

# 《Rootkit：系统灰色地带的潜》

## 图书基本信息

书名：《Rootkit：系统灰色地带的潜伏者》

13位ISBN编号：9787111441786

10位ISBN编号：7111441788

出版时间：2013-10-1

出版社：机械工业出版社

作者：Bill Blunden

页数：600

译者：姚领田,蒋蓓,刘安,李潇

版权说明：本站所提供下载的PDF图书仅提供预览和简介以及在线试读，请支持正版图书。

更多资源请访问：[www.tushu000.com](http://www.tushu000.com)

# 《Rootkit：系统灰色地带的潜》

## 内容概要

《Rootkit：系统灰色地带的潜伏者》共分四部分。第一部分（第1~6章），全新阐释rootkit本质、rootkit与反取证关系、安全领域态势，以及反取证技术的策略、应对建议和攻击优势。之后，从硬件、软件（系统）、行业工具和内核空间方面介绍rootkit调查过程和利用反取证技术破坏调查过程的策略，使你对取证和反取证有全新了解。第二部分（第7~8章），主要介绍rootkit如何阻止磁盘分析和可执行文件的分析，而调查人员如何利用有效的工具和策略来分析辅助存储器（例如磁盘分析、卷分析、文件系统分析以及未知二进制分析）中可能留下的rootkit痕迹，并对内存驻留和多级释放器技术及用户态Exec（UserlandExec）理念进行了深入剖析。第三部分（第9~15章）主要详解攻击者利用rootkit破坏数据收集过程和造成“一切安好”的假象的前沿实用策略：阻止在线取证、内核模式策略、更改调用表、更改代码、更改内核对象、创建隐秘通道和部署带外rootkit。第四部分（第16章），高屋建瓴地重新总结了rootkit的核心策略，以及如何识别隐藏的rootkit、注意事项和如何处理感染等。

# 《Rootkit：系统灰色地带的潜》

## 作者简介

Bill Blunden，资深计算机安全专家，从事相关研究10余年，对rootkit有非常深入的研究。目前从事网络安全设备代码和ERP中间件的相关工作。活跃于计算机安全类社区，常与计算机安全领域多名世界级安全专家交流探讨。在学术生涯中走过不少弯路，因此对计算机安全有异于常人的观察角度和体会。

## 书籍目录

译者序

献给“孙悟空”

前言

第一部分基础知识

第1章 清空思想

1.1不速之客

1.2提炼一个更确切的定义

1.2.1攻击循环

1.2.2rootkit在攻击循环中的角色

1.2.3单级释放器与多级释放器

1.2.4其他部署方法

1.2.5确切的学术性定义

1.2.6不要混淆设计目标与实现

1.2.7rootkit技术--力量倍增器

1.2.8金·费尔比式比喻：破坏与毁坏

1.2.9为何使用隐身技术？rootkit不能被发现吗

1.3rootkit不等于恶意软件

1.3.1感染源

1.3.2广告软件和间谍软件

1.3.3僵尸网络的兴起

1.3.4引入：愚人飞客病毒

1.3.5恶意软件与rootkit

1.4谁在开发和使用rootkit

1.4.1市场营销

1.4.2数字版权管理

1.4.3不是rootkit，而是种功能

1.4.4法律实施

1.4.5商业间谍

1.4.6政治间谍

1.4.7网络犯罪

1.4.8谁开发了颇具艺术感的rootkit

1.4.9rootkit的道德性

1.5慑魄惊魂：战场伤员分类

1.6总结

第2章 反取证综述

2.1事件响应

2.1.1入侵检测系统（和入侵防御系统）

2.1.2异常行为

2.1.3发生故障

2.2计算机取证

2.2.1rootkit不是隐身的吗？为什么还要进行反取证

2.2.2假定最糟糕案例的场景

2.2.3取证技术分类：第一种方法

2.2.4取证技术分类：第二种方法

2.2.5在线取证

2.2.6当关机不再是种选择

2.2.7关于拔掉电源插头的争论

- 2.2.8崩溃转储或者不进行崩溃转储
- 2.2.9事后检查分析
- 2.2.10非本地数据
- 2.3AF策略
  - 2.3.1数据销毁
  - 2.3.2数据隐藏
  - 2.3.3数据转换
  - 2.3.4数据伪造
  - 2.3.5数据源消除
- 2.4AF技术的总体建议
  - 2.4.1使用定制工具
  - 2.4.2低且慢与焦土策略
  - 2.4.3避免特定实例攻击
  - 2.4.4使用分层防御
- 2.5不明身份者具有优势
  - 2.5.1攻击者能够专注于攻击
  - 2.5.2防御者面临制度性挑战
  - 2.5.3安全是一种过程（而且还是一种令人讨厌的过程）
  - 2.5.4持续增加的复杂度
- 2.6总结
- 第3章 硬件概述
  - 3.1物理内存
    - 3.2IA-32内存模型
      - 3.2.1平面内存模型
      - 3.2.2分段内存模型
      - 3.2.3操作模式
    - 3.3实模式
      - 3.3.1案例研究：MS-DOS
      - 3.3.2这不是浪费时间吗？为什么学习实模式
      - 3.3.3实模式执行环境
      - 3.3.4实模式中断
      - 3.3.5分段和程序控制
      - 3.3.6案例研究：转储IVT
      - 3.3.7案例研究：用TSR记录击键
      - 3.3.8案例研究：隐藏TSR
      - 3.3.9案例研究：为TREE.COM命令打补丁
      - 3.3.10小结
  - 3.4保护模式
    - 3.4.1保护模式执行环境
    - 3.4.2保护模式分段
    - 3.4.3保护模式分页
    - 3.4.4地址扩展分页
    - 3.4.5进一步研究页表
    - 3.4.6进一步研究控制寄存器
  - 3.5实现内存保护
    - 3.5.1通过分段实现保护
    - 3.5.2界限检查
    - 3.5.3类型检查
    - 3.5.4特权检查

3.5.5受限指令检查

3.5.6门描述符

3.5.7保护模式中断表

3.5.8分页保护

3.5.9总结

第4章 系统概述

4.1Windows系统下的物理内存

4.1.1失落的大陆（内存）

4.1.2Windows如何使用物理地址扩展

4.1.3页、页帧、页帧号

4.2Windows下的分段和分页

4.2.1分段

4.2.2分页

4.2.3线性地址到物理地址的转换

4.2.4一个更快的办法

4.2.5关于EPROCESS和KPROCESS的讨论

4.3用户空间和内核空间

4.3.14GB调优（4GT）

4.3.2各得其所

4.3.3跨越篱笆

4.3.4用户空间剖析

4.3.5内核空间动态分配

4.3.6地址窗口化扩展

4.3.7PAE、4GT和AWE的对比

4.4用户模式和内核模式

4.4.1执行方式与执行位置

4.4.2内核模式组件

4.4.3用户模式组件

4.5其他内存保护特征

4.5.1数据执行保护

4.5.2地址空间布局随机化

4.5.3GS 编译选项

4.5.4SAFESEH链接器选项

4.6本机API

4.6.1中断向量表的发展

4.6.2进一步研究中断描述表

4.6.3通过中断进行系统调用

4.6.4SYSENTER指令

4.6.5系统服务调度表

4.6.6枚举本机API

4.6.7Nt\*()系统调用与Zw\*()系统调用

4.6.8系统调用的生命周期

4.6.9其他内核模式例程

4.6.10内核模式API文档

4.7引导过程

4.7.1BIOS固件启动

4.7.2EFI固件启动

4.7.3Windows启动管理器

4.7.4Windows启动加载器

- 4.7.5初始化执行体
- 4.7.6会话管理器
- 4.7.7wininit.exe
- 4.7.8winlogon.exe
- 4.7.9启动过程概括
- 4.8设计决策
  - 4.8.1藏在人群中：类型0
  - 4.8.2主动隐藏：类型1和类型2
  - 4.8.3跳出边界：类型3
  - 4.8.4前景展望
- 第5章 行业工具
  - 5.1开发工具
    - 5.1.1诊断工具
    - 5.1.2磁盘映像工具
    - 5.1.3更快速救灾：虚拟机
    - 5.1.4工具综述
  - 5.2调试器
    - 5.2.1配置CDB.exe
    - 5.2.2符号文件
    - 5.2.3Windows符号
    - 5.2.4激活CDB.exe
    - 5.2.5控制CDB.exe
    - 5.2.6有用的调试器命令
    - 5.2.7检查符号命令（x）
    - 5.2.8列举已加载的模块（!m和!mi）
    - 5.2.9显示类型命令（dt）
    - 5.2.10反汇编命令（u）
    - 5.2.11显示命令（d\*）
    - 5.2.12寄存器命令（r）
  - 5.3KD.exe内核调试器
    - 5.3.1使用内核调试器的不同方法
    - 5.3.2物理宿主机-目标机配置
    - 5.3.3准备硬件
    - 5.3.4准备软件
    - 5.3.5启动内核调试会话
    - 5.3.6控制目标机
    - 5.3.7虚拟宿主机-目标机配置
    - 5.3.8 有用的内核模式调试器命令
    - 5.3.9列举已加载模块命令
    - 5.3.10 ! process扩展命令
    - 5.3.11寄存器命令（r）
    - 5.3.12使用崩溃转储
    - 5.3.13方法1：PS2键盘技巧
    - 5.3.14方法2：KD.exe命令
    - 5.3.15方法3：NotMyFault.exe
    - 5.3.16崩溃转储分析
- 第6章 内核空间中的玄机
  - 6.1KMD模板
    - 6.1.1内核模式驱动程序：全局概览

- 6.1.2 WDK 框架
  - 6.1.3 真正最小的 KMD
  - 6.1.4 处理 IRP
  - 6.1.5 与用户模式代码通信
  - 6.1.6 从用户模式发送命令
  - 6.2 加载内核模式驱动程序
  - 6.3 服务控制管理器
    - 6.3.1 在命令行使用 sc.exe
    - 6.3.2 编程使用 SCM
    - 6.3.3 注册表踪迹
  - 6.4 使用导出驱动程序
  - 6.5 综合利用内核中的漏洞
  - 6.6 Windows 内核模式安全
    - 6.6.1 内核模式代码签名
    - 6.6.2 KMCS 的应对措施
    - 6.6.3 内核补丁保护
    - 6.6.4 KPP 的应对措施
  - 6.7 同步
    - 6.7.1 中断请求级
    - 6.7.2 延迟过程调用
    - 6.7.3 实现
  - 6.8 总结
- 第二部分 事后分析
- 第 7 章 阻止磁盘分析
- 7.1 事后调查：概述
  - 7.2 取证副本
  - 7.3 卷分析
    - 7.3.1 Windows 下的存储卷
    - 7.3.2 手工分析卷
    - 7.3.3 应对措施：破坏分区表
    - 7.3.4 Windows 下的原始磁盘访问
    - 7.3.5 原始磁盘访问：突破常规
  - 7.4 文件系统分析
    - 7.4.1 恢复删除的文件
    - 7.4.2 恢复删除的文件：应对措施
    - 7.4.3 枚举可选数据流
    - 7.4.4 枚举可选数据流：应对措施
    - 7.4.5 恢复文件系统对象
    - 7.4.6 恢复文件系统对象：应对措施
    - 7.4.7 带外隐藏
    - 7.4.8 带内隐藏
    - 7.4.9 引入：FragFS
    - 7.4.10 应用层隐藏
    - 7.4.11 获取元数据
    - 7.4.12 获取元数据：应对措施
    - 7.4.13 改变时间戳
    - 7.4.14 改变校验和
    - 7.4.15 识别已知文件
    - 7.4.16 交叉时间差异与交叉视图差异



7.4.17识别已知文件：应对措施

7.5文件签名分析

7.6总结

第8章 阻止可执行文件分析

8.1 静态分析

8.1.1扫描相关人工痕迹

8.1.2验证数字签名

8.1.3转储字符串数据

8.1.4检查文件头

8.1.5反汇编和反编译

8.2破坏静态分析

8.2.1数据转换：加壳

8.2.2加壳：加密程序

8.2.3密钥管理

8.2.4加壳：压缩程序

8.2.5加壳：变形代码

8.2.6定制工具的需求

8.2.7关于加壳的争论

8.2.8数据伪造

8.2.9虚旗攻击

8.2.10数据源清除：多级加载器

8.2.11深度防御

8.3运行时分析

8.3.1运行环境

8.3.2手工与自动运行时分析

8.3.3手工分析：基本概要

8.3.4手工分析：跟踪

8.3.5手工分析：内存转储

8.3.6手工分析：捕捉网络活动

8.3.7自动化分析

8.3.8运行时复合分析

8.4破坏运行时分析

8.4.1跟踪的应对措施

8.4.2API跟踪：规避迂回补丁

8.4.3API跟踪：多级加载器

8.4.4指令级跟踪：攻击调试器

8.4.5断点

8.4.6检测用户模式调试器

8.4.7检测内核模式调试器

8.4.8检测用户模式调试器或者内核模式调试器

8.4.9通过代码校验和检测调试器

8.4.10关于反调试器技术的争论

8.4.11指令级跟踪：混淆

8.4.12混淆应用数据

8.4.13混淆应用代码

8.4.14阻止自动化

8.4.15应对运行时复合分析

8.5总结

第三部分 在线取证

## 第9章 阻止在线取证

### 9.1 在线取证：基本过程

### 9.2 用户模式加载器

#### 9.2.1 UML破坏现有的API

#### 9.2.2 关于加载器API模块的争论

#### 9.2.3 纵览Windows PE文件格式

#### 9.2.4 相对虚拟地址

#### 9.2.5 PE文件头

#### 9.2.6 导入数据节 (.idata)

#### 9.2.7 基址重定位节 (.reloc)

#### 9.2.8 实现独立的UML

### 9.3 最小化加载器踪迹

#### 9.3.1 数据节育：献给The Grugq的颂歌

#### 9.3.2 下一步：通过漏洞利用程序加载

### 9.4 关于独立PE加载器的争论

## 第10章 用C语言创建shellcode

### 10.1 用户模式shellcode

#### 10.1.1 Visual Studio工程设置

#### 10.1.2 使用相对地址

#### 10.1.3 寻找kernel32.dll：通往TEB和PEB的旅程

#### 10.1.4 扩展地址表

#### 10.1.5 解析kernel32.dll导出表

#### 10.1.6 提取shellcode

#### 10.1.7 危险空间

#### 10.1.8 构建自动化

### 10.2 内核模式shellcode

#### 10.2.1 工程设置：\$(NTMAKEENV)\makefile.new

#### 10.2.2 工程设置：SOURCES

#### 10.2.3 地址解析

#### 10.2.4 加载内核模式shellcode

### 10.3 特殊武器和策略

### 10.4 展望

## 第11章 更改调用表

### 11.1 在用户空间挂钩：IAT

#### 11.1.1 DLL基础

#### 11.1.2 访问导出例程

#### 11.1.3 注入DLL

#### 11.1.4 走查磁盘上PE文件的IAT

#### 11.1.5 挂钩IAT

### 11.2 内核空间的调用表

#### 11.3 挂钩IDT

#### 11.3.1 处理多处理器：方案#1

#### 11.3.2 裸例程

#### 11.3.3 关于挂钩IDT的问题

### 11.4 挂钩处理器MSR

#### 11.5 挂钩SSDT

#### 11.5.1 禁用WP位：技巧#1

#### 11.5.2 禁用WP位：技巧#2

#### 11.5.3 挂钩SSDT项

- 11.5.4SSDT示例：跟踪系统调用
- 11.5.5SSDT示例：隐藏进程
- 11.5.6SSDT示例：隐藏网络连接
- 11.6挂钩IRP处理程序
- 11.7挂钩GDT：安装调用门
- 11.8挂钩的应对措施
  - 11.8.1检查内核模式挂钩
  - 11.8.2检查IA32\_SYSENTER\_EIP
  - 11.8.3检查 INT 0x2E
  - 11.8.4检查 SSDT
  - 11.8.5检查IRP处理程序
  - 11.8.6检查用户模式钩子
  - 11.8.7解析PEB：  
第1部分
  - 11.8.8解析PEB：  
第2部分
- 11.9反应应对措施
  - 11.9.1假设最坏的案例
  - 11.9.2最坏案例应对措施#1
  - 11.9.3最坏案例应对措施#2
- 第12章 更改代码
  - 12.1跟踪调用
    - 12.1.1迂回实现
    - 12.1.2获取NtSetValueKey()的地址
    - 12.1.3初始化补丁元数据结构
    - 12.1.4对照已知签名核实原始机器码
    - 12.1.5保存原始序言和尾声代码
    - 12.1.6更新补丁元数据结构
    - 12.1.7锁定访问并禁用写保护
    - 12.1.8注入迂回
    - 12.1.9序言迂回
    - 12.1.10尾声迂回
    - 12.1.11事后总结
  - 12.2破坏组策略
    - 12.2.1迂回实现
    - 12.2.2初始化补丁元数据结构
    - 12.2.3尾声迂回
    - 12.2.4将注册表值映射到组策略
  - 12.3绕过内核模式API记录器
    - 12.3.1故障安全规避
    - 12.3.2更上一层楼
  - 12.4指令补丁应对措施
- 第13章 更改内核对象
  - 13.1隐形的代价
    - 13.1.1问题#1：陡峭的学习曲线
    - 13.1.2问题#2：并发性
    - 13.1.3问题#3：可移植性和指针运算
    - 13.1.4特有技术：DKOM
    - 13.1.5对象

- 13.2再访EPROCESS对象
  - 13.2.1获取EPROCESS指针
  - 13.2.2EPROCESS相关域
  - 13.2.3UniqueProcessId
  - 13.2.4ActiveProcessLinks
  - 13.2.5Token
  - 13.2.6ImageFileName
- 13.3DRIVER\_SECTION对象
- 13.4令牌对象
  - 13.4.1Windows授权
  - 13.4.2定位令牌对象
  - 13.4.3令牌对象中的相关域
- 13.5隐藏进程
- 13.6隐藏驱动程序
- 13.7操纵访问令牌
- 13.8使用No-FU
- 13.9内核模式回调
- 13.10应对措施
  - 13.10.1交叉视图检测
  - 13.10.2高级枚举：CreateToolhelp32Snapshot()
  - 13.10.3高级枚举：PID暴力
  - 13.10.4低级枚举：进程
  - 13.10.5低级枚举：线程
  - 13.10.6相关软件
  - 13.10.7域校验和
- 13.11反应应对措施
  - 13.11.1最好的防护：饿死对手
  - 13.11.2评论：超越双环模型
  - 13.11.3最后一道防线
- 第14章 隐秘通道
  - 14.1普通恶意软件通道
    - 14.1.1互联网中继聊天
    - 14.1.2对等通信
    - 14.1.3HTTP
  - 14.2最坏案例场景：截获所有数据内容
    - 14.2.1协议隧道
    - 14.2.2DNS
    - 14.2.3ICMP
    - 14.2.4外围设备问题
  - 14.3Windows TCPIP栈
    - 14.3.1Windows Sockets 2
    - 14.3.2原始套接字
    - 14.3.3Winsock内核API
    - 14.3.4NDIS
    - 14.3.5不同任务使用不同的工具
  - 14.4DNS隧道
    - 14.4.1DNS查询
    - 14.4.2DNS应答
  - 14.5DNS隧道：用户模式

## 14.6 DNS隧道：WSK实现

### 14.6.1 初始化应用程序的上下文

### 14.6.2 创建内核模式套接字

### 14.6.3 确定本地传输地址

### 14.6.4 绑定套接字与传输地址

### 14.6.5 设置远程地址（C2客户端）

### 14.6.6 发送DNS查询

### 14.6.7 接收DNS应答

## 14.7 NDIS协议驱动程序

### 14.7.1 创建并运行NDISProt6.0示例

### 14.7.2 客户端代码概要

### 14.7.3 驱动程序代码概要

### 14.7.4 Protocol\*()例程

### 14.7.5 缺失的特征

## 14.8 被动的隐秘通道

## 第15章 转到带外

### 15.1 附加处理器模式

#### 15.1.1 系统管理模式

#### 15.1.2 流氓管理程序

#### 15.1.3 白帽成员对策

#### 15.1.4 流氓管理程序与SMM rootkit

## 15.2 固件

### 15.2.1 主板BIOS

### 15.2.2 ACPI组件

### 15.2.3 扩展ROM

### 15.2.4 UEFI固件

## 15.3 远程管理设施

## 15.4 不太明显的备用方案

### 15.4.1 板载闪存

### 15.4.2 电路级伎俩

## 15.5 总结

## 第四部分 结束语

## 第16章 rootkit之道

### 16.1 核心策略

#### 16.1.1 尊重你的对手

#### 16.1.2 五指穿心掌

#### 16.1.3 忍耐强行夺取的欲望

#### 16.1.4 研究你的目标

### 16.2 识别隐藏之门

#### 16.2.1 对付专有系统

#### 16.2.2 监视内核

#### 16.2.3 重要特点：硬件是新软件

#### 16.2.4 充分利用现有研究

### 16.3 建筑领域的训诫

#### 16.3.1 首先加载，深度加载

#### 16.3.2 为自主性而奋斗

#### 16.3.3 Butler Lampson：策略与机制分离

## 16.4 设计rootkit

### 16.4.1 隐身与开发努力

## 《Rootkit：系统灰色地带的潜》

16.4.2使用定制工具

16.4.3稳定性很重要：致力于最佳实践

16.4.4逐步提高

16.4.5容错移转：自我修复的rootkit

16.5处理感染

## 《Rootkit：系统灰色地带的潜》

### 精彩短评

- 1、有史以来翻译的最烂的一本书
- 2、难得的一本关于rootkit的专业书籍，几年前利用它的英文版走入rootkit的世界，终于有了英文版，而且还是新版，翻译的也不错。赞一个。
- 3、很不错的一本书。不过在看的时候得结合一些个人的理解，死抠字面意义是很难懂的。
- 4、力荐
- 5、不错入门

## 《Rootkit：系统灰色地带的潜》

### 精彩书评

1、首先，我不是说这本书是本烂书。作者也许功力深厚，作者也许是个大牛，可惜选错了译者！从目录和大概看，作者的本意是给入门的rootkit编写者看的，内容也不错，还介绍了一些rootkit的前沿发展，可惜选错了译者！翻译文字极尽晦涩拗口之能事，甚至还不如谷歌翻译的好不好！再看看译者就明白了，群译啊！群译啊！群译啊！群译啊！16章的书有姚领田，蒋蓓，刘安，李潇，张振项等共计24人分包翻译，翻译不好谁也不担责任，果然是三个和尚没水喝啊！评价：极差。性价比极低，极度不建议购买。如果英文能力不错还是建议去看原版吧！



# 《Rootkit：系统灰色地带的潜》

## 版权说明

本站所提供下载的PDF图书仅提供预览和简介，请支持正版图书。

更多资源请访问：[www.tushu000.com](http://www.tushu000.com)