基于 UEFI 固件的攻击验证技术研究*

孙 亮[1]， 陈小春[1]， 王 冠[2]， 郑树剑[1]
(1.中电科技(北京)有限公司,北京 100083;
2.北京工业大学,北京 100124)

[摘 要]随着信息技术飞速发展,木马病毒和黑客攻击也不断花样翻新。目前,已经出现了以固件木马为手段,对路由器、防火墙、服务器进行渗透,对我国信息安全造成现实威胁。固件木马的特点是先于操作系统启动,不易被杀毒软件发现和清除。对固件木马攻击方式进行研究,设计和实现了基于 UEFI 固件的攻击验证原型系统,模拟真实的固件木马运行环境,验证固件木马对计算机的攻击效果。

[关键词]UEFI;固件木马;攻击验证
[中图分类号]TP309.5 [文献标志码]A [文章编号]1009-8054(2016)07-0089-05

VerificationTechnology based on UEFI Firmware Trojan

SUN Liang[1], CHEN Xiao-chun[1], WANG Guan[2], ZHENG Shu-jian[1]
(1. ZD Technologies (Beijing) Co., Ltd, Beijing 100083, China;
2. Beijing University of Technology, Beijing 100124, China)

[**Abstract**] With the rapid development of information technology, various computer virus and hacking attacks emerge in an endless stream. At present, by means of fireware Trojans, the routers, firewalls, servers and other equipment are attacked and penetrated, thus causing great threat to the state information security. What is more, the firmware Trojan usually starts earlier than operating system, so it is not easy for virus-killing software to find and eliminate it. Firmware Trojan attack is studied in this paper and the prototype system based on UEFI firmware attack verification system is designed and implemented. Actual operating environment of firmware Trojan is simulated and the attack of firmware Trojan upon computers also verified.

[**Key words**] UEFI; firmware Trojan; attack verification

0 引言

随着信息技术的飞速发展,以移动计算、云计算、物联网为代表的互联网产业呈现出空前繁荣的景象。但是,危害信息安全的事件也随之不断发生。从近期披露的安全事件和资料来看,终端渗透和攻击的技术也一直在更新和变化。目前,已经出现了通过固件木马等工具,针对特定网络、特定终端进行攻击和渗透的专业化组织。可以看出,仅仅依靠操作系统中运行的杀毒软件或防火墙,已经不能满足当前终端安全和网络安全的需要[1]。

固件是计算机中不可缺少的重要部件,是连接计算机基础硬件和系统软件的桥梁。基本输入输出系统(Basic Input Output System, BIOS)是计算机上最重要的固件之一。在开机上电后,固件会对 CPU 中的寄存器、计时芯片、可编程中断器及 DMA 控制器的状态检查,同时初始化设置主板芯片组、动态内存、显卡及相关外围的寄存器。在以上设备正常运行的前提下,固件将负责引导操作系统。

从开机上电到操作系统加载的过程中,固件拥有着系统极高权限。一旦固件中存在安全漏洞或被植入木马,将会为计算机带来严重的威胁。

通过在固件中增加后门程序,如使用系统管理模式(System Management Modal,SMM),可在操作系统无法察觉的情况下,隐蔽地获取计算机数据、破坏系统正常运行。这些后门程序难以被操作系统下的杀毒软件探测和删除[2]。美国商贸部专门发布了 sp800-147、sp800-147b、sp800-155 三个指南,用于指导对 BIOS 的安全保护。美国国防部也专门就此发布备忘录,要求美军计算机采购时,固件需要满足相应的保护标准。

因此,本文对固件木马的特征进行了分析,设计了攻击验证的原型系统,并对关键技术进行了验证。该系统支持 X86 计算机平台和国产计算机平台,支持扩展相应的外设,包括 USB 接口/PCI-E 等接口的可信卡,能够在此基础上进行进一步扩展,检测

*收稿日期:2016-02-20
北京市教委科技项目:可信云计算安全体系及关键技术研究(No. 007000546615020)

和发现固件的安全漏洞,并开发相应的补丁。

1 UEFI 固件安全威胁分析

1.1 固件的软件硬件二象性及安全威胁分析

固件在形态上类似于硬件,是"固化"封装在芯片中、安装在电路板上的器件;固件在功能上又属于软件,固件芯片内是机器语言编写的控制程序。这种"二象性"使得固件平滑地完成了将机器的控制权由处理器硬件转交给操作系统的任务。固件的二象性构成了计算机软硬件的重要桥梁,但也对计算机安全构成了潜在的威胁。

一方面,固件的物理形式是芯片,可以安装在计算机主板、硬盘、显卡等硬件设备中[3]。如果把固件木马以非常隐蔽的方式固化到计算机主板或扩展板卡上的芯片中,并在特定条件下激活,将会立即执行渗透攻击和情报收集工作,用户在日常使用计算机的过程中却很难察觉。

另一方面,固件是运行在计算机底层的软件,是先于操作系统启动而运行的。因此,固件中的恶意代码是无法被杀毒软件和安全工具发现的。固件的运行过程是具有极高权限,能够对硬件、文件系统、操作系统和特定软件进行篡改和破坏,具有极强的攻击能力。

1.2 固件运行过程安全威胁分析

固件的运行过程分为四个阶段,包括硬件初始化阶段、驱动服务执行阶段、启动设备选择阶段和操作系统运行阶段:

1)硬件初始化阶段。

执行由汇编语言代码实现的固件组件,主要实现对硬件平台和固件初始启动代码进行安全验证,初始化系统缓存,为后续的硬件初始化工作准备必要的存储空间;同时,初始化处理器、内存、芯片组和其他芯片及设备。

硬件初始化阶段是固件首先运行的阶段,是保证计算机硬件和固件完整可信的基石。如果固件在这个阶段中运行了恶意代码,将能够破坏这个阶段对硬件和固件的安全校验,破坏整个计算机的安全运行环境。

2)驱动服务执行阶段。

执行设备驱动程序,安装及初始化与设备、总线、服务等相关的协议。驱动服务执行阶段为后续操作提供协议/服务接口,提供给固件自身及操作系统和应用软件调用。

驱动服务执行阶段固件运行的核心阶段,固件会加载硬件驱动和应用程序。如果计算机主机中接入了包含固件木马模块的硬件,其木马模块将在这个阶段在固件层进行加载。在该阶段中,固件能够加载和执行文件系统驱动,对硬盘分区和文件系统进行识别和分析,能够将操作系统中的木马植入操作系统,并实现随操作系统自启动[4]。

3)启动设备选择阶段。

根据系统中预先设置的配置规则或者用户本次的选择,寻找承载操作系统的设备(如 USB 设备、硬盘、光盘、网络等)并加载操作系统。

启动设备选择阶段提供了 UEFI SHELL 运行环境、UEFI 驱动及操作系统加载的功能。因此,也存在操作系统加载过程中被劫持的攻击漏洞。

4)操作系统或应用软件运行阶段。

固件将机器的控制权正式移交给操作系统。此时,仍然有部分固件服务/协议可用。

在该阶段中,如果固件中存在可以执行的恶意代码,操作系统中的杀毒软件将无法觉察。

通过对固件运行过程分析可以看出,固件安全首先要保证在硬件初始化阶段,对固件进行完整性验证;其次,需要在驱动服务执行阶段对加载的硬件和固件驱动进行验证,禁止加载未经确认的固件驱动。这两种安全措施能够在一定程度上防止固件木马的加载和执行。

2 固件木马攻击的特点

固件层的木马比操作系统的木马更具有破坏性,更加难以发现和清除。CIH 病毒是第一款对物理设备进行破坏的病毒,其在 v1.2 版本中加入了破坏硬盘和 BIOS 的代码。CIH 在 1999 年4 月 26 日全球大规模爆发,导致全球近六千万台电脑遭到破坏,甚至无法启动[5]。

文献[6]提出了针对智能电源芯片的固件进行攻击、重新编写控制器程序,可以屏蔽电池过充、过热等报警信号,并可导致电池爆炸等严重后果。

早期的 BIOS 采用汇编语言编写,并且各大厂商对 BIOS 源码进行严格的保密。因此,对固件进行攻击的难度比较高。

John Heasman 在 2006 年黑帽大会中,提出了利用高级电源管理接口(Advanced Configuration and Power Interface, ACPI),使用 ASL(ACPI Source Language)语言,实现在固件中部署 Rootkit 恶意代码[7]。该方式的特点是使用了 ACPI 的专用语言 ASL,简化了通过汇编语言编写病毒的难度。其特点是硬盘上没有痕迹,重装系统和更换硬盘仍然能够存在,难以发现,难以清除。在 2007 年黑帽大会上,John Heasman 又提出基于主板 PCI 板卡上的扩展固件进行 Rootkit 的攻击[8]。该方法首先在 PCI 板卡固件中植入恶意代码。而后,在 BIOS 运行时,这些恶意代码会从 PCI 板卡中读取和自动加载。

文献[9]提出了通过固件中的系统管理模式(System Management Mode, SMM)中隐藏 Rootkit 恶意代码。SMM 是特殊的处理器操作模式,可以执行特权指令和 IO 操作。该模式只能被固件调用,并且对操作系统完全透明。

目前,基于固件的攻击不再作为单一的攻击手段,而是作为

多层次复合攻击的基础工具。如 BMW 病毒,能够感染 BIOS 及 MBR,在通过 Windows 系统加载恶意代码。用户即使重装系统、格式化硬盘甚至更换硬盘都无法彻底清除病毒[10]。

同时,固件的攻击已经不止对主板 BIOS 进行攻击,而且包括了对外接设备的固件进行破坏。如近期出现的方程式病毒[11],已经开始针对硬盘中的固件进行攻击,能够绕过操作系统的监控,直接对硬盘数据进行修改。

文献[12]提出了对显卡固件进行攻击的方法。该木马能够对用户的键盘敲击进行记录,并将信息存储在 GPU 中。这种针对外设板卡的攻击,实现了跨平台隐藏 rootkit 的效果,包括 Windows、Linux、Mac OS 等操作系统。

可以看出,与操作系统攻击和网络攻击不同,固件攻击的的主要特点包括:

(1)固件木马难以清除。固件木马会驻留在被攻击终端的主板或板卡上,通过重装系统、硬盘格式化或更换硬盘无法清除固件木马。

(2)固件木马可对硬件直接进行攻击。固件具有对计算机底层硬件进行访问和控制的权限,在固件中植入的恶意代码可在开机上电时,通过提升超出门限的电压或配置破坏性的参数,对硬件进行破坏性配置,造成硬件损坏、无法开机,甚至导致电池爆炸等严重后果。

(3)操作系统难以发现固件层攻击。固件中的恶意代码在操作系统运行之前执行,往往难以被杀毒软件发现。此外,操作系统也难以发现运行在系统管理模式的固件恶意代码。

(4)固件木马可以对操作系统进行直接攻击。在固件执行过程中,可以对硬盘分区、文件系统进行解析,对操作系统中的文件进行复制、删除、修改。也可以通过固件木马,将特定的攻击载荷写入操作系统,并随操作系统进行自启动[13]。

可以看出,与操作系统攻击和网络不同,固件木马是驻留在主板固件芯片之中,攻击方式具有跨平台特性,操作系统难以发现其攻击行为,重装系统甚至更换硬盘无法清除。随着攻击方式的多样化、复合化、复杂化,固件木马作为专用于潜伏隐藏的渗透工具进行应用。

3 固件木马攻击验证系统总体设计

3.1 固件木马攻击验证系统总体结构

固件木马在操作系统下难以发现和清除。因此,固件木马攻击验证系统需要首先提供固件木马可以运行的硬件及软件环境,以激活固件木马的攻击动作,抓取相应的固件层执行过程和运行状态,才能深入了解固件木马的攻击特征,找到应对策略。

固件攻击的主要方式包括以下三种:

(1)直接针对固件进行攻击,如对固件或某些模块进行篡改。

(2)通过外设板卡进行攻击,如在显卡、硬盘中植入恶意代码。

(3)通过固件对操作系统进行攻击,如在固件运行阶段加载文件系统,对硬盘中的文件进行篡改。

因此,固件攻击验证系统需要对固件进行读取和解析,判断固件镜像和固件模块是否正确。其次,需要对固件加载外设板卡驱动和其他驱动模块的执行过程进行记录。再次,需要记录固件层访问操作系统的执行过程。

图 1 展示了固件攻击验证系统的逻辑架构,包括硬件层、固件层、操作系统层和服务层四个层面:

(1)硬件层。由于固件与硬件平台是紧密相关的,每一款硬件平台都需对应特定的固件。因此,固件攻击验证系统需要提供多种平台的基础硬件支持,包括 X86 平台、国产处理器平台、可信密码模块和其他外设。这些外设能够根据不同的需求进行相应的组合和定制,能够模拟局域网或单台终端的硬件实验环境。

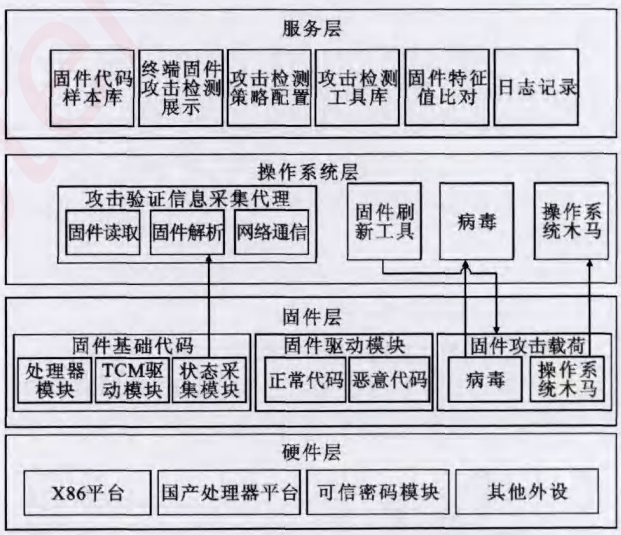

图 1 固件攻击验证系统逻辑架构

(2)固件层。为支持对固件进行攻击验证,需要将固件划分为三个部分,一是固件基础代码,二是固件驱动模块,三是固件模拟攻击载荷。

固件基础代码包括对处理器和芯片组初始化、操作系统引导等基本功能。同时,该部分还包括固件状态采集模块,用于记录固件运行过程中加载的硬件信息、固件驱动模块的状态信息及特征值。状态采集模块的目的是在固件层放置一个探针,对固件运行的过程进行记录。

固件驱动模块中包括了固件加载的所有驱动模块,包括以 OPROM 方式加载的固件驱动。在固件驱动模块中可以加载某些预设恶意代码,用于模拟固件层漏洞攻击。攻击方式包括对硬件平台芯片的破坏性参数配置,对文件系统的复制传输等。

固件攻击载荷是固件层中预留的模拟攻击的恶意代码。固件木马在执行过程中会识别硬盘分区和文件系统后,将恶意代码写入操作系统,评估攻击效果。

(3)操作系统层。操作系统层中包括了固件攻击验证信息采集代理、固件镜像和固件刷写工具。

固件攻击验证信息配合固件中的固件状态采集模块,将固件文件、固件运行状态等信息通过网络发送到后台服务端进行分析。

固件刷写工具用于将固件镜像刷入主板固件芯片。固件镜像包括了可用于攻击验证的多种固件镜像。在固件运行过程中,会将木马和病毒写入操作系统并执行自启动。

(4)服务层。服务层中包括固件代码样本库,固件攻击检测展示、固件攻击检测策略配置、固件验证工具库、固件特征值比对和日志记录等功能模块。

固件代码样本库包括了已经采集的并且已授权的固件代码特征值,用于对固件样本进行验证。

固件攻击检测展示用于对攻击效果和检测结果进行展示。

攻击检测策略配置用于配置固件中的安全策略,如是否采用攻击载荷,选用何种攻击载荷等。

攻击验证工具库用于存储和提供固件攻击工具。用户可以通过配置界面,选择固件攻击方式和相应的工具。

固件特征值比对模块用于对信息采集模块获取到的信息进行特征值比对。

日志记录功能模块用于对固件攻击检测的过程进行记录,用于后续的固件漏洞效果分析。

3.2 固件木马攻击验证系统拓扑结构

固件攻击验证系统包括攻击验证靶机、攻击实验终端和攻击验证服务器三类终端,其网络拓扑如图 2 所示。

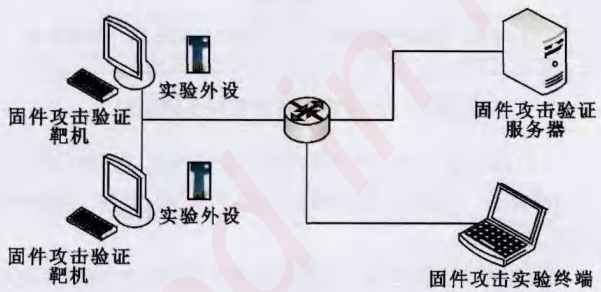

图 2 固件攻击验证系统拓扑

(1)固件攻击验证靶机。该类终端是可根据攻击场景需求定制的 X86 及国产计算机主机。在靶机中已经安装了固件信息采集代理和定制的固件。该类终端根据不同的实验需求,可配插相应的板卡。

(2)固件攻击验证服务器。该类终端的作用包括两个,一是用于对固件攻击的效果进行跟踪记录,二是用于对靶机的固件进行安全检测。

(3)固件攻击验证终端。该类终端主要部署了支持 X86 平台和国产处理器平台的固件开发工具和调试环境。开发和调试环境中集成了编译工具、调试工具及扩展设备。

固件攻击验证系统能够模拟通过固件漏洞,对多计算机终端进行远程攻击的场景。

3.3 固件攻击验证系统工作原理

固件攻击验证系统的工作原理如图 3 所示,其工作过程主要包括设计阶段、准备阶段、植入阶段、执行阶段和评估阶段共五个阶段。

(1)设计阶段。用户首先需要登录固件攻击验证服务器,根据攻击验证需求,设置攻击策略,并选择相应的木马工具,包括固件刷新工具、固件木马、工具载荷完成固件攻击策略、流程和工具的设置后,将自动生成可执行的攻击验证脚本。

(2)准备阶段。用户在设计阶段完成后,会将攻击验证脚本推送到固件攻击终端。固件攻击终端将会根据攻击验证脚本配置软件运行环境、下载木马工具,并按照脚本中的攻击策略进行执行。

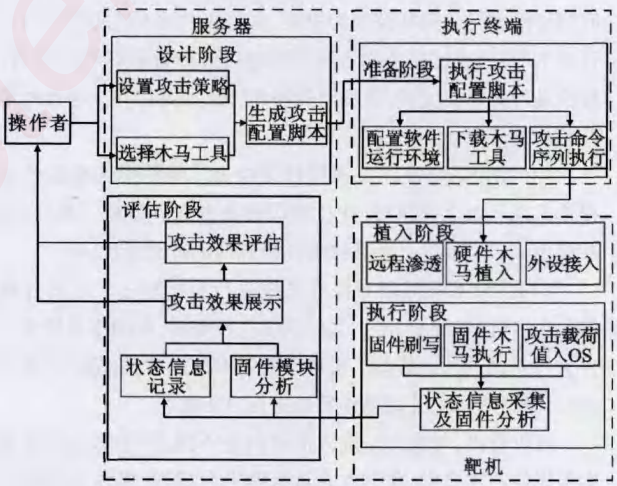

图 3 固件攻击验证系工作原理

(3)植入阶段。固件刷新工具、木马工具和攻击载荷首先需要推送到固件攻击靶机,并进行植入。植入的方法包括固件镜像完全更新、固件攻击载荷部分更新、固件木马外设板卡接入三种方法。

(4)执行阶段。固件刷新工具将固件木马、攻击载荷刷入固件。在计算机重启过程中,固件木马模块将被加载和执行。固件木马能够识别硬盘、分区和文件系统,可以将攻击载荷写入操作系统,并随操作系统自启动。

(5)评估阶段。在固件木马执行过程中,固件层采集模块还将对固件运行过程的状态进行记录,并将状态记录传入操作系统,并最终传输到服务器中进行状态记录和分析,最终生成评估报告。

4 固件木马攻击验证系统的实现

在对固件木马攻击验证系统进行分析和设计的基础上,已经研发了原型系统对攻击效果进行验证。该系统主要包括三个部分,包括固件模拟攻击模块、攻击验证服务器、攻击验证客户端。

1)固件模拟攻击模块。

通过对固件将空间划分为三个部分,一是固件核心镜像,用于完成固件的基本功能;二是固件木马模块,用于完成将模拟植入的固件攻击载荷回写操作系统;三是固件攻击模拟载荷,用于模拟要回写操作系统的攻击软件。固件空间划分如图4所示。

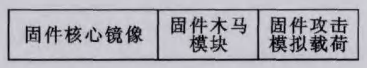

图4 固件空间划分示意

2)攻击验证服务器。

攻击验证服务器用于选择固件攻击工具,定制固件攻击策略,展示固件攻击效果。

3)攻击验证客户端。

统计验证客户端用于获取被攻击计算机信息采集和回传给攻击验证服务器,用于展示攻击效果。

图5展示了固件攻击验证系统的攻击选项配置界面,通过该配置界面,能够定制需要植入的木马和预定的攻击路径。图6展示了该原型系统能够对固件的关键信息进行验证,并在信息异常时,进行记录和提示。

图5 固件攻击验证系攻击选项界面

图6 固件攻击验证系统攻击效果

5 结语

本文提出了基于UEFI固件的攻击检测技术,并设计和实现了以验证固件木马攻击效果为目标的固件攻击验证系统。固件攻击验证系统能够在上电开机和操作系统运行过程中,通过固件层和操作系统的状态采集模块对固件木马的执行过程和状态进行记录;操作系统中的攻击验证客户端还将对固件模块进行读取和解析;在服务器中还部署了固件样本代码库,会将固件解析后的模块特征值进行比对。

下一步,我们将在已有的基础上继续进行相应的研究,以该原型系统为基础,对固件安全漏洞进行评估、检测,并研究相应的防护机制。

参考文献:

[1] 池亚平,许盛伟,方勇.BIOS木马机理分析与防护[J]. 计算机工程,2011,37(13):122-124.

[2] 唐文彬,祝跃飞,陈嘉勇.统一可扩展固件接口攻击方法研究[J].计算机工程,2012,38(13):99-101.

[3] 张京生,韩劲松. 硬盘固件病毒的工作原理及防治方法[J]. 北京信息科技大学学报:自然科学版:2013,28(01): 42-45.

[4] 郭致昌,张平,庞建民等. 基于行为特征的BIOS Rootkit检测[J]. 计算机工程,2011,38(02): 251-253.

[5] 李越,黄春雷. CIH病毒的分析与清除[J]. 计算机科学,2000,27(05):104-105.

[6] Charlie Miller. Battery Firmware Hacking: Inside the Innards of a Smart Battery [J]. In Black Hat , 2011.

[7] John Heasman. Implementing and Detecting an ACPI BIOS Rootkit[J]. In Black Hat DC, 2006.

[8] John Heasman. Implementing and Detecting an PCI Rootkit [J]. In Black Hat DC, 2007.

[9] Shawn Embleton and Sherri Sparks. SMM Rootkits: A New Breed of OS Independent Malware, Presented at BlackHat USA, Las Vegas, NV, USA, 2008.

[10] BMW病毒感染量突破5万 遭全球杀毒厂商围捕[J]. 电力信息化,2011,9(10):103.

[11] Equation: The Death Star of Malware Galaxy. http://securelist.com/blog/research/68750/equation -the death-star-of-malware-galaxy/

[12] Vasiliadis G, Polychronakis M, Ioannidis S. GPU-Assisted Malware[J]. International Journal of Information Security, 2010:1-6.

[13] 杨培,吴灏,金然. BIOS安全防护技术研究[J]. 计算机工程与设计,2008,29(15):3840-3842.

作者简介:

孙 亮(1980—),男,博士,工程师,主要研究方向为网络安全、可信计算、固件;

陈小春(1980—),男,硕士,高工,主要研究方向为网络安全、可信计算、固件;

王 冠(1968—),男,博士,副教授,主要研究方向为信息安全、可信计算、云计算;

郑树剑(1982—),男,硕士,工程师,主要研究方向为网络安全、固件、操作系统内核。

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

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

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

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

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

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

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

Programming:

1. 计算机软件基础数据结构 - 算法