

基于嵌入微 LINUX 的开机安全认证系统

马富达, 蔡皖东

(西北工业大学 计算机学院, 陕西 西安 710072)

摘 要: 提出了一种高安全性的开机安全认证方法, 该认证系统在硬盘上嵌入一个微 Linux 系统作为认证系统的载体, 并且使用 USB 移动介质作为 KEY 盘来保存硬盘启动扇区、分区表、以及登录密钥等系统启动的必要信息, 只有同时拥有密钥和 KEY 盘的合法用户才能通过验证并登录计算机系统. 这种方法实现了重要信息如密钥和启动信息等与硬盘原始数据的独立存储, 而且可以应用基于微 Linux 系统下性能优良的 MD5 和 DES 等加密算法, 从而增强了开机认证的安全性.

关键词: 开机认证; 嵌入 LINUX; 安全启动; 计算机数据安全

中图分类号: TP309

文献标识码: A

文章编号: 1000-7180(2010)07-0181-04

Boot Security Authentication System Based on Embedded Micro-LINUX

MA Fu-da, CAI Wan-dong

(College of Computer Science, Northwestern Polytechnical University, Xi'an 710072, China)

Abstract: This paper proposes a highly secure method of boot security authentication, involving one embedded micro-Linux system stored in the disk driver being the authentication carrier and one USB removable media being a KEY startup disk to save boot data and key disk. This approach implemented the independent storage of important information such as key, start-up and disk original data. Also, it can employ high-performance MD5 and DES encryption algorithms based on micro-Linux system, thus greatly increasing the security of the boot authentication and effectively protecting the disk data.

Key words: boot authentication; embedded micro-LINUX; security start-up; computer data security

1 引言

计算机开机认证技术^[1-2]是重要的计算机系统安全保护措施, 对于用户身份的合法性认证和保护计算机系统与数据的安全发挥着重要的作用. 目前普遍采用认证方法是在计算机的 BIOS 和操作系统级上设置密码进行身份认证. 然而, 这两种方法都存在易破解的缺陷, 通过一些简单的方法就可以清除或绕过所设的密码. 例如 BIOS 密码可以通过对主板放电来清除, 操作系统密码可以通过将系统磁盘挂接到另一台计算机上来绕开或破解所设密码, 并

随意获取硬盘中任何数据或文件.

当前的另一种方法是对硬盘加锁, 普通的硬盘锁都是通过改写硬盘引导扇区和分区表来实现. 认证程序只能写在硬盘前 64 个扇区, 因此可利用空间较小且一般用汇编语言实现, 难以应用复杂而成熟的加密算法来保证密钥和启动信息安全, 并且密钥和认证程序以及原始扇区信息都在原始的硬盘上存储, 使得破解相对容易.

对于以上存在的问题, 必须通过增强现有开机认证技术的方法来加以解决, 设计和开发出一套新的开机认证方法来提高开机认证的安全性, 有效地

保护计算机系统的数据安全.文中提出一种在硬盘上嵌入微 Linux 系统作为认证系统载体,并且使用 USB 移动介质作为 KEY 盘来保存启动信息和密钥的磁盘数据保护方法^[3].这种方法实现了重要信息如密钥和启动信息等与硬盘原始数据的独立存储,而且可以应用基于微 Linux 系统下的成熟有效的加密算法 MD5 和 DES^[4-5]等,从而增强了开机认证的安全性,有效的保护计算机系统安全.

2 开机认证系统的基本原理

2.1 计算机启动过程分析

计算机加电后,主机将 CMOS 中的 BIOS 引导进入 CPU 运行,BIOS 加载普通的驱动(如键盘和鼠标驱动)并检测设备(POST),在设备检测完毕之后,察看引导盘.引导盘的选择取决于 BIOS 中的启动顺序设置.如果启动顺序为先从光盘启动后从硬盘启动,那么 BIOS 将首先检测光驱中是否有光盘,如果有是否其第一个扇区为引导扇区,如果没有或者第一个扇区不是引导扇区,那么 BIOS 将检查硬盘,并察看其第一个扇区是否为引导扇区,如果是则将引导扇区调入内存并将控制权交给它,这个引导扇区叫做主引导记录(Master Boot Record).主引导记录(见表 1)为 512 字节长,它包括代码部分和分区表部分,代码部分在前部,分区表部分在后部.其分布为:512 字节的前 139 个字节(从 00h 到 8Ah)是可执行代码,紧接着的 80 个字节(从 8Bh 到 DAh)包含了错误提示信息,最后的 66 个字节包括了 64 个字节的分区表和 2 个字节的标志位 55AAh.代码部分查找分区表,并找到一个活动分区,活动分区的标志位为 80h,一般处于 64 位分区表的最开始处.通过调入活动分区并将控制权交给它就可以实现操作系统的引导.

表 1 硬盘主引导记录结构

0000-0088	Master Boot Record 主引导程序	主引导程序
0089-01BD	出错信息数据区	数据区
01BE-01CD	分区表 1(16 字节)	分区表
01CE-01DD	分区表 2(16 字节)	
01DE-01ED	分区表 3(16 字节)	
01EE-01FD	分区表 4(16 字节)	
01FE	55	结束标志
01FF	AA	

2.2 开机认证系统的基本原理

具有磁盘数据保护功能的开机安全认证系统是

文中提出的一种高安全性的开机安全认证技术,在 BIOS 和操作系统之间嵌入一个微 LINUX 系统进行认证管理,该环节在初始化时对磁盘扇区对前段 2M 数据(包括引导区)以及硬盘其他分区的分区表信息进行特殊处理,使之处于不可用状态(也不能作为副盘挂接在其他计算机上使用),并且将用户密钥和磁盘启动信息存储在 USB 移动介质上.在计算机开机启动时,用户必须插入认证 USB KEY 设备,在 BIOS 启动完成后自动转入嵌入的微 LINUX 进行身份认证,只有认证通过,才能恢复原有的磁盘引导区以及分区表信息,并转入引导原操作系统.这种方法将密钥和原始磁盘启动信息提取出来加密存储在移动介质上,实现了重要信息的分开存储,提高了安全性和破解难度.

本系统通过嵌入微 LINUX^[6]来取代普通硬盘锁的汇编认证程序,一方面微 LINUX 系统驱动完备可以很好的支持各种计算机和 USB KEY 设备;另一方面,可以在微 LINUX 平台下使用更复杂的加解密算法(本系统目前采用 MD5 和 DES)来加密密钥和磁盘启动信息来提升系统的安全性.这种方法以微 Linux 微认证系统载体,将 USB 移动介质作为启动计算机系统的钥匙,极大的增强了计算机系统的安全性.开机认证系统的基本原理见图 1.

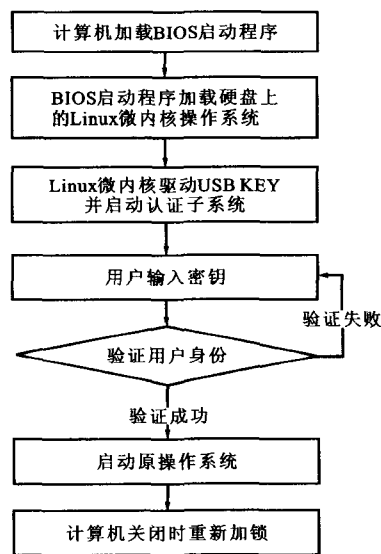


图 1 开机认证系统的基本原理

3 开机认证系统的设计与实现

开机认证系统主要包括硬盘预处理子系统,身份验证和解锁子系统,以及加锁子系统构成.其中硬盘预处理子系统主要负责整个系统的初始化,数据

的备份恢复以及重要信息的加密等;验证和解密系统是构架在微 Linux 系统之上,是本系统的核心部分,负责对用户身份的验证以及通过验证后恢复硬盘的完整信息;加锁子系统负责在 Windows 关闭前重新对硬盘进行加锁,从而保证下次启动计算机时仍必须通过身份验证.这里给出了各个子系统的核心部分的详细设计与实现.

3.1 微 Linux 的设计与剪裁

由于本系统需要在硬盘中嵌入微 Linux 作为认证系统的工作载体,而又由于应用的特殊性,其制作过程与所要达到的标准并不完全等同于普通的微 Linux 系统的剪裁过程.

(1) 整个系统小于 2M.在本系统中微 Linux 内核作为认证系统的载体需要在对硬盘加锁时嵌入到硬盘最前端的扇区,在认证通过后又要用原始硬盘信息覆盖掉微 Linux 内核,因此,在加锁和解锁的过程分别要对微 Linux 内核大小的扇区空间进行读写操作.为了计算机的启动速度不会明显被认证系统拖延,必须保证内核所占空间尽可能小,本系统设计其小于 2M,实验结果表明这一大小完全能够满足启动速度的要求.

(2) 保留 USB 驱动模块.认证系统需要在认证阶段读取 USB KEY 中的信息,所以要保留 USB 驱动部分,其他部分则做最大限度的剪裁.

(3) Ramdisk 技术的应用.微 Linux 系统采用 Ramdisk 技术,即微 Linux 系统启动后将整个根文件系统全部挂接在内存虚拟磁盘中,可以脱离硬盘独立运行在计算机内存里,这样可以避免认证系统读写磁盘时与 Linux 产生冲突.

其他部分诸如 Linux 系统的引导、根文件系统的设计、内核的剪裁过程等在这里就不再赘述.

3.2 硬盘预处理子系统

硬盘预处理子系统在 Windows 下采用 VC++ 编写,主要完成系统的预处理以及其他子系统的安装工作,这里磁盘的读写工作均通过调用 Windows API 完成,因此可以对硬盘的任意扇区进行读写.其核心步骤如下:

(1) 通过 Windows API CreateFile 获取硬盘及移动设备句柄,然后调用 API ReadFile 读取硬盘扇区前 2M 数据以及各个分区的分区表信息,将其采用标准 DES 加密后调用 API WriteFile 将其存储到 USB KEY 中.

(2) 将密钥信息采用 MD5 加密存储到 USB 移动介质中.

(3) 用制作好的可引导微 Linux 系统(带有 syslinux 引导程序)覆盖掉硬盘最前端 2M 的扇区.

(4) 遍历硬盘所有分区的分区信息,将其用随机生成的假信息替换,同时将分区表所在硬盘位置信息用 DES 加密后保存到 USB KEY 中.用户身份验证通过后将根据该信息来查找分区表的正确位置并将其写回硬盘.

磁盘预处理子系统的核心算法如下(以读取硬盘启动扇区并加密存储到 USB KEY 为例,其余类同):

```
hHardDiskDev = CreateFile(
    theApp.gsHardDisk, //硬盘设备号
    GENERIC_WRITE | GENERIC_READ, FILE_SHARE_READ | FILE_SHARE_WRITE,
    0,
    OPEN_EXISTING,
    0,
    0); //获取硬盘操作句柄
SetFilePointer(hHardDiskDev,
    512 * 0,
    NULL,
    FILE_BEGIN); //设置硬盘读操作指针
bResult = ReadFile(hHardDiskDev, Buffer, 512, &num, 0);
DESEncrypt(Buffer); //对数据进行 DES 加密
hFlashDiskDev = CreateFile(
    theApp.gsFlashDisk, //USB KEY 设备号
    GENERIC_WRITE | GENERIC_READ, FILE_SHARE_READ | FILE_SHARE_WRITE,
    0,
    OPEN_EXISTING,
    0,
    0);
SetFilePointer(
    hFlashDiskDev,
    512 * piSectorIndex,
    NULL,
    FILE_BEGIN); //设置 KEY 盘写操作指针
bResult = WriteFile(
    hFlashDiskDev,
    Buffer,
    512,
    &dwRet,
    0); //将加密后的数据写入 KEY 盘
```

3.3 验证和解锁子系统

验证和解锁子系统是本系统的核心部分,由于采用嵌入微 Linux 系统的方式可以使整个系统应用

安全性更高的加密算法以及更通用的驱动程序,使得系统的安全性和兼容性得到增强.验证子系统工作在微 Linux 平台之上,采用标准 C 语言与 Shell 脚本结合的方式编写,C 语言编写的用户程序负责数据的处理,Shell 脚本则负责控制验证程序与微 Linux 操作系统的交互.由于微 Linux 系统包含 USB 驱动模块,所以可以通过 dd 命令对磁盘扇区进行读写,其所得到的数据再交给用户程序进行处理,处理过程包括数据的加解密,密钥的比较等.其中加解密部分则采用成熟的 DES 和 MD5 加密算法实现.该子系统的工作流程如图 2 所示.

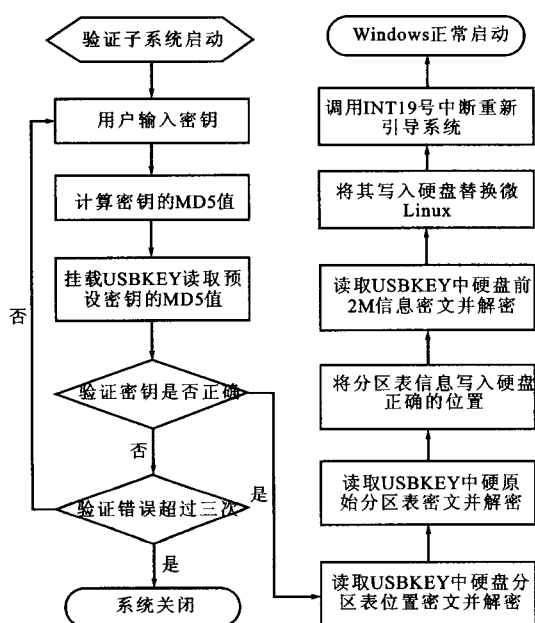


图 2 验证和解锁子系统工作流程

(1) 验证用户身份.验证程序首先要求用户输入密钥,并计算其 MD5 值,然后与 USB KEY 中预设的密钥 MD5 值进行比较,如果相同则通过验证,不同则要求用户重新输入密钥,如果三次不能通过验证则系统自动关机.其中磁盘数据的读写采用 dd 命令的方式进行,下面给出用 dd 读取硬盘引导扇区的例子,读写其他扇区数据与之类似.

```
dd if = /dev/hda of = disk.mbr
```

```
bs = 512 count = 1
```

(2) 恢复硬盘原始数据.通过验证以后,验证程序根据 USB KEY 中的分区位置信息找到分区表正确位置并将 USB KEY 中正确分区表信息解密后写入硬盘.最后将硬盘前 2M 数据用 USB KEY 中存储的原始信息解密后进行替换(由于微 Linux 采用

Ramdisk 技术完全独立运行于内存之中,可以取得对硬盘所有扇区的绝对读写权).

(3) 最后,调用 INT 19 号中断重新引导系统,Windows 就可以顺利启动了.

3.4 加锁子系统

在 Windows 系统关闭前要对系统重新加锁,从而保证下次系统启动时仍然必须通过验证才能登录系统.加锁子系统加锁过程主要是将微 Linux 系统重新写回硬盘,并且破坏硬盘所有分区表信息,这一过程实现类似于硬盘初始化过程,这里不做赘述.问题的关键在于加锁程序需要用户在关机时自动启动.本系统通过 HOOK 关机的 API 函数 ExitWindowsEx 来实现,该算法的主要过程如下所示:

```
void WINAPI HookOneAPI(
    LPCTSTR pszCalleeModuleName,
    PROC pfnOriginApiAddress,
    PROC pfnDummyFuncAddress,
    HMODULE hModCallerModule)
{
    //查找记录,查找是否存在目标 API 函数
    //ExitWindowsEx 所在的 user32.dll
    for (pImportDesc -> Name; pImportDesc++)
    {
        ...
    }
    //寻找所要 HOOK 的 ExitWindowsEx 函数
    pThunk = (PIMAGE_THUNK_DATA)
        (hModCallerModule + pImportDesc -> FirstThunk)
    for (pThunk -> u1.Function; pThunk++)
    Beginfor:
        pProcFuction = &pThunk -> u1.Function;
        if (*pProcFuction == pfnOriginApiAddress)
        BeginIf:
            //如果地址相同,则为所要 HOOK 的函数,
            //对其进行改写,将其指向自定义函数.
            WriteProcessMemory(
                GetCurrentProcess(),
                pProcFuction,
                &(pfnDummyFuncAddress),
                sizeof(pfnDummyFuncAddress),
                NULL);
        return;
    EndIf
    EndFor
}
```

最后对这个钩子函数进行安装,并且在自定义关机函数里实现加锁即可.

(下转第 189 页)

预期目标,因此 RFID 标签数据的安全性得到了保障.

5 结束语

文中提出了一种基于杂凑函数的 RFID 标签数据安全的研究方案,在本方案中 RFID 标签仅需要一个 hash 函数电路,成本较低,比较适合目前的实际情况和成本控制目标.同时,在保持低成本的前提下对该方案抵抗各种类型的攻击进行了分析,并通过 BAN 逻辑对该方案中的认证功能部分进行了形式化分析.结果表明本方案能够对读写器和标签的合法身份进行有效的双向认证、能够有效的抵御重传攻击、哄骗攻击、位置跟踪等.通过与现有的安全协议技术比较具有保护全面、低成本、低的计算量、便于扩展、适合于分布式数据库环境,能够有效的解决当前 RFID 标签数据面临的安全问题,非常适合用于建立仅具有有限资源的低成本标签的 RFID 系统的安全机制.

参考文献:

- [1] 彭昭,刘威,马选斌,等. RFID 系统安全与隐私问题研究[J]. 微电子学与计算机,2007,24(8):129-131.
- [2] 肖珊,郎为民,胡东华. 射频识别(RFID)安全解决方案研究[J]. 微计算机信息,2008(14):210-211.
- [3] 王连强,吕述望,韩小西. RFID 系统中安全和隐私问题的研究[J]. 计算机应用研究,2006(6):16-18.
- [4] Li Zhanglin, Lu Guizhang, Xin Yunwei. Application of reencryption technology in RFID security[J]. Computer Engineering,2008(3):160-162.
- [5] Juels A, Rivest R, Szydlo M. The blocker tag selective blocking of RFID tags for consumer privacy[C]// The 8th ACM Conference on Computer and Communications Security. New York, ACM Press, 2007:103-111.
- [6] Lin Guibin, Wang Yonghua, Zhan Yiju. RFID secure protocol based on random sequence[J]. Computer Engineering,2008(20):151-153.
- [7] Yuan Shuguang, Dai Hongyue, Lai Shengli. Hash-based RFID authentication protocol[J]. Computer Engineering, 2008(12):141-143.
- [8] Michael Burrows, Martin Abadi, Roger M Needham. A logic of authentication[J]. ACM Transactions on Computer Systems,2005,8(1):18-36.

作者简介:

李秀芝 女,(1982-),硕士研究生.研究方向为智能信息处理.

霍华 男,(1968-),博士,硕士生导师,副教授.研究方向为智能信息处理、光纤通道技术、嵌入式系统等.

马林 男,(1984-),硕士研究生.研究方向为光纤通道技术.

(上接第 184 页)

4 结束语

文中实现了基于嵌入微 Linux 内核的开机认证系统,并且采用 USB KEY 的方式分离存储硬盘重要信息及密钥,在密钥等重要信息的存储方面采用了成熟且加密效果优良的 MD5 与 DES 算法,从而增大了破解难度,很大程度上保障了计算机系统的安全.

参考文献:

- [1] Kephart J O. Biologically inspired defenses against computer viruses[C]// Proc. Int'l. Conf. on AI (IJCAI-95). Morgan Kaufmann, San Francisco, 1995:985-996.
- [2] White S R, Kephart J O, Chess D M. Computer viruses: a global perspective[C]// Proc. Fifth Int'l Virus Bulletin Conf. Virus Bulletin Ltd. England, Abingdon, 1995:165-181.
- [3] 廖金祥. 硬盘引导替换技术及其应用[J]. 武汉冶金科技大学学报,1998,21(1):1998.
- [4] Love R. Linux 内核设计与实现[M]. 2 版. 陈莉君,译. 北京:机械工业出版社,2006.
- [5] 胡向东. 应用密码学[M]. 北京:电子工业出版社,2006.
- [6] 赵炯. Linux 内核完全注释[M]. 北京:机械工业出版社,2004.

作者简介:

马富达 男,(1985-),硕士研究生.研究方向为网络信息安全.

蔡皖东 男,(1955-),教授,博士生导师.研究方向为网络信息安全.