# 进程0的LDT0代码段描述符分析

2018年10月12日 23:36:57 [Icoding\_F2014](https://me.csdn.net/jmh1996) 阅读数：47更多

所属专栏： [操作系统高级教程](https://blog.csdn.net/column/details/29164.html)

 版权声明：本文为博主jmh原创文章，未经博主允许不得转载。 https://blog.csdn.net/jmh1996/article/details/83034195

LDT0的值为

····

{ \

{0,0}, \

/\* ldt \*/ {0x9f,0xc0fa00}, \

{0x9f,0xc0f200}, \

},

···

其中第二项{0x9f,0xc0fa00}为代码段的描述符。

然后看task中对ldt的定义:

struct task\_struct {

/\* these are hardcoded - don't touch \*/

long state; /\* -1 unrunnable, 0 runnable, >0 stopped \*/

long counter;

long priority;

long signal;

struct sigaction sigaction[32];

long blocked; /\* bitmap of masked signals \*/

/\* various fields \*/

int exit\_code;

unsigned long start\_code,end\_code,end\_data,brk,start\_stack;

long pid,father,pgrp,session,leader;

unsigned short uid,euid,suid;

unsigned short gid,egid,sgid;

long alarm;

long utime,stime,cutime,cstime,start\_time;

unsigned short used\_math;

/\* file system info \*/

int tty; /\* -1 if no tty, so it must be signed \*/

unsigned short umask;

struct m\_inode \* pwd;

struct m\_inode \* root;

struct m\_inode \* executable;

unsigned long close\_on\_exec;

struct file \* filp[NR\_OPEN];

/\* ldt for this task 0 - zero 1 - cs 2 - ds&ss \*/

struct desc\_struct ldt[3];

/\* tss for this task \*/

struct tss\_struct tss;

};

其中 ldt的类型为结构体:desc\_struct  
可以得知desc\_struct的定义为：

typedef struct desc\_struct {

unsigned long a,b;

} desc\_table[256];

就是两个long的整数，而每个long都是4个字节,32位的。

ok,我们把{0x9f,0xc0fa00}展开完整(8个字节):0x0000 009f,0x00c0 fa00;其中第一个数在低32位,第二数在高32位  
也就是:  
高：  
0x00c0  
0xfa00  
低：  
0x0000  
0x009f  
转换为bit:  
c为1100

63-48:00000000 11000000

47-32:11111010 00000000

31-16:00000000 00000000

15-00:00000000 10011111

根据段描述符的定义:  
15-00为段限长:0x009f,其单位根据G=1,为说明单位是4k。0x9f \* 4k=636k。所以段限长为636kb.  
31-16(最低16位):00000000 00000000 ,39-32(次低8位):00000000,63-56(高8位):00000000 合并起来构成段基址：00000000 00000000 00000000 00000000,就是地址0.  
它的DPL为46-45位:11,也就是3

再看看内核的GDT 代码段描述符号:

gdt: .quad 0x0000000000000000 /\* NULL descriptor \*/

.quad 0x00c09a0000000fff /\* 16Mb \*/

.quad 0x00c0920000000fff /\* 16Mb \*/

.quad 0x0000000000000000 /\* TEMPORARY - don't use \*/

.fill 252,8,0 /\* space for LDT's and TSS's etc \*/

第2项0x00 c0 9a 00 00 00 0f ff 为内核代码段  
高:  
0x00c0  
0x9a00  
低:  
0x0000  
0x0fff

63-48:00000000 11000000

47-32:10011010 00000000

31-16:00000000 00000000

15-00:00001111 11111111

它的31-16(最低16位)为00000000 00000000 ,39-32(次低8位)为00000000,63-56(高8位)为00000000 合并起来构成段基址：00000000 00000000 00000000 00000000  
DPL:00,也就是0

因此，进程0的代码段基址与内核的段基址是相同的，都为0.