数量。

免补系统是指当应用受到破坏时可以不予修复的系统。通过一个简单的计数器可以查看由于缺乏修复而日益积累的风险。

虽然该过程对 DoD 是有意义的，但在商业领域不被接受，或许将来这种政府出台的标准

会被业界接受并执行。阻碍人们接受 IAVA 过程的最主要的因素可能是漏洞管理产品供应商间缺乏统一的标准，以及缺乏动力对系统漏洞状态实施如此严格积极地管控。如果所有供应商和安全研究员均采用如 OVAL 的标准来对漏洞检查进行编码，并且如果 CVE 及 OVAL 库审查和接受漏洞的流程更高效，那么也许会促进人们逐渐接受该标准过程。

IAVA 效果远未达到完美，在以下方面均存在问题：

- ❑ 通知内容可能质量不高，增加了修复的难度，因为人们需要寻找更多信息，从而延迟了打补丁的过程。
- ❑ VM 及配置管理工具供应商间缺乏统一标准增加了识别过程的难度，每种工具识别软件缺陷的方法不同，可靠性不一。OVAL 在推进漏洞检查标准建设方面很有价值，这对国防部和大型公司均是如此。
- ❑ 如果工作人员有更为优先的考虑，他可能会滥用“免补”权限，并使用该机制来应用这些优先权。有些情况下，只需多做一些工作即可消除需要“免补”的情况，但受制于资源而无法实现。
- ❑ “免补”会成为一种漏洞。例如，服务于外国政府的某个攻击者想利用漏洞攻击某系统，他可能会篡改 IAVA 数据库，在某种情况下向该系统发送“免补”指令，从而导致这些系统无法修复。
- ❑ 通知发布流程也可能存在漏洞。在上述情况下，IAVA 消息系统中包含一连串的分层链接为攻击者提供了篡改来自漏洞系统的消息内容和路由。合规消息也可能在源头或传输过程中被篡改。

人们可能会问这样的问题：既然 IAVA 这么好，我们为什么总是听到政府系统遭到一些普通攻击的破坏的消息？这可能不是过程本身的问题，而是执行过程中的事。重要的一点是，IAVA 过程中没有包含统一的策略，而检查技术策略和标准合规性与检查漏洞的难度不相上下。这又将我们带回到关于标准化的话题，各不相同的漏洞管理系统使用的审计方法和语言都不一致，这使得集中的、积极的控制变得更难，而漏洞的利用更为容易。

6.6 数据分级

制定策略是实施漏洞管理项目的重要步骤，而有一个稳健的安全操作基准也同样重要。

对数据进行合理的分级将更有助于漏洞管理项目的成功，但很多公司没有这样做。安全管理员只有在了解数据敏感等级和相应的威胁等级的情况下才能更加有效地监测和管理风险。

6.6.1 案例研究：Big Tyre Corporation

Big Tyre Corporation，即 Big T，是一家全球性的重型卡车和建筑轮胎的生产销售商，在 12 个国家或地区设有办事处。总部的关键计算设施过剩，且每个国家办事处的总部都拥有自己的企业资源计划（ERP）和会计系统。该公司的生产系统由芝加哥的一个独立的 IT 团队负责管理，并在上述各国均有管理和维护队伍。由于不断地有其他轮胎制造公司被兼并进来，所以通常在一个国家设有多个销售和营销系统，并且随地域而异。而这些销售系统从没有进行过整合，因为它们的优先级很低而工作负荷很高。

在规划设计漏洞管理系统时，漏洞扫描似乎很简单，所有策略和过程均实施了，管理层高层也大力支持，产品经理热情很高。初次扫描主机后，发现很多 ERP 系统基本补丁缺失，于是迅速为这些问题分配了优先级，并据此进行了修复。由于修复工作本身的复杂性及与其他内部系统集成在一起，修复过程进展很慢。很多 IT 资源由于被占用而长期无法获得，修复过程便成为一项长期工程（半年到一年）。

在此期间，其他系统中也发现了漏洞。内部安全审计人员的一项随机调查显示，一种新模型轮胎的敏感设计数据被存放在中国本土的营销系统中，该系统漏洞百出，补丁缺失，而几个月前人们就了解到这一情况，却一直未进行修复，因为他们认为该系统并非关键系统。

安全主管对于这些敏感信息存放在这样脆弱的系统上感到十分担心，为此又请一位取证分析人员做了进一步的调查研究。经过一周的调查，结果显示，该主机至少在一年前就处于十分危险的状态了。

而这些敏感数据在该主机上存放已经 3 个月，谁也不确定数据是否已被复制并散布到外界，但可以确定的是，已经有一台公司外的主机使用了一个专门编写的 HTTP 脚本向中国的该主机发出请求，并进行了数据交换。

分析

在这一环节，公司 CIO 应该要求对公司所有敏感数据进行核查，在所有主机上确认是否存在这些敏感数据。用于处理各级别信息的适当标准应要求所有主机统一遵循，并使这些主机达到合规标准。既然漏洞扫描过程已经开始，每台主机都应根据数据分级标准和所处理的业务重新评估其各自所处理的业务数据的风险等级。

当前的修复工作仍会继续，但对于正在接受修复和计划不久即将接受修复的主机，需要先对其进行分级，以对修复的优先顺序进行必要的调整。完成对主机的评估和其上数据的分级后，这些信息会被加载到漏洞管理系统中漏洞评分的计算中，以帮助重新确定修复优先级。

这样做当然工作量会很大，并且是在漏洞管理系统部署后进行的，如果事先未对资产价值进行评估的话，这种工作量会更大。

6.6.2 数据分级流程

数据分级对于针对某一特定目标决定采取何种安全控制手段非常重要。在漏洞评估过程中，数据分级有助于确定优先次序，有助于确定资产价值和执行风险评估，而资产价值又是确定修复优先级的重要因素。本文中我们只想强调数据分级对于漏洞管理过程有着重要的作用，但并不打算用大量篇幅来介绍它，已有大量文章和书籍对其进行了更为深入的讨论。下面列出了数据分级的基本步骤：

- 1) 评估业务需求 / 影响：关键任务应用程序、正常运行时间要求、电力中断敏感性（1 ~ 10）、数据损坏的影响、数据保留时间、管理性限制、民事处罚。
- 2) 制定分级层次：确定层级数量（为简便起见不要过多）、恰当命名各层级（公共、仅限内部、限制级、保密、私密等）。
- 3) 为在使用过程中调整数据分级制定一个变更管理过程。分类信息是动态的，随着使用广泛程度的提高及业务优先次序的变化，数据的重要程度也会随之而变化。当这种情况发生时（通常是在业务策略的关键时期），就要确定哪些数据受到了影响，并对其进行重新分级。
- 4) 确定受分类层级影响的资产：一些资产上存放或处理不止一种层级的数据。如果一项资产上存放了多种级别的数据，则以最高级别为准。

6.7 风险评估

漏洞管理系统对组织开展风险评估可以提供很大帮助，但前提是先获得组织资产和业务的相关信息。风险评估结果能很好地反映组织环境受威胁程度，这将有助于确定修复工作的优先次序，以及漏洞管理系统整体部署的时间安排。漏洞评估和风险评估相辅相成、相互促进，一并实施时往往能获得更好的效果。虽然风险评估有多种手段，如可操作的关键威胁评估和漏洞评估（OCTAVE）、中央计算机与电信局（CCTA）、风险分析与管理方法

(CRAMM)、安全专员管理与分析项目 (SOMAP), 但基本的目标是一致的。没有必要对每一种方法都进行详细的了解, 但通过对风险评估方法的基本回顾, 能够对漏洞评估有一个更清晰的认识。

将漏洞评分用于风险管理过程, 其结果是以数值表达的安全态势, 且这些分值会有一个对应的解释, 这将在本章稍后讨论。以下是计算安全态势分值的流程:

- 1) 获取某目标的漏洞分值。
- 2) 将该目标中所有资产价值汇总, 得到一个总值。
- 3) 评估外部和内部资源及目标之间的安全控制措施的分值。
- 4) 计算威胁面和资产风险。

6.7.1 信息收集

为确定组织的整体风险态势, 风险评估需要收集资产、威胁、漏洞、控制措施等有关信息。由于组织和环境均不是静止的, 所以组织的风险态势也是动态变化的。漏洞管理会为风险评估过程提供重要信息。

要进行风险分析离不开对业务运营环境的深入了解。如果能了解组织结构、业务模块、网络设计 (内外连接点) 及资产清单, 将会使风险的识别和评估更为有效。

从技术角度看, 既有资产清单和配置数据库是信息收集的重要一环。如果没有配置数据库, 那么可以使用漏洞系统收集此类信息, 但这种情况下, 仍有必要确认发现的每项资产的业务功能或要求。来自漏洞管理系统的数据可能是很原始的, 只提供了主机的配置信息, 而不会提供有关这种配置是否符合业务需要, 以及在运行什么样的关键应用的信息。

在确认资产时, 注意不要漏掉外包系统和相关的信息资产, 因为工作人员如果不是直接操作和监测这些资产的话, 就很容易将其忽略。每个目标的资产价值的总和就等于该目标的总价值, 该价值不是目标的清算价值, 而是其暴露的价值, 即处于风险中的价值。一项资产可能在组织运营的好几个环节都面临风险。

安装了经适当配置的软件的漏洞管理程序可以提供很多风险评估所用的关键数据, 不仅包括前面讨论的基本资产数据, 还包括其安全状态。安全状态包括以下三个组成部分:

- 资产价值: 资产价值被输入到漏洞管理系统中以支持风险评估, 该值可以用多种方法评估得到, 本章稍后会介绍这些方法。

- ❑ 资产安全配置：大多数漏洞管理系统能够审计安全配置项的状态。漏洞管理会通过某种策略或标准更为严格地检查这些配置项的合规性，这一功能有时会利用附加工具自动完成。
- ❑ 漏洞状态：资产的整体漏洞状态会根据漏洞管理产品供应商设定的规则进行评估。

6.7.2 安全控制评估

全面了解策略、过程和标准将有助于对风险环境有一个更清晰的认识。策略可能无法反映组织的变更需要，或者反过来说，公司的行为可能也并不会总是遵从策略或标准。在风险评估过程中，有必要量化目标针对其潜在威胁所采用的安全防护措施的价值，如表 6-1 所示。威胁来自内部和外部两个方面，这样就需要计算两种安全态势值，即安全态势－外部威胁（*SPe*）和安全态势－内部威胁（*SPi*）。以下步骤可以用来确定访问控制的取值：

- 1) 针对目标点和内部及外部威胁，根据策略和标准列出安全控制措施清单。
- 2) 对适当的管理手段取值为 1，不恰当的取值为 0。
- 3) 分别汇总内部和外部风险控制措施的得分，并分别取其倒数。
- 4) 对每个 *SPi* 和 *SPe*，用漏洞系统给出的漏洞评分乘以第 3) 步中所得的值。
- 5) 将第 4) 步中得出的值乘以目标资产价值，即得到了总的风险值。

这一过程可以表示为下面的等式形式：

表 6-1 目标控制评估清单

控制手段	内部 (CI)	外部 (CE)
账户密码强健可靠	1	1
数据本地访问控制	1	1
加密存储	0	0
防火墙	0	1
网络入侵防御系统	0	1
汇总	2	4

$$SPe = \frac{1}{\sum_{e=1}^n C(e)} \times V$$

$$SPi = \frac{1}{\sum_{i=1}^n C(i)} \times V$$

C 是指控制手段 $1-n$ 的取值, 具体为 1 或 0, V 是来自漏洞管理系统的漏洞评分。

该过程中假设内部和外部威胁的严重性相同。虽然业界的统计数据不停在变化, 但其实总体上, 内部威胁管理手段要少于外部威胁管理的, 也即人们更重视外部威胁。

例如, 组织中有一台服务器负责处理工资数据。因为工资数据属于机密性很高的数据, 所以其安全控制措施的得分, 在满分 100 的情况下应达到 85。(后面会讲到如何评估资产价值。) 在本例中, 假设安全策略和标准规定了表 6-1 中所列的安全措施, 对于这些要求实施的安全措施中的每一项, 如果存在则得分为 1, 如果没有则得分为 0。

来自漏洞管理系统的数据显示漏洞评分为 75 (得分范围为 0 ~ 100), 因此, 内部 (SP_i) 和外部 (SP_e) 安全态势的评分分别为:

$$SP_i = \frac{1}{2} \times 75 = 37.5$$

$$SP_e = \frac{1}{4} \times 75 = 18.75$$

非常有趣的是, 本例的结果也非常符合常见情况, 内部安全态势得分要高于外部, 这是因为内部威胁防范措施中未安装防火墙和入侵防御系统 (IPS)。风险管理员需据此决定是否需增加更多的内部防护或漏洞修复手段, 是否真正采取行动还要取决于风险分析师的分析结果, 并要考虑诸如目标价值及其他影响, 如该业务单元模块对企业整体的重要性, 目标系统或网络的未来规划等, 这些特征值可能都是无法量化的。

6.7.3 业务需求

作为信息收集的扩展, 了解业务需求有助于风险分析人员评估当前的风险状态。很多情况下, 将数据暴露在风险中不符合业务要求。在业务混乱时期, 有时资产会被置于先前未制定业务要求的位置, 这会导致不必要的风险, 因为数据分级未能满足资产用的安全控制措施。通常, 这是由业务要求与资产的保护措施不匹配造成的。

如图 6-2 中所示的例子, 当机密数据和相关的应用程序被置于系统中之后, 资产的价值提高了, 但对资产的防护却未相应提高, 其结果是资产面临的真实风险提高了。安全组织不知道这种新部署, 认为资产价值为中, 数据级别仍然为低。这就是未将新的业务要求告知漏洞管理过程的结果, 因此资产估值未能得到及时的修改, 风险评估也未得到相应的变更。

最终，通过审查业务要求和资产上运行的应用程序和数据发现了不一致的地方，于是基于这种新的要求重新进行评估后，部署了新的安全控制措施，并最终降低了漏洞威胁和目标面临的风险。

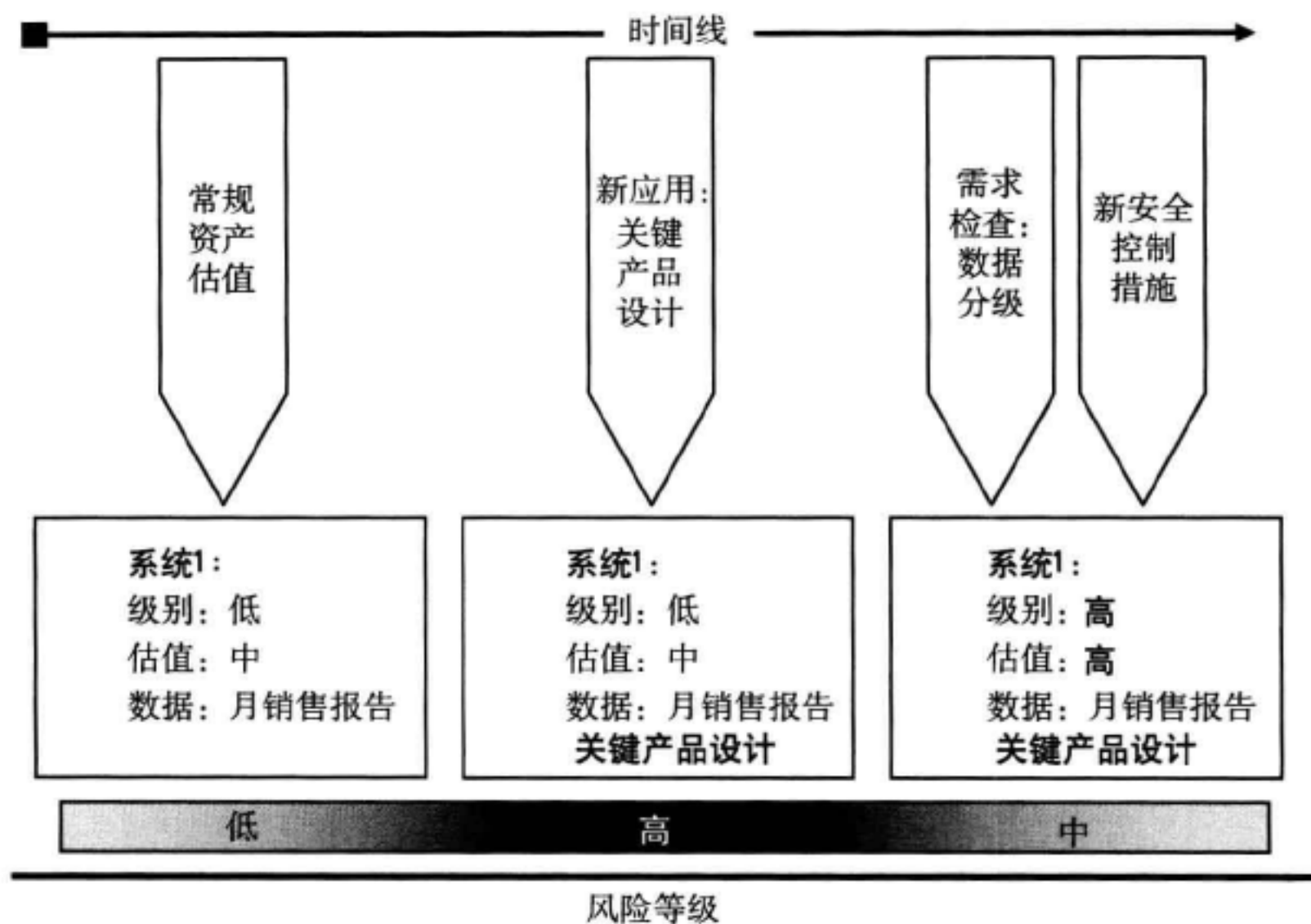


图 6-2 风险随时间发生变化

6.7.4 资产估值

了解资产的相对价值可能比较困难，但对于确定总体风险却非常重要。资产价值有几种表示方式：如可以使用具体的货币金额，或使用类似从 0 到 1.0，或低、中、高等级别来度量，前面所讨论的数据分级在此处就起到了非常重要的作用。在对数据分级前要对其进行评估。管理高级别数据的资产评估值也会较高，数据分级因此也成为资产估值中的一个重要因素。

一些组织倾向于利用某种涵盖更多风险影响要素的方法。通常的做法是制定一种能够衡量保密性、完整性和可用性（CIA）三个方面的度量标准，并对这一度量标准赋予一个最大分值，如满分 100，各项按一定比例分配这 100 分，这样最终的估值就不会超过 100。图 6-3 展示的就是这种方法，在本例中，总分 100，资产估值为 70，可以认为处于中高水平，这当然也要取决于组织确立的标准。

另一种评估组织价值的方法相比之下更为困难，是通过衡量保密性、完整性和可用性（CIA）受到的破坏程度进行评估的。这种方法需要大量的开发和努力完成评估，人们总体上

倾向于选择前一种简单的方法，因为它与漏洞评分系统能够很好地兼容。

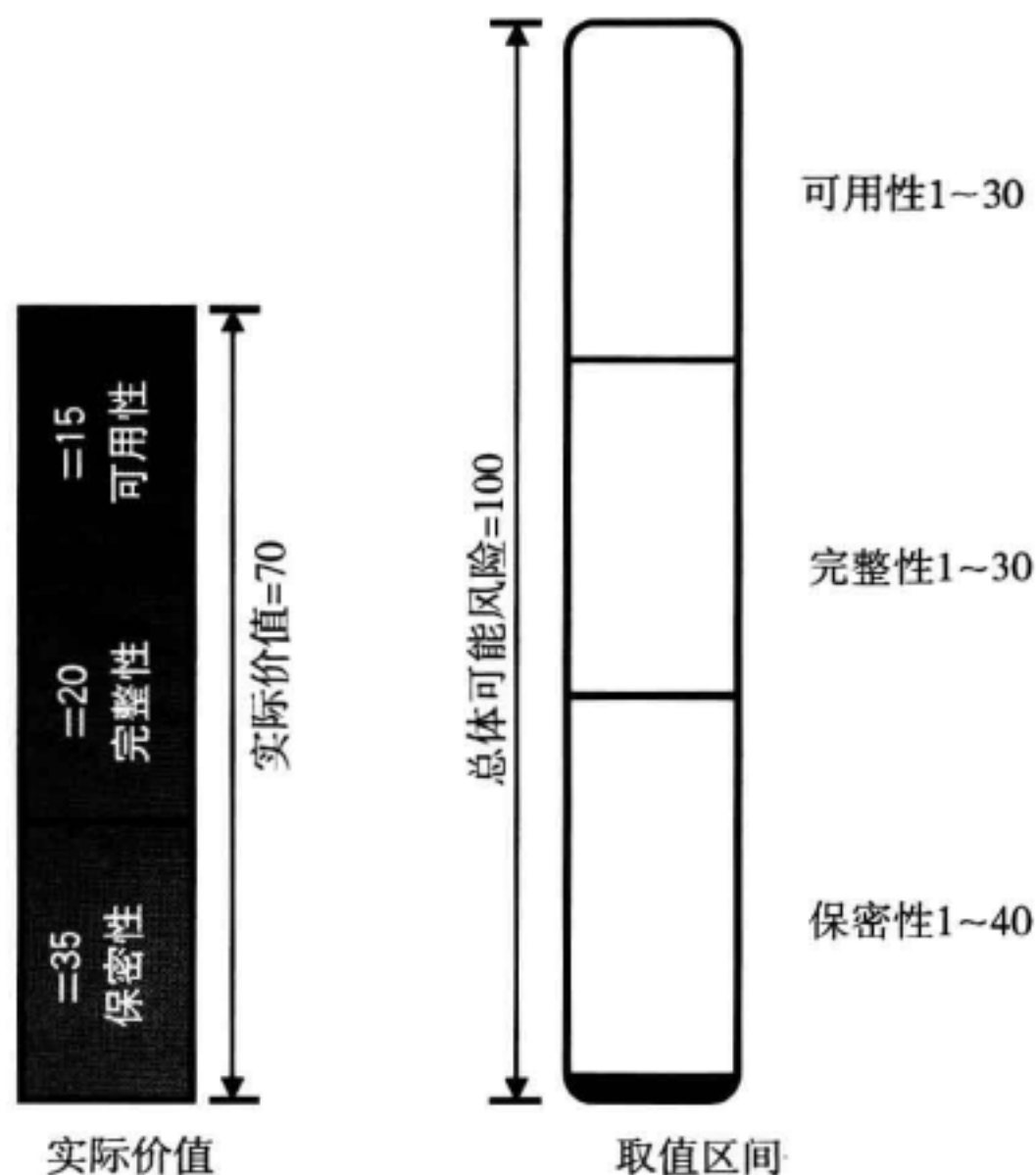


图 6-3 估值比例方法

不管采取哪种方法，最后的评估信息都会用于确定资产的优先级，以指导将来的安全投资。这些信息也是资产总体风险取值的重要组成部分。数据分级对资产估值起着极为重要的作用，关于这一点在前面已进行了讨论。

6.7.5 漏洞评估

这是漏洞管理系统工作的一部分。面对一张汇总了各项资产的漏洞的清单，风险分析员或漏洞管理人员必须决定哪些漏洞需要立即修复，哪些可以延迟或忽略。通过将以前收集的信息融合进目标的风险评分中，系统报告能很清楚地反映出优先次序，很多漏洞管理系统允许在计算和报告中插入某个值或某个标准，并以此来反映漏洞严重程度和修复的紧迫性。

在先前计算目标安全态势的例子中，可以通过修复更严重的漏洞来提高系统的总体性能，从而使内部和外部安全态势达到可接受的水平。相比额外增加安全控制措施，这种方法的性价比更高。

6.7.6 安全控制措施有效性评估

这一步对内部安全控制措施进行评估，检查其是否符合业务风险要求。在技术和流程方面，也会针对对其解决当前环境威胁的能力进行评估。

漏洞管理的有效性来自于对威胁环境变化的适应性，以及针对最新发现的漏洞进行升级的能力，这包括以下三个主要属性：

- 及时性：升级频率要高，升级要及时、迅速。
- 准确性：错判是最浪费时间的，需要继续执行许多步骤才能发现漏洞并不存在。这对于那些只会存在于一些目标上的漏洞更是如此。
- 自动性：新漏洞检查指令的接收和开展应实现自动化，并最大限度地减少干预和延迟。

6.8 小结

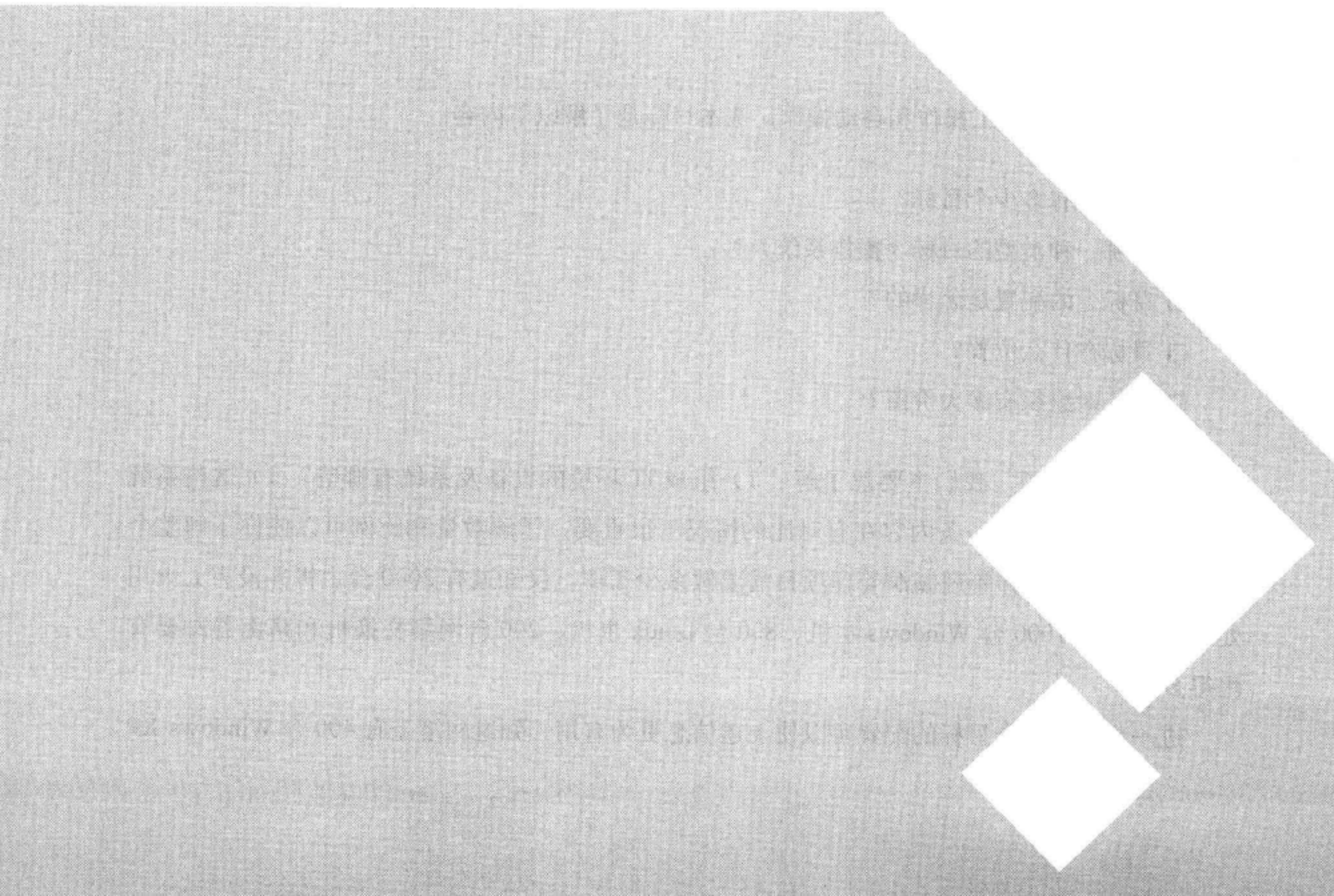
通过阅读本章内容，我们可以知道对于漏洞管理来说，过程是骨架，技术是肌肉，二者相互依赖，缺一不可。过程成功与否取决于其能否实现既定目标，通过组织的设计和功能的调校漏洞管理过程，则能够以对现有操作的最小干扰来实现组织的目标。

组织要得到一套合理的过程有很多途径。私营企业主要依赖于与现有过程相适应的定制过程，这些过程有很多共同点，基本上都与本章中所描述的过程类似。对于成熟的，以 ITSM 为中心的操作，漏洞管理流程可以很好地适应，并与服务管理定义相结合。军事领域是另一个完全不同的领域，但同样是通过漏洞采取“主动防护”的理念而取得相同的结果。

最后，任何漏洞管理流程的成功都有赖于透明的实施和广泛的接受。操作透明能够解除人们对于漏洞管理流程目标和方法的疑惑，弱化漏洞管理会威胁系统性能的看法。参与性提供给所有关键参与者一个利用漏洞管理流程优化结果得到既定收益，从而获得成功的机会。

第7章

执行、汇报与分析



7.1 介绍

通过前面介绍可知，漏洞管理过程和技术要发挥作用离不开准确的、可指导操作的信息。阅读本章后，你将能够确定你的组织需要哪种报告，以及什么样的报告内容和格式能够让漏洞管理过程更加有效。上一章我们讨论了过程，这些过程必须通过类似于输入的可执行信息来支撑和监督。当漏洞管理管理员需要决定 IT 基础设施中最需要修复的区域时，相关信息必须随时可以得到。执行修复技术任务的工程师需准确知道要修复什么和如何修复，只是简单地知道某系统存在严重漏洞是不够的。同样，管理员也不能想当然地认为当系统在接受扫描并创建时间标签时，整个过程是有效的。

本章将介绍漏洞管理流程的关键环节，并讨论各环节都需要什么样的信息。发现、评估、框架描述、修复和管理是最受人们关注的几个环节。我们省略了审计环节，并不是说该环节不重要，而是因为该环节是通过技术手段自动实现的，不要求提供报告。另外，审计是否有效在很大程度上受到处理报告的行为的影响。

7.2 发现报告

发现过程兼有人工操作和自动操作，基本目标是了解以下内容：

- ☐ 环境中有多少个目标？
- ☐ 是哪一种类型的目标（操作系统）？
- ☐ 目标上的配置是怎样的？
- ☐ 目标在什么位置？
- ☐ 目标对组织有多大价值？

通过这些问题，我们主要想了解：1) 组成 IT 环境的设备及系统有哪些？2) 这些系统的漏洞情况如何？第一条内容在有对比的情况下很重要，了解数量和比例可以使你了解整个环境的规模，并掌握管理漏洞管理项目需要做多少工作。仅知道有 2000 台主机并没有太大用处，但知道有 1000 台 Windows 主机、800 台 Linux 主机、200 台网络交换机和路由器却要有用得更多。

进一步了解每个目标的配置可以使上述信息更为有用。知道网络上有 400 台 Windows XP

主机很有用，但如果知道这 400 台 Windows XP 机器中，有 300 台安装了 QuickTime 就更好了，这是因为你知道了会有机器需要打 QuickTime 的漏洞补丁，并且这些安装的 QuickTime 很可能不是标准的企业台式机硬盘镜像版本。知道 QuickTime 有多少个版本也很有用，这样你就可以决定是否将所有机器升级到相同的新版本，从而使打补丁更为容易，也可以有效地降低总体风险。

配置项（CI）是指配置细节和所安装的软件，从中可以看出机器的安全状态，可能你所在的组织的配置管理数据库（CMDB）中已经含有了这些信息。CI 和 CMDB 都是配置管理中经常用到的术语。漏洞管理与配置管理密切相关，因为漏洞管理总是试图确定一个系统的 CI 状态。配置管理的作用是监测和评估 CI 的变更对系统的影响，当然，这样做的主要目的是维持目标所提供的服务性能的稳定。

目标系统的位置也是需要重点考虑的因素，其作用不可低估，其复杂程度也会比人们预想得要高。美国国防部（DoD）丢失的笔记本电脑中就包含了私人认证信息（PII）。在 2008 年的报告中，美国国防部总审计长就说：“虽然国防隐私办公室保证不会公布 PII，但如果仍无法找到电脑，未经授权公布 PII 的可能性也是存在的。”从此例中可以看出，如果连电脑在什么地方都不知道，就假定信息被公布的风险低，这是很可笑的。报告还说：“一项对 50 个关于保管如电脑等电子设备的 DSS 资产记录的审查结果显示，这 50 个记录中只有 23 个记录是准确的。DSS 内部管理工作不到位。”如果无法找到该资产，仅知道资产存在漏洞是没有用的，而且，丢失的资产本身就是一个漏洞。

有四种类型的位置：

- 物理位置：这种位置无需多言，它表明了资产所在的大楼、楼层、房间、层架。这在需要更换某个固件或硬件的时候非常重要。漏洞修复需要更换硬件看起来不可思议，但确实存在这种情况。一个通常的原因是硬件及其驱动程序已经过时，已经到了生命周期的最后阶段，供应商已不再升级软件驱动了，而漏洞研究人员仍会对这些产品进行测试，因此这种设备用得越多，发现漏洞的可能性就会越大。
- 受益区域：这种位置与公司所有者或客户有关，是指可以直接从系统提供的服务中受益的群体，这也是系统乃至所有组织的 IT 部门存在的主要原因。受益位置是在评估损失或变更的影响时应重点考虑的因素。如果你的公司正在运作一项业务，该业务的主要供货商使用一个系统来接受部分订单或深入关键产品的服务，那系统损失带来的影

响就会很大，而如果一个系统只是向临时咨询者提供静态的产品目录，那系统出现问题时所造成的影响就会很小，但通常都不会是后面这种简单的情况。

受益者通常是漏洞管理服务最应争取的重要支持者，但受益者通常都会忽略漏洞直到漏洞被攻击利用，而这之后，也仅仅是对未能事先修复好漏洞而进行一些惩罚。此外，这个受益者对停机维修少有耐心。从政策上讲，这是一个需要与之密切合作，并不断向其告知危险和现状的群体。

□ 财务区域：财务区域与受益区域类似，是指为系统的安装和维护付费的群体，他们的目的在于满足受益人的要求。该区域的用户非直接受益于系统，该区域通过会计法规界定，支持该代码的个人都需被告知漏洞的潜在影响，他们也会协助说服受益者，变更是必要的。

根据业务结构的不同，财务区域与受益人可能是同一群体，当然也不总是这种情况，例如，一个为系统存储数据付费的组织（财务位置）可能也是从数据使用中获益的组织（受益人），但并不总是这种情况。

□ 架构位置：架构位置对于风险评估极为重要，这是从资产的可访问性上定义的资产位置。DMZ 就是架构位置的一个很好的例子。DMZ 是位于两个安全网关间的一个网段。知道资产或目标的位置有助于确定真实的威胁水平。如果目标位于两个配置完好的防火墙及相对应入侵防范系统设备之间，那它受到威胁的可能性就很低。如果安全设备不允许 TCP 端口上的漏洞服务应答进入，那风险也会很低。此外，如果一个普通的防火墙或路由器位于互联网和一个在提供 HTTP 服务方面存在漏洞的目标之间，那么风险就可能会很高。尽可能详细地了解结构位置的情况非常重要。有很多配置方式都能导致严重的威胁。

清单程序输出的是用于验证 CMDB 或要存入 CMDB 的一系列设备和配置的列表。了解现有的资产及其配置有助于工作人员理解和掌控复杂环境。管理员、工程师和主管人员如能更快、更准确地评估事件对正常操作的影响，将对变更管理、配置管理及事件管理过程大有裨益。

图 7-1 是一张设备清单，我们对其内容进行一些简化，仅包含了资产的名称，安装的软件及版本号，还有硬件组件。这张清单能够从两个方面为工程师和规划师提供帮助：一是该清单

能够帮助规划师评估环境的复杂性，甚至是评估维护和替换功能时所需付出的工作量；二是能够帮助工程师通过确认设备潜在的兼容性和配置项选择相似的复制功能。

资产配置报告		
域名: xyzcorp.com		日期: 2010 年 8 月 2 日
名称	软件	硬件
Acctg.xyzcorp.com	Windows 2003 Server IIS 5.0 ASP.Net Java 1.5	RAID 控制器 8GB 内存 5X750GB 硬盘 2X1GB 以太网卡 2X3GHz 64-bit CPU
Webinsidel.xyzcorp.com	Apache JRun	英特网服务器 1
Loadball.xyzcorp.com	SuperBalance 1.0 应用程序安全模块	8X1GB 以太网 1X100MB 以太网
Asw10-5	SuperSwitcher 2.1 固件	ASW 主板

图 7-1 资产配置报告

调查资产所用的技术不同，信息的准确性也会有很大差异。对于任何一种技术来说，自动识别各种资产是很难的，不仅因为产品存在各种不同的版本及品牌，产品的相似性和各种各样的识别方法也极具挑战性。总的来说，在该领域，使用代理的发现技术比无代理发现技术效果要好。

7.3 评估报告

清单被广泛应用于风险管理活动中，清单本身也需要被评估。有时，清单会包含资产价值，而这一资产价值会在评估过程中被收集。评估时需收集两种价值，一是更换设备及内部所有软件、硬件的直接费用，包括购买相同或相似性能的组件的费用及安装、组装和测试等一系列替换操作的成本。收集这一价值是为了确定目标发生重大故障时替换该目标所需的费用。

第二是使用该资产所能带来的收益值，受益人位置的重要性即在于此。了解谁是受益人及其在组织内外的位置能够帮助评估者确定受益人获得的价值。如果最终资产丢失或服务中断，那也是运行该资产的公司造成的损失，外部组织只损失其付费使用的资产功能所带来的收益。网站上的广告商即是很好的例子，广告商付费在网站上做广告，当该站点受到破坏或服务中断时，广告商便丧失了广告收益。

设备的拥有者或经营者直接从资产本身获取收益，这些人构成了所谓的财务区域，用一

个代表财务会计单元的代码来定义。不管是在书面上还是现实中，该单元都因对设备提供直接支持而承担着相应风险，如果该设备已缴纳了保险，保险公司将根据保险合同承担直接损失。

结合资产的价值和各种风险测量方法可以评估资产所承受的风险，并有助于处于不同位置的团体了解风险对其各自业务的影响。了解资产所处的物理位置有助于对资产进行评估，这是因为物理位置的变化会影响风险水平。为此，应对公司的各个位置分类以代表一定事件的风险。例如，置于深海石油探测平台上的具有某种配置（CI）的计算机会被认为处于高风险环境中，遭受损失的可能性很大。

在日复一日对同一资产的管理过程中，工程师会了解到资产的位置、拥有者及资产价值，并可以据此确定修复和维护操作的优先次序，也可以借助这些信息评估某事件的危害性。图 7-2 即是这样一种报告，其中包含了资产的名称、准确的物理位置、财务价值及对业务的关键程度，这些因素会被用于确定优先操作次序。例如，“acctg.xyzcorp.com”服务器是其位置上折余价值最高的资产，在自动洒水系统出现故障时，应首先保障该服务器不受水的侵害。

资产估值与位置报告					
域名: xyzcorp.com			日期: 2010 年 8 月 2 日		
名 称	物理位置	机架	折余价值 / 美元	关键度	描 述
acctg.xyzcorp.com	Water 大街 25 号 4 层计算机机房	3C/12	150000	高	Accts Rec 服务器
mail01.xyzcorp.com	Water 大街 25 号 4 层计算机机房	5A/2	50000	中	内部邮件服务器
webinside1.xyzcorp.com	Main 大街 100 号 2 层计算机机房	6A/4	50000	低	内部网服务器 1
webinside2.xyzcorp.com	Main 大街 100 号 2 层计算机机房	6A/6	50000	低	内部网服务器 2
webinside3.xyzcorp.com	Main 大街 100 号 2 层计算机机房	6A/8	50000	低	内部网服务器 3
loadbal1.xyzcorp.com	Main 大街 100 号 2 层计算机机房	7A/10	32000	高	负载均衡 1/2
loadbal2.xyzcorp.com	Ocean 大道 5 层设备间	7A/11	30000	高	负载均衡 1/2
asw-10-5	Main 大街 100 号 2 层设备间	5	30000	中	接入交换机
asw-10-6	Main 大街 100 号 3 层设备间	7	30000	中	接入交换机

图 7-2 资产估值与位置报告

而一台放置于中西部办公室大楼 10 层的计算机遭受巨大损失的可能性就很小。对物理环境风险的评估可用风险等级如高、中、低（分别对应具体的数值区间）来表示。风险评估过程需要一份报告，该报告应包含所发现的资产的清单、各资产的估值及物理风险水平等详细信息。在确定了物理位置后，我们可以评估该位置遭受灾害、攻击及随机事故的风险。这些因素会被包含在一份非常有效的评估报告中，设施的拥有者及拥有者可据此采取适当的措施降低这些风险。

图 7-3 是一份总结性的物理风险报告样例，通过该报告，资产管理人員可以迅速掌握风险暴露情况，及需对哪些资产进行更为严格的监测。报告还提供了资产拥有者的信息，了解这一信息有助于就资产风险进行沟通，协商降低风险的措施。关键度分布图显示了环境中各种资产的相对重要程度，理想情况下，应将关键资产的数量降到最低，将损失影响也降到最低。这是很困难的，但该统计图能够帮助风险管理人員找到一个平衡点。

资产价值分布图有助于评估物理资产损失造成的更换费用，最小化资产成本对风险管理总是很有用的，对业务来说也是如此。组织中高价格的资产数量越少，潜在的替换成本也会越低。

图 7-3 也显示了整体的资产风险暴露情况。关键度只表示一个程度而非一个具体的值，它很重要，虽然它没有量化价格或评估业务所造成的确切的（甚至估计的）成本值，但它显示了该资产对公司运营的重要程度。如果该数据来自公司拥有者及利益相关者提供的调查，那么就更加可靠了。在这样一个调查中，可能会用 1 ~ 5 的等级划分来标注资产的关键度：1 表示资产如果停止运作不会造成什么影响，5 则表示会导致关键的业务操作无法运行。

图 7-3 的关键度 - 暴露度图中的 X 轴表示的是暴露度：高、中、低，Y 轴表示的是处于各暴露程度的目标的比例。图中，高价值、高暴露度的资产占到总体资产的 38%，高暴露度、中等关键度的资产占比为 22%，其余为高暴露度、低关键度的资产。理想情况下，该图表应显示随着暴露度的提高，高关键性资产的数量会下降（即呈现下倾），高暴露区域中的低关键度资产占比例应最高。图表中所说关键度主要是指资产丢失（损坏）对业务运营的影响程度，当然丢失价值相对低的资产但对公司影响却很大的情况也是存在的。

另一个考查损失的方法是将资产价值与暴露度相对比，这是从资产本身受到损失的角度进行审视而不是从对公司运营造成影响的角度。图 7-3 中也展示了可视化的价值暴露度图，在此图中，人们会对高价值、高暴露度的资产更为关注，代表这些资产的点位于图的右上角，并具体标注为“cli-clust”、“Web01”、“sql04”和“ora02”。则在资产风险评估和制定风险缓解策略时，这些主机就具有更高的优先级。标注为“cli-clust”的目标是面向客户的应用群，提供的是收入生成服务，该服务器显然有很高的价值，同时，在本报告中显示的物理（风险）暴露度也很高。

你可能想知道，为什么该服务器和其他类似的高价值服务器会被置于这样一个暴露度极高的物理位置上。该报告可以识别应对哪些主机进行评审以将其放置到一个暴露度较低的环境中。在一个每天都会增加、更换、移动服务器的 IT 环境中，每周查看一下类似 7-3 所示的

图及整个报告是非常必要的。

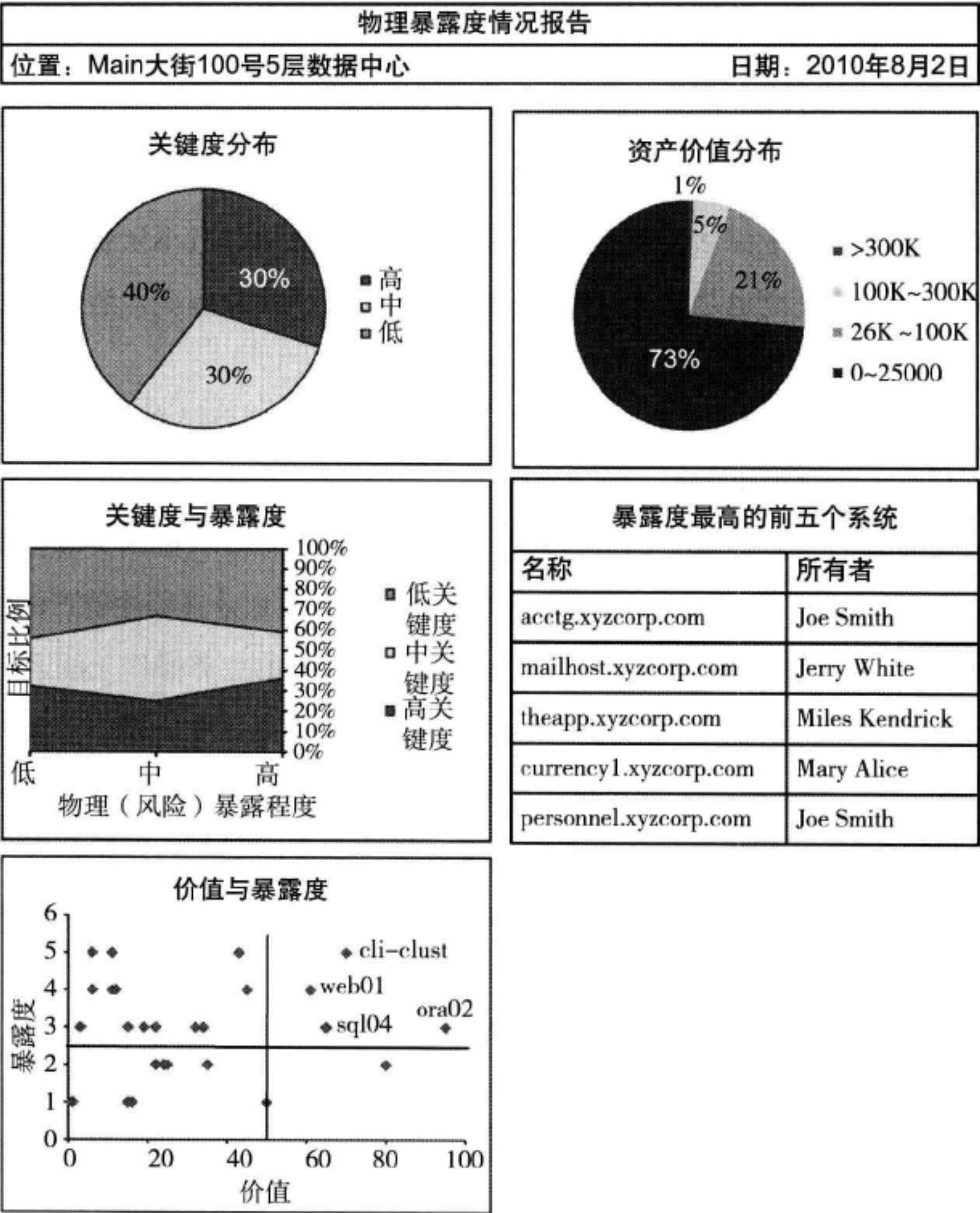


图 7-3 总结性物理暴露度报告示例

7.4 框架报告

框架报告旨在从架构的角度解读环境，包括资产的数量、重要性和价值等。例如，了解位于公共 DMZ 的关键主机与非关键主机的数量非常重要，同样，弄清多少台主机被用于后勤办公，多少台主机被用于直接创造效益对于后续的管理和维护也很重要，高层管理人员也可

以据此了解到各类资产的应用是否与公司的业务优先级相一致。

如果公司管理人员确定目前公司的顶级优先权是保持来自在线资源的当前资金收入流（相比于在线销售效果），那就需要集中大量的主机和网络设备。另一方面，如果公司为了减少对人员的依赖程度，将后台的操作自动化确定为优先目标，那么就需要在内部放置更多的主机以提高运营效率。

这只是架构描述阶段的两个很简单的标准的示例，为了更加精确地根据业务目标监测环境状态可以增加更为细致的衡量指标来更加精确地根据业务目标监测环境状态。应牢记漏洞管理程序的流程的各个环节原本就是要服务于业务目标的，将在项目章程中进行详细阐释。

图 7-4 展示了一份 IT 资产一致性报告，这对 IT 管理高层衡量支出水平或对应某一业务部分的资产数量非常有用。通常，根据年度预算和支出报告来确定普通业务实践是否符合业务的优先级，不过，在经过多年的资金投入后，审查一下目前的基础设施现状是否与公司的优先级目标相匹配，往往更为有效。该报告包含了两种基本信息：数量分布，即某一资产群组或网络中的主机数量；价值分布，即某一网络或群组中的全部资产的价值，这些值会分别以绝对数量和百分比的形式列出。本例中，绝对数量和价值所占比例最大的资产位于生产和工程部门，这种位置完全符合典型的技术－产品－驱动的业务模型。而在销售和营销模型中，则会重点关注 IT 资产和服务的递送以完成订单和销售。

IT 资产一致性（操作区）				
名 称	数量分布	数量百分比 /%	值分布	值百分比 /%
会计	4	12	170	6
人力资源 / 工资	6	18	130	4
生产	9	27	900	30
订单履行	2	6	600	20
工程	12	36	1200	40

图 7-4 IT 资产一致性示例

图 7-5 是资产价值 / 漏洞数量及偏差报告，融合了框架和审计数据。该报道有两种用途：一是告知 IT 管理人员网络间的主机分布及其平均价值，资产管理通过该报告可以迅速识别哪几台主机承载了过大的价值。在一份报告列表中，一个拥有上百台主机的网络中，提供相似功能的主机的估值应不会相差太多。为更容易地看出偏差信息，我们也制作了相关的图表。

图 7-5 中的“网络价值 / 漏洞偏离”显示了主机平均估值与漏洞平均数量间的偏差。理想情况下，不论各个网络中的主机数量是多少，每个网络中的平均漏洞数量都应当是相同的，因为影响漏洞平均数量的所有因素都应当是一致的，如复杂性、配置、流程、软件安装及技术选择等。如果所有的位置、功能及过程都是一致的，资产的平均估值应该也是基本一致的。该图表最大的用处在于：当一个网络中主机的平均估值与其他网络发生了较大差异时能够及时地显示出来。另外，当过程制定的优先次序与漏洞修复需求相一致时，漏洞的平均数量应与主机估值成反比。也就是说，如果主机估值高，那漏洞修复的优先级就高，如图中标注了数字 2 的“孟买”一项；但如果某网络中主机平均估值低，那漏洞修复的优先级也低，在这种情况下漏洞的平均数量可能就会更多，如图中标注了数字 1 的“巴黎”一项。

IT 资产价值 / 漏洞数量（网络）			
网络 / 位置	主机	平均价值	平均漏洞数量
纽约	627	532	35
香港	310	475	37
芝加哥	200	530	40
洛杉矶	190	522	37
伦敦	462	458	34
巴黎	355	620	39
孟买	280	498	45
东京	170	445	34

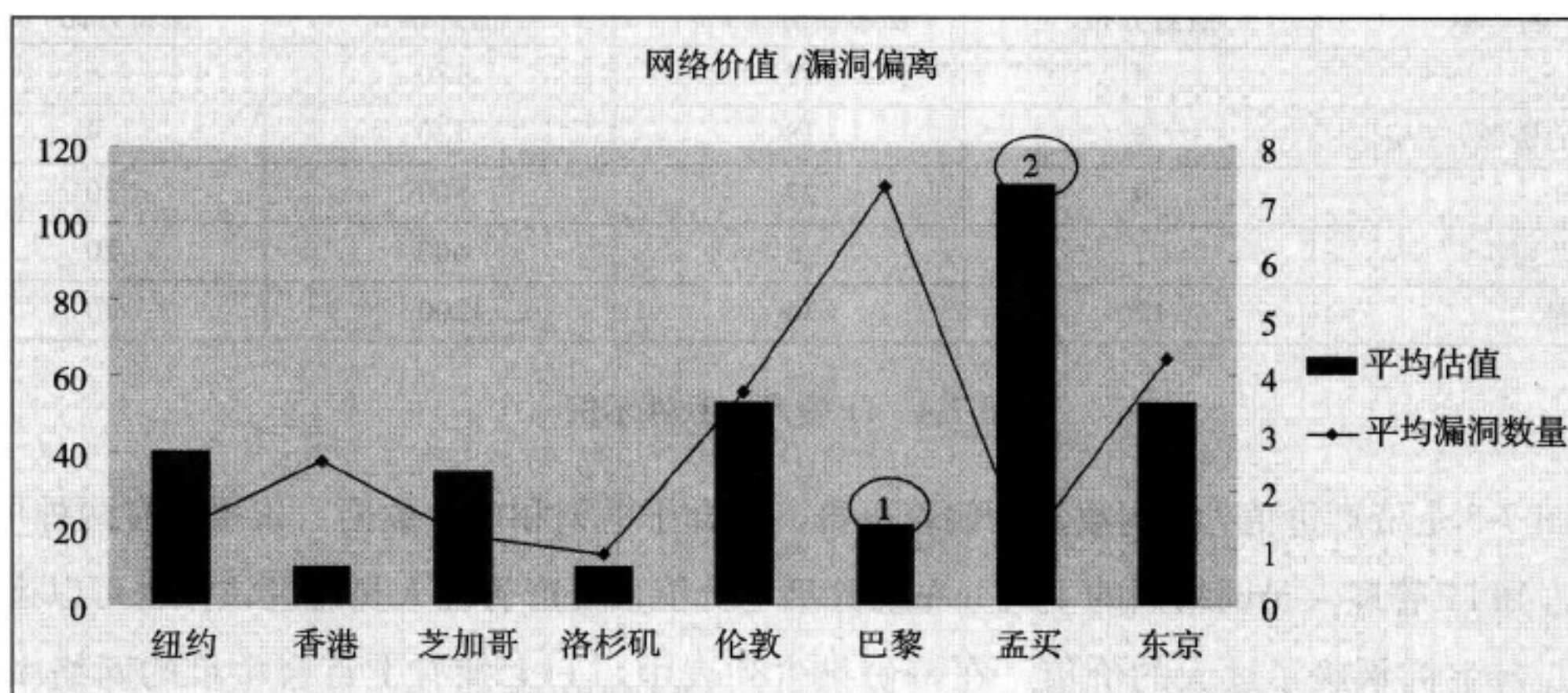


图 7-5 资产价值 / 漏洞数量及偏差报告

面对这样一张图表，IT 管理人员应重点关注的是，是否估值高的网络中会存在含有大量漏洞的情况，因为这种情况意味着组织风险极大地增加了。如果在该报告中确实发现了这种情况，就需要对该网络的漏洞趋势报告进行严密的审查，在本章后面我们将讨论一个更为精确的确定漏洞优先级的方法。

7.5 审计报告

审计过程是极具操作性的。漏洞修复活动每天或每小时都在进行，一旦请求响应，审计就会自动执行。审计过程中除了总体审计的进展报告和整体的成功率、失败率，其他内容很少提供给管理者。审计阶段需要两种类型的报告：当前的审计状态报告和审计时间表报告。当然这是对使用独立应用程序或代理的主动扫描技术而言的，被动漏洞扫描技术则会以另一种方式提供各个数据收集器的状态信息，对性能的审视角度也会有所不同。

7.5.1 主动扫描审计报告

1. 审计状态报告

对于主动扫描来说，图 7-6a 中所示的当前审计状态报告显示当前正在接受审计的几个网络。每个网络均有各自的扫描配置，规定了应用于每个扫描的具体参数，配置的名称很好地说明了参数，这些参数对于系统管理人员进行规划和制定组织策略都非常关键。IP 范围虽然不是那么重要，但它能告知操作人员具体哪个网络正在接受审计。审计状态回答了与当前漏洞操作相关的三个重要的问题：

- ☐ 哪些审计正在进行并即将结束？
- ☐ 哪些审计刚刚结束？
- ☐ 哪些审计将要开始？本例中所示时间范围是前后 24 小时。

该示例报告显示了某一时间点的情况。部分信息已属历史性信息，但当快速评估对环境的影响时仍有作用，更早的信息对于分析由于技术原因而失败的审计结果仍有意义。

2. 审计时间表报告

该报告用于追踪主动扫描资源的使用情况，如果一个组织的网络分布很广，且各网络的

接入情况、带宽限制、扫描器数量等情况不同，那么有一份有关当前扫描资源的分布情况的报告对于规划扫描很有用处。图 7-6a 是一个报告样本，其中两列内容对于可用资源的规划扫描很关键：上次扫描持续时间和主机。这两列内容有助于设置性能期望值。因为扫描应当在一个对业务影响最小，并且可接受扫描（在线）的目标机数量最多的一定时间窗口内进行，掌握扫描所需的时长以及时间安排有助于确定扫描资源的分配及发现限制条件。

例如，报告中阴影显示的三个网络（图 7-6a 中的卡拉奇、大阪和上海）所需的扫描时间较长，这可能是由于当地正在进行其他扫描或是目标网络的限制而导致的带宽受限造成的。观察发现，由于同一扫描设备所执行的两个网络扫描任务的时间窗口有重叠间隔，所以我们应该为其中一个网络扫描任务重新安排扫描时间。

7.5.2 被动扫描审计报告

对于被动漏洞评估来说，可以忽略分析网络流量时对环境造成的影响。于是，这导致监控对操作产生的影响没有太大变化，但管理漏洞和清单的要求却立刻发生了较大变化。图 7-6b 是一份各种被动漏洞分析器状态的实时报告样例，选取度量标准的方式可以有多种，这里所用的只是一种可能的衡量标准的组合。我们假设每个网络的某一物理位置上均有一台扫描设备，设备名称即监控网络的名称，根据操作环境的变化，图中显示了三种类型的网络及可能的度量方式。

对于某一特定的时间窗口，该报告有以下几列：执行分析任务的网络 / 设备的名称、前一时间段内出现的新目标、当前观察时间段内的网络协议、发现的新漏洞及分析器使用的峰值和平均情况。

“纽约 - 财务”是一个财经网络，分析器平均利用率相对较低，当按月和年来计算时，峰值的出现具有周期性。由于主机的增加和移除频率相对适度，新目标的数量相对较低。“法兰克福 - DMZ”是一个公共的 DMZ，很多用户都可以访问它所提供的资源，网站繁忙时分析器的使用峰值很高（90%）。由于该网站是面向公众的，所以分析器的平均利用率也很高。不过，这种环境不经常变化，因此新目标的数量一直较少，当这一数字发生变化时，变更管理过程应当能够快速进行调节。之后，利用率变化趋势报告将有助于确定当前网络中扫描设备的性能是否能满足变化的业务技术环境的需要。

当前审计状态 / 时间表 (+/-24 小时)					
网络	IP 范围	状态 / 时间	配置	上次扫描时间	主机数量
纽约	10.30.8.0/22	完成	高 - 周五	1:20	430
孟买	192.168.70.0/24	完成	2Mb- 周五	2:45	612
法兰克福	192.168.1.0/24	25%	高 - 周末	0:59	181
杜塞尔多夫	192.168.50.0/24	30%	高 - 周末	2:07	210
巴黎	192.168.55.0/24	75%	6Mb- 周末	1:19	167
卡拉奇	192.168.85.0/24	01:00	256K- 周末	9:52	140
大阪	172.16.100.0/24	14:00	1Mb- 周六	11:31	202
上海	192.168.3.0/24	16:00	2Mb- 周六	12:44	287

a) 当前审计状态与审计时间表报告

当前审计状态 - 窗口: 24 小时					
设备	新目标	协议	新漏洞	利用率峰值	平均利用率
纽约 - 财务	7	8	54	60	20
纽约 - DB	1	2	6	90	54
孟买	15	16	7	25	18
法兰克福 - LAN	25	8	82	65	30
法兰克福 - DMZ	2	4	14	73	52
杜塞尔多夫	14	24	96	60	18
巴黎 - DMZ	0	3	221	90	72

b) 各种被动漏洞分析工具状态实时报告示例

图 7-6 扫描审计报告

“法兰克福 - LAN”设备显示大量的新主机被发现，这在允许大量用户无线接入的公共环境中非常常见，这些用户可能是网站客户，也可能是全球性公司的驻外员工。在一些公司中，业务发展的趋势是允许任何人利用合适的设备接入任何网络，这带来了新的安全挑战，其影响反映在这份报告中即为出现了异常的大量的新主机。分析师或系统操作员能够通过每日监控新主机的数量，了解工作环境的技术表现如何。

最后，图 7-6b 中的“巴黎 - DMZ”一项显示没有新主机出现，这是正常的，但发现了一些新漏洞，这一般要么是因为漏洞研究者发现了新漏洞，要么是因为是一台或多台主机的配置发生了根本改变，这就是与变更管理程序保持一致的重要原因。如果更改配置或安装新软件造成了漏洞，那么在更改过程中就应予以修复。同样，如果发现了新漏洞，系统工程团队就应给予更多关注。漏洞数量在报告中好比是一个危险信号，其危险程度取决于系统的操作活动。

7.5.3 审计趋势分析

1. 漏洞趋势报告

如前所述，如果某一给定网络中的主机值与修复优先次序不一致，那么就应该审查一下漏洞趋势了。漏洞趋势报告中所示的是未经分析（原始）的已发现的漏洞的数量及每台主机上的漏洞平均数量，原始计数会说明网络的整体漏洞活动的层次，但平均值会告诉我们可以知道修复功能能否跟得上新发现漏洞的步伐。注意：该报告是带有关键性能指标的修复跟踪报告的替代品。

图 7-7 是一个漏洞与目标报告，显示了 20 周的时间长度内某网络或整个组织的漏洞趋势，该报告可用于同一团队管理和修复的其他网络或主机群组，可用于识别那些由于运行困难或意想不到的环境变化而引起非正常增长的区域。在这个例子中，目标的数量持续增长，但相对于业务增长来说，这种增长不算异常。但漏洞数量增长速度却非常引人注目，这可能是由于环境中增加了更为复杂的主机及应用，这种现象一般会显示为漏洞绝对数量的快速增长，而非持续向上的趋势。

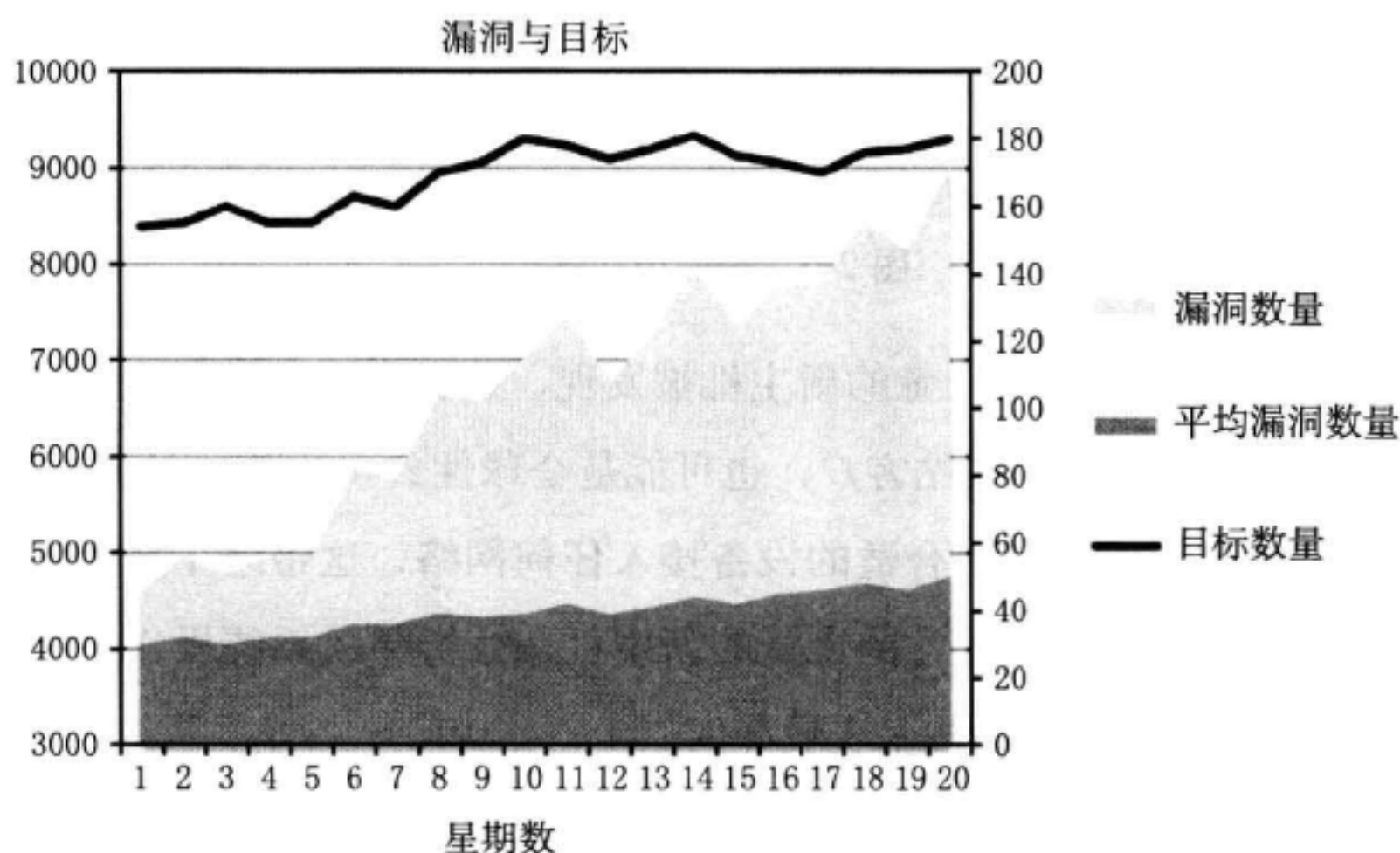


图 7-7 漏洞和目标报告：持续 20 周内的重要发展趋势

新目标的增加与主机数量的增加保持线性一致性是可能的，即保持每个目标上的平均漏洞数量相同，为方便查看，我们增加了一条表示平均漏洞数量的线。可以看到，该线是稳步上升的。如果工作人员未能成功维持新接入的系统附加的负载，就会出现这种趋势。一般，这是由于缺乏更高效的修复能力造成的；也可能是由于新主机部署后，未及时地对监测和控制

系统进行相应的配置以对其执行打补丁或其他自动修复任务造成的。

2. 网络风险趋势报告

该报告能够反映给定网络或目标群组的风险趋势，虽然很简单，但对评估 IT 基础设施及业务的总体风险非常重要。报告的阅读者为中高级风险管理人员和合规管理人员。不要将该报告与前述的漏洞趋势报告混淆，后者主要处理一些数据，不显示总体风险。风险趋势报告是一种风险评估工具，它在综合了漏洞、危害程度、估值及暴露程度等因素的基础上，以一条曲线来显示风险趋势。

由业务区域或 IT 安全团队来审查此报告有助于资源分配及进行更为详细的风险分析。一般来说，该报告是用于反映组织中的各个区域，如生产、产品开发区，或某一办公室、部门、或功能区，这些区域的划分并不一定要按照为配合漏洞审计系统而配置的网络规范进行，相反，这种划分可以为漏洞审计系统产生的数据提供一个不同的视角。

要创建该报告，最重要的是要确定绘制数据的公式。下面用于绘制报告的样表中，其风险值的计算基于特定的参数：暴露程度取值为 1～5，1 代表暴露程度最低，5 代表最高；资产估值范围为 1～100，100 代表该资产对组织最有价值，如果损坏、丢失、被盗或致使无法使用，造成的损失也最大；网络的漏洞评分为 0～100，100 代表极易被利用，可以通过它远程访问所有目标。风险计算公式如下：

$$\text{风险} = \text{vscore} \times \frac{ex}{5} \times \frac{av}{100}$$

其中，*vscore* 是指漏洞评分，*ex* 指暴露程度，*av* 指资产估值。

上述计算方法的基本思想是用漏洞评分乘以总体暴露程度和相关的资产估值除以各自取值范围的分数。此例中的暴露程度用 1～5 来衡量，需要将其折算为 0～1 之间的一个分数以保证最后的结果不超过 100。同样，资产估值也是以 0～1 之间的分数（*av*/100 的形式）参与计算。

图 7-8 中的趋势报告显示了公司的工程区存在风险。尽管每月的风险值都有波动，但整体风险程度是随时间明显增大的。风险值在从一个月进入到下一个月时出现波动的现象告诉我们，虽然进行了主动修复，但风险并未被完全消除。更为重要的是，我们从风险趋势线上可以看出，风险是在增加的，这在某些图表中可能不会显示的那么清楚。很显然，应当对工程区进行审查，寻找风险增长的原因。

工程				
风险趋势 月份	漏洞评分	暴露程度	资产估值	风险
1 月	75	3	40	18.00
2 月	77	3	40	18.48
3 月	78	3	40	18.72
4 月	78	3	40	18.72
5 月	75	3	40	18.00
6 月	74	3	45	19.98
7 月	76	4	45	27.36
8 月	77	4	45	27.72
9 月	79	4	45	28.44
10 月	79	3	45	21.33
11 月	80	3	47	22.56
12 月	82	3	47	23.12
1 月	83	4	47	31.21
2 月	85	4	48	32.64
3 月	88	3	48	25.34
4 月	90	3	49	26.46
5 月	90	4	49	35.28
6 月	92	4	48	35.33

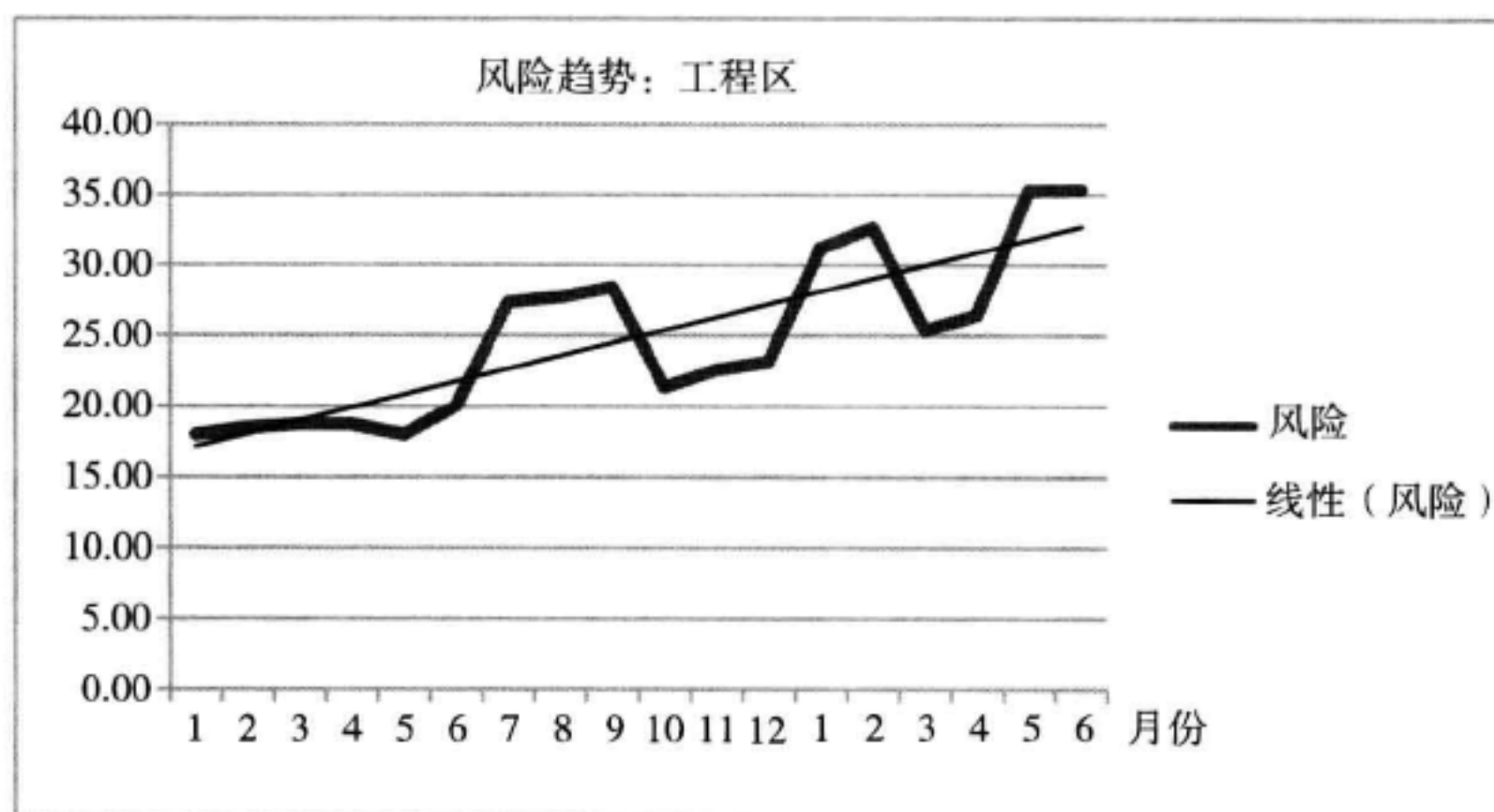


图 7-8 风险趋势样表

看一下详细数据就会发现，图 7-8 中资产估值是上升的，而同时漏洞评分也在升高。分值在一月份为 75，此后的 18 个月一路攀升，到第二年的 6 月份达到 92。从表中还可以看到，

漏洞评分在前 8 个月处于良好的控制之下，从 9 月份开始上升，并且几乎再没有下降。此外，从 7 月份开始，由于资产估值的提高使得风险值增加，这说明可能有新的、更昂贵的资产加入进来，并引起了漏洞评分的提高，这可能是由新资产复杂程度更高导致的。这种情况多是由于工程或安全人员对净资产掌控不足造成的，接下来需要进行修复，并对影响主机的特殊漏洞进行检查。

7.6 主动扫描：时间安排与资源

当使用代理或应用程序进行主动扫描时，制定扫描时间和进行资源管理都是非常重要的。如前面讨论的，当对某网段进行主动扫描时，就会对网络、目标、各种设施组件，甚至可能是整个业务运行都产生影响。为合理调控这些因素，系统操作员及过程管理员必须能够掌握一定的信息来制定适当的扫描参数和扫描时间。

7.6.1 审计参数

审计参数是按照网络规定的审计执行规范设置的一套规则和限制条件，这些参数包括但不限于：

- ☐ 带宽；
- ☐ 协议；
- ☐ 端口；
- ☐ 漏洞集；
- ☐ 认证。

每个参数都会极大地影响审计的效果、持续的时间及对设施产生的作用。

1. 带宽

漏洞扫描对基础设施最直接的潜在影响就表现在带宽上。工作时间内即使一个弱的 WAN 连接上的网络带宽量，其消耗的网络带宽也可能会显著影响一项或多项重要服务的服务水平。要避免这些影响，调整审计时间即可，但这可能会造成很多台式机或便携式计算机无法接受审计（因为它们处于关机状态），我们会在第 8 章讨论如何安排扫描时间。

2. 协议

执行审计时，协议是一个关键因素，知道哪些协议与业务相关性最高，哪些协议有可能发现最多的漏洞而造成的影响却最小也是非常重要的。在很多公司里，使用最多的协议是 TCP 和 UDP，除非应用程序专门指定要用 UDP 协议，否则一般都会使用 TCP。例如，很多市场数据系统由于性能的原因都要求使用 UDP。UDP 中没有双向握手过程，因此不保证信息的成功传递，但其对网络资源的消耗与 TCP 相比要小得多。

3. 端口

除 DNS 使用 UDP 53 端口外，其他 95% 的应用和服务一般均使用 TCP，因此，对很多公司来说完全不使用 UDP 是可行的，这样可以省略全程审计，从而缩短审计所需时间。

对审计结果影响更大的是对特定端口上所要进行评估的协议的选取。这是为求得最大审计效率的一种折中做法：如果选择了过多的协议，扫描对基础设施的影响将是巨大的；但如果选择的协议数量太少，扫描就会漏掉某些主机和漏洞。一种较好的判断选取哪种协议的方法是在环境中选取一些目标样本进行扫描，这些目标样本使用了所能想到的尽可能多的协议，然后查看扫描结果，确定目标上的哪些协议是活跃的。

常用的协议，如 HTTP/HTTPS、FTP、SMB 在几乎所有的环境中都使用，但也会使用到其他一些不为人们熟悉的用于流媒体、P2P 和 Web 服务的协议。另一方面，也可能有不通用的，只针对某个特定平台的协议。例如，在一个完全使用微软产品的环境下，Novell 协议就是不适用的。如果业务中普遍使用的是 Sybase，而 Oracle 不是标准数据库，那就应注意供应商通常使用的端口。

同样，为保持一致性，应审计主要数据库供应商的所有协议，以发现 IT 团队不提供支持的不兼容、不一致的产品。下表是一个公司最常用的 TCP/UDP 端口：

FTP data	20	用于 FTP 服务器传送和接收数据
FTP control	21	控制从 FTP 服务器流入 / 流出的数据，客户端 / 服务器在此端口建立连接，并且数据的传输使用其他频带（TCP-20）
SSH	22	该认证、加密协议用于通过命令行远程接入系统，也用于文件复制
TELNET	23	在命令行通过非加密渠道远程访问系统的常用方法
SMTP	25	用于服务器从其他系统接收电子邮件，常用于邮件服务器而非客户端
DNS	53	通常与 TCP 结合使用时，用于区域转换，而非标准化查询；与 UDP 结合使用时，该端口用于请求域名解析
HTTP	80	HTTP 是网络服务器上最常使用的协议，该端口用于接收网页接入请求

(续)

POP3	110	该端口用于接收从监听服务器的信箱中收集邮件的请求，属于一种邮件存储-推送协议，当邮件被送至客户端后会从服务器上删除。服务器与客户机间传送的内容一般不加密，但会有简单的认证过程
PRC	135	该服务允许其他计算机访问和运行本地机器上的程序，该端口上的请求处理存在大量漏洞
IMAP	143	允许用户在支持该协议的任何主机上统一查看和操作电子邮件，而不是简单地从主机获取和删除邮件，查看后邮件仍存放在服务器上
SNMP	161 (UDP)	用于管理使用 SNMP 的设备，几乎可以接入到网络上的任何设备
HTTPS	443	与 HTTP 相似，该端口监听使用 SSL 认证的加密网页的请求。
SMB	445	该协议用于文件共享，常用于基于微软产品的系统，也用于 UNIX 和 Linux 系统（名称为 SAMBA）
IMAP over SSL	993	功能与 IMAP 相同，但有加密功能
ASP.net session state	42424	微软的 ASP.net 平台使用该端口与网络服务器交流会话状态信息，这样网络应用程序可以跟踪业务处理状态，而不用在客户端计算机上保留信息记录（cookies）

4. 漏洞集

选择要对哪些漏洞执行检查也会对审计结果产生影响。对所有可能的漏洞进行检查虽然是很自然的想法，但有些情况下，可能需要对某类漏洞排除审查。有些主动漏洞管理系统对应用设有专门的审查，但可能会对这些应用程序产生不良的影响。例如，某个目标的安全策略可能会规定，如果登录五次不成功就会锁住该账户。但漏洞扫描过程一般都会使用暴力破解方式来测试 SNMP 社区字符串（SNMP community string）和密码的强度，如果对某个业务应用访问远程系统所使用的服务账户实施这种漏洞检测，就会导致服务账户被锁，进而导致应用中断。

已知有一些种类的主机扫描会引起支持这些主机的网络设备失效，这有时与参与监测扫描程序与目标间连接状态的交换机和路由器的工作方式有关。网络设备在实际使用中的确切反应是不可预知的，但供应商一般会知道并有说明。

5. 认证

之前在讨论独立漏洞扫描时也提到过认证扫描，对目标的认证扫描能够揭示关于目前漏洞的更多信息。但有时从一种“黑客的视角”来审查目标可能会有用，因为入侵者一般无法获得证书，他们会从外部的某个有利位置进行强力扫描，并可能会获得更多信息，对目标的影响也会更小。这种扫描非常快，但可能会受到防火墙和 IPS 设备的阻挠。取消登录步骤甚至是取消对此类扫描的拦截，有助于获取最多的信息。

在进行认证时，证书数量越少越易于管理。目录服务，如 Active Directory，可用于创建和发布专门针对某主机或读取关键系统信息的证书，这些证书的权限应仅限于完成某项任务。

牢固的密码及完备的账户登录处理过程会发挥重要作用。此外，证书应定期更换，但不能太频繁以免影响审计操作，一般为半年到一年一次。一些业务单元可能会要求系统使用不同于其他业务单元的一系列证书，这属于管理和策略问题。

7.6.2 时间安排

如果扫描强度可能影响业务操作，那对主动扫描资源的规划使用就非常关键。像在第5章中讨论过的，主动扫描会影响主机和网络的性能，引起防火墙和路由器等设备中断服务，扫描设备的成本也可能会因为一些原因变得十分昂贵，而并不仅是购买成本。为此，合理调度这些设备的使用和对特定目标的扫描非常重要。

只有当台式机和便携式计算机开机和联网时才能对其进行扫描，也就是说扫描只能在工作时间进行。服务器由于全天都在运行，所以时间的选择更多地应该考虑对业务的影响。某些服务器全天都在为客户服务，因此只能选择在业务不太繁忙时对其进行扫描。

扫描器的布置和使用通常有三种策略。一是集中摆放和管理，并对所有目标区域进行审计；二是添加额外的扫描器置于每单位带宽主机数量最多的地方，以减少对网络流量的影响；三是综合利用前两种方法，以取得最佳效果。

中到大型公司（20 000 ~ 100 000 个目标点）一般可以部署 30 ~ 50 个扫描器，安排这些扫描器的使用就像安排列车时刻表。不幸的是，漏洞管理供应商对此问题还未妥善解决，因此有必要进行细心的规划。

7.7 审计趋势与性能报告

审计趋势与性能报告对安全主管、技术主管或漏洞主管监控各过程、系统及工作人员表现来说都是至关重要的。该报告能提供持续的性能监测数据，使用户可以发现并处理系统缺陷。本节将讨论两种报告：一是显示漏洞过程趋势的基本报告，二是有助于识别干扰来源的高级报告。

7.7.1 基本报告

首先定义漏洞管理过程甚至是具体的修复过程，如果你深入领会了本书的精神，那么可能早已完成了这一步。图 7-9 是一个漏洞管理过程的概要图，显示了主要的功能模块，阴影项

是每步骤中输入和输出的数据元素。

确定每步的输入与输出。图 7-9 中的过程在每步均有输入和输出，并用箭头进行了标识。这些数据并不完全是直接产生于漏洞管理过程，有的输入来自组织中的其他过程，或是由安全研究人员及供应商开发和提供的。输入项中的“发现漏洞”是指世界各地的研究院发现的与该组织有关的所有漏洞。输出项是在组织每个目标的 IT 基础设施上发现的漏洞。后者会被输入到下一个阶段——“确定优先级”。

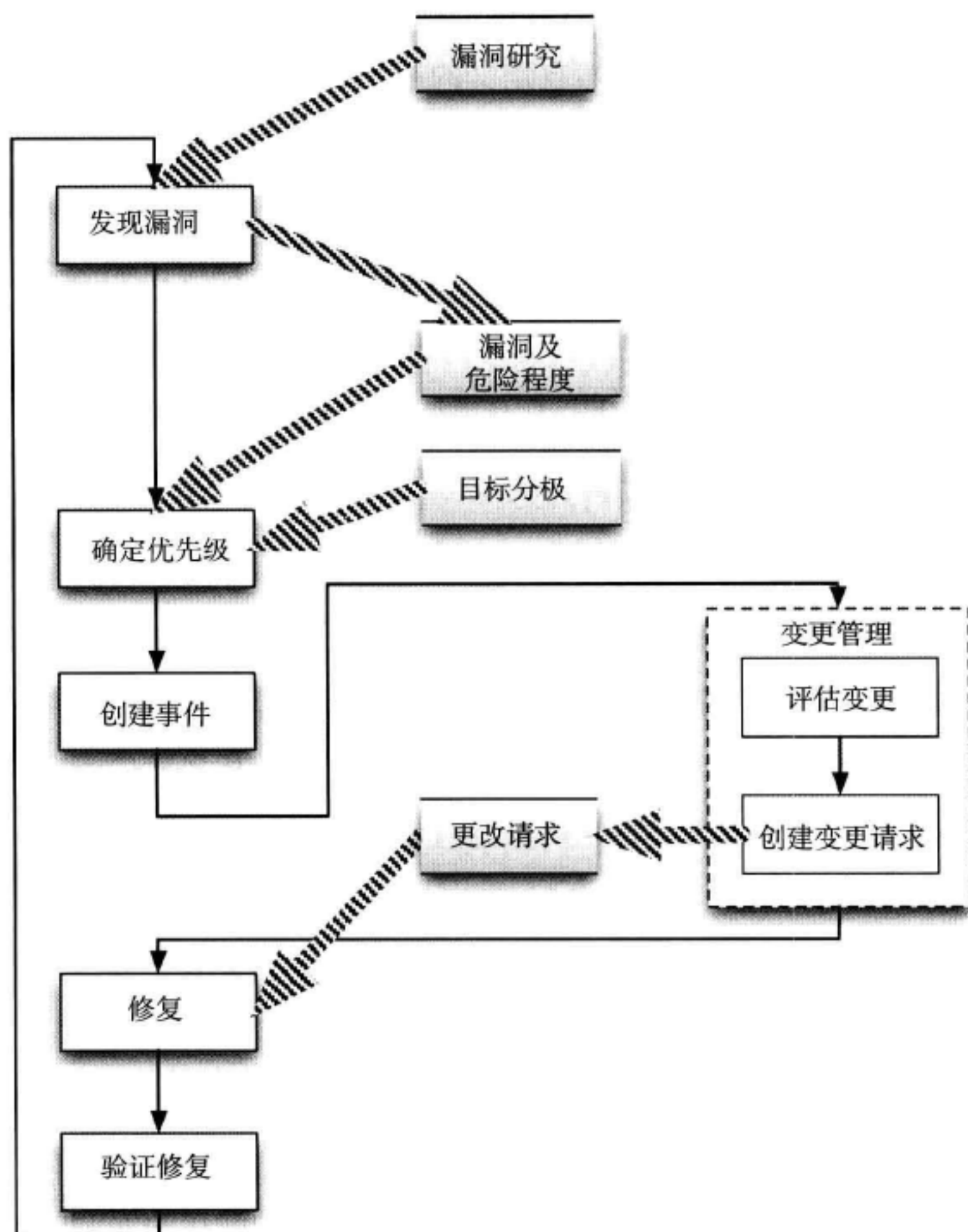


图 7-9 漏洞流程与数据流简图

下一阶段的输入项还包括每个目标的分类级别（潜在影响）及各漏洞的危害程度。“确定优先级”的输出是根据需要进行修复的优先级列出的漏洞 / 主机列表的组合。设定优先级是一

个手动或手动控制的过程，手动控制过程中生成优先级列表是自动的，而其输入项是受管理员调控的。

该过程输出的是变更控制标签或事件标签，这取决于组织如何规划运营支持系统。在我们的例子中，是创建了一个事件并将其分配给适当的变更管理人员或拥有者。在变更管理过程中，漏洞管理以外的活动目的是使变更与其他 IT 操作相协调。

最终，修复过程使用来自变更管理系统的标签作为输入项。修复过程完成后应对漏洞修复效果进行一些验证，验证的结果是该过程的输出项及一个“干净”的系统。

确定影响过程性能的重要变量。虽然自定义的过程和组织要求会有所不同，但有一些基本的度量项是共有的。根据组织运作周期的不同，这些度量项合适的评估时间频率会有所不同，对绝大多数公司来说，一般是每周、每月或每季度。这些度量项应根据不同的业务和 IT 领域进行分组。例如，如果组织中的每个业务单元有单独的 IT 群组，那么查看业务单元所对应的度量项及其评估时间将会是十分有益的，能够使过程更为统一有效，并使结果更可预期。管理风格和文化不同会极大地影响过程的有效性，导致过程性能的较大差异。

这些信息最终会被用于创建性能趋势分析所需的关键报告。以下报告在管理典型的漏洞管理过程中一般会：

- 根据运行区域区分配置型漏洞：运行区域是指执行某项功能的技术和安全组织的某个部分，如信息系统、网络服务器管理，或台式机管理等。这些区域每个都能够对漏洞的数量及危害程度造成显著影响，应关注它们对组织整体风险的影响。例如，网络交换机可能持续显示由未正确配置 SNMP 服务而导致的漏洞，这可能意味着未能遵从标准或要求对标准进行修订。
- 根据危害程度区分运行区漏洞：了解每个业务运行区内各种不同危害等级的漏洞数量有助于发现优先级中存在的问题。
- 根据危害程度和 / 或种类区分的已发现与已修复的配置型漏洞：漏洞修复和规避的顺序对于修复和配置过程作用的发挥有很大的影响。例如，我们可以从发现的弱密码（一种配置项）的数量与得到修复的数量的对比中看到某种潜在的漏洞的处理情况，即了解有多少漏洞是在配置中就规避掉的，有多少是在后续的修复中清除掉的。这种衡量方法既可以针对特定漏洞使用，也可以针对类别或危害程度。这种方法还有助于发现过程未能满足整体优先级制定要求的情况。

- 根据目标分级区分已修复漏洞配置：与前一种类似，了解根据业务重要性区分的每种主机的漏洞情况有助于将注意力集中到关键主机的漏洞管理活动的优先级的制定上来。在执行配置和修复过程时，资源分配的优先级应与之相匹配。不同的 IT 团队共同管理重要主机的情况是很常见的。这种方法也有助于识别技术管理团队在制定优先级的能力方面存在的缺陷。

推而广之，该报告可以定制为资产估值乘以漏洞评分，虽然这个值可能并不十分准确，但能为确定修复的优先级提供一定参考，这种以业务的价值为依据的方式要优于以简单的数据分级为依据的方式。这是非常必要的，因为资产估值并不总是与数据分级一致，当资产分级（和估值）标准并不统一时就会出现这种情况。虽然数据分级应遵从资产估值，但资产所有者可能并不认同该估值，因为组织内的各成员对资产价值的看法并不一致。

- 根据修复类型区分漏洞：这种方法提供了修复漏洞的方法用到了什么样的补丁或修复哪些配置的相关信息。可以进一步确定各操作组的漏洞数量，以确定哪里需要资源。例如，信息系统可能会遇到大量需通过打补丁解决的漏洞，因此需要对该系统加强补丁管理或是增加一个补丁审查过程。
- 新漏洞：各种漏洞衡量方法都不能离开整体的漏洞环境，套装软件中可能会有更多的新漏洞被发现，因此修复也会越来越多。这种情况下应考虑其他的衡量方式，以避免错误地认为是过程缺陷导致了这种变化。如果某一区域的漏洞数量的增长率与所发现的全部漏洞的增长率相同，那其根本原因不太可能是内部程序缺陷。
- 修复后无法通过验证的漏洞：在我们的示例过程的最后一步中，在变更或事件标签结束之前需对修复后的目标进行验证，这种方法可以直接查看修复过程的有效性。
- 每种应用的漏洞数量：指在一段时间内每种应用中出现过的漏洞的最大数量，这有助于理解环境的复杂性，发现那些更为脆弱，需要更多修复的应用。这些信息可用于识别运行某项应用的真实成本，并可能使人们重新评估某供应商的产品的可用性。例如，如果套装应用 XYZ 价格很高，但在半年到一年的时间内发现漏洞很多，这说明该软件的安全质量不高，于是人们会对该应用进行评估，确定修复成本会不会太高，是否应换一家供应商，或要求目前的供应商提高信用。

7.7.2 高级报告：控制图

迄今为止，所有的审计报告提供的都是短时间内的信息，被用于管理日常的操作。而趋势报告会包含更多信息，并对战略性安全规划起到重要作用。如了解4个月或1年内发现的新主机的数量可以知晓业务周期内重要但未经沟通的更改。通过趋势可以识别过程缺陷，从而为管理整个漏洞管理程序提供帮助。

任何过程都有自己运行的范围，通过测量某过程的输出，并用统计的方法进行描述，我们就可以确定该过程的正常分布或运行范围，即过程能力。任何超出该范围的结果意味着过程出现了故障，或是过程有需要提升的空间。对于漏洞管理，我们可以测量发现或修复的漏洞数量随时间变化的情况。如果目标的数量或分级未有重大更改，而发现或修复的漏洞数量变化明显，那就需要查看配置管理或漏洞管理过程了。

监测过程统计数据变化最常用的工具是控制图。在产品制造流程中，该图会显示一段时间内某种不合格的组件的数量。对于漏洞管理应用来说，我们关注的是重要主机上的漏洞的数量随时间变化的情况。该图有两条有关漏洞发现的“控制界限”线，称为上控制界限（UCL）和下控制界限（LCL），超出这些界限也就意味已超出了“统计控制”，即结果或性能上出现了意想不到的变化，这被称为特殊原因变异。如果该变化未超出上控制界限和下控制界限，我们称之为普通原因变异，它反映了任何过程的内在变动水平。通常，人们查看短期控制图，看到一个信号，然后做出反应，并试图修复该问题，而实际上这并不是一个问题，而只是过程正常变化的一部分，这种情况经常发生。只有当超出控制界限，即发生特殊原因变异时，才需要处理。

对于漏洞管理，普通原因变异可能会表现为每周三关键漏洞的数量迅速增加，这可能是由于供应商定期发布新漏洞，并将其纳入到漏洞管理系统的审计中引起的。另一方面，如果漏洞数量的激增突破了上控制界限，这说明程序可能存在某种缺陷，从而引发大量漏洞。我们如何能发现这些变化并确定其根本原因？以下是几个基本步骤：

1) 收集所需的基本数据，主要包括以下三种：

- ☐ 漏洞总数量及单位时间内的平均漏洞数量；
- ☐ 单个目标或群组的平均漏洞评分；
- ☐ 达到或未达到某种要求的目标的数量。

2) 绘制控制图。一旦确定了想要分析的数据类型, 就可以绘制控制图以寻找特殊原因。在调查研究的基础上所绘制的适当的图表能够反映出变化的原因。很多情况下能够从图表中明显地看出原因, 但有些时候则需要通过仔细调查过程的步骤。绘制控制图有一些基本步骤。

3) 选择要分析的群组 and 子群组。如果想分析各业务单元中漏洞评分低于某一水平的关键主机的数量, 那群组就是关键主机, 子群组就是业务单元, 数据元就是不满足标准的数量。如果想分析某个网络上每台主机的平均漏洞评分随时间变化的情况, 那么群组是所有主机, 子群组是网络, 数据元是主机平均评分。

4) 确定每月整个网络的平均漏洞评分。这很简单: 将过去 12 个月的网络评分进行平均即可。这个结果值为被称为 X 标杆, 它是穿过控制图的一条直线, 被称为中线。借助于中线, 用户一眼就能看出评分的月度变化。

5) 确定 UCL 和 LCL。控制界限是从统计角度定义的系统所能容忍的数据的高界限和低界限, 如果某个数据点达到或超出该界限, 我们就应该寻找原因。确定 UCL 和 LCL 比计算平均值要复杂一点, 需要按以下几个步骤进行:

第 1 步: 确定评分的月际差异。例如, 1 月评分减 2 月评分, 2 月评分减 3 月评分等。12 个月的时间段时会产生 11 个差值。为使每个差值为正, 可以使用绝对值。

第 2 步: 将这 11 个值进行平均, 所得值为 R 标杆。

第 3 步: LCL (下控制界限) 等于 $X \text{ 标杆} - (2.66 \times R \text{ 标杆})$ 。

第 4 步: UCL (上控制界限) 等于 $X \text{ 标杆} + (2.66 \times R \text{ 标杆})$ 。

图 7-10 显示的是东京的每月漏洞评分, 这些评分的平均值是 X 标杆值。最后一列是评分月度差异的绝对值, 这些数值平均后得到 R 标杆值。最后将这些值代入 UCL 和 LCL 计算等式即可得到边界值。

利用这些数据可以创建控制图, 如图 7-11 所示, 漏洞分析师根据该图表可以发现重要信息。很明显, X 标杆值在 11 月份上升到接近 UCL, 这意味着有超出常规的事件发生, 管理良好的程序一般不会产生这种变化, 应进行调查并确定原因。

调查结果对于避免出现整体风险的上升很有意义。以下是几种可能的造成风险上升的原因:

- 以前发现的高危漏洞未能成功修复: 当某一严重漏洞出现在大量系统中时, 就很容易发生这种情况。它对风险评分的整体影响很大, 与负责漏洞修复的 IT 管理人员进行进一步的沟通有助于确定原因。

- 供应商公布了极大量的关键漏洞：有时，软件商会将发现的漏洞集中起来一次性发布，很多风险管理人员认为这种作法十分有害，因为这就要求这些大量的漏洞要在发布之后而被攻击者利用之前的极短时间里得到修复。检查一下其他报告，看公司范围内每台主机的漏洞数量是否增长可以迅速确定此类原因。

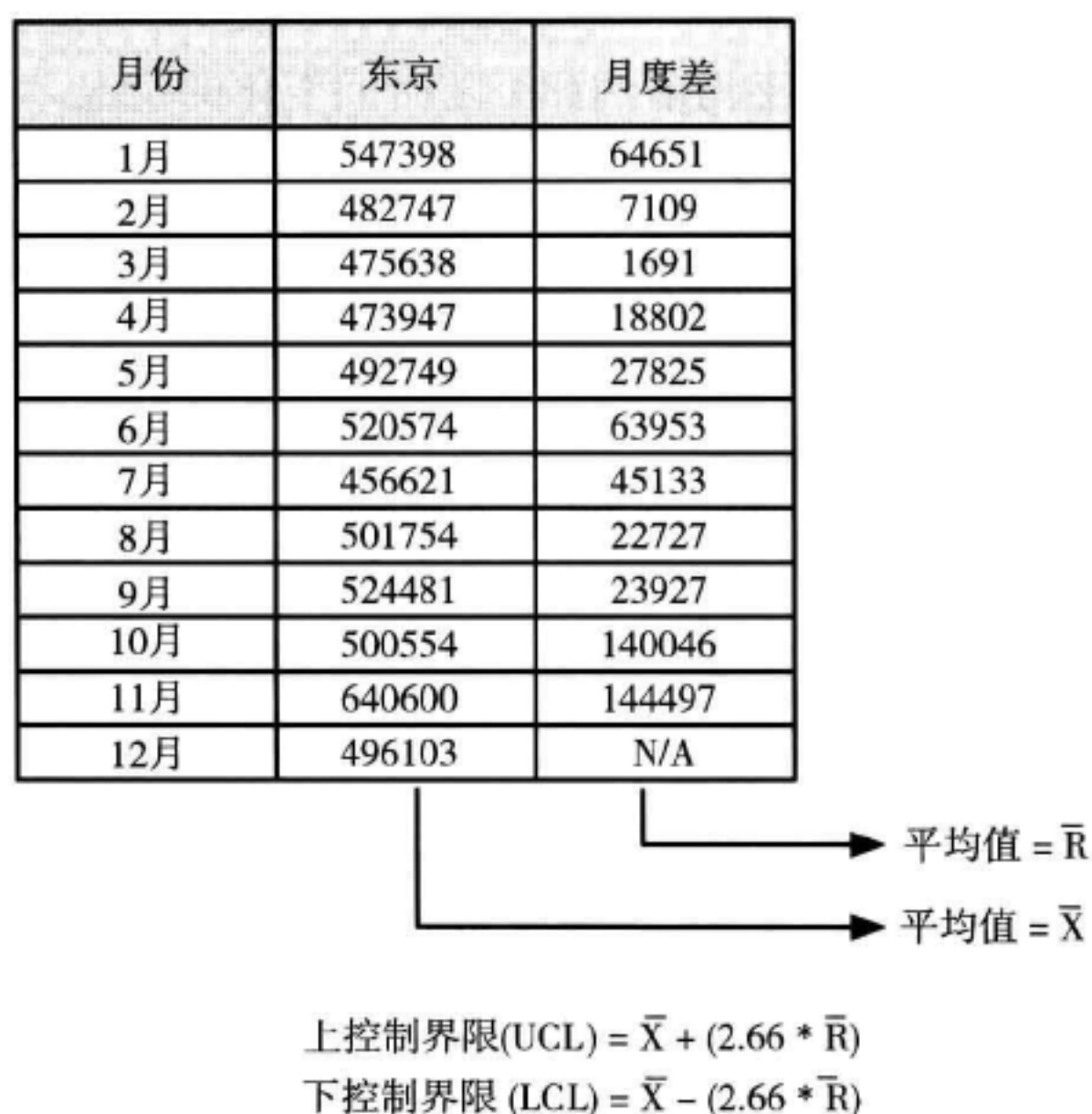


图 7-10 东京月度漏洞评分表

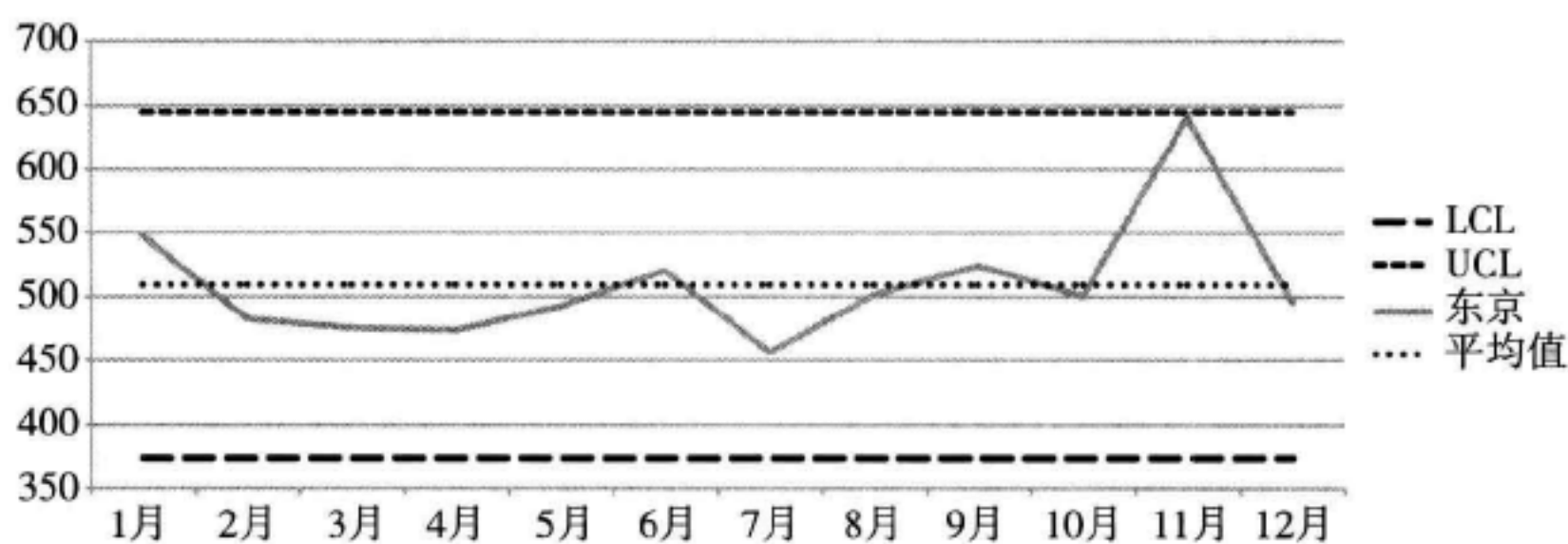


图 7-11 东京月度漏洞评分图

- 供应商发布了一批新核查项目，而有一些可能是相关的且严重的：这是导致评分异常的另一个常见原因。解决这一问题的方法是在评估其影响之前不执行这些新审查项，或简单地与风险管理人员就可能的结果进行沟通。

- 发布了专门针对某些系统（独特的环境）的漏洞：一个特定的网络或组织，在本例中为东京，可能会发布一些新的、严重的漏洞，这些漏洞可能只与日语字符集或在用的某种特殊软件有关，在公司中的其他区域都没有发现。因此，确定问题发生在某一区域还是整个公司是很必要的。

除了由于某些事件造成的漏洞评分超出控制界限的事件之外，以下几类情况也可能意味着过程出现了问题：

- 连续 7 个点在平均漏洞评分线一侧出现；
- 连续 7 个点显示漏洞评分的持续升高；
- 超过 90% 或低于 40% 的标绘点位于控制界限区域的中部 1/3 的范围内；
- 漏洞评分图凌乱，无规律。

7.7.3 介绍漏洞群：控制性能报告

漏洞并不总是明显和离散的。迄今为止的绝大多数漏洞管理实践以及本书中提到的漏洞管理实践都是围绕两个主题进行管理的：1) 前线（识别和修复）和 2) 引领（优化和管理过程）。然而，我认为还有另外一种等级的漏洞需要识别和修复，但现在尚无正规的方法。

这一等级的漏洞就是所谓的漏洞群。我们常会看到很多中低级别的漏洞由于其自身危害程度低或其所在的资产估值低而未被给予优先考虑。由于组织专注于处理更高危害的漏洞，这些低风险级别的漏洞通常得不到修复。而这些漏洞所造成的联合威胁却未被评估，尤其是当这些漏洞属于同一类别，用同一种技术或存在于特定的软件产品中时。此类漏洞危害程度不大的技术的普遍应用可能会导致巨大的危害。

例如，安全标准可能会规定所有的文件共享都需要访问认证，但一份专业的“漏洞群分析”（swarm analysis）报告显示一类被称为“访问控制”的漏洞评分很高，经过深入调查，安全管理人员发现原因是大量的文件共享允许所有人写访问。

如果安全意识方面的培训达不到预期的效果，那么可以考虑借助于技术手段。在本例中，Active Directory 中的组策略对象（GPO）能够防止继续创建这种开放式的文件共享，这是针对关键漏洞问题的一种无成本的解决方案。

有人可能要问，这些漏洞在以前讨论的诸如“漏洞报告”这类更细的报告中为什么没

有被发现呢？原因很简单，因为这些报告只是为辅助修复危害程度最高的漏洞而创建的，而像常用的台式机之类估值低且漏洞危害度也较低的资产则不具有优先性。在很多位置，某些应用程序的一些配置要求可能会被忽略，如果组织中存在大量的此类漏洞，一旦病毒利用这些漏洞入侵并进行传播，组织就可能遭到严重破坏。因此，我们可以通过分析每种控制类型的聚合结果以及某种漏洞的共同特征来识别这种危害性低但聚集起来却可以产生严重威胁的漏洞。

安全管理人员可以利用控制性能报告评估内部控制的有效性。安全控制手段在部署时需要遵从一定的标准进行配置，有的安全控制手段会被要求配置成已安装软件的一部分。对于组织缺少的合规项，控制性能报告会打破它们的类别。以下是一些合规类别：

- ☐ 认证与访问控制；
- ☐ 存储与备份；
- ☐ 恶意软件管理；
- ☐ 登录与不可否认性。

通过对某一目标网络或组织在以上几项上进行估值或漏洞评分，安全或风险管理员可以确定目前使用的控制手段是否有效，从而指导风险管理员调查当前程序，以提高控制使用效果或修改策略以取得相同的安全值。

关于此前的例子，我们来看一个关于认证和访问控制类别的报告。该报告会显示整个认证和访问控制类的漏洞评分，但该评分不应是此类中每个漏洞评分的简单相加，因为这些漏洞除去共有的分类特征之外可能并没有什么相关性，即它们之间不应是简单的累加关系，所以这样得出的结果可能会产生误导。

为简单起见，我们使用 CVE 数据作为识别漏洞的一种标识符。一些漏洞可能会跨越某些技术或硬件平台，但 CVE 已将这些全部考虑在内。

文件共享对任何人允许写访问是一种非常糟糕的做法，当然这也是一种漏洞。有些应用程序可能会要求这种配置，因此并不将其作为一种严重漏洞，但在有时就可能非常严重了。这种漏洞可能会存在于同时运行 Windows 和 UNIX 操作系统的多系统上。这种漏洞通常不予修复。为引起人们对这种广泛存在的漏洞的重视，我们将这种漏洞的数量进行了汇总，并表示为使用 SMB 协议的系统总量下的百分比。

如果我们有 100 个使用 SMB 的系统，其中 30 个系统含有这种漏洞，这种漏洞的“群评分”就是 30。由于不清楚对组织的总体影响，该数字并不足以将该问题的严重级别表达清楚。如果总体系统数量为 1000，使用该算法，就只有 3% 的主机是存在漏洞的，那受影响的系统的比例就低多了，危害程度也就低多了，但漏洞的实际情况可能比受影响系统的百分比所体现出来的更为严重，因为存在漏洞的这 30 台主机的估值可能比剩下的 970 台都要高。

为更准确地表述这种影响，我们必须将这些受影响资产或系统的价值相对于全部系统总价值的比例考虑进来，因为系统的价值并不是相同的，受影响系统的价值可能大于（或小于）未受影响的系统。我们可以利用以下公式：

$$S=1+\frac{\sum V_x}{\sum V_t} \times \alpha$$

其中， V_x 是漏洞系统的价值， V_t 是网络或组织单元中所有系统的价值， α 指单个漏洞的评分。

这个等式用受影响资产的价值总和占目标区域内全部主机的总资产值的比例作为受影响资产的价值取值，并将该值与某一具体漏洞的评分相乘。因此，如果某一漏洞的评分 10 分，这是一个较低的分值，受影响主机的价值占全体主机总价值的比值为 52%，则最终的结果就是 $1 + 0.52 \times 10 = 15.2$ 。

该数值实在太小，即便在一份常规的漏洞报告中也无法引起人们的注意。这一类漏洞最好用一份独立报告或是漏洞总结报告中的一个独立章节来描述会更好。这样一份报告可能会类似于图 7-12 中为 XYZ 公司所写的那样。

该报告中有五个漏洞，单独评估时其危害性都相对较低，但其联合影响却很大。弱 SNMP 社区字符串就是很好的例子。在打印机上经常使用这种字符串，一般也没有引起大的安全问题，不过这取决于打印机的制造商及其在网络中所处的位置。因此，这些漏洞很容易被忽视，但其数量可能已经很多。有时这些漏洞会完全被忽略，因为在打印机上到处都是这种漏洞。结果，当同样的漏洞出现在服务器上时，也没有引起人们的注意。而上面的一份报告则能体现出服务器相比其他设备（如打印机）的更高价值，并能够对这些漏洞整体所造成影响的严重性进行评估。

XYZ 公司		
漏洞群报告		
网络：旧金山		
2009 年 12 月 15 日		
第 1 页，共 1 页		
业务单元：Widget 设计	阈值	
	量：5	评分：2
漏洞	量	评分
0001——SMB 共享允许所有人写访问	30	15.2
0045——SQL 服务器弱 SA 密码	6	18.0
0133——SMB 零会话列举	82	14
0227——弱 SNMP 社区字符串	18	9.2
1472——SMB 共享允许所有人执行访问	41	6.1
总漏洞群：5		

图 7-12 XYZ 公司通用漏洞报告

7.8 合规性报告

漏洞审计监测的另外一个重要项目是策略合规性。漏洞管理程序正在试图添加这种功能，以使组织不仅能够实现安全目标，而且可以监测是否与策略和标准相符合。

一般来说，合规性报告要求有与目标系统上的配置项相关的策略说明。系统会审计这些配置项，并记录与策略一致的数量，由此而生成的报告主要有三个作用：

- ☐ 显示每台主机、网络或组织单元与策略的合规性程度；
- ☐ 确认需要更正的配置项；
- ☐ 显示系统和组织合规性表现的历史记录。

7.8.1 系统合规性报告

该报告可以发现与某一策略不合规的系统，从而通过采取一些补救措施使之合规。该报告与详细的漏洞报告非常相似，只是不需要那么详细，以下是该报告的一些关键要素：

- ☐ 策略报告名称、日期和目标网络；
- ☐ 主机名及 IP 地址；
- ☐ 系统有多少条策略和标准不相符；

❑ 系统的拥有者或管理员。

合规性管理员或安全管理员利用该报告可以迅速发现并集中精力修复那些合规性最差的系统和网络，如图 7-13 所示。

<div> <div>XYZ 公司</div> <div>策略合规性报告</div> <div>网络：芝加哥主网</div> </div> <div>2009 年 9 月 10 日</div> <div>第 1 页，共 3 页</div>		
业务单元：部件分销商		
<div> <div>合规目标数：2</div> <div>不合规目标数：3</div> </div>		
系统	不合规	合规
SERV01.widget.net (10.1.2.30)	2	4
SERV02.widget.net (10.1.2.31)	1	5
NET-SW.widget.net (10.1.2.2)	0	6
EDG-RT2.widget.net (10.1.2.1)	3	3
EDG-RT3.widget.net (10.1.2.3)	0	6
总计	6	24

图 7-13 系统合规性报告样页

此外，还需要增加一个更为详细的报告，或一个与该报告结合来显示各个系统的详细状态的报告，这样一份报告能使各系统的管理员更好地修复各配置项（CI）。例如，假设一台主机需要安装防病毒软件，但最好的做法是取消所有人对文件共享的写访问权限，这种情况下就应首先修复防病毒软件项，以下是所需的报告内容列表：

- ❑ 策略报告的名称、日期和网络；
- ❑ 主机名和 IP 地址；
- ❑ 策略项编号及描述；
- ❑ 合规性状态。

图 7-14 是摘自系统合规性报告中的一张样页，这是一份肯定的推断（positive assertion）报告，即该报告不仅显示出哪些不合规，还显示出哪些合规，这样对于目标状态就没有任何疑问了，而这也是萨班斯－奥克斯利法案对审计的普遍要求。

XYZ 公司

策略合规性报告

网络：芝加哥主网

2009 年 9 月 10 日

第 56 页，共 90 页

系统：Cust-Web01.XYZCORP.COM (10.2.3.4)

合规：4

不合规：1

策略项	合规性状态
4.1 使用如 SSL 的强加密安全协议	合规
5.1.1 确保防病毒软件能够检测到、删除恶意软件，包括间谍软件和广告软件。	合规
6.1 保证所有系统组件和软件都安装了供应商提供的最新安全补丁，在发布后的一个月 个月内安装了其他的相关安全补丁。 • MS06-11 补丁未安装 • MS07-15 补丁未安装	不合规
8.5.10 要求密码长度至少为 7 个字符	合规
8.5.11 使用包含数字和字母的密码	合规

图 7-14 系统合规性报告样页

7.8.2 合规性执行总结

执行总结报告评估组织现状与特定 IT 管理目标的合规程度。例如，IT 管理员可能会设定目标，要在年末使 95% 的系统达到 PCI（支付卡业界）标准。可将从主机收集到的数据和定期生成的报告与这些标准进行对比来生成合规性执行总结报告，规范的报告至少应包含以下内容：

- ❑ 不合规的主机的数量及比例；
- ❑ 合规的主机数量及比例；
- ❑ 每台主机上缺失的合规项的平均数目；
- ❑ 每个网络或组织分组中合规和不合规的主机的数量及比例。

IT 管理员据此可以看出哪些群组在合规性方面做得最好、哪些最差，这有助于通过资源的分配提高最差的群组与标准的合规性程度。

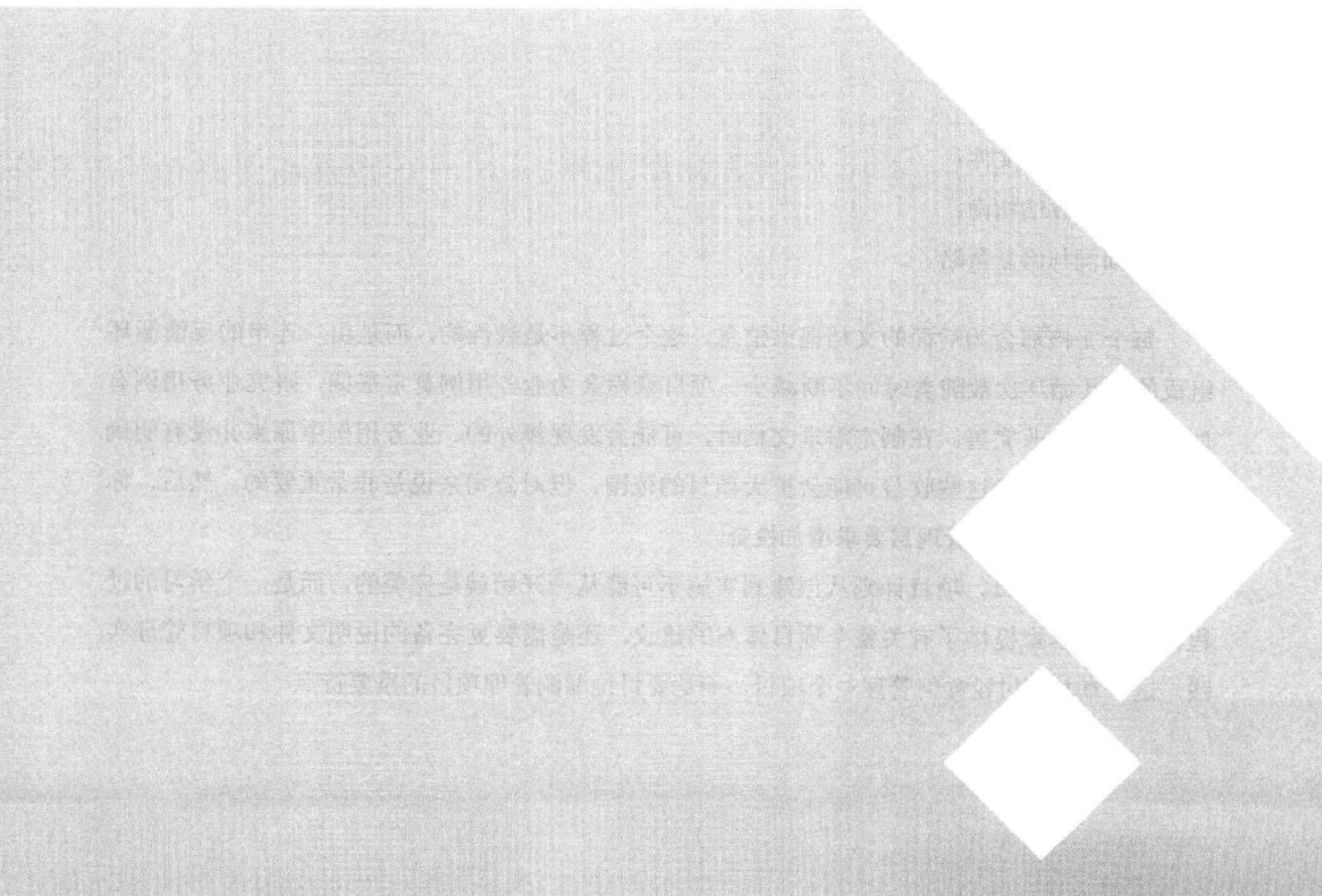
7.9 小结

从本章可以清楚地看出，这些容易理解的和安排有序的信息对管理漏洞管理非常重要，漏洞评估产品的本质就是信息收集系统，很多人未能意识到这一点，认为这些产品不会提供多少有用信息。这些系统是由技术人员规划和实施的，目标配置细节是他们的弱项，我认为这是导致系统的报告功能偏低的原因。

只收集大量详细信息但却不进行数据挖掘是没有太大意义的，这些信息所能提供的价值远远超出技术本身，而项目管理员应当是负责挖掘其中价值从而给企业带来收益的人。回顾一下本章的内容，就会更加意识到生成的这些报告能够使过程更加完善、更加高效，从而带来更优质、更高效、更可靠的修复，最终达到提高安全性的目的。既能识别单个目标的漏洞，也能识别一组或一群目标上的漏洞，这种能力能够改变优先级，为公司真正节省成本。具体来说，需要通过全球性配置更改来修复一个小漏洞已经不是一件令人兴奋的事情了，但如果需要重复不断地从同一目标上清除相同的蠕虫来修复同一个漏洞，那就更加令人沮丧了。

第8章

规 划



8.1 介绍

到目前为止，我们已经讨论了漏洞管理的各个方面及其与漏洞管理过程的整体关系，同时也描述了漏洞管理在公司风险管理和安全态势维护方面发挥的其他作用，还讨论了实施漏洞管理的各阶段工作，如制定策略、规划流程和提出要求。现在需要做的是制定工程规划，检查清单及实施策略以使该程序能够运行起来。

本章将提供一些建议，引导你从制定检查表、工程计划和实施策略等方面一步步了解如何在一个大型的、全球性公司里开发一个完整的漏洞管理项目。主要包括以下内容，在实际操作中可能需要根据项目需求加以选择和调整：

- ☐ 漏洞管理项目章程；
- ☐ 业务用例；
- ☐ 需求文档；
- ☐ 安全架构提案；
- ☐ 项目计划；
- ☐ 征询方案；
- ☐ 实施计划；
- ☐ 运作流程文件；
- ☐ 资产评估指南；
- ☐ 漏洞与修复策略。

每个文档都会为后面的文档提供信息，这个过程不是线性的，而是由一连串的反馈循环组成的，其循环次数随着时间不断减少。项目章程会为业务用例奠定基调，研究业务用例有助于章程的修改完善。在制定需求文档时，可能会发现额外的、业务用例中原来并没有明确说明的收益，明确这些收益可能会扩大项目的范围，但对公司来说是非常重要的。然后，你可以调整业务，并向管理层要求增加投资。

对于任何项目，项目计划从创建到实施不可能从一开始就是完美的，而是一个学习的过程，即使在本章提供了有关整个项目体系的建议，还是需要更完备的说明文件和项目管理实践。这一章并不讨论如何管理一个项目，而是要讨论漏洞管理项目的重要特点。

8.2 章程制定

漏洞管理项目章程非常重要，它设定了项目目标和业务准则，居然前者的重要性不言而喻，但其实后者更为重要。要使项目成功，高层管理者必须理解和认同该项目确实具有真正的业务价值。

8.2.1 介绍：业务价值

业务价值必须在章程介绍的最开始部分被清晰地表述。它回答了一个关键问题：“我为什么要在这上面花钱？”其价值体现在很多方面，例如：

- 举出实际案例，说明哪些东西可能会遭到破坏，以及为什么一定要对其进行修复。这些内容可能出自其他公司发现的一些漏洞列表，并且这些漏洞对公司造成了巨大的破坏。同样重要的是，还应证明漏洞破坏为公司增加的成本和人工劳动大于漏洞管理项目本身，这在 Philip B. Crosby 的书《Quality Is Free: The Art of Making Quality Certain》(McGraw-Hill, 1979) 中有详细的探讨。
- 在业务背景下，举例和描述会对以下各项造成威胁的事件：
- 业务模式的可行性。
- 收入来源或增长率的连贯性。
- 赢利，即扣除事件成本后的净收入。
- 通过将耗时耗力的工作常规化，漏洞管理能提高运营效率。
- 尽可能地量化收益，实施项目造成成本支出是很正常的，但如果高层管理者看不到项目可以带来的收益，他们很可能就不会通过项目提案。所以应当在项目章程中写明收益目标，即将公司的关注重点包含进来。

8.2.2 目的和目标

章程中应包含项目目的和目标的描述，并且应该是两个独立的部分。一定要记住目的是指希望从一系列的行为中得到的结果或收益；而目标是指希望实现的具体事项，一般是可量化的。

典型的目的表述如下：“准确地识别和报告漏洞及其修复状态。”这告诉我们想实现什么，但却无法真正量化，要达到这一目的可能需要一个或更多的行为，甚至目标。

以下是一个更为具体的关于目标的陈述：“修复接入 XYZ 网络的非关键目标机上至少 75% 的重要漏洞。”这项陈述非常清楚，完全合格，利用这种目标可以对项目成果进行衡量。

8.2.3 范围

范围指项目在以下方面所达到的程度：

- ☐ 所要求的资源。
- ☐ 受影响的网络与主机。
- ☐ 受物理（安装的硬件）和逻辑（被扫描与监测的）影响的位置。
- ☐ 创建和修改的程序。
- ☐ 更改的系统（主要是为了适应修复活动而进行的更改）。
- ☐ 购买的产品种类。
- ☐ 要求的服务种类（如系统集成）。

8.2.4 假设

起草这一部分时要非常谨慎。这通常是一系列假设，都用短句进行陈述。当不确定一些具体细节时，这些表述可能会比较笼统。如果你知道应购买哪种漏洞管理系统，或由谁来操作该系统，应该将这些情况都详细地列举出来。也许你现在还不知道这些内容，那么只能在陈述时假设会有人来全职或兼职地管理该系统。以下是一些其他可能的假设：

- 1) 一般表述：一名全职管理员将负责系统的运行和监测；具体表述：目前在伦敦的一名高级安全管理员将兼职负责监测和运行系统。
- 2) 一般表述：漏洞扫描应定期进行，以在漏洞被发现的 30 天内完成识别和修复；具体表述：每周进行漏洞扫描，关键漏洞立即提交修复。
- 3) 一般表述：此外，需要对变更管理系统进行一些小的开发，以跟踪修复行动；具体表述：需要两个开发人员花费大约 100 个工时对 XYZ 变更系统进行更改。

我们注意到，在接受高级管理人员审查时，陈述越笼统，出现的问题可能就越少。但是，使用明确、具体的陈述能更好地阐明预期和预算。哪些问题应该陈述的笼统一些，哪些应该陈述的具体一些，取决于管理层的态度和文化因素。

8.3 业务用例

在公司中如果有必要，业务用例应着重关注风险管理实践，而非量化各种概率。如果你提到漏洞被利用的概率，读者会理所当然地迅速质疑该数字，这是一个不可能胜出的争论，因为没有针对业务模型、组织或行业的可靠数据支持，所以数据很难不被质疑。创建业务用例时可利用以下技巧：

1) 重点写影响：在安全领域，损失造成的影响都有完好的存档。这些损失难以核算，但比写一个概率效果要好得多。在计算机安全出版物、FBI、计算机安全协会的计算机犯罪与安全调查等处可以找到大量此类的损失记录资料。高级管理人员很难开口说从这么多机构的广泛调查数据中得出的每个安全事件的平均损失数额是错误的或不合理的。

2) 无形风险：公司声誉是重要的资产，是由市场和销售人员经过多年的潜心经营建立起来的，具有信誉价值和营销价值。公司在业务和客户中的形象一旦受损，很难恢复。下面是一些增进业务用例亲和力的技巧：

- ☐ 部分审查业务用例的人员应具有销售和营销经历，只有这样他们才能理解漏洞管理的价值正在经受风险。
- ☐ 审查者应是高层人员，可以影响决策过程和预算。
- ☐ 包括其他公司漏洞被利用从而造成损失的例证（如 PetCo.com, TJ Maxx）。
- ☐ 要证明系统和数据存在损坏或损耗是很严重的，即使这些缺陷未被恶意使用，仅公众知晓这些情况已经很危险了。
- ☐ 在某些州，对客户数据泄露的惩罚性赔偿是很高的，如果你的公司也涉及此类业务，要高度重视。

8.4 需求文档

做过正规 IT 项目的人都知道需求文档的重要性，该文档会使公司的目标、流程及系统为目标方案的功能服务。这些需求可能会超出系统规格，也可能将必须开发和更改的相关程序包括在内。以下是需求文档的一些主要内容：

1) 功能需求：详细描述软件和硬件作用。

功能需求可能包括以下内容：

- 漏洞检测类型；
- 可调参数；
- 漏洞报告要素；
- 报告类型（管理、执行、风险总结、修复等）；
- 系统接口（变更和事件管理系统、标记）；
- 网络与群组定义；
- 主动扫描认证能力；
- 外部数据库需求；
- 备份和恢复能力；
- 审计日志特点；
- 安全与隐私（加密与高可用性）。

2) 流程要求：

- 应采取哪些步骤；
- 每个步骤由谁负责执行；
- 每个步骤的输入与输出；
- 与每个步骤都交互的现有过程。

3) 图和表：

- 实体模型报告，显示包含哪些要素、分组情况、关键要素的排序。不要太具限制性，但要抓住最关键要素。
- 基本流程图：流程图表化，将现有各内部流程包含进去，以便检验集成各种工具和步骤的能力。
- 总体设计：该图将显示如何以及在什么地方放置各组件，这有助于阻止创建和接受与策略不符的系统。

这些要求中有很多来自选择技术流程的细节，见第 5 章。

8.5 安全架构建议

虽然已经讨论了技术和漏洞管理架构，但仍有一些关键的组件必须包含在安全架构建议中以获得各关键团队的关注和批准。其中，最重要的团队包括网络运行、网络工程、安全、合规、系统管理及系统工程。这些团队都会受到系统架构不同程度的影响。

在整体的项目规划中，安全架构方案的制订应当在调查了各种可用技术，并列出了想要咨询建议的供应商名单之后进行。通过与各个供应商的基本问答交流，可以掌握以下内容：

- ❑ 漏洞系统的种类（主动、被动、代理或联合）；
- ❑ 网络需求；
- ❑ 主机扫描需求（如果可以）；
- ❑ 主机兼容性（支持的操作系统）；
- ❑ 负面作用（网络或主机）；
- ❑ 主动扫描系统认证、代理安装、特殊网络设置的要求；
- ❑ 部署方法（特别是有代理的：如何安装、分布及配置）。

在对即将部署的系统有了较好的了解后，还需知道基本架构需求，这些需求有助于决定如何将漏洞管理系统与现有各组件集成。稍后，其他团队在审查协议的架构时还会提出其他的需求。例如，经过审慎地考量，系统组认为部署代理最好的方式是通过补丁管理系统，而不是使用 Windows 服务器升级服务（WSUS），如果是这样，新需求就产生了，供应商需要满足该需求。

以下是制定架构方案的基本步骤总结：

- 1) 确认安全、基本网络及合规方面的需求。
- 2) 审查并选择几个供应商。
- 3) 起草架构方案，包括各种需要的图表，技术人员一般喜欢图表方式。
- 4) 重新审查所有关键技术及业务团队。
- 5) 为新需求收集反馈。
- 6) 改进该架构，直到所有相关团队都同意。

8.6 RFP

作为技术选择流程的一部分，应制定产品需求，这些需求应在征询方案（RFP）中高度清晰表述。至少，应先确定系统的类型（主动、被动、代理或联合式）。RFP 应包括以下部分：

- 1) 目标。
- 2) 操作环境。
- 3) 现有系统与流程。
- 4) 位置与规模。
- 5) 网络类型与带宽。

6) 用户组及角色（描述每个组的作用和所需数据，提供具体用例对于供应商更好地理解系统功能对各用户组角色的适用情况非常重要）：

- ☐ 高级经理——系统性能；
- ☐ 安全工程师——漏洞分析；
- ☐ 系统工程师——修复；
- ☐ 风险管理员——各位置、网络、组织总体风险。

- 7) 投标流程及规则。
- 8) 沟通（主要联系人及联系方式）。

8.7 实施计划

一旦选定了产品，确定了流程，就需要制定实施计划以保证按时按要求交货。该计划非常重要，因为网络中的很多漏洞评估技术都是干扰式的。例如，如果使用主动扫描设备，那么之前所讨论的一些技术就需要进行调整。如果选用代理，则需要使用各种基线桌面图像对其进行测试。以下是实施计划中需注意的更为重要的事项。

- 1) 现场准备清单——实施现场所需的各种必需品，主要包括以下内容：

- ☐ 存储架空间（所需的存储架空间）；
- ☐ 电源（120 V/240V、50Hz/60Hz、接头类型）；
- ☐ 网络连接（带宽、物理连接类型、寻址）；

- ☐ 安全（防火墙或访问控制列表设置、入侵防御系统免责条款、目标主机安全设置）；
- ☐ 网络设备（明确哪些网络设备模型不适合配备主动性扫描设备，这些网络设备模型可以是制造商明确指出对审计有不利反应的任何固件或硬件）；
- ☐ 对现场工作人员的最低技能要求；
- ☐ 每个场地设置唯一的联络点，统一进行实施和升级。

2) 实施安排——对每个施工现场，应记录以下信息：

- ☐ 计划实施日期；
- ☐ 预计的安装时间；
- ☐ 单点联络的人员、电话和电子邮件地址；
- ☐ 需要哪些专门的指定的资源；
- ☐ 首次审计的目标日期。

3) 安装流程——用图示（确切行动）详细说明安装设备的操作流程；不要对安装人员的知识和技能做假设，因为不同的地方会有差异。这些过程应当说明需要在安装过程中的适当时机用来与业务和漏洞管理系统管理员进行通信的内容。

4) 项目实施介绍——处于重要位置的参与者解说项目细节是有必要的，因为大多数将直接或间接实施及使用这些技术的人员都对该技术知之甚少。项目实施介绍内容应包括该技术如何发挥作用，对工作流的影响，每个参与角色的作用，及特定现场的时间安排等；还包括解决有关收集数据及工作人员隐私等安全相关问题。有可能受影响的工作群组包括：

- ☐ 网络工程师；
- ☐ 本地及全球帮助中心；
- ☐ 系统所用者及管理员；
- ☐ 其他系统支持人员；
- ☐ IT 安全工作人员；
- ☐ 本地 IT 管理员。

8.8 操作流程文档

一旦确定了漏洞管理活动的流程，必须以两种可理解的方式将其记录存档。流程管理员需理解各个步骤，及其通过群组界面与其他流程的关联关系。具体来说，就是漏洞管理流程文档中需包含升级路径，输入数据及输出数据。

该文档的另一种格式类似于操作说明书，它以清单的形式解释各个操作步骤，介绍了操作者在可能遇到的各种情况下的做法。如果现有流程已在运行当中，则要么让新流程采用相同的形式，要么升级当前流程，将新步骤包含进去。

例如，定义一个新网络的流程可能如下所示：

漏洞系统新网络过程

用例：新网络

描述：为适应一个新的用户 IP 地址段，在组织中部署一个新网络

所需数据：IP 地址范围、CIDR 块、网络上目标的风险水平，商定的审计窗口（频率、开始时间、持续时间），允许占用的带宽，排除的目标 IP

操作：

在漏洞管理系统中定义网络规范

限定被排除的 IP 地址

规定审计参数（时间安排表、带宽）

创建防火墙和 IPS 所防御内容的安全变更要求

完成上述安全变更时运行测试审计

评估测试审计结果，当需要时可调整参数

与安全管理员及目标所有者沟通审计参数变更情况

激活定期审计程序

流程结束

并非所有流程都这样简单，有些需要对当前流程进行更改。以下是一些可能导致更改的因素：

- ❑ 毫无疑问，新流程与已经存在的流程之间所需的通信对于变更管理项的升级和讨论非常必要。
- ❑ 为当前工作人员增加了责任。任何新系统几乎都会给某些人增加责任，但由于避免了安全漏洞，免除了相关的应对操作，最后得到的将会是更好的服务和工作人员长期成本的降低。
- ❑ 向现有系统中输入新的漏洞管理数据可能会要求不同的处理流程。例如，如果事件管理与漏洞管理系统间无自动接口，就需要手工创建事件。
- ❑ 如果新漏洞管理系统与现有系统存在不兼容情况，则可能需要强制添加或更改已有的人工操作步骤。

8.9 资产估价指南

为保持估价的一致性，任何负责为资产进行估价和确定风险参数的人员都应有一份评估标准指南。一些组织选择使用带公式和规则的电子表格，以简化评估过程。这种表格实际上是一系列封闭式问题，其答案诸如是或否，或 1 ~ 5 标注的等级。

在采用为了简便且极少有管理员了解相互作用的风险的评估方法的情况下，评估会变得更加快速和简单。不需要多少分析就可以为每项资产确定风险影响等级。可以直接将处理客户信息、财务数据、知识产权的系统确定为高等级，而将其他系统确定为低等级。以下是资产估价表中可能包括的一些问题：

- ☐ 该资产是否传输、存储或处理个人身份信息（PII）？
- ☐ 如果该资产无法运行，或由于违反服务等级协议（SLA）而中断运行，会对公司造成何种程度的损失：低、中、高？
- ☐ 如果该资产存储或处理的数据崩溃，会对业务造成何种程度的破坏：低、中、高？（注意：下面的等级划分取决于公司规模和风险承受能力。）

低 = 少于 10 000 美元；

中 = 10 001 ~ 100 000 美元；

高 = 100 001 美元或更高。

这种方法依赖于安全控制措施的实施方式、成本的变动以及控制措施的复杂性。例如，有些公司对所有系统施加同样严格的安全控制措施，并对关键系统增加额外的安全控制。

这种方法在大型公司中不一定适用，因为大公司的安全支出会很高，实施的时间跨度会很大。

8.10 漏洞管理策略

任何系统和程序在开发和部署前都要考虑开发策略，策略的制定应基于相关资产的估价及对当前业务运行环境的风险分析。例如，修复操作的财力投入和优先级取决于资产的业务价值。策略高于标准，策略在一个较高的层次上定义解决关键问题的方法，标准则依据策略内容提供更为具体详尽的信息。以下是漏洞管理策略中应包含的一些项目：

- 要求所有资产均被分级或估值。应制定一个明确的总体分级标准，分级机制由支持标准确定。
- 指定负责资产分级的人。这些人可以被称作“资产所有者”或“主要利益相关者”，甚至“系统管理员”（system manager）。
- 指定修复流程。必须合理地制定修复操作的优先级顺序和预算，这项工作是前面提到的资产所有者或类似职位人员的责任。
- 确定一个主要负责人员，负责处理与修复或分级活动有关的争端及问题。该人员应熟悉业务需求、策略及有问题的系统，可以是高级风险管理员或安全管理员。

由于修复时间及审计频率这类问题变动性太大，难以写入策略中，因此通常写在标准里。此类问题相关标准的制定原则是：漏洞的检测和修复是依据当前的风险等级来确定的。

8.11 部署策略

漏洞管理系统的部署复杂度不是很高。但是，在不同的组织机构中，遇到的挑战也会有所不同。

8.11.1 基本策略

部署漏洞管理系统的基本方法，或者说部署任何系统的基本方法——要在较小的可控的范围内开始部署，然后逐步扩大范围。随着实施部署的员工的知识经验的积累和信心的增长，他们能够逐渐应对更大用户群的系统部署。一开始就进行庞大的、复杂的部署操作会挫伤员工的积极性，而首先部署复杂程度低的网络则能迅速看到收益，并极大地鼓舞员工士气。以下是一些有关策略的建议：

- 小处入手。最初可以是本地的几百个用户，这样可以立即看到收益。
- 各个测试，全网铺开。如果不进行一定程度的测试，则无法估计网络设备和主机会受到的影响，没有任何两个网络是完全相同的，但有时会非常相似，这也能够尽可能减少测试工作量。
- 监测每次部署所覆盖的目标及未解决问题的数量，如果后续的部署中出现了更多或同样的问题，应设法先解决，然后再开展后续部署，而不要任其发展。

- 指定负责处理服务器和 DMZ 部署的资源，这些系统一般来说更为重要，为防止服务中断，应有合适的资源负责测试和监控。
- 大型组织应避免线性部署，因为这样工期可能会很长。这可以通过细心规划及高级管理层设定截止日期来尽量缩短部署时间。

8.11.2 基于风险的策略

基于风险的策略会稍微困难一些，但也更令人满意。其思路很简单，就是先在公司最关键的网络中部署漏洞管理系统。这样主机的数量是有限的，审计范围的增长速度也迅速地下降了。以下是利用这种策略最初安装时的一些选择：

- 面向公众的 DMZ 一般都是很重要的，因为它们是公司品牌的电子门面。从一开始就确保将漏洞数量保持在最低水平会为公司的信誉带来益处。
- 任何公司里执行重要任务的系统都很容易掌握：如果没有这些系统，公司将无法创造收入，收集应收账款，支付工资及进行关键操作。降低这些系统的风险就是漏洞管理部署的最初胜利。
- 后端数据库系统一般保存着上游系统所需的数据。要优先保证将数据库系统的漏洞数量维持在最低水平，因为数据库关系着价值。
- 类似于执行重要任务的系统，沟通系统是业务运行的心脏和动脉。应付账款系统终端停止运行一两天公司还可以接受，但如果有一天无法发送电子邮件、打电话或发送即时信息，业务运行很快就会停止下来，拥有多处办公地点的大公司更是如此。

从最大的最关键的数据中心开始着手安装设备和漏洞管理服务是最好的，那里有大量的关键系统及工程人员、系统管理员及支持性服务，这使得部署、监测及对问题的响应会更高效，多数问题的识别也会更快，将来在其他数据中心及办公室的部署也会更为顺利。

要在一批服务器上部署主动网络扫描，可以先从一台提供备份的单独的服务器开始，然后评估其结果。Active Directory 环境下的域名控制器是一个理想的开始地点。通过最初对一台域名控制器进行审计并评估其结果，可以发现存在哪些不良的影响，此后即可在更大范围的域名控制器上展开部署。当从目录管理中获得了足够的经验和充分的信息后，就可以在其他基础设施系统上展开部署。

几周以后，由于漏洞管理技术已被当成基础设施的一部分，向其他重要业务系统的部署

转移就容易得多了。比起一个使用新系统、没有业绩的、独立的安全或审计小组，公司利益相关者会更倾向接受他们所信赖的基础设施合作伙伴的建议。

最后，涉及的是那些重要但不显要的系统。台式机和便携式计算机之于漏洞管理就像个体客户（与商业客户相比）之于电话公司。单独的工作站及其用户和用户组是高维护、低风险的。工作站对审计的技术要求与关键服务器一样，但是面临的直接风险却小得多，这并不是说台式电脑就不需要审计。只要有一台电脑存在关键漏洞，攻击者就会利用它进入到其他更关键的系统。

而如果关键系统成功通过了审计，那么桌面系统通过审计也将比较容易。此外，如果扫描工作站时未使用代理，那就不需过于担心中断干扰。以下是需要向台式机系统管理员强调的几点：

- ❑ 不要在目标机上安装软件。
- ❑ 通过关键系统的使用经验说明操作的影响非常小。
- ❑ 所需的只是让认证系统管理员提供审计凭证，而这些管理员可能已经在使用漏洞管理系统。
- ❑ 你可以迅速对一些样本系统进行审计，然后逐一部署 VLAN。支持人员可以在部署时评估其影响。
- ❑ 漏洞管理技术已经出现了一段时间，其稳定性和可靠性是众所周知的。

8.11.3 改进的时间表

任何部署策略都必须能够灵活应对前一阶段中发生的变化，及时进行调整。虽然初期已经制定了一个一致统一的时间表，但实施过程中出现意想不到的问题也是很正常的。

例如，最初的部署可能要使用在 Active Directory 系统上生成的凭证，该凭证很快就会被复制到全球所有的域名控制器上。但这一操作开始后，可能发现过去两年中被兼并的其他公司保留着自己的目录服务及相关的认证系统，而这些管理程序从来没成为该项目计划的一部分。而使问题更为复杂的是被兼并公司与母公司在好几处办公地点使用相同的物理基础设施。项目管理员将不得不为这些兼并的公司分别制定部署计划，而这些计划的制订流程大部分是相同的，换句话说，项目管理员不得不重复这些工作。项目管理员也许会认为被兼并公司的系统应在报告中予以单独考虑，这样修复操作可以由不同的管理员来执行。由于项目管理员

和漏洞管理开发团队未考虑到这种情况，导致漏洞管理系统和所有其他相关程序都需要重新评估，以找到解决方案。

8.12 部署标准与进展报告

由于项目的投资方要知道他们在为什么付费，因此需要监测部署是否成功，以及进展情况，并与投资方进行沟通，这对任何的项目和操作都是适用的。首先应该定义成功的标准，以下标准应被置于项目章程中。

- 常规审计的系统的比例：制定这一标准依赖于知道组织内系统的总量或估计的总量。
- 每个网络上未被审计的目标的估计比例：这一标准的制定有一定技术性和难度，它可能是未执行全面审计或认证审计的目标的比例，也可能是网络上被审计的系统数量除以全部已知目标数量。这一比例有助于风险管理员确定有多少风险他还不知道，这相当于统计学中的误差边际，有助于确定整体组织风险评估的可靠性，也可以用来衡量整体的项目进度。
- 评估完成章程规定系统数量所需的时间：这是项目管理问题，该时间由网络管理员和漏洞管理员共同估计。
- 最初部署后每周新审计的系统数量：来自漏洞系统每周的数据，能够反映项目进展趋势，有助于确定项目是否取得新进展。
- 每周遇到的问题和解决情况列表：用于加强沟通、促进问题解决的项目管理办法。
- 接受常规审计的系统中已开始进行修复活动的系统比例：指变更管理系统项的数量除以漏洞系统创建的事件数量。
- 与漏洞管理相关的修复活动的系统比例：是指与漏洞管理相关的变更管理系统项的数量除以组织中评估系统的总数量。

8.13 小结

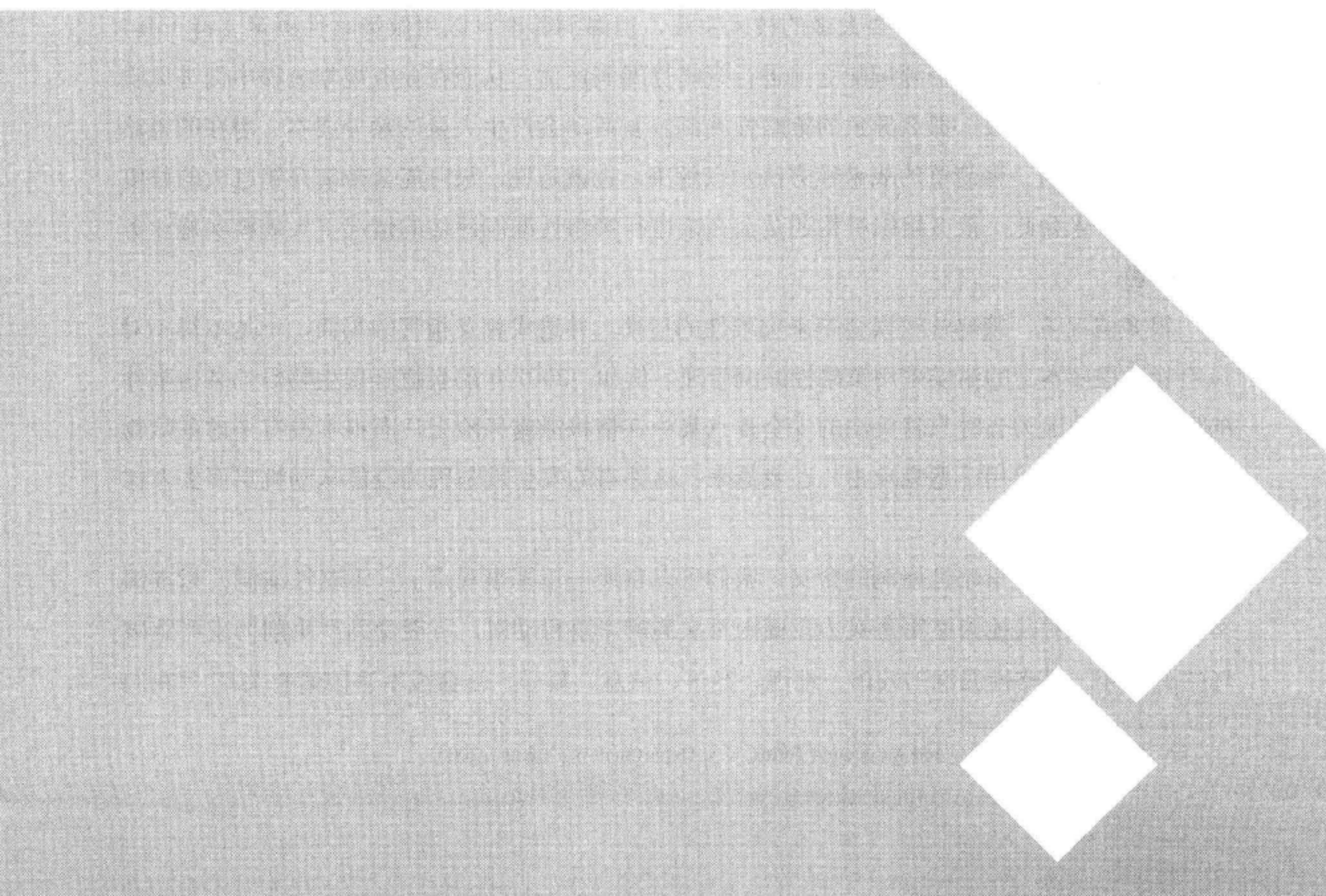
规划的好处无可争议，规划漏洞管理项目重要的是要遵从公司内已确定的结构流程。由于规划活动是一项结合了风险管理和基础设施的项目，利用本章讨论的方法进行充分的沟通

交流是很重要的。创建业务用例和需求的最初几步不仅能促进各合作组织就各阶段目标进行沟通协商，还能培养各组织的参与感和融入感，有助于项目的成功。审计系统和流程容易在人们之间造成交流壁垒，而各组织之间的融洽和谐有利于消除这种隔阂。审计或监测工具听起来总是让人很紧张，因此需要增大其开放性，增进人们对它的了解，改变这种现状。

在实施过程的后面几个阶段，各种衡量标准的显示数据让参与者们感到很满意，不仅让他们确实感受到项目的进展，也让他们感到他们的投入没有浪费。这样一来，项目经理就有可能将一个原本面临许多阻力的项目转化为一个参与方都支持的项目。漏洞管理在赢取参与方认同与支持的过程中起着特殊的推动作用，使得项目实施者得以将精力集中在流程和沟通上，而这两者对项目规划有着重要的指导作用。

第9章

策略性漏洞



9.1 介绍

目前我们已经从技术、流程和管理的角度对漏洞管理有了一个全面的了解，下面将从一个更宏观的、策略性的层面来研究漏洞的呈现形式及修复方法。识别策略性漏洞没有可用的专门技术或程序，这需要实践经验和实事求是的心态。管理人员不仅要从攻击者使用的方法，还要从其动机和目标来了解攻击者。在对策略性漏洞进行分析时，具体的受攻击目标和攻击事件本身并不那么重要，本章将在更宏观的层面，从业务策略和技术策略的角度来探讨一些抽象的术语。

在分析策略性漏洞时，需要时刻牢记漏洞管理与其他重要 IT 及业务功能的基本关系，这一点很重要。图 9-1 显示了这种关系，并且还有漏洞管理系统向各个功能传送的信息类型。风险管理与业务策略处于 IT 策略层，而变更管理、事件管理、及配置管理处于 IT 操作层。在技术层面上，漏洞管理既是其他环节，如风险管理的信息提供者，同时也是风险管理的一个组成部分，并非独立存在的实体。这使得漏洞管理兼具了操作性和策略性，这一双重特性使漏洞管理与本章后面将要讨论的一个环节相耦合。

策略性漏洞的评估不需要太多的技术手段，但漏洞管理可以用做策略性决策支持工具，并影响决策过程，应在资源调配之前进行策略性漏洞评估，从而在业务规划流程中促进风险管理。如果不这样做，那么导致的策略性漏洞修复活动会产生大量的额外成本。潜在的策略漏洞很少能找到一种简单的战术性方法加以解决，也就是说，很可能会带来开销巨大的修复工作，但尽管如此，很多组织机构还是会在未进行策略性漏洞评估的情况下大规模实施业务资源部署。

很多情况下，策略性错误会带来灾难性的后果，并造成相互指责的局面，于是不得不对风险管理程序本身的错误采用策略性漏洞管理。例如，2001 年的炭疽杆菌生物恐怖袭击事件的发生，就是因为当时只将关注的安全重点集中在情报泄露风险上，而根本没有考虑危险物质本身也可以直接用于恶意攻击。在我看来，这件事的发生就是因为没能从动机层面去考虑潜在的攻击方式。

为进一步阐述策略性漏洞的含义，我们可以参照一下军事观点：“（策略性漏洞）指本国军事力量在面对其他国家军事实力范围内可实施的军事行动时，可能受到严重削弱或严重破坏的脆弱性。策略性漏洞与政治、地理、经济、信息、科学、社会或军事因素有关。”^①我们

^① Dictionary of Military and Associated Terms, U.S. Department of Defense, 2005.

可以将这个定义修改一下，以使之能更好地适用于非军事领域。我们可以将“国家军事力量”改为“业务”，将“其他国家”改为“竞争对手、批评者或社会活动家”。这样就得到：“（策略性漏洞）指公司业务在面对竞争对手、批评者或社会活动家能力范围内可施加的一些活动时，可能受到严重削弱或严重破坏的脆弱性。策略性漏洞与政治、地理、经济、信息、科学或社会因素有关。”

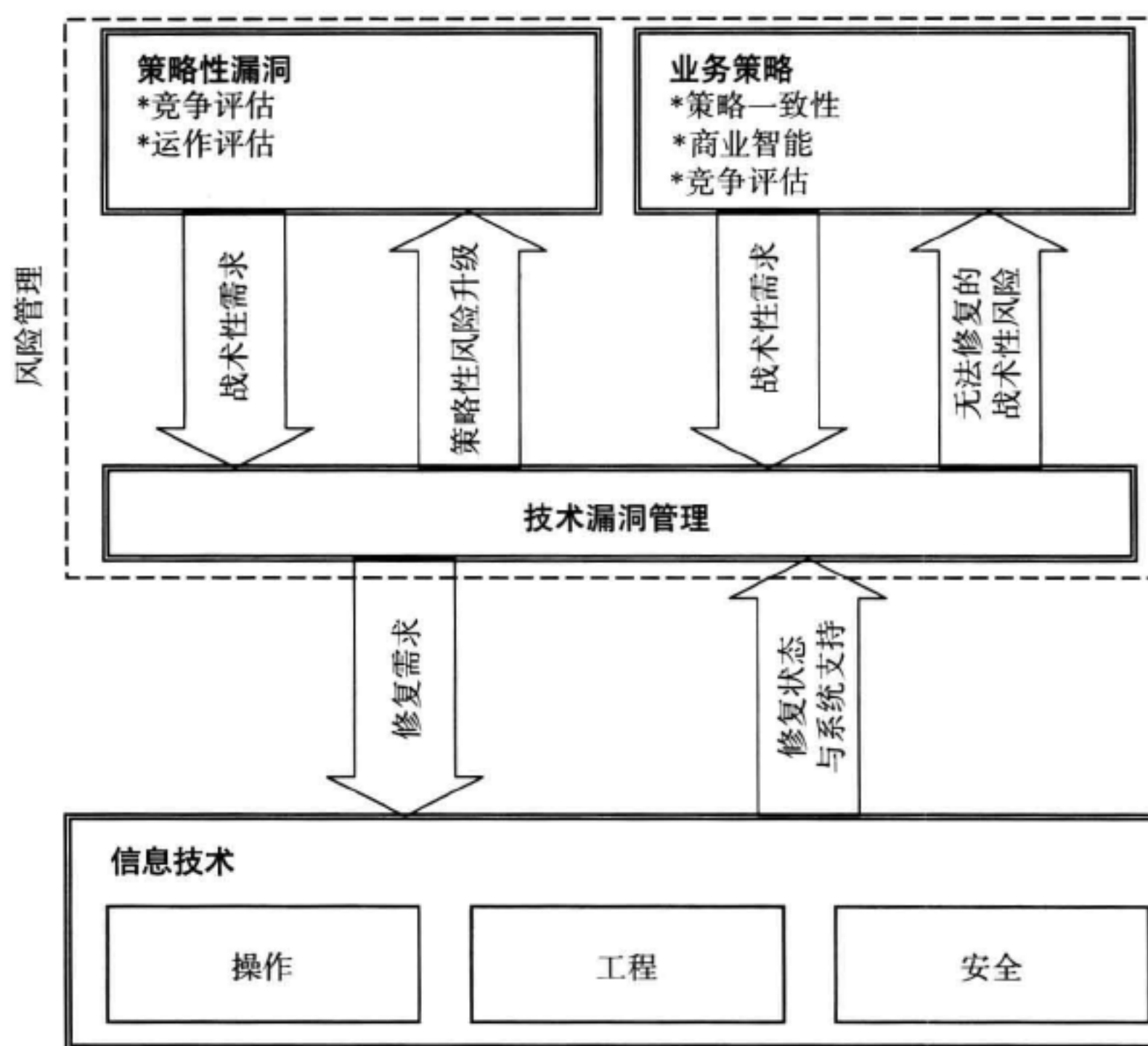


图 9-1 策略性漏洞与技术领域互动

检查威胁来源和威胁目标是识别漏洞的有效方法，如表 9-1 所示，可根据攻击区域将目标分类：

- **策略**：如果遭到攻击，其执行公司任务的能力会下降。良好的适应能力及应急预案可有效地化解此类攻击。
- **持续资源**：如果遭到攻击，组织持续运行的能力将受到显著影响。这类攻击有时可以依靠灾难恢复流程和技术来减轻危害，而更多时候则只能依靠预防。
- **领导层**：此类攻击会损害领导执行其职责的可信性和有效性。公共关系计划及适应性

方案是两种可行的化解方法。

表 9-1 漏洞类型

威胁来源	威胁目标
外部威胁 • 自然灾害 • 恐怖袭击 • 黑客活动 人员 • 雇员 • 客户 • 恐怖分子 供应链 • 供应商 • 原材料供应商 • 生产商	策略 财务 • 现金流 • 股份 • 专利信息 持续资源 基础设施与信息 • 计算机系统 • 网络 • 供货商 领导层 公共宣传 • 信誉 • 声誉 • 股份

以下是处理策略性漏洞的三个要点：

- ❑ 对存在上述威胁目标或威胁来源的操作系统实施持续、重复性的漏洞检测。例如，一家网上木材交易公司决定专门买卖一种只能在生态敏感区找到的木材，这很可能触怒一名精通计算机网络技能的环保人士，这些环保人士会将公司最薄弱的基础设施作为攻击目标。巴顿将军说的好：“要想在战争中取胜，不要去打击武器，而是要打击敌人的精神，找到你能力范围内的，能给敌人造成最大伤害的事，然后就去做吧。”这正是正是公司的漏洞管理员和风险管理员必须考虑的。漏洞管理员的优势在于对公司及其运作更为了解和熟悉，因此可以设计更持久、可靠的防御措施。
- ❑ 利用各种内部和外部策略管理外部因素。很多情况下，公司在制定策略时只考虑如何使该策略可行，而没能考虑可能产生的漏洞。应考虑可替代的方法以最大程度降低可能的外部威胁。本章将讨论组织外部环境的各个方面，以及如何分析这些方面，如何降低其影响。
- ❑ 无论其章程如何，内部漏洞对任何组织来说都是一种挑战，必须得到控制。一般遇到的内部漏洞包括雇员欺诈、业务程序缺陷、产业间谍等。这些漏洞通常针对组织的基本产品或服务，并能威胁组织的生存。

以下章节将讨论策略性漏洞是如何生成、如何识别及如何规避的。在阅读这些例子时，

想一想策略性漏洞管理作为所有业务活动的普遍参与者及作为风险管理团队的组成部分，是如何发挥作用的。

本章描述漏洞管理如何指导业务和技术策略，还将讨论漏洞管理充当风险减缓角色是如何发现现行策略中出现的漏洞。

9.2 操作环境

IT 操作环境的状况是漏洞管理结构与实施的重要影响因素，因此我们需要明确“IT 环境”的含义及环境的特点如何影响程序漏洞。

IT 环境包括两个部分：内部环境和外部环境。内部环境包括业务的所有方面及相关的技术操作：

- ☐ 人员技能及流动；
- ☐ 计算机系统及相关配置；
- ☐ 新业务程序及相关系统更改；
- ☐ 业务并购比率、目标规模及业务模式；
- ☐ 应用类型及关键漏洞发现率；
- ☐ 特定技术平台的普及率。

这些要素与 IT 运作关系最为密切，在某种程度上是可以通过直接行动进行控制或管理的。例如，若某一关键 IT 技术的操作人员存在频繁流动的情况，通过分析可以发现并减少这一情况。若某种特定的标准系统配置存在问题，则可以制订计划在全球范围内修正这种配置。

外部业务环境指公司 IT 活动外的、影响业务运作和安全态势的因素：

- ☐ 业界规定；
- ☐ 业界遵从的标准（如 PCI 标准）；
- ☐ 供应商的可靠性，其软件已在公司广泛部署；
- ☐ 外界对业界（或公司）的普遍看法和社会舆论；
- ☐ 面向公众的业务数量。

这些因素有一个共性，就是都很难或不可控制，但在适当的环境下也有可能对其产生一

些微小的影响。但通常，改变这些因素的成本收益率都很低。最后两项直接影响公司的总体风险状况，并可能会使某一漏洞变得更为严重，需要尽快修复。外部因素，如供应商的可靠性、业界民意及对公共用户的开放程度会共同作用，从而使得某一目标业务中的漏洞看上去比其他业务中的更为严重。例如，网络使用量高的烟草商要比只有公司用户的会计师事务所更关心网络服务器软件的缺陷。

业界规定会影响 IT 运作是众所周知的。例如，萨班斯法案的确立极大地增加了安全和合规性工作人员的负担，并增加了审计和评估成本。虽然业界通常试图影响此类立法，但其与政府间的复杂关系很容易造成一种官僚作风：只管向业界实施规定，而极少考虑规定将给业界造成什么影响。业界和政府谈判的焦点是政治性的，因此会避免所制定政策的特殊性、承诺性和可说明性。想到即将实施的规定所要带来的影响，策略性漏洞管理员所能期望实现的最好结果是尽快、尽可能地提高效率。

当将审计结果用以决定是否应继续为某种产品发放许可时，标准的制定与规定相似，PCI 就是一个很好的例子。这些标准的更改对技术架构的影响很大，成本也很高。透彻理解这些标准有助于发现哪些地方将来可能会变更，并对组织产生影响。由此，可以对这些潜在的变动进行规划，保证标准适应变动的灵活性。

9.3 管理外部因素

在讨论外部因素时，适应性是一个经常出现的概念，这是因为对于外部因素来说，试图改变要比接受和预防困难得多——风险管理员通常不愿接受这一看法。以下两个概念在 IT 界已出现多年，并在很多应对情形中很有帮助：

- 模块化：IT 组件和程序可以被封装到有指定的输入和输出的功能模块中，从而降低在对其进行移动、操控和修改的过程中对组织的整体影响。
- 耦合：这一概念被广泛应用于程序、系统和组织设计中。当各程序被更为紧密地耦合在一起时，其功能的发挥会更多地依赖于其他程序而不是简单的输入和输出。松散耦合支持模块化，因为松散耦合的程序不必关心输入数据的产生，而只关心在需要的时候存在即可。

模块化和松耦合增强了业务操作应对环境变化的灵活性和适应性。当出现有竞争力的新

的业务模式时，因为组织有备而来，所以可以迅速转向，适应这种竞争。Kyocera 公司就是一个很好的例子：组织策略在面对策略变动挑战时提供了良好的适应性。Kyocera 公司创始人 Kazuo Inamori 创造了一种文化：能在适应外界变化的同时保持卓越的业务经营，这一模式被称为阿米巴管理系统。

当出现一项业务需求，如开发新产品时，就新建一个阿米巴系统，并在产品开发完成后解散。而系统中的资源，包括人员、设备、财务等会被现有或新建立的阿米巴系统选用，这样，资源就可以有效地重复利用，并在不需要的时候去除。这对于结构来说是一种高适应性、低成本的方法。如果某项业务功能处于策略性威胁中，因为功能很小所以它可以适应该威胁。但如果无法适应，则该功能可以被移除而不会对组织造成太大影响，因为它与公司其他部分的系统性连接很少，这降低了公司风险，加强了对不利事件的应对能力。

另一种特性是尽可能少地依赖于通用基础设施和服务的潜在的设计能力。中心组织可以提供任何所需的基本的设施与服务，除了核心服务，因为可能会对规模经济效益产生重大影响。将两个阿米巴宽泛地连接在一起是不必要的，普适性的网络和系统设计也是不必要的，这样可以将漏洞群的扩散风险降到最低。普通的、共享的服务，如工资及福利成为攻击目标的可能性很低。

可以在很多层面上使用阿米巴来管理风险。在不确定哪种策略可行时，减少承诺是有利的。例如，如果 20 世纪 70 年代的一家公司致力于制造 Betamax 磁带，而在当时 VHS 是一种策略性威胁，这样为每种技术创建一个阿米巴有助于在高科技竞争中降低漏洞带来的风险（高科技在这一案例中为录音时间的长度）。而当 VHS 成为人们喜欢的格式后，Betamax 的阿米巴就可以解除了，并将资源用于支持更为成功的技术。企业里肯定会有资源重叠的地方。

索尼公司并没有采用适应性策略来应对科技产业的竞争压力，而是十分笃信地为自己选择了一条单一的途径，这就成为了公司的一个策略性漏洞。为了降低这一漏洞可能带来的风险，他们又进一步地试图控制客户这一外部因素，但这是很困难的。采用效率更高、适应性更强的方法会更容易成功。

9.4 控制内部漏洞

策略性漏洞在组织内随处可见，这是因为在系统开发生命周期中，只考虑了当前环境下的风险。在制定策略时一般没有考虑到环境的动态性，以及环境为应对竞争可能发生的不利

变化。这是典型的无法预见的漏洞威胁，当最初假设成立的基础发生变化时，这一威胁就会出现。不管在什么环境下，策略性漏洞最主要的特点是其意味着对组织健康的一种威胁。策略性漏洞是与公司目标或政府本身紧密相关的。

9.4.1 业务模式

人们总倾向于在理想条件下制定基本业务策略，而对可能发生的情况不加考虑。但环境是变化的，对于一项业务策略来说，现在正确的将来不一定正确。这对软件、新的产品和服务中的漏洞也是如此。美国在线（AOL）公司就遇到了此类问题。在互联网风行之前，AOL就意识到了使用电脑和现代交流工具的人们需要组建在线社区并交流信息的需要，因而其业务模式在多年内都非常成功，直到市场上出现了另一种更廉价、更开放、更广泛的模式。

20 世纪 80 年代，一个叫 Quantum Link 的小公司利用 BBS 实现了在线信息共享，此后，公司将策略转向迎合各种新出现的计算机交流平台。BBS 早期最大的吸引力就是在线游戏，但后来开始提供各种新颖、易用的交流工具。从 2001 年开始，互联网的普及打破了全球交流的障碍，BBS 的用户开始减少。很多现在的和潜在客户发现，AOL 提供的社区服务（已关闭）收费较高且使用价值低。

为求生存，AOL 与提供独家内容的时代华纳公司合并，现在回想起来，这种做法好像很合情合理。由于技术发生了变化，业务模式也需要大幅度改动。但其实技术的变化来的并不算快，甚至是可以预见的。早在 1996 年，该技术的前景就应引起策略性漏洞管理员的注意，若更早地采用新技术，是可以减少技术变化带来的影响的，从而使公司成为市场的引领者而非衰落者。在开发阶段设计和废弃各式可能的业务模式产生的成本与核心业务衰落却无法拯救所造成的损失相比要小得多。

从上述案例中可以对策略性漏洞的含义有一个更清楚的认识。策略性漏洞涉及的范围很广或很深，需要通过调整来补救和减轻。漏洞管理不仅仅是对内部业务技术的审计和对漏洞的查找。外部环境和竞争性市场也为业务战略自身的漏洞造成了威胁。

9.4.2 业务程序

程序很容易被客户和雇员之类的人控制，成为欺诈和钻营系统的渠道。当实现某小型业务功能的一个不太重要的程序被利用了，所造成的损失不会大到威胁公司生存。只有造成大规模影响的才称得上是漏洞战略。例如，在一家彩票公司中，如果有人发现了一种方法，每

次都能选中中奖号码，那这一程序缺陷就会成为巨大的策略性漏洞，将导致巨大损失。

9.4.3 复杂性

安全领袖 Bruce Schneier 受到普遍认可的一个观点是：复杂性是安全的敌人。虽然他那时是针对加密算法提出的，但对安全策略也同样适用。复杂性使漏洞变得隐蔽，难以在运营策略制定时被发现。

1. 漏洞方案

小细节能造成大影响，复杂的供应链就是一个很好的例子。如果多个不相关的组织参与到关键组件开发的各个阶段中，那即使是最小的供应商从中做了一些手脚，也会给行业或某公司造成广泛的影响。例如，LCD 电视生产商 A 公司从 B 公司购买 LCD 显示器，而 B 公司从 C 公司购买液晶，C 公司从 D 公司购买关键原材料铟，D 公司位于加拿大，世界各地绝大多数的铟都是由该公司提供的。如果来自加拿大的 D 公司供货中断，那整个供应链的价格都会迅速上升，但 A 公司没有看到这种情况，也没有采取措施减少对单一资源或单一产品的依赖。

2. 规避策略

供应链一般很长、很复杂，多方参与其中，虽然松散，但却满足了客户的需求。近年来，网络技术的出现减少了中介的参与，供应链变短了，但这也产生了负面效应，即公司与上游供应商隔离开来，而上游供应商往往掌握着更多供应信息。

例如，如果发现供应商依赖于单一来源的某种关键原材料，那漏洞管理员就应考虑寻找依赖其他替代性原材料源的供应商，以防意外事件发生，这相当于在存在薄弱环节的供应链上创建备用方案。在 LCD 生产商例子中，A 公司最好寻找一个替代性的 LCD 显示器供应商，并且其铟来源应不同于前一家供应商。

该方案展示的是内部漏洞，因为这主要是由内部策略决议引发的，而该决议通过评估和规避手段是可控的，而不是要靠制定响应方案来解决的。

9.4.4 反应方案

一些内部策略性漏洞无法前瞻性修复，而其危险又是不可接受的，这时知道什么行为会对漏洞造成威胁非常重要。策略性漏洞的存在会诱发恶意的攻击行为，留心关注这些问题，制定出备用方案，就能在攻击发生时及时应对。

例如，公司会设计多种将产品和服务交付给用户的途径。如果该业务模式中的漏洞就在交付方法上，当最初的交付方式受到攻击时，公司可以立即采用其他的替代方法。下面是一个例子。

1. 漏洞方案

Spam Free Corporation 为大公司开发出了一套独特的垃圾邮件过滤方法，该方法以软件作为服务模型，电子邮件需先通过 Spam Free 公司的网关，经过处理后才被推送到用户的邮件服务器。这种方法不需要额外增加硬件或软件。垃圾邮件过滤系统的源程序很小，对一两个用户很有效，但要处理大量用户的邮件通信流量时，则需增加专用的服务器。垃圾邮件过滤软件可以从处理大量用户邮件的经历中不断学习，从而提高程序整体的可靠性。Spam Free 的过滤软件也采用这种模式，没有多少新意，除了其过滤算法特别高效，还能够对众多用户收到垃圾邮件的次数进行统计。

2. 规避策略

意识到其他公司只需突破微小障碍就可以进入该市场，Spam Free 开发了一种过滤垃圾邮件的新服务模式，该模式要求用户在他们的邮件网关上安装 spam 过滤模块，该软件间或会向中央服务器反馈启发式（经验）数据或从中央服务器接收更新。由于用户邮件无需再经过 Spam Free 的公司网关，邮件的传送速度加快了。但该方法的缺点在于软件必须安装在客户站点。

当原来的业务运营了半年并拥有超过 125 个客户之后，一个竞争对手进入了市场，以更低的价格提供类似的服务。Spam Free 现在似乎不得不直接进行价格竞争，而这种做法会直接造成收益损失。但事实上 Spam Free 并没有这样做，而是开始以更低价格推销新服务，并向用户介绍新服务的优点。结果更多的用户喜欢这一新服务，认为这比竞争对手的服务更好。

在这一案例中，规避方案变成了竞争优势，新服务一直到出现了竞争威胁才被启用，是因为其支持成本高而收益低。在制定新业务策略时会有很多替代选择，很多都是由于收益的原因而未被采用，但是花一点时间和资源对其进行完善，这些替代性策略都能用做应对竞争环境变动的备选方案。

9.4.5 漏洞方法论与变更

公司技术资产的部署要求发挥最大效用。虽然服务器、网络，甚至整个业务功能看似运

行良好，时间长了也可能产生漏洞，并最终引发风险。图 9-2 显示了在漏洞长时间未被发现的情况下，漏洞威胁水平是如何随着时间的流逝逐渐升高的。漏洞存在的时间越长，出现潜在威胁的风险就越大。这是由于业务功能相关的目标资产的市值会随时间变化，并且漏洞利用技术也在不断成熟。

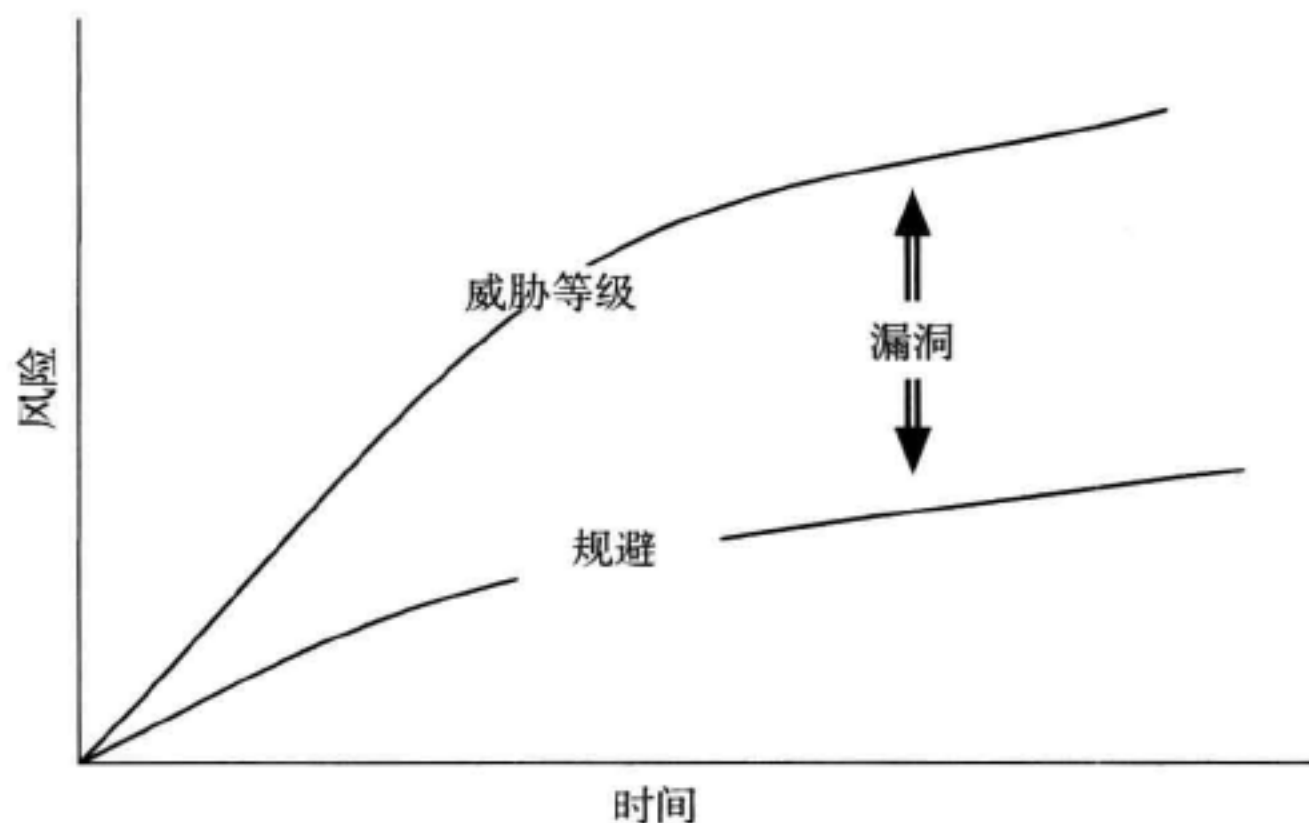


图 9-2 威胁等级与规避

例如，某种产品的供应链会随着时间的流逝变得更加依赖于某种关键资源。或者，新技术的出现使得企业原先所依靠的技术在市场中变得不再有竞争力，就像 AOL 案例中展示的那样。图 9-2 中的第二条线突出显示了修复或规避措施如何严重落后于威胁风险的发展，两者之间的差距显示了不断增大的风险。造成修复或规避措施不能及时跟进的最可能原因——未能制定出一种能对资源价值随时间进行重估的漏洞管理策略。

图 9-2 反应的问题就是在于缺乏应对威胁的方法。像 AOL 例子那样，探索开展新业务要比识别威胁成本更低。将 AOL 例子对应到图 9-2 中，“威胁等级”具体指“网络”，“规避”具体指“探求新的业务模式”。

另一个真实的例子是微软对开源软件的应对。微软 CEO 曾明确表示源代码开放的 Linux 操作系统是一个威胁。最好的防御方法可能就是开发高性价比、易于实施和操作的 management 工具。

公司如不能迅速适应环境的变化，则将遭受巨大的损失，这种例子也很多。IPX 协议的使用就是一个很好的例子。IPX 协议曾在 20 世纪八九十年代非常流行，由于 Novell 牢牢控制着该协议的使用，用户转而寻找更具竞争性的资源。在技术飞速发展的情况下，IPX 由于缺乏防范，导致人们最终选择采用开放标准 TCP/IP。

Novell 有两种选择：采用 TCP/IP 让原有用户放弃使用 IPX 或发起防御，驱走威胁。

Novell 适应速度很慢，而微软却将 TCP/IP 集成到了其快速普及的操作系统中。

此外，联网计算机数量的快速增长远远超出了 Novell 的反应速度。一边继续支持 IPX 的遗留系统，同时将生产线移向 IP，这种工作负荷超出了公司的承受能力。虽然 Novell 现在仍然是一个很成功和不断发展的公司，但其市场占有率已经小得多了。

对于 Novell 的管理，公平来讲，当存在广泛的支撑平台时，确实很难发现底层技术中的漏洞。由于 IPX 事件，如今微软已经成为全球主流操作系统。

漏洞的增加往往是由于未能分配足够资源以识别和修复造成的。为解决这一问题，不仅要看到整体的风险链，也要对具体资产的风险及其价值保持清醒的认识。在该例子中我们可以看到，处于风险中的资产及整个的威胁链都在不断地发生着变化。任何风险识别模型都应提供一些概率计算功能，或至少能够预测最坏情况。以下几点对于管理策略性漏洞很有用：

- ☐ 持续了解什么对公司最重要；
- ☐ 组织一个漏洞联合小组，这可能包括来自法律、事件管理、人力资源、IT、操作系统等部门的人员代表；
- ☐ 保证投资与预算的方案与潜在影响相匹配；
- ☐ 站在利益相关者的角度不断评估漏洞。

9.4.6 复杂性

如果环境复杂，要发现漏洞就更加困难，因为它们藏于细节中。如果降低复杂性，漏洞的识别和修复就会变得容易一些，但由于缺乏一些关键方法，保证在环境中实施应用的安全性就更具挑战。例如，一个只有两个安全层级的简单网络架构，对于漏洞监测和实施安全应用而言无疑很容易。

简单的网络设计中可能包括两个基本层面的安全：互联网 DMZ 和内部网。要保护其安全，在边缘、核心及每台主机上加装安全组件即可，这种架构很容易管理，但在真实的复杂网络环境下对复杂的业务要求来说是不现实的。

极其复杂的网络架构管理难度将更大，当受到攻击时情形也更难以描述，漏洞源的识别和修复都变得更为困难。不过，根据不同的技术环境，还是会有一些有效的处理方法。

1. 集中化和区域化

全球各地的网络基于当地不同的业务模式和发展状态，呈现出各式各样的结构，但都要

求为扩大经济规模进行一定的企业兼并以及对管理和监控给予更多关注。如果一家公司有数百个统一安装了专用销售程序的销售站点，一些站点可以通过互联网进行访问，而其他只对公司内部用户开放，那么这种公司就是进行兼并的首选。这种方案的特点是拥有统一的业务或应用访问模式，其访问方式在整个公司是完全一致的，除了每种应用的功能是针对特定用户群体的要求专门开发的。但由于该公司的销售站点分布在全球各地，漏洞的检查和事件的监测都更为困难。

一种更为主流的方法是将应用程序服务器集中到某个位置或地区，从而更有利于统一地、持久地管理监测和漏洞评估。集中采购及更有效地利用网络资源也可能会实现规模经济，这样做在安全方面最大的好处是很多未被当地 IT 人员检测到和修复的漏洞会得到训练有素的专门小组的更仔细、全面的修复，公司也会从基于网络流量分析的漏洞评估中受益。

2. 标准化

如果由于政治或技术的原因无法实现区域化或集中化方法，可以考虑标准化和审计。虽然不管采取什么策略都需要标准化和审计，但在分散化的方法中其作用更为重要。

标准的制定必须使用多种语言，并要支持多种环境。标准化中最困难的是使所有 IT 区域都按同一基准操作。以下是必须标准化的项：

- ☐ 漏洞评估的方法与频率；
- ☐ 修复流程（如创建标签、确认、关闭）；
- ☐ 用于支持修复的变更管理程序；
- ☐ 主动扫描参数；
- ☐ 基于主机评分的漏洞管理服务水平。

常规的审计对于使用标准和策略确认合规性是必要的，这种审计可以通过查看自动生成的报告来完成。要亲自去远程站点进行查看的情况是很少的，除非确实出现一些疑难问题。

9.5 规避原则

虽然这听起来好像纯粹是网络和系统架构师的事，但架构所受到的威胁很少能够得到充分考虑，这是因为系统架构师更关心运行操作的成功，而不是防御。安全架构师则对攻击更为关注，看法也更为实用。

网络架构师发现了一种有效控制流量的路由方法，对攻击者而言则找到了一个可以通过破坏路由和 ARP 表发动攻击的新目标。为更好地利用设备，网络架构师对服务器进行多处备份，一处放在内部网络，一处置于 DMZ。而备份协议对于攻击者而言，则又看到了一条通往防火墙背后诸多可利用漏洞的渠道。

架构中的漏洞并不总是这么明显，尽管每个组件的位置和功能都已被明确定位，工程师还会考虑该架构可能要调整适应的其他情况，从而保持对业务需要变更的灵活性，但这可能会损害最初的安全设计意图。以下原则有助于构建更安全的技术架构：

- ❑ 控制实现各功能目标的途径。制定这条原则是因为设计中考虑过多的灵活性会很容易造成漏洞。设计时应该只允许实现执行所需功能必需的方法途径，不过，对方法的界定就没有那么严格了。但这并不是说不能通过路由协议更改或升级网络路由，而是要求将路由限定在可接受范围内。
- ❑ 主动管理技术系统中的所有活动。同样，这也不是不要灵活性，但应将活动控制在一定的操作参数范围内，就像在网络的各个要点安装防火墙以限制资源、目标和使用的协议一样，计算机系统的各组件都需有此类限制。从应用的参数作用域验证等级到网络存储系统的访问控制均需如此。
- ❑ 备份不是重复。在系统中创建备份并不意味着需要复制系统。如果一个系统存在漏洞，那复制系统只会使这种风险加倍。很多产品被制造商建议作为复制品的方式来应用。一般来说，厂商会认为，复制是为了通过冗余提供高可用性，但设计者却并未因此考虑进一步提高其安全性（健壮的安全目标）。虽然用两种不同的方式编写同一个应用并期望其性能一致不太现实，但用多种配置方法实现同一种目的是可行的，虽然这样做在很多情况下不易管理，但可以减少受到某些针对某种产品或某个版本的漏洞的威胁。
- ❑ 为公司所有重要活动设置迅速响应功能。策略性漏洞的一个重要特点是处理延时会放大漏洞攻击所造成的影响，迅速响应可部分化解该问题。应急方案和灾难恢复（DR）规程可提高公司的响应能力，但这些计划要不断根据威胁情况进行修正，并测试其有效性。
- ❑ 漏洞总会存在。在漏洞识别中需要持续保持警惕，如果漏洞没有被资产所有者发现，就会被攻击者发现。资产所有者有一个关键优势，就是可以自由地查看资产的评估情况，从而能够了解资产的市值。唯一能在这一点上与之匹配的就是来自公司内部攻

击者。

- 漏洞并不总是“无生命”的，人也可以成为重大的漏洞源。任何安全策略都需包含对人员漏洞的修复方法，这是指在实施应用程序时要持续地、逐步地提高操作人员的安全意识。但绝大多数公司却只愿意花费大量的钱去发现和修复技术漏洞，而不去培训和提高员工的安全操作意识。而事实上，后者的性价比往往更高，并且几乎不需要任何专门技术或程序。
- 驱动业务的是流程，而不是技术。虽然本书主要从技术层面论述漏洞管理，但本章将重点声明，漏洞并不仅存在于技术之中。业务流程确定了业务目标是如何实现的，而流程当中也存在漏洞。忽略这些漏洞等于不给面向公众的网络服务器打补丁。

9.6 了解对手

俗话说，知己知彼，百战不殆。对于漏洞管理而言，防御措施本身也是一种受攻击对象，并且这些防御措施的灵活性并不足以应对狡猾多变的攻击者。因此，为保证每项资产不被攻击利用，必须加强和调整资产本身的防御性能。想象一下如果一支军队的枪支都被缴获并被调转枪口指向军队本身，这会怎样的情景。若这种情况发生在漏洞管理中，那么资产就成了使风险加倍的累赘。而这种优势正是攻击者企图获得的。

9.6.1 至 9.6.5 节讨论了了解对手的一些重要原则，以及如何确定优先修复哪种漏洞和选择哪种修复方法。

9.6.1 优点与缺点

不要因为对手少就认为他们弱。利用现代技术，对手能够掌控弱势的网络资源并将其转而用于攻击你。更糟的是，存在设计缺陷的技术能够发挥将一把单发的手枪变成一排大炮的作用，DoS 就是这种例子。可考虑以下策略：

- 将系统架构分解成多个保护区。有些修复操作即使不必进行广泛地测试也有可能造成服务中断，对于这种情况，分割保护区的方法能有效地为关键漏洞的修复提供更多的时间。在制定各保护区的优先级时，应当对其性能有充分地了解。
- 一个人不必掌握很多黑客技能就能发起一场拒绝服务（DoS）攻击。与此类攻击者邻

近的区域内，如果含有任何 DoS 漏洞，都应尽快得到修复。此类攻击随时会变化，今天有效的方法明天可能不适用，域名系统攻击就是这样。对于那些支撑公司在世界上的商业形象的重要资产，可以通过维护一台授权服务器的方式对其加以保护。DoS 攻击经常将 DNS 服务器、面向 Web 的服务器和网络设备作为攻击目标。

- 安全区越靠近因特网，区域内漏洞修复的优先级就应越高，可以考虑将该区域内的漏洞的威胁等级都加倍。虽然该区域内的资产本身可能并没有那么高的价值，但公司声誉和公共形象一旦遭到破坏是很难恢复的。
- 对处理关键数据的目标采取严格的访问限制以减少攻击的可能性。数据库服务器就是一个很好的例子。数据库服务器存储着对公司极为重要的信息，因此必须被置于公司的防御区内，且进出这些区域的数据流应尽可能小。可以额外增加安全设备以保证系统和数据安全。
- 采用互助支持防御措施。与建立防御区相似，特定安全区内的安全特点会被用于支持其他区域的安全，从而有利于尽快识别感染者。互助支持防御有效是因为攻击在到达最终目标前经常要经过多个区，因此会受到每个区安全措施的影响。对于漏洞来说，这有两种好处：一是它加强了在未发现和修复自身漏洞的情况下阻止攻击的能力，二是通过向关联服务通报该目标的威胁等级，漏洞数据有可能有助于防御。

9.6.2 现实事件

过去和当前发生的现实事件，虽然看起来不相关，但会为漏洞策略员提供有价值的参考。动机总是相同的，不同的只是方法。恐怖主义、公平感、光荣、复仇、保护他人或拯救世界的妄想会将一些人误导入背叛、邪恶及为伤害他人寻找理由，几百年来一直如此。以下是需要注意的一些重要方面：

- 最敏感的区域往往是员工压力最大的地方，因为某些员工必须能够访问关键系统。管理员可以通过妥善管理员工，减缓员工的压力来减少漏洞的发生，这样做的成本与可能发生的内部员工监守自盗的系统攻击行为相比，一般会较小。
- 发现并消除发起恶意行为的动机。如果你所在的组织拥有信用卡号码这类信息资产（出售可以赚钱），那么应该给能够访问这些数据的员工足够高的薪水以高薪养廉。此外，要确保对所有员工都能实施监控，但不要有侵犯性。将员工组织起来，共同抵制

将那些想共谋盗窃或破坏数据的人；还有一条在 IT 管理中经常被忽略的原则：最小权限原则，这一原则能避免职员中出现试图提升访问权限，而这些高访问权限并非工作所需的情况。

9.6.3 目的与目标的对比

攻击者试图达到的目标往往难以预测，他并非全无理性，或缺少智慧。攻击者使用的攻击媒介不一定直接与目标相关，但对于实现目标同样有效。

如果某石油公司最可能的攻击者关心环境或安全问题，那么其攻击手段可能不会是直接将油倾倒入环境中，而可能是破坏供应链支持系统，引起业务流中断，从而导致更长远、更高昂的损失。其长期的影响可能是对石油公司声誉的破坏。石油公司是和石油和供应链绑定在一起的，公司业务的特点是要有大型的集中式的基础设施及运输系统，而这些都会成为攻击目标。以下是对可能成为攻击目标的事项所能采取的预防措施：

- 改变资产的属性以分散漏洞。一个简单的例子是将数据分散存储到公司内不同安全级别的其他区域。这是进行数据分类，并根据风险等级将数据置于适当的安全区的基本程序。这使得攻击者得到全部有用数据集（目标）的难度增加，从而降低未经授权的个人成功访问的可能性，同时也会降低单个数据源的价值。攻克各个数据源所需的方法截然不同，从而增加了攻克整个目标的难度。石油公司要想以较好的性价比来实现这一点可能很难，但对信息资产类公司来说是可行的，因为后者的资产集中程度较低。
- 将数据逻辑集中点最小化到最小的用户组。这是对前一条的补充，以确保当需要从多个不同来源收集数据进行处理时，只传输和使用必需的数据元素。临时存储点和不安全的输出被最小化。如此一来，当攻击者需要来自两个以上数据源的信息时，就会受到诸多限制，对攻击行为形成阻碍。
- 可从以下角度评估数据破坏或系统瓦解的影响，即竞争对手、攻击者或不满的员工会怎么利用这种方式对公司造成伤害。公布于众的数据入侵事件将会影响公司的形象和信誉，无法提供服务会直接影响用户或使关键的资金流中断，从而使竞争对手获利。部分数据和系统分级程序应考虑资产对竞争对手和攻击者的价值，而不仅是该资产能给公司带来的直接收益。
- 找出销售流程中可能会被竞争对手发现有缺陷或不完善的环节和方法，这在 IT 界尤为

重要，因为技术上的任何细微差别都可能成为竞争中的不利因素。任何接触过技术销售员的人都能理解这点。如果销售员问你正在考虑其他哪些产品，那攻击策略就很明显了。

9.6.4 时间放大效应

在漏洞攻击中，时间的影响最容易被忽视。对基本信息安全攻击事件的长期、高度公开宣传能够破坏一个公司甚至一个产业的声誉，零售业与支付卡行业就是此类例子。如果一家大型零售商遭受攻击，所造成的破坏是暂时的，但所带来的影响却需要很长时间才能消除。当同一家或一家相似的零售商再次遭到攻击时，人们会很容易认为第一次被攻击的零售商及相似的公司也遭受了破坏，人们会考虑是否还要与这些置客户数据于危险之中或潜在危险之中的公司做生意。长此以往，该零售商及整个行业的信誉都将受到不可挽回的损害。

支付卡行业尤其如此。由于信用卡号码和客户信息是行业的关键资产，信息的保护被置于优先地位。虽然有些数据攻击不属于信用卡公司的控制范围，但是涉及这些关键资产的数据外泄，都将随着时间逐渐减小公司的业务增长潜力。有多少人关心他们拥有的号码或信用卡，以及在购物时对信用卡的使用？

支付卡行业为此制定了 PCI 标准。虽然提高安全性的意图良好，但不幸的是人们将注意力都集中到如何符合标准而不是更高的安全性上，但该行业已经尽力了。

人们该如何处理那些危害性与时间成正相关的潜在漏洞呢？考虑以下方法：

- ❑ 制订计划快速平静地消除潜在的负面舆论。如果公司没能保护资产安全或监控外部因素变化，迅速承认过失并呈交应对行动计划。最好能将这一流程标准化。
- ❑ 确保数据过期不用后就被清除。一些公司长期保留不再使用的客户数据，这就成为潜在的漏洞。之所以造成这种无限期保留情况，要么是因为公司从来没有制定过关于过期数据如何处理的有关规定，要么是因为公司认为这些数据也许可用于将来的某个业务模型中，却没有意识到这种无限期保留数据所带来的危害远比可能带来的便利要大得多。
- ❑ 通过加入安全协会与业界公司联合共保安全。一个行业的所有公司都对行业公共安全感兴趣，分享安全措施并不会泄漏商业秘密，相反会增强行业整体的公信力。

9.6.5 政治环境加剧攻击

立法者在公开谴责公司未能成功保护数据或数据内容本身时，会不经意地为数据盗窃攻击提供信息，从而加剧攻击形势。这种错上加错的局面使得失窃的信息被攻击者一用再用，充分挖掘和利用了其价值。Big Oil 公司的攻击者就是利用了这种情况，通过立法者在谴责公司时公开的信息，进一步找到公司的利润或供应信息并将其公之于众。

另一个例子与信用报告机构丢失客户信息有关。这些信用报告机构本是个人财产或个人信息数据的保管者，但实际上却未能成功保护客户信息。政府官员为消除用户担心，严肃惩处了信用报告机构，并要求其加强立法。政府这一举措表明该公司或整个行业都存在着严重的漏洞，并且还没有严格控制漏洞的相关措施。

受牵连的各方，由于未能有效管理程序和系统中的关键漏洞，其声誉都受到了严重影响。这些事件中的攻击者可能是一名拥有访问权限的员工，甚至可能是想非法购买信息的机构，而这会间接地鼓励信息攻击行为。

9.7 小结

策略是指公司或政府确定如何达到其目的的方法。这些目的是随时间而变化的，环境也是这样。策略性漏洞管理是管理由这些变化引发的风险的一种方法。就像计算机系统上安装的软件，并非在一个静止的环境中运行，同时也会出现因最初的设计者未预测到的使用情况而导致的缺陷，策略也是这样。

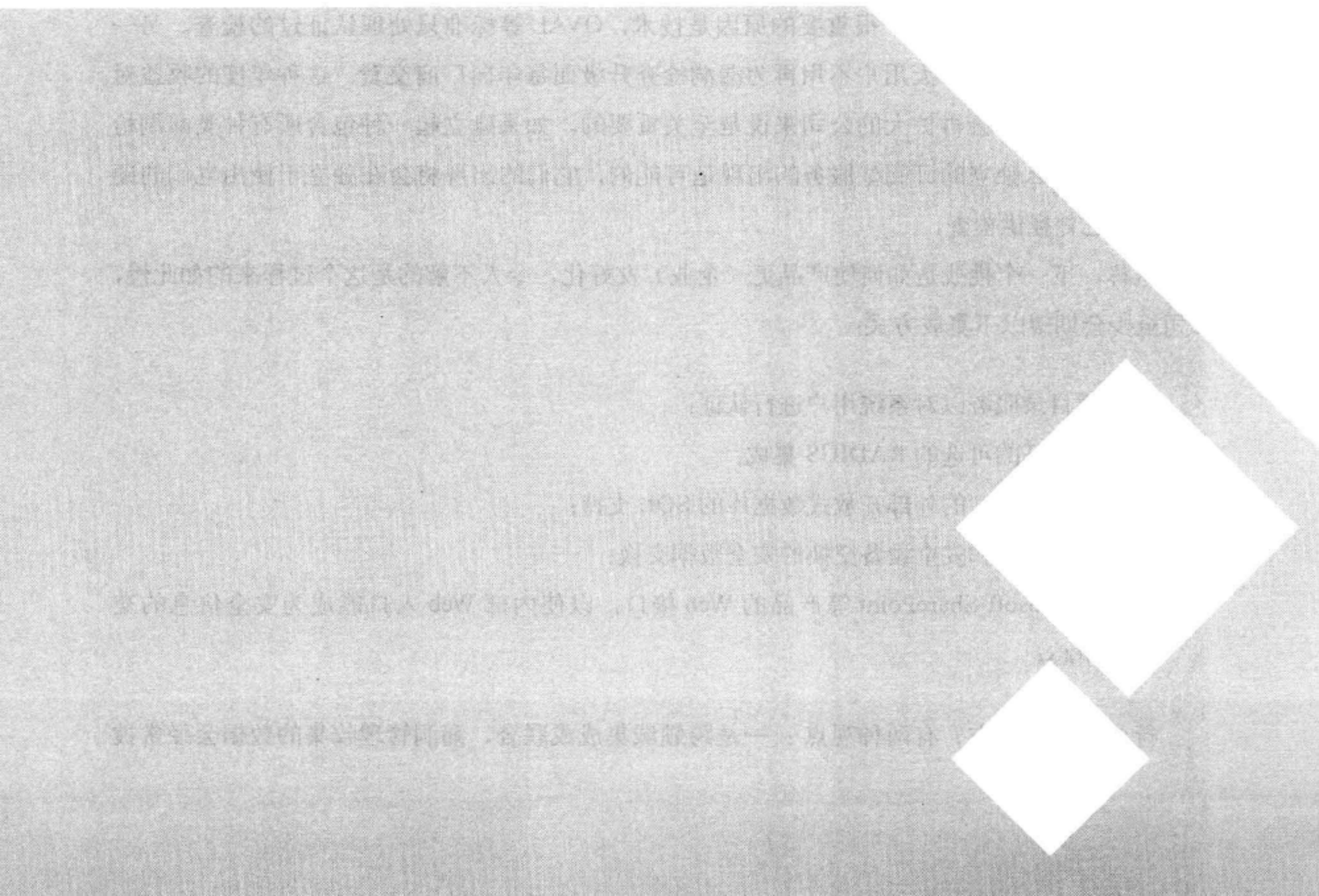
以下几点有助于成功识别和修复策略性漏洞：

- ❑ 通过相互关系和流程密切关注业务策略制定过程；
- ❑ 不断站在竞争对手或其他潜在敌人的角度看待问题；
- ❑ 关注环境变化对现行策略的影响；
- ❑ 将制定新策略作为创造机会以规避漏洞的手段；
- ❑ 持续关注和了解什么对公司最重要；
- ❑ 与存在共同问题的其他关键部门的成员组建联合团队；
- ❑ 使投资与潜在影响相匹配。

这些方法需要引导、创造性、交流、坚持及变通的开放性。虽然这些不是技术方面漏洞管理所严格规定的科学的程序，但对最大程度提高企业的效能很有必要。策略性漏洞管理对公司财务“保健”的作用是不可低估的，公司的高级管理层应了解这一点并接受策略性漏洞管理。

第10章

总 结



10.1 介绍

漏洞管理管理流程看起来很复杂，但也能带来巨大的益处。毫无疑问，漏洞、配置错误以及补丁在未来一段时间内将一直存在。在漏洞管理行业形成的早期，重复的扫描、修复和验证操作被认为是主动防御性的安全措施，其出发点就是要早于对手发现和清除漏洞，这在很长时间里成为了一种共识。

很多厂商介入以后，竞争转而变成谁能识别和检查出最多的漏洞。

像入侵检测系统（IDS）一样，误报是一类值得注意的问题。在主要漏洞管理销售商提供的宣传资料里，那些精心斟酌的宣传介绍让人实在难以确定哪家产品最为适合。公司用户必须不断进行测试，并将测试结果与需求对比，但最后，所选定的产品也就仅能达到满足需求的程度。很明显，精确性和全面性不是漏洞管理系统最重要的衡量因素。

行业下一步要增强漏洞扫描器的功能，或者说增加用户自定义漏洞参数功能。OVAL 和 CVSS 协议提供了这种框架，但为了保持竞争力，并突出自己的产品，开发人员会坚持原来的执行方法，并使系统保持一定程度的封闭性。直到现在，几乎没有厂商提供 XML 接口以接纳如 OVAL 之类的标准，一个很重要的原因是技术，OVAL 等标准只处理认证过的检查。另一个原因可能是担心有一天用户不用再为漏洞检查升级而每年向厂商交费。这种年度的收益对于维持一个基本不会再扩大的公司来说是至关重要的。如果建立起一种包含所有种类漏洞检查的标准，那么独立的订阅型服务的出现是可能的，它们的出现将会在独立于使用它们的硬件和软件之外提供检查。

这样，下一个挑战是如何使产品更（企业）友好化，令人不解的是这个过程来的如此慢，公司至少会期望以下集成方式：

- ☐ 集成目录服务以对系统用户进行认证；
- ☐ 用于认证的可选的 RADIUS 集成；
- ☐ 用于应用模型的外部开放式数据库的 SQL 支持；
- ☐ 与其他网络和安全设备交换的安全数据交换；
- ☐ 与 Microsoft SharePoint 等产品的 Web 接口，以使内部 Web 入口能成为安全信息的交换中心。

行业将走向何方？有两种观点：一是跨领域集成或联合，漏洞管理收集的数据会经常被

导入其他区域，如配置数据库及安全事故与事件管理（SEIM）中，因此其需求是显而易见的，已经通过流程在这方面采取了一些措施，但还需要提高；二是更紧密结合各种技术，很多技术肯定会受益于漏洞管理数据，这种互惠已经有所体现。

10.2 跨领域机会

漏洞管理行业正在努力寻找其定位。公司购买漏洞管理不是想让它成为包容一切的安全管理平台，这样的策略对用户来说难以接受。其他的策略是作为相关功能的扩展，如合规性监测、风险评估、配置管理。这些领域从技术角度而言意义重大，但从流程角度来看存在着很多风险和挑战。合规性管理有意义，因为其功能与漏洞管理密切相关。但还需要创建更多全面的、可定制的合规性检查，这些检查应当能与现行的大多数通用合规框架相匹配。

由于执行合规性检查的一般不是技术人员，因此合规性管理技术应简化。在用户界面定义策略的简单方法应秉承一个原则，即不必知道待测目标系统的相关特点及功能。其他漏洞扫描器配置项（CI），如时间表、带宽、目标系统资源、核查升级等也需简化，以便非 IT 人员也可以操作。

风险管理无疑是业界已经开始但却远未完善的一项功能。如本书所述，漏洞管理对风险管理功能有重要贡献，但更高层次的风险管理功能却留给了其他厂商，或是被忽视了。风险管理产品存在的问题是很多公司并不正式确认这项功能，从而造成无法分配所有权。另外，由于分析人员无法得到准确、翔实的数据，风险管理是以相对抽象的方式进行的，而漏洞管理系统或许能提供这些细节。

阻碍风险管理 / 漏洞管理产品成功的因素是双向的。风险管理商没有收集所需数据的方法和技能，因而难以与那些拥有这些方法和技能的手竞争。而漏洞管理厂商则是不愿进入到一个规模和收入有限的市场。因此，可行的解决方案是两者需要依靠定制的方式结合在一起。

配置管理是另一个漏洞管理可以发挥重要作用的领域。扫描系统可用于为配置管理数据库（CMDB）收集原始数据（通过扫描系统），一些漏洞管理产品非常适合监测多条配置项的变化，类似于合规性监测功能。此间主要的障碍是需要跨领域知识，要从 ITIL 的角度开发和管理一个 CMDB，这两个领域对漏洞和修复的理解差异很大。

10.3 跨技术机会

能为公司创造价值的其他更为技术性的方法是存在的。在安全领域，向 IPS 提供漏洞数据就是一个机会。对于 IPS 来说，用于发现攻击的高速检测网络流量将是一个资源瓶颈，因为这些设备不断被调用来检测网络是否存在成百上千种可能的攻击，并确定是否阻断该网络流量，因此设备性能非常重要。虽然技术已经在以前的 IDS 基础上有了巨大进步，但延迟和吞吐量仍是一个问题，尤其是当再购买一台新设备的开销巨大时。IPS 必须在多个层上通过协议将数据分包，或将数据帧组装起来以分析其内容，判断是否存在攻击。这项工作非常繁重，特别是在不能造成网络延迟的情况下。

此外，为简化部署而对带宽性能的更高要求也给 IPS 厂商造成了压力。公司想用一种技术检测整个网络上的所有攻击，这经常需要在接入点或核心交换机上安装某种设备，使大量网络流量经过 IPS。为提高这种设计的性能，厂商尝试利用群组设备、增加负载均衡器，并尽量使每个设备组达到最大工作量。

一切到位以后，成本和复杂性开始提高了，为此，安全分析员开始筛选风险评估数据，以确定哪些攻击签名需要检测，哪些不需要。如果系统已经有完备的防御措施，不存在相应漏洞攻击的情况，是无需进行检测的。如果你所有的 Web 服务器对某些特定攻击都没有漏洞，那就可以从 IPS 中将这些攻击检测去掉以减小工作量，从而降低延迟。与可以省略的大量扫描检测工作相比，将无需检测的漏洞攻击数据送到 IPS 是一件相对轻松的事。

但也有障碍，如标准化。解决方法是普通漏洞检测（CVE）、OVAL，以及其他标准。如果 IPS 签名和漏洞核查可以引用相同的标识符，那么就可以定期升级了。另外一个挑战是要确保所传输的只是适应于该网络段的漏洞数据。漏洞数据可能有几百兆，但适应于该 IPS 功能的可能只有几 K，应制定交换 IPS 地址范围或漏洞适用主机类型的标准方法，这类似于两种技术间的双向握手。

10.3.1 代理

在写这本书时，由于安装和维护代理的难度大，市场对漏洞管理行业使用代理的兴趣开始下降，但兴起了一种用于终端安全程序的安全代理。当执行网络访问控制（NAC）的预许可程序时，如果系统请求访问，控制器可以提供一种兼容的、小型的临时性代理，即众所周知的即时代理。该代理在 NAC 过程中被装入并执行，用于验证请求连接的系统的合规性和安

全状态。

这一思路很简单，但 NAC 架构策略就复杂得多了，变化也较大。以下说明足以限定漏洞管理的范围：

- 1) 主机连接到网络访问交换机，该交换机只允许在隔离 VLAN 上进行通信。
- 2) NAC 控制器扫描隔离 VLAN 上的系统合规性，可以采用主动扫描，但效果有限；也可以使用即时代理，以进行更为深入的分析。
- 3) 主机的安全状态被回传给控制器以验证与策略的一致性。
- 4) 如果主机符合安全要求，且含有证书，则被允许进入到 VLAN，代理会自动从主机内存中移除。如果不符合要求，该代理会向用户发送信息，告知用户该修正哪些地方。如果需要补丁和配置项，修复可以自动化完成。那么，程序返回到步骤 2)。

与其他技术相比，代理具有很大的优势，因为代理可以在连接网络之前对主机进行漏洞和合规性评估，这里使用的是最后的已知策略，如果连接策略未有变化，那么就不需要再做什么了，允许访问。该方法可能与前面的例子不同，但从概念上讲是相同的。

10.3.2 补丁管理

补丁管理是集成漏洞管理技术的最理想的应用，二者非常适合，因为大量套装软件漏洞是通过补丁来修复的，这甚至也可能适用于定制的、内部软件。

考虑到漏洞识别之后的一系列程序（创建更改标签、手动执行变更（打补丁）、关闭标签）的复杂性，自然会想到补丁管理的自动化。

当然，考虑到对系统的影响，并不是所有补丁都能自动安装，管理员应制定管理规则。管理员应根据系统或应用类型或同时根据二者确定哪些补丁可以自动安装。例如，网络浏览器加装补丁的风险一般较低，但 Java 服务器则较高，因为功能变化会引起目前应用的中断。

总的来说，这种技术集成有可能实现增加价值，节省时间。

10.3.3 应用渗透测试

漏洞扫描通常是渗透测试的前提，扫描得出的漏洞列表提供了可以对应用程序进行渗透测试的目标项。攻击通常是针对应用本身，而不是底层的基础设施软件。例如，一旦扫描结果列表显示某网络服务器在 TCP 80 端口上应答，就可以直接去检查数据，因为此时检查网络服务器软件的漏洞虽然有趣但意义不是太大。

渗透测试的下一步骤是尝试各种应用弱点攻击，如 XSS（跨站脚本）和 SQL 注入等，谁知道最多的通用应用软件编码错误及其产生的漏洞，谁就能迅速识别和利用这些错误和漏洞。

以下是其他一些容易被利用的漏洞：

- 格式字符串：在系统未提供用户输入过滤功能的情况下，可以用此方法向 Web 表格中的输入栏填充一定顺序的字符串，这些字符串能引起某种编程语言执行其他特定的功能，从而泄露涉密信息。
- 用户名破解：这是一种暴力技术，可以使用一个自动化工具很容易地进行。通过输入多个用户，并查看错误响应的不同，攻击者可以确定什么用户名是有效的。得到用户名后，攻击者可以再破解密码。

这些漏洞很容易被修复，但需要对安装了应用程序的各个网段的各台电脑进行全面检测，检测过程非常耗时，因此使用漏洞扫描器自动地执行这一过程是非常好的选择。

10.4 流程缺陷

漏洞管理流程的标准化要达到工具、人员的例行公事化的效果，还有很长的路要走。各公司肯定都有自己独特的方法去适应内部交叉流程。但漏洞管理流程的结果太难预测，期望的结果很难得到确认。以下是一些并不被经常问到的问题，但它们确实是急需解决的：

- 该流程会为组织增加多少安全性？
- 关键系统上的漏洞修复时间是多长？
- 修复每个漏洞的平均成本是多少？
- 我如何能知道流程在以最佳方式运行？
- 我公司漏洞管理绩效与业界其他公司相比情况如何？

这些问题都很难回答。很多其他技术领域很清楚哪些衡量标准有意义，相关组织会收集信息支持最佳的性能标准，但怎样才是最佳？最佳的确切含义是什么？最佳就是能熟练操作一个程序，并使产生的积极结果符合利益相关者、管理人员和商业伙伴的最重要的需求。这一定义非常适用于绩效管理。Robert S. Kaplan 和 David P. Norton 在《Harvard Business Review》中介绍了综合评价卡方法。

综合评价卡方法就是将组织（各程序）当前的行为与其长期策略挂钩。在漏洞管理中，我们可以试着将识别和修复漏洞的各程序与组织的策略和目标挂钩。在各种不同的情况下，需要一个不同的策略。风险管理功能还应有其他高级管理者的参与，从一个更高的层面来确认哪些策略适用于取得组织的安全目标和处理组织关心的安全问题，这些都会反映在综合评价卡上。

考虑一下上述 5 个问题中的第一个：该流程会为组织增加多少安全性？漏洞管理项目如何向高级管理员回答这个问题？这是一个开放问题，可以有很多种答案。这些答案可以以策略示意图的方式呈现。我们从按主题划分的组织策略目标示意图入手，下图就是一个示意图样例。

主 题	目 标	衡量方法	目 标	方 法
财务	赢利	较低的 IT 财务暴露风险	每年减少 25% 的关键漏洞	探测优化、自动修复
客户	个人数据低风险	快速修复关键漏洞	平均修复时间在一周内	优先处理关键漏洞
内部	将员工盗窃数据风险降到最低	限制授权员工访问敏感数据	策略合规性达到 5%	探测和修复特权扩大漏洞
学习	员工对安全的支持更有力	全面安全意识培训	完成 98% 的安全意识培训	基于计算机的培训

该示意图显示了财务、客户、内部和学习四个主题。在客户主题中，其战略目标是将客户个人数据的风险降到最低，很明显，这会极大影响客户及公司形象。这在漏洞管理中的衡量方法是监测关键漏洞的修复速度，成功的标志是平均修复时间在一周以内，而方法是优先处理关键漏洞。

虽然很多设备供应商都提供了衡量方法，但几乎没有哪种方法是能在整个行业通用的，即能判断最好的公司能够提供衡量标准 X、Y 和 Z 来确切地回答以上五个问题。有时，组织需要其他定制的衡量标准来应对自己业务模型中的具体问题。例如，一家处理医疗记录的公司需要的是衡量病人数据系统的安全状态的标准，而不是衡量支付信息系统的标准。简而言之，完全不同的内部组织评分卡要求也不同。有时，一些公司会请专门机构如 SANS 或 NIST 来担任其安全领导层，这些机构会制定一些标准或参考条件，用于衡量漏洞管理项目的总体性能。而现在，应制定的是符合各种特定情况要求的标准。

漏洞管理程序需要首先按要求界定系统成果，随后是衡量各步骤在实现上述效果中的有效性，以此制定正确的数据收集与汇报机制。

10.5 运行环境的变化

漏洞管理所在的技术环境变化迅速。节能、分布式计算、远程办公及服务器整合趋势正在为系统设计者和产品定价模型带来挑战，也使漏洞的识别更为复杂。

10.5.1 省时

对大型网络来说，主动扫描非常耗时。要想找到某台主机，仅仅使用简单的 ICMP（网间报文控制协议）应答，结果是不可信的。因此，还需采用其他手段，如向各种常用端口发送 TCP-SYN 数据包。有时，即使使用了这些方法，搜寻了成百上千个 IP 地址，也仍有可能找不到目标主机，这一般是网络寻址机制的问题。

需要针对已知存在的系统使用向中央管理系统查询准确数据的技术方法，以加速扫描过程。在微软环境下，WMI 可被用于识别 Windows 主机，其 IP 地址及所运行的服务，从而极大加快扫描过程。

也可以结合使用其他方法，如动态主机配置协议（DHCP）数据库和配置管理系统等，这些信息源不应被看做会降低扫描的准确性，相反，它们能提供迅速发现目标并选择合适的认证机制的手段，从而实现成功审计。

10.5.2 节电

如第 7 章所述，在非工作时间执行漏洞审计时，很多系统由于已关闭或处于节电模式而无法成功进行。如果在 WAN 链接上执行主动审计，即使开启了网络唤醒功能，那些主机也不可能被唤醒。这是因为只有在本地网段上向特定 MAC 地址发送专门的数据包才会激活网络唤醒功能。远程的 TCP/IP 连接无法携带这种信息，这使我们只能：

- ❑ 安装一个本地扫描器，配置成在审计前 30 秒向目标发送此种数据包；
- ❑ 在目标主机上运行可按时唤醒主机接受审计的漏洞管理代理；
- ❑ 在工作时间运行慢速、低带宽，资源消耗少的扫描，不过完成扫描可能需要几天或一周的时间；
- ❑ 协调 IT 人员在某个时间窗口保持所有目标机开启，这最耗电，无疑也更可取。

最有效的方法是在本地网络上安装主动扫描设备，但这有两个困难，一是供应商会因额

外增加设备而收费，但该设备使用率又不高，因此会降低该技术的性价比；同时将一件设备通过各国海关运送到数十个国家也不太现实。

另一种方法是供应商为自己的产品创建虚拟机，以安装在达到最低硬件要求的主机上，这些虚拟的漏洞扫描器可以通过电子方式传送，免除了政府干涉，保持了性价比。供应商也可能会以为虚拟机提供支持和升级服务为由收取安装费用，但这相比于第一种方法安装硬件设备所带来的贬值问题和维护开销要低得多。

10.5.3 分布式计算

分布式计算为我们带来了挑战，而漏洞管理产品供应商并没有对这一问题提供支持。VPN 连接、更改安全区、Web 服务、软件即服务模式（SaaS）增加了漏洞评估的难度，各供应商显然对这些 IT 环境的改变尚无充分的准备。

VPN 连接很好理解，它会对非基于代理的漏洞评估方法形成挑战。持续扫描 VPN IP 空间不可取，因为即使并未建立通道也会增加基础设施负担。公司网络的复杂性则可能会导致下面这种情况：某个目标建立了连接到某个网段的通道，而该网段的扫描仪所使用的参数与该目标所在网络完全不同，这一现象即为变更安全区。有时，这种方法是可取的，符合本地站点安全策略的；但在其他情况下可能是不合适的，例如当该目标断开通道连接重新回归到所在网络时，可能与所在网络不一致。

解决该问题需要能够将一条安全策略适用于一个目录组或计算机名称模式，例如，一个会计计算机用户位于目录中的会计组，则会被要求遵守严格的合规性策略并被分配相应的扫描参数。当该用户到了一个很远的地方，并在就近的一个营业部连接 VPN 网关时，该计算机所在群组会在漏洞扫描中触发会计群组策略参数的应用。

同样，当对一个 VPN IP 地址池进行漏洞评估时，发现程序会被执行。在此过程中，通道会被持续建立和摊销。我们已经知道了连接状态和所有连接到该 VPN 网关上的机器，为什么还要执行发现过程呢？因为这里需要一些简单的互操作，以使 VPN 网关可以向漏洞管理系统通报目标与网络间的连接。然后，漏洞扫描仪会判断是否应在此时此地执行审计。

网络服务也是被各供应商忽略的一个领域。业务应用，或更抽象一点，自动业务功能已经不再建立在单台主机上了。各种网络服务散布在世界各地，安全等级各不相同，通过功能的联合提供所需的业务服务。单台主机的风险从技术上可以评估，但业务应用的风险却不能。这是因为业务应用是由各个不同的目标提供的服务组合而成的，应该有一种方法

来将各个目标定义为整体应用的某个部分。然后，风险评估报告需计入所有这些服务的安全态势。

相反，提供网络服务的目标的漏洞很容易评估，但其风险却不然，因为没有对该目标基于主机的分类级别使用目标的服务进行风险分析。例如，一台提供数据压缩的网络服务主机处理来自 A 系统和 B 系统的数据，A 系统的数据可从公共渠道获得，B 系统的数据则是关键业务的机密信息。如果该网络服务主机存在漏洞，那对系统 A 和系统 B 的真正风险是什么？如果漏洞被用于发起拒绝服务攻击（DoS），使得网络服务不可得，所造成的风险是什么？漏洞管理系统必须能够关联这些关系，并生成风险管理和漏洞管理流程所需的报告。

网格计算属于分布式计算，也为漏洞管理工具带来了挑战。由于处理过程是在多台非特定的网格成员机器上进行的，特定业务功能所受到的威胁是很难确定的。漏洞管理系统需要知道网格，这样才能评估某个网格的整体风险。此外，漏洞管理系统还应提供网格状态报告，因为网格的计算机成员的状态是不明确的，所以报告应该是在应用级别或类别层次上的状态描述。

另一种对漏洞管理行业的分布式计算挑战是软件即服务（SaaS）模型。目前对作为服务提供的软件还没有进行漏洞研究或漏洞评估的有效方法。如果进行漏洞评估可能会威胁服务提供商的安全，更大可能是破坏服务。

目前缺乏广为接受的标准化的方法向网络服务用户传达漏洞状态，使之可以评估其机密数据承受的风险。如果没有这种方法，很多企业用户不可能接受广泛使用这种服务。设想一下，如果跨国银行的员工使用供应商在网上提供的软件，数据泄露的危险是完全不可预期的，而且也不会有审计结果用于验证合规性。

在这一标准中，漏洞的严重程度应当在利用和整体影响所需的访问类型上下文中进行考虑。该评分方法类似于 CVSS，只是信息中不包含具体漏洞。该评分方法会根据访问利用方式将漏洞划分等级。其划分结果会类似于下面的例子（见下表）：

访问方式	DoS	控制	数据访问	信息
远程	5	3	4	2
本地	3	3	1	3

在第一行中，该例子显示在远程访问漏洞中：DoS 风险为高、系统控制风险为中、数据访问风险为中高，信息泄漏风险为中低。第二行显示的是本地访问漏洞的各种风险等级。

10.6 报告

目前业界尚无成熟的漏洞管理案例以帮助各公司优化其当前流程和更快地识别广泛的趋势和威胁。要做到这一点，缺少信息当然是不行的，本书前面讲过，系统要能创建多个报告。但从销售商手中购得的系统多数都不提供报告自动生成功能。很显然，向没有漏洞管理的公司出售新系统需要有基本的报告功能，基本的就足够，因为出售高级报告功能似乎也收益甚少，一般只对保持长期客户有用。我认为可以把这作为使自己区别于其他产品的一个卖点。除发现漏洞和基本的报告功能外，程序、部署和项目管理都是很好的专业服务机会。

10.7 服务水平协议

虽然 IT 服务管理的接受程度越来越高，但购买服务的客户并不多，这可能是由于缺乏需求。但鉴于一些漏洞危害程度很高，需要更快速地响应，供应商会争相承诺开发和发布新漏洞检查服务以达到某种水平。这种服务水平至少能做到在供应商已经开始攻克超过某一 CVSS 分值的新漏洞的 24 小时内发布报告，也应包括在发现漏洞的一周内报告其开发工作进展，以决定是否在接下来的一周或晚些时候发布检测信息。

这种方法的价值在于风险管理员可以据此决定是否应人工识别和修复关键主机上的漏洞，还是等待自动检测程序进行审计和检查，这是业界尚未解决的一个风险黑洞。

另一种需要供应商考虑的服务水平衡量方式是支持电话最初响应时间，这属于问题管理，是一项基本功能，如果关键性漏洞探测系统没有从问题中恢复的可靠性方法，那么关键风险识别工具的作用就大打折扣了。

10.8 小结

漏洞管理正在进入成熟期。其最初的重点是漏洞识别和修复，而现在程序的卓越性能已经成为下一个目标。漏洞管理可以实现卓越性能，并能集成进其他风险管理和 IT 管理程序，为此，供应商、标准组织和用户必须致力于发现漏洞管理的局限和存在的扩展机会。和其他准则一样，漏洞管理必须要集成到其他组织流程中，并支持该组织流程。

供应商在解除 IT 管理员和风险分析师的技术担心方面卓有成效。漏洞工具市场尚未接近饱和，而在已安装了这些工具的环境下的漏洞管理已发展成熟。在很多方面，漏洞管理服务于公司基本业务和政府需要的潜力尚待发挥，产品的灵活性远不能满足未来的需要，发挥技术和程序的最大潜能需要企业的眼光和承诺。本书所讲的内容只是拉开了一个序幕，风险管理还将面对现实世界里不断出现的新型挑战。