基于 UEFI 的安全模块设计分析

霍卫华
（中国电子科技集团公司第三十研究所，四川 成都 610041）

【摘 要】传统 BIOS 是用汇编语言编写的，一般固化在主板的 CMOS 芯片中，利用电池来维持 BIOS 的硬件配置信息，维护和发展都很困难。文章简单分析了基于传统 BIOS 技术的安全模块软件系统设计和基于 UEFI 技术的安全模块软件系统设计，针对两种体系的软件维护和安全性进行了对比分析。

【关键词】基本输入输出系统；统一可扩展固件接口；应用程序接口
【中图分类号】TP39 【文献标识码】A 【文章编号】1009-8054(2008)07-

A Secure Module Analysis of UEFI

HUO Wei-hua
(No.30 Institute of China Electronic Technology Group Corporation, Chengdu Sichuan 610041, China)

【Abstract】Traditional BIOS is complied by assembly language, and generally firmwared in the coms chip of the main board, maintaining the hardware configuration information of BIOS by battery, and thus resulted in difficulty for maintenance and development.This paper gives a brief analysis on secure module's software system of the secure module based on traditional BIOS and UEFI, the project maintenance and security is contrasted and discussed, between traditional BIOS and UEFI, and all this can be taken as a reference for the designers in secure module design.

【Keywords】BIOS; UEFI; API

0 引言

计算机的体系结构上强调计算效率和易用性，促进了其产业的发展，但却给安全性留下了巨大的隐患。常用的计算机从芯片、BIOS（Basic，即基本输入输出系统）到操作系统（OpeSy，简称）等核心技术都采用国外的技术，尤其是伴随着计算机产业发展余年，负责软件与硬件接口的 BIOS 核心技术对中国信息安全领域发展来说，如同雷池一般无法介入。

国内传统的安全模块或称密码模块等通常只能在操作系统运行后进行内核层或应用层的安全保护，属被动式防御，这无疑为中国的信息安全产品遗留了严重的安全隐患。

1 传统安全模块设计

1 传统 BIOS 简介

BIOS 是计算机硬件的低级别抽象，它主要负责计算机硬件进入操作系统之前的初始化工作，在系统启动时主要是检查硬件与系统设备是否正常。传统 BIOS 是用汇编语言编写的，一般固化在主板的 CMOS 芯片中，利用电池来维持 BIOS 的硬件配置信息，维护和发展都很困难。Intel 的 EFI 首席工程师 Mark 在 IDD oF 上指出了 BIOS 许多缺点。例如，BIOS 的设计者无法预计 BIOS 可以使用多久，因此也就在设计上留下了隐患；生产商可随心所欲地修改和配置；用户通过 BIOS 配置系统和诊断问题时，往往起不了太大的作用。传统的 BIOS 具有很多缺陷，通常，BIOS 的配置十分繁琐，对于初学者而言操作十分困难，要想支持新硬件设备，就需要不断升级 BIOS 的版本，而该操作也是十分困难的。BIOS 连接着硬件与操作系统，这个连接没有标准，不同生产商之间的 BIOS 也不相同。

1 总体框架

通常，安全模块的开发除硬件设备的基础开发外，软件

系统的开发是基于操作系统内核层和应用层进行的。在操作系统的支撑基础上，对安全模块的硬件驱动程序、安全服务接口和安全应用系统进行相应的定制开发，安全模块系统框架如图 所示。

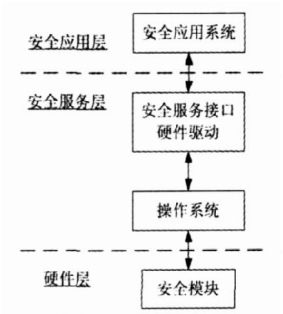

图1 传统安全模块总体框架

1 硬件驱动程序

硬件设备的驱动程序通常采用 V 模式或 W 模式设计开发。 W D M （ W i n d ，即 w W i D驱动程序 vw 模型 ）是] W i n 和 oW wi sn d 及以后版本使用的 O 驱动程序设计规范，也是主流的驱动模式。使用 W 使得 M 硬件驱动程序更加稳定，让操作系统对硬件控制更加有效。

1 安全服务接口

安全服务接口分内核层和应用层两种方式，其中以应用层居多。内核层安全服务接口多以内核驱动程序方式提供给安全应用系统使用，通常，设计采用厂家自定义的 A 函数访问列表进行安全模块的访问控制。应用层安全服务接口多以 A P I （ A p p l i c a t 库方式提供给安全 o 应用系统使用，它支持任何 W i n d 的编程软件使用。D 安全应用系统通过调用该 A 实现对安全模块的访问控制。

A 是以动态链接库 （ D y n a m i ，简称 ID 的 n 磁盘分区 方式提供，通常，设计采用厂家自定义的 A 或部分标准接口规范进行安全模块的访问控制。这里说的标准接口规范包括 R 的S A K C S # 1 1 （ P u b l i c 和 K e C 、y 北桥、 C 内存和硬盘的初始化工作；紧接着载入 S t a n d a 微软的 C S P （ C r y p t o g 等。a p h i c E 驱动程序，当 iE 驱动程序被载入运行后，系统便具有控制所有硬件的能力。在 E 规范中引入了 G 磁盘分区系统 （ G ，其磁盘的分区数不再受 M 的限制，在 M 结构下只能存在 个主分区 。） E 系统分区可以被 E 系统存取，用于存放部分驱动和应用程序。t 是在 fM Xa8平台 e E 系统中的一个特殊模块，它将为不具备 E 引导能力的操作系统提供类似于传统 B 的系统服务，以实现旧系统的兼容性。

2基于 U 的安全模块设计

2 ．和 U 简介F ll

E F I （ E x t e n s 可扩展固件接口F ）i[4] r 是由 I 推出的e 一种在未来计算机系统中替代传统 B 的 O 升级方案，其主要目的是为了提供一组在 O 加载之前，在所有平台上一致的、正确指定的启动服务。 E 采用模块化、动态链接和 语言风格的参数堆栈传递方式构建系统，比 B 更易于实现。此外， E 驱动程序可以不由运行在 C P 上的代码组成，而是由 E 字节代码编写而成，保证了在不同 C 架构上的兼容性。

2 年，0 I5 n开放 eE 源代码，并将其交托一个工业联盟管理。同时， E 变更为 U E F I （ U n i f i m e 层进行的 E ）U 在开 F i r m w a，统一e 可扩展固件接口 e ），[5] 0 联盟E d 机时的作用是初始化用户计负责相关技术标准的制订。 2 年 （月，7 U 联盟颁布了 U E F 算机。但它和传统 B 的不O 版本规范。2要使用 U ，E 要求主板和操作系统都必须支持 U 标准。目前支持 U 的操作系统有 W i n V i和 s W t i a n S e r 等.e r

U 架构如图 2 所示。中间灰色的 型区域 （ 即 EE 初始 I 化模块， D 即D驱动 块执行环境 ，）以及 d E 驱动程序、平台驱动程序、 C 即兼容性(支持模块 和 U 架构驱动等，就是传统 B 所提供的功能范围。 B 的芯片容量只有 2 字节6，还远不够 U 使 F I 用。 U 是以小型磁盘分区的形式存放在硬盘上的。 U E F I 的安装使用光驱引导系统，然后对磁盘进行划分 U 独 F I 用的磁盘空间。i Un 的存储空间大约为 r 5f 到 M1 eB0，）O M B 具体视驱动文件多少而定。在这部分空间中，包含了 P、E I D、X UE 驱动程序、 C 以及 MU 高层应用和 G U I D 磁盘分区等部分。r a r y

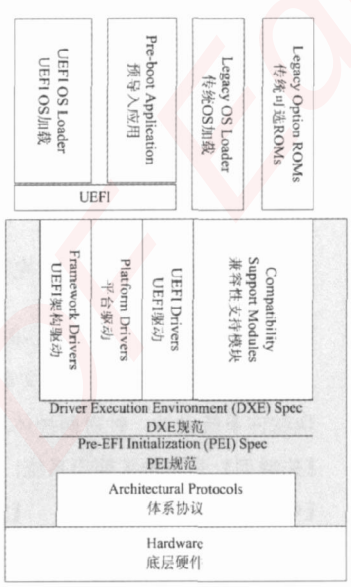

图2 UEFI框架

在实现中， P和E D通常被集成在一个只读存储器中，在硬件系统开机时，最先执行的是 P 初始化程序，它负责

2 总体框架

文中基于 U 协议的安全模块系统框架如图 所示。基于 U 协议的安全模块软件系统开发是基于 U 的模块驱动 （ m o d u r

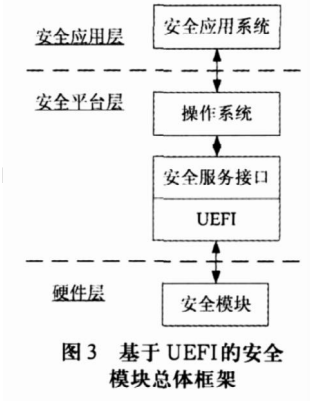

图3 基于UEFI的安全模块总体框架

硬件的完整性，还会加载硬件在 U 中的模块驱动，而不用操作系统进行驱动加载工作。因为 U 架构使用的模块驱动基于 E F I 。B 文件不由 C 直接执行 o d e 操作，而需要 U 层进行翻译，它直接安装在 U 环境中。I 对硬件设备的控制由 U 负责，IU 向操作系统直接提供硬件操作的接口，不需要操作系统再调用驱动。

E F I 直接和 t We i 的 C dAo 联系，意味着无论操作系统是 W i 还是 d lo iy，只要有 xE F I C 支持，只需一份驱动程序就可自动支持所有操作系统平台。此外，对于安全模块的安全服务接口升级操作也很容易，只需进入 U 界面，更新模块驱动程序就可完成，不需要对每个操作系统分别进行升级操作，方便系统维护。

3 两种体系架构比较

表 是对两种体系架构下安全模块软件系统的基本结构、工程维护和安全性进行了比较分析。

表 1 体系框架对比

内 容	传统安全模块设计	基于 UEFI 的安全模块设计
结 构	在操作系统的环境上提供应用，需分别实现硬件驱动和安全服务接口两部分软件功能。	在 UEFI 的环境上提供应用，只需实现安全服务接口(内置硬件驱动)，提供统一的软件功能。
软件维护	受操作系统的限制，在不同操作系统下，硬件驱动和安全服务接口的软件工程均不同，维护量大。	不受操作系统的限制，只需管理统一安全服务接口的软件工程，支持不同操作系统的应用，维护量小。
安全性	传统 BIOS 过于自信硬件芯片的安个，容易受病毒攻击。 安全模块的硬件驱动和安全服务接口受操作系统的限制，对操作系统自身的安全隐患无法保障，安全性相对被动。	UEFI 的安全性优于传统 BIOS，EFI 驱动部分不存放在 EFI 的 GUID 分区中，即使分区中的驱动程序遭到破坏，也可以用简单方法得到恢复。 安全模块的安全服务接口集成在 UEFI 的内部，不受操作系统自身的安全隐患限制，安全性相对主动，结合可信计算等技术可提供安全主动防御。

4结语

目前，U 凭借自身的技术特点，可以广泛运用于嵌入式应用、网络电脑、网络客户端电脑等产品，其应用已由服务器领域扩展至 P 领域。与此同时，E 技术向消费电子、家用设备领域的延伸也从未停止。

2 年，国家信息产业部和国家科技部分别设立重点项目，B 支持发展国产 B 技术与产品。目前，在信息产业部的指导下，中科院软件中心有限公司、南京百敖软件有限公司等国内各个厂商正在积极制订中国 B 规范与标准，确保在这一新领域国内企业能够不受知识产权方面的制约。涉足信息安全行业的各开发厂商也积极参与了 U 的应用，I 这必将推动新一代安全计算机产业的体系创新，提高信息安全行业的自主创新能力。

参考文献

[1] M S D N L i b r a, r 2y 0[OM 1] -. 4
[2] C a n t 设备驱动程序开发指南 i[dM o] w. s 孙义，马莉波，国雪飞，等译 北京：机械工业出版社，2 0 0 1 .
[3] R i c h t e r J . P r o g r a [M] . M i c r, d1 s9 o9 f9 t C o r p o
[4] I n t ,e El x tC eo mr sp io br la et i Fo in r s p e c i f i c ,a2 t0 i0 o2 n- 1v 2e -r 1s [i Ro]n
[5] U n i, fU ineidf i Ee Fd I E Ixntce .n I n t e r f a c e s, p2 e0 c0 i7 Qi1 c- a2 t3 i|

上接第 9页 0)

于管理，安全性高；中心服务和安全规则统一管理，可管理性和可控性强；统一规划和部署，互通性和交互性可扩展。

(使用方便：产品定位于基层电子政务，只要可上互联网的地方就可部署；系统易于安装和使用；可传输的数据类型广泛。

5结语

在机关信息化建设中，基层电子政务建设占据重要的地位。由于机关工作的特殊性和基层政务信息的敏感性，基层电子政务信息安全系统必须与电子政务系统建设同步规划、同步建设。同时，由于基层电子政务网络环境的特殊性，解决基层电子政务系统网络安全问题必须从实际出发，全面分析存在的安全隐患，有针对性地制定安全策略，并以此为基础采取有效的安全措施，才能确保基层电子政务系统网络信息安全。

参考文献

[孙靖] 电子政务需要怎样的安全方案 [信息安全与通信保密， 2 ,0 (0 7 3) 4 - 3 5 .
[国家密码管理局 商用密码通用产品名单 [北京：国家密码管理局， 2 0 0 :7 /. /h wt wt wp . o s c c D o c / 2 1 / N e w s _ 1 1 3 1 . h t
[卢锦泉，苏一丹 网络管理中安全策略与安全信息共享模型研究 [广西民族大学学报 自然科学版 ,) 2 ,0 0 5 (:0 12 8) - 2 1 .
[卢世凤，刘学敏，刘淘英，等 基于策略的管理综述 [J] . 计算机工程与应用， 2 ,0 (0 9 8)5 - 8 9 .

嵌入式资源免费下载

总线协议：

1. 基于 PCIe 驱动程序的数据传输卡 DMA 传输
2. 基于 PCIe 总线协议的设备驱动开发
3. CANopen 协议介绍
4. 基于 PXI 总线 RS422 数据通信卡 WDM 驱动程序设计
5. FPGA 实现 PCIe 总线 DMA 设计
6. PCI Express 协议实现与验证
7. VPX 总线技术及其实现
8. 基于 Xilinx FPGA 的 PCIE 接口实现
9. 基于 PCI 总线的 GPS 授时卡设计
10. 基于 CPCI 标准的 6U 信号处理平台的设计
11. USB30 电路保护
12. USB30 协议分析与框架设计
13. USB 30 中的 CRC 校验原理及实现
14. 基于 CPLD 的 UART 设计
15. IPMI 在 VPX 系统中的应用与设计
16. 基于 CPCI 总线的 PMC 载板设计
17. 基于 VPX 总线的工件台运动控制系统研究与开发
18. PCI Express 流控机制的研究与实现
19. UART16C554 的设计
20. 基于 VPX 的高性能计算机设计
21. 基于 CAN 总线技术的嵌入式网关设计
22. Visual C 串行通讯控件使用方法与技巧的研究
23. IEEE1588 精密时钟同步关键技术研究
24. GPS 信号发生器射频模块的一种实现方案
25. 基于 CPCI 接口的视频采集卡的设计
26. 基于 VPX 的 3U 信号处理平台的设计
27. 基于 PCI Express 总线 1394b 网络传输系统 WDM 驱动设计
28. AT89C52 单片机与 ARINC429 航空总线接口设计
29. 基于 CPCI 总线多 DSP 系统的高速主机接口设计
30. 总线协议中的 CRC 及其在 SATA 通信技术中的应用
31. 基于 FPGA 的 SATA 硬盘加解密控制器设计
32. Modbus 协议在串口通讯中的研究及应用
33. 高可用的磁盘阵列 Cache 的设计和实现
34. RAID 阵列中高速 Cache 管理的优化

35. 一种新的基于 RAID 的 CACHE 技术研究与实现
36. 基于 PCIE-104 总线的高速数据接口设计
37. 基于 VPX 标准的 RapidIO 交换和 Flash 存储模块设计
38. 北斗卫星系统在海洋工程中的应用
39. 北斗卫星系统在远洋船舶上应用的研究
40. 基于 CPCI 总线的红外实时信号处理系统
41. 硬件实现 RAID 与软件实现 RAID 的比较
42. 基于 PCI Express 总线系统的热插拔设计
43. 基于 RAID5 的磁盘阵列 Cache 的研究与实现
44. 基于 PCI 总线的 MPEG2 码流播放卡驱动程序开发
45. 基于磁盘异或引擎的 RAID5 小写性能优化
46. 基于 IEEE1588 的时钟同步技术研究
47. 基于 Davinci 平台的 SD 卡读写优化

VxWorks：

1. 基于 VxWorks 的多任务程序设计
2. 基于 VxWorks 的数据采集存储装置设计
3. Flash 文件系统分析及其在 VxWorks 中的实现
4. VxWorks 多任务编程中的异常研究
5. VxWorks 应用技巧两例
6. 一种基于 VxWorks 的飞行仿真实时管理系统
7. 在 VxWorks 系统中使用 TrueType 字库
8. 基于 FreeType 的 VxWorks 中文显示方案
9. 基于 Tilcon 的 VxWorks 简单动画开发
10. 基于 Tilcon 的某武器显控系统界面设计
11. 基于 Tilcon 的综合导航信息处理装置界面设计
12. VxWorks 的内存配置和管理
13. 基于 VxWorks 系统的 PCI 配置与应用
14. 基于 MPC8270 的 VxWorks BSP 的移植
15. Bootrom 功能改进经验谈
16. 基于 VxWorks 嵌入式系统的中文平台研究与实现
17. VxBus 的 A429 接口驱动
18. 基于 VxBus 和 MPC8569E 千兆网驱动开发和实现
19. 一种基于 vxBus 的 PPC 与 FPGA 高速互联的驱动设计方法
20. 基于 VxBus 的设备驱动开发
21. 基于 VxBus 的驱动程序架构分析
22. 基于 VxBus 的高速数据采集卡驱动程序开发
23. Vxworks 下的冗余 CAN 通讯模块设计

24. WindML 工业平台下开发 S1d13506 驱动及显示功能的实现
25. WindML 中 Mesa 的应用
26. VxWorks 下图形用户界面开发中双缓冲技术应用
27. VxWorks 上的一种 GUI 系统的设计与实现
28. VxWorks 环境下 socket 的实现
29. VxWorks 的 WindML 图形界面程序的框架分析

Linux：

1. Linux 程序设计第三版及源代码
2. NAND FLASH 文件系统的设计与实现
3. 多通道串行通信设备的 Linux 驱动程序实现
4. Zsh 开发指南-数组
5. 常用 GDB 命令中文速览
6. 嵌入式 C 进阶之道
7. Linux 串口编程实例
8. 基于 Yocto Project 的嵌入式应用设计
9. Android 应用的反编译
10. 基于 Android 行为的加密应用系统研究
11. 嵌入式 Linux 系统移植步步通
12. 嵌入式 CC++语言精华文章集锦
13. 基于 Linux 的高性能服务器端的设计与研究
14. S3C6410 移植 Android 内核
15. Android 开发指南中文版
16. 图解 Linux 操作系统架构设计与实现原理（第二版）
17. 如何在 Ubuntu 和 Linux Mint 下轻松升级 Linux 内核
18. Android 简单 mp3 播放器源码
19. 嵌入式 Linux 系统实时性的研究
20. Android 嵌入式系统架构及内核浅析
21. 基于嵌入式 Linux 操作系统内核实时性的改进方法研究
22. Linux TCP IP 协议详解
23. Linux 桌面环境下内存去重技术的研究与实现
24. 掌握 Android 7.0 新增特性 Quick Settings
25. Android 应用逆向分析方法研究
26. Android 操作系统的课程教学
27. Android 智能手机操作系统的研究
28. Android 英文朗读功能的实现
29. 基于 Yocto 订制嵌入式 Linux 发行版
30. 基于嵌入式 Linux 的网络设备驱动设计与实现

31. 如何高效学习嵌入式
32. 基于 Android 平台的 GPS 定位系统的设计与实现
33. LINUX ARM 下的 USB 驱动开发
34. Linux 下基于 I2C 协议的 RTC 驱动开发

Windows CE:

1. Windows CE.NET 下 YAFFS 文件系统 NAND Flash 驱动程序设计
2. Windows CE 的 CAN 总线驱动程序设计
3. 基于 Windows CE.NET 的 ADC 驱动程序实现与应用的研究
4. 基于 Windows CE.NET 平台的串行通信实现
5. 基于 Windows CE.NET 下的 GPRS 模块的研究与开发
6. win2k 下 NTFS 分区用 ntldr 加载进 dos 源代码
7. Windows 下的 USB 设备驱动程序开发
8. WinCE 的大容量程控数据传输解决方案设计
9. WinCE6.0 安装开发详解
10. DOS 下仿 Windows 的自带计算器程序 C 源码
11. G726 局域网语音通话程序和源代码
12. WinCE 主板加载第三方驱动程序的方法
13. WinCE 下的注册表编辑程序和源代码
14. WinCE 串口通信源代码
15. WINCE 的 SD 卡程序[可实现读写的源码]
16. 基于 WinCE 的 BootLoader 研究
17. Windows CE 环境下无线网卡的自动安装
18. 基于 Windows CE 的可视电话的研究与实现
19. 基于 WinCE 的嵌入式图像采集系统设计
20. 基于 ARM 与 WinCE 的掌纹鉴别系统

PowerPC:

1. Freescale MPC8536 开发板原理图
2. 基于 MPC8548E 的固件设计
3. 基于 MPC8548E 的嵌入式数据处理系统设计
4. 基于 PowerPC 嵌入式网络通信平台的实现
5. PowerPC 在车辆显控系统中的应用
6. 基于 PowerPC 的单板计算机的设计

7. 用 PowerPC860 实现 FPGA 配置
8. 基于 MPC8247 嵌入式电力交换系统的设计与实现
9. 基于设备树的 MPC8247 嵌入式 Linux 系统开发
10. 基于 MPC8313E 嵌入式系统 UBoot 的移植
11. 基于 PowerPC 处理器 SMP 系统的 UBoot 移植
12. 基于 PowerPC 双核处理器嵌入式系统 UBoot 移植
13. 基于 PowerPC 的雷达通用处理机设计
14. PowerPC 平台引导加载程序的移植
15. 基于 PowerPC 嵌入式内核的多串口通信扩展设计
16. 基于 PowerPC 的多网口系统抗干扰设计

ARM：

1. 基于 DiskOnChip 2000 的驱动程序设计及应用
2. 基于 ARM 体系的 PC-104 总线设计
3. 基于 ARM 的嵌入式系统中断处理机制研究
4. 设计 ARM 的中断处理
5. 基于 ARM 的数据采集系统并行总线的驱动设计
6. S3C2410 下的 TFT LCD 驱动源码
7. STM32 SD 卡移植 FATFS 文件系统源码
8. STM32 ADC 多通道源码
9. ARM Linux 在 EP7312 上的移植
10. ARM 经典 300 问
11. 基于 S5PV210 的频谱监测设备嵌入式系统设计与实现
12. Uboot 中 start.S 源码的指令级的详尽解析
13. 基于 ARM9 的嵌入式 Zigbee 网关设计与实现
14. 基于 S3C6410 处理器的嵌入式 Linux 系统移植
15. CortexA8 平台的μC-OSⅡ及 LwIP 协议栈的移植与实现
16. 基于 ARM 的嵌入式 Linux 无线网卡设备驱动设计
17. ARM S3C2440 Linux ADC 驱动
18. ARM S3C2440 Linux 触摸屏驱动
19. Linux 和 Cortex-A8 的视频处理及数字微波传输系统设计
20. Nand Flash 启动模式下的 Uboot 移植
21. 基于 ARM 处理器的 UART 设计
22. ARM CortexM3 处理器故障的分析与处理
23. ARM 微处理器启动和调试浅析
24. 基于 ARM 系统下映像文件的执行与中断运行机制的实现
25. 中断调用方式的 ARM 二次开发接口设计

Hardware:

1. DSP 电源的典型设计
2. 高频脉冲电源设计
3. 电源的综合保护设计
4. 任意波形电源的设计
5. 高速 PCB 信号完整性分析及应用
6. DM642 高速图像采集系统的电磁干扰设计
7. 使用 COMExpress Nano 工控板实现 IP 调度设备
8. 基于 COM Express 架构的数据记录仪的设计与实现
9. 基于 COM Express 的信号系统逻辑运算单元设计
10. 基于 COM Express 的回波预处理模块设计
11. 基于 X86 平台的简单多任务内核的分析与实现
12. 基于 UEFI Shell 的 PreOS Application 的开发与研究
13. 基于 UEFI 固件的恶意代码防范技术研究
14. MIPS 架构计算机平台的支持固件研究
15. 基于 UEFI 固件的攻击验证技术研究
16. 基于 UEFI 的 Application 和 Driver 的分析与开发
17. 基于 UEFI 的可信 BIOS 研究与实现
18. 基于 UEFI 的国产计算机平台 BIOS 研究

Programming:

1. 计算机软件基础数据结构 - 算法
2. 高级数据结构对算法的优化
3. 零基础学算法
4. Linux 环境下基于 TCP 的 Socket 编程浅析
5. Linux 环境下基于 UDP 的 socket 编程浅析