内核移植及其保护方法①

王俊杰, 段俊瑞, 朱耀麟

(西安工程大学 电子信息学院, 西安 710048)

摘 要: 内核移植工作是一项具有实际应用价值和各项配置较为复杂的工作, 故提出一种依据内核本身特性实现内核更新的方法. 现有大部分内核移植的方法需要利用 module-init-atools、mkinitrd、lvm2 和 device-mapper 等工具进行, 而该文提出的移植方法摒弃了外部工具, 依据内核自身特点实现新内核更新及其策略保护. 首先, 进行内核的配置, 对必要模块进行分析配置; 然后利用内核特性进行的模块化编译和 grub 界面分析; 再次, 鉴于对内核的健壮性要求, 采用改进后 MD5 算法机制实现对内核的保护. 最后实验研究表明, 该内核移植方法便捷实用, 优化的保护机制对内核具有重要的应用价值.

关键词: Linux; 模块; kernel; grub; MD5

Method of Kernel Transplantation and its Protection

WANG Jun-Jie, DUAN Jun-Rui, ZHU Yao-Lin

(College of Electronics and information, Xi'an Polytechnic University, Xi'an 710048, China)

Abstract: The work of kernel transplantation has practical application value and the configuration more complex tasks. So this paper comes up with a new method of kernel updates on the basis of the kernel itself characteristics. Most of the existing kernel transplantation methods need to complete with the module init-tools, mkinitrd, lvm2 and device mapper and other tools, external tools are abandoned in this paper. It is to achieve the new kernel updates and its strategy is on the basis of the kernel protection characteristics. First of all, for kernel configuration, we analyze the necessary module configuration. Then it uses the command interface to the kernel of modularization compile and grub analysis. Again, in view of the robustness requirements for the kernel, the kernel is realized with the MD5 algorithm which can improve the mechanism of protection. Finally experimental research show that the kernel transplantation method is convenient and practical, and the improved protection mechanism of the kernel has important application value.

Key words: Linux; modules; kernel; grub; MD5

Linux 是一种实时性的操作系统, 可以在 x86, sparc, amd64, ppc, ppc64 等多种硬件平台上稳定运行, 同时是支持硬件平台最多的操作系统. Linux 的最大优势在于其代码开源, 开发人员只要遵循 GPL 协议, 就可以根据系统硬件以及应用需求对内核进行必要配置, 进而更改和开发并用以满足系统的应用需求. 本文 X86 平台为例, 根据数据存储的模块化开发设计思路, 深入分析实现了 3.10.26 内核的移植, 并在此基础上对内核的保护措施进行了优化.

基于 module-init-tools、mkinitrd、lvm2 和 device-mapper 等工具进行内核移植技术是一项被广泛应用的方法. 该方法具有一定的优势, 其具体实现过程是: 首先除了需要下载新内核(3.10.26)外, 还需要将这 4 个工具安装到系统中, 用以辅助新内核; 其次, 需要使用 module-init-tools 来完成对安装内核模块文件的支持, mkinitrd 软件用来实现为新内核的 iscsi 设备做的新 initrd 映像, lvm2 用以实现对系统磁盘的管理, 而 device-mapper 则用来管理系统的真实或则虚拟的

① 收稿时间:2014-06-22;收到修改稿时间:2014-07-14

块设备; 最后利用明文密码实现对 grub 文件的设计并完成新内核更新.

该方法虽然能实现系统新内核, 但其具有一定技术瓶颈: 其一, 系统工具要求过多, 系统资源不能发挥其作品用; 其二, 需要依据各工具的特性各步操作, 一旦系统没有某个工具, 则需要重新来完成, 这给新内核的更新带来了极大地不变; 其三, 在内核版本过高的情况下, 指定的工具不能实现必要的文件配置; 最后, 简单的明文 MD5 算法实现的密保机制很容易被破解.

因此, 根据以上技术局限性, 本文提出一种依据内核自身特性和文件属性实现新内核及其密保的方法. 该方法既解决了系统多工具和版本问题, 也使得内核的相关文件的特性发挥到极致, 同时也优化了系统的密保机制. 在本文中, 首先, 介绍了内核更新领域的相关工作; 然后, 详细论述了本文方法的设计思路与实现过程; 最后对研究工作进行实验验证和总结.

1 内核相关工作

功能模块的编译是内核配置中至关重要的一项, 其需要在指定目录下进行操作, 并且根据应用需求考虑将 modules 编译进内核还是在需要的时候在动态的编译进内核, 以实现系统的最优化. 内核移植是一般将下载的内核存放在/usr/src 目录下进行移植操作, 首先要切换至该目录下, 进入内核存放目录, 进行内核配置(本文若无特殊说明, 操作均在 /usr/src/linux-3.10.26 下完成). 首先使用指令 make mrproper 来清除之前的内核配置文件.config 和源代码树, 作用就是在进行编译内核前将原来配置文件全部删除, 从而确保在源代码目录下没有错误的.o 文件, 以免影响新内核的编译. 其次就是配置内核, 这里采用 make menuconfig 配置方式进行内核配置[1].

在选择各项时, 用户不会看到提示要保存的信息, 而这些信息会在用户所有配置选择完成准备退出图形编辑界面时才会提示是否需要保存这一次的配置. 用户在提示需要执行确定操作, 否则会前功尽弃, 导致的直接后果就是需要再次的重新配置, 影响任务进度.

2 内核移植方法

系统的模块支持是内核能否成功移植的一项至关重要的指标, 如果系统中没有装载某个模块, 不但不能满足用户的应用需求, 同时也会给系统运行时会带来不同程度的编译问题. 以文件系统为例, 如果系统中文件系统的模块选择不当, 将会直接影响到系统的更新, 在编译时会出现 kernel panic 而系统更新失败. 下面是对必要内核模块的详细配置, 从而实现满足需求的最小化系统[2].

2.1 模块配置

内核中有很多功能模块, 在配置时需要根据需要选择不同模块.

首先, 选择启用对可加载模块的支持. 对于经常要使用的驱动程序应该模块化, 需要的时候使用指令 insmod 加入核心, 不需要的时候则使用指令 rmmod 移除核心, 或者用 lsmod 查看当前所加载的模块.

其次, 需要选择磁盘的设备驱动模块, 对于 iscsi 硬盘, 选择选择对该类硬盘的驱动支持和网络支持. 此外, 也需要提供 mapper 设备支持.

最后是文件系统. ext2 是 linux 很早就开始用的文件系统, 支持反删除, 也就是说如果用户误删除文件, 在操作正常的情况下是可以删除的, 并且支持大文件; 而 ext3 文件系统是由 ext2 文件系统发展来的,应用于 linux 的日志文件系统, 同样支持大文件, 但不支持反删除操作. 所以, ext2 和 ext3 是 linux 的标准文件系统, 需要将这 9 项编译进模块, 如果直接编译到内核, reboot 内核启动时会出现: mount: error 19 mounting ext3 的错误提示.

表 1 模块配置

Choice	Modules
可加载模块的支持	Forced module loading
	Module versioning support
加载设备驱动模块	Block devices->Loopback device support
	>SCSI low-level drivers--->BusLogic SCSI support ->Multiple devices driver support (RAID and LVM)--->Device mapper support ->Graphics support--->Support for frame buffer devices ->Network device support--->Ethernet(10 or 100Mbit)---><*> AMD PCnet32 PCI support
文件系统	<M>Second extended fs support [*] Ext2 extended attributes [*] Ext2 POSIX Access Control Lists [*] Ext2 Security Labels [*] Ext2 execute in place support <*> Ext3 journalling file system support [*] Ext3 extended attributes [*] Ext3 POSIX Access Control Lists [*] Ext3 Security Labels

各模块的详细提供如表 1 所示. 各需求功能模块编译保存退出后会在当前目录下生成一个配置文件.config,

该文件是本次配置的生成的配置文件, 不可更改.

2.2 内核编译的实现

内核编译是一项繁琐而复杂的工作, 稍有不慎就会出现功能模块编译失败, 从而导致前功尽弃. 首先使用 make dep, 该指令是建立编译时所需的从属文件, 在内核没有编译过的情况下可以直接跳过这一步, 然后执行 make clean 来清除内核编译的目标文件. 未曾编译过的内核也可以跳过这一步继续下一步操作. 开始真正的编译内核 make bzImage, 该指令只会按行输出相关编译信息; 如果需要查看编译过程中的详细信息,可以在该指令后加上 V=1, 即指令 make bzImage V=1, 此时会输出编译的详细信息.

内核编译成功后, 会在 arch/x86/boot 目录中生成一个新内核的映像文件 bzImage; 编译可加载模块 make modules, 该指令是将内核各模块的*.c 文件编译为*.o 文件, 各模块编译成*.mod.o,在执行 makefile 后生成*.ko 文件,相当于多次执行 install *.ko. 接下来需要安装可加载模块 make modules, 将编译好的内核模块安装到系统的指定目录中[3]; 安装内核模块 make modules_install; 将新内核安装到系统中并自动用新内核必要的配置 make install.

2.3 grub 界面的分析与设计

资深软件工程师 M. Tim Jones 在 Inside Linux boot process 中绘制的系统启动流程如图 1 所示, 该图详细的解释了加电启动时系统的启动流程. 针对系统开机启动时, 屏幕界面显示的密密麻麻编码, 达不到界面的应有效果, 所以, 系统会采取相依机制来规避这种问题. 另外, 为防止一些黑客非法破坏系统, 用户还可以在启动界面增加密码保护机制, 从而使他人无法肆意更改内核启动配置, 这样就进一步增加了内核的健壮性和可恢复性.

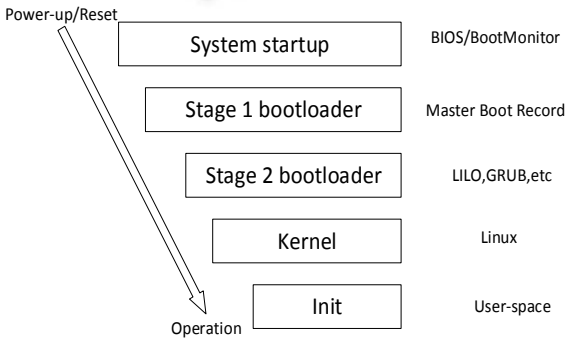

图 1 系统启动流程

启动界面的设置文件是/etc/grub/conf 文件, 该文件会在系统内核更新完成后进行自动添加, 但是如果用户不采用手动进行设置的话, 重新启动后新的内核是不能够运行的[4]. 以下是界面各项的设计和说明.

Default 项:

内核按 title 划分, 并且 title 行的内核名称可以根据用户需要进行自定义. Default 的等号右侧需要的是数字, 该数字可以为 0,1,.... 这些数字按 title 行的内核不同从 0 自上而下依次递增进行定义. 本文中 default=0 对应内核 CentOS (3.10.26), default=1 对应内核 CentOS (2.6.32-220.el6.x86_64).

Password 项:

为保证用户的方便和安全方面的需求, 防止其他用户进入编辑 kernel 菜单进入 single 模式启动系统. 在启动系统后, 用户需要修改模式的话, 系统就会给出提示输入 password, 这样就对用户的内核作了进一步加密. 例如在系统故障无法正常启动时, 用户可以在 grub 界面通过修改模式为 single 模式, 进入系统查看问题[5].

根据上述的配置, 可以实现新内核的移植工作, Password 项密保机制的优化输出项, 需要按优化后的操作进行必要输入. 所以, 密保机制的设计原理就相当重要了.

3 内核保护机制的优化

系统内核的保护被视为系统 s 安全的重中之重, 一旦系统内核被人恶意篡改, 那么整个系统都会面临无法估量的后果[6]. 所以, 要实现系统的安全, 需要在系统中实现对内核以及整个系统的一种保护机制, 常用的实现算法是 MD5.

众所周知, MD5 实现有明文密码和加密密码 2 种. 本文在加密密码设计的原理基础上, 实现一种改进的加密保护机制, 以此来提高了系统的健壮性能.

3.1 MD5 优化原理

MD5 以 512bits 分组来处理用户输入的信息, 且每一个分组又被划分为 16 个 32 位子分组, 经过一系列处理后, 算法的输出为四个 4Bytes 分组的级联后生成的 128 位散列值, 如图 2 所示.

根据 MD5 算法原理, 拿到待加密信息后, 首先需要对信息进行补充, 在末尾填充一个 0x80, 以及无数个 0x00, 使得填充后信息的总长度(Bytes)对 64 求余的

结果等于 56. 之后再填充一个以 8Bytes 表示的填充前信息长度(bits).

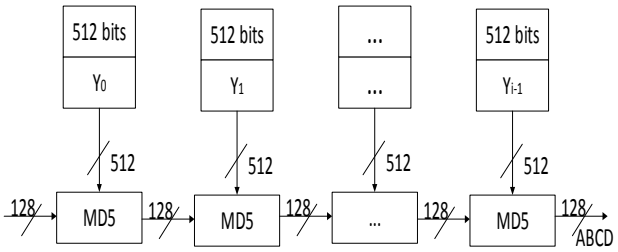

图 2　MD5 原理

接下来进入主循环的四轮运算. 每一轮都会调用 16 次相同的函数, 但参数不同. 每一轮所掉用的函数均不相同. 虽然有 4 种函数, 但参数的种类均一致. 参数 a, b, c, d 第一次进入主循环的初始值分别为:

a = 0x1234567　b = 0x89ABCDEF　c = 0xFEDCBA98

d = 0x7654321

之后再进入主循环时, 它们的值为上一个 64Bytes 分组的计算结果. 第五个参数*M. 它分别代表了每个子分组(数组指针 unsigned int *M[16]). 第六个参数是个常数, 函数中控制数据循环左移多少位. 第七个参数也是常数, 不过他有来历. 它是 2^32*abs(sin(i))的整数部分, i 从 1~64.

每个函数的实现过程.

```
#define FF(arg1, arg2, arg3, arg4, arg5, arg6, arg7)    \
    arg1 = arg2 + ((arg1 + F(arg2, arg3, arg4) + arg5 + arg7) << arg6)
#define GG(arg1, arg2, arg3, arg4, arg5, arg6, arg7) \
    arg1 = arg2 + ((arg1 + G(arg2, arg3, arg4) + arg5 + arg7) << arg6)
#define HH(arg1, arg2, arg3, arg4, arg5, arg6, arg7) \
    arg1 = arg2 + ((arg1 + H(arg2, arg3, arg4) + arg5 + arg7) << arg6)
#define II(arg1, arg2, arg3, arg4, arg5, arg6, arg7) \
     arg1 = arg2 + ((arg1 + I(arg2, arg3, arg4) + arg5 + arg7) << arg6)
```

大部分都是一样的, 区别只在于内部调用的 F()、G()、H()、I(). 最后再来看看 F()、G()、H()、I()这四个函数是如何实现的.

```
#define F(X, Y, Z)    (((X) & (Y)) | ((~X) & (Z)))
#define G(X, Y, Z)    (((X) & (Z)) | ((Y) & (~Z)))
#define H(X, Y, Z)    ((X) ^ (Y) ^ (Z))
#define I(X, Y, Z)    ((Y) ^ ((X) | (~Z)))
```

经过主循环的这四轮运算, 将 a、b、c、d 加回到上一个 64Bytes 分组的运算结果上, 就得到了当前 64Bytes 分组的运算结果. 所有分组计算完之后, 最后的运算结果如果为大端系统, 直接将 a、b、c、d 连在一起输出就好了; 如果为小端系统, 需要进行转换.

3.2 算法实现步骤

1) 初始化 a、b、c、d;

2) 从标准输入读取待加密字符串, 每次读取 64 Bytes;

3) 因为填充的信息在待加密信息的末尾, 所以只要还没读到待加密信息末尾, 前面的 64Bytes 分组(零个或多个)正常进行主循环运算;

4) 当读到的待加密信息不足 64 Bytes 时, 需要对待加密信息进行填充,填充后继续进行主循环运算;

5) 小端系统将运算结果转换后输出; 大端系统直接级联输出.

3.3 实验结果与分析

首先需要用户以 root 权限在 CLI 下的任意目录下执行指令/sbin/grub-md5-crypt, 该指令输入确认后会生成一行乱码(*), 相关操作如图 3 所示. 在 grub 文件下以 password --md5 *格式添加后保存退出即完成会加密密码的设置.

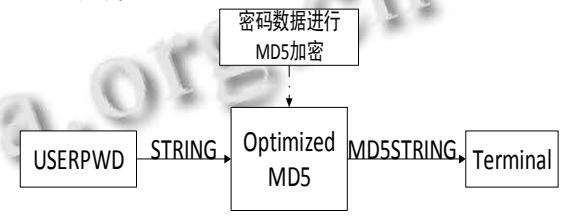

图 3　操作流程

```
[root@localhost ~]# /sbin/grub-md5-crypt
Password:
Retype password:
$1$XgYrn1$Q1vKcXLWs10tbrPcgXQJc/
```

两者模式的不同之处在于: 密文密码, 对于以单用户模式登录的非法用户, 通过查看 grub 文件就就可以获取系统的登录密码, 从而可以以正常方式登录系统; 而对于加密方式而言, 即便是登录后查看了 grub 文件, 其获得的也仅仅是一行乱码, 无法进行正常的登录. 另外, 在 grub 文件中也可以加上 lock 行, 其作

用是每次系统在启动时都会要求输入 grub 密码才能进入系统. 如果不添加, 则需要在修改 grub 时才要求输入密码.

4 结语

本文在内核移植的基础上, 摒弃 module-init-tools、mkinitrd、lvm2 和 device-mapper 等内核移植工具, 依据内核自身的特点进行更新, 使得移植更加简便快捷, 在一定程度上实现了内核的高效性[7]. 然而随着内核的开发, 其保护机制变得更加重要. 基于此, 在传统的 MD5 算法基础上作出改进, 实现 128bits 散列值的密保机制. 结合优化后的效果来看, 该机制密保变得更加具有实际应用价值. 从内核的角度考虑, 优化的 MD5 保护机制, 是一种贴近实际应用需求的方法.

参考文献

1 杨浩,王正元,孟庆民,张艳彬.Little Kernel 分析与移植.中国科技信息,2012,(22):69–71.
2 黄聪会,陈靖,张黎,李东阳.软件移植理论与技术研究.计算机应用研究,2012,29(6).
3 李小航.虚拟嵌入式开发环境中的 Linux 内核移植与剪裁.科技广场,2011,(9).
4 钟约夫,王瑞勋.桌面虚拟化应用中虚拟环境评估与规划的研究.自动化与仪器仪表,2011,(1).
5 Bill Blunder,杨涛译.虚拟机的设计与实现.北京:机械工业出版社,2003:89–110.
6 Bovet DP,陈赫君译.深入理勰 Linux 内核.北京:中国电力出版社,2007.
7 王保平.GRUB 实现多系统统一引导的应用研究研究.微计算机应用,2008,11(29).