qiuy_mac L65 2015-06-26 18:25 Linux 计算机专业课

实验代码

“操作系统原理与实践”实验报告

基于内核栈切换的进程切换

问题一

(1) 为什么要加4096？

因为应用程序从用户态进入内核态时，其内核栈必须是一个空栈，即*esp* 必须位于申请到的一页内存的末端（**高地址**），一页内存为4*k*，此处的基本单位是 **字节** ，*4096* 8bit=4*k* ，***正好将****tss* ***中内核栈的*** esp*的值设置到一页内存的末端。应用程序一进入内核，就从一页内存的末端向低地址进行压栈操作。

(2) 为什么没有设置tss 中的ss 0？

因为所有进程都共用 **进程0** 的*tss* ，而进程0的*tss* 在初始化时已经设置好了*ss0* ，所以不需要再设置*tss* 中的*ss0* 了。

问题二

(1) 子进程第一次执行时，*eax=0* 。这样fo rk()函数就可以通过返回值区分子进程和父进程。在*fork.c* 中的*copy_process()* 函数中语句*(--krnstack)=0* 做了这个工作。

(2) *ebx* 和*ecx* 来自*fork* 系统调用执行*int 0x80* 时，压入栈的*ebx* 和*ecx* 的值，它们是系统调用的参数，将它们写到子进程的内核栈中保证了子进程和父进程的一致性。

(3) *ebp* 同样来自系统调用及处理过程中的压栈的参数，*ebp* 指向内核栈的栈底，这样设置在执行*popl %ebp* 时，可以将内核栈的基址指针指向内核栈栈底，为后续的指令找到正确的栈底。

问题三

这样在寄存器的隐藏部分就有了段基址和段限长，方便日后的操作，提高执行指令的效率。必须要等到切换完*LDT*之后，这样才能保证寄存器的隐藏部分是新的进程的段基址和段限长，如果在切换*LDT*之前重新设置*fs=0x17* ，那么寄存器的隐藏部分将是之前被切换出去的进程的段基址和段限长，是错误的。