



2023-02-23, 08:37

LINUX

Operating System

从零开始 DEVELOPING YOUR OWN 自制操作系统 OPERATING SYSTEM FROM SCRATCH O Brave New Kernel

安装 GDT

OPERATING SYSTEM DEVELOPMENT TUTORIAL SERIES

EP 5-6



指令 lgdt : 加载 GDT 的地址到 GDTR

用法: lgdt (%reg)

寄存器 %reg 的值是
GDTR 的值的地址

WHY?

GDTR 的大小为 48bits
而 x86 的操作数宽度和所有
通用寄存器只有 32bits
所以 GDTR 的值只能放在内存，而后由 CPU 直接从内存
读出。

Recall:

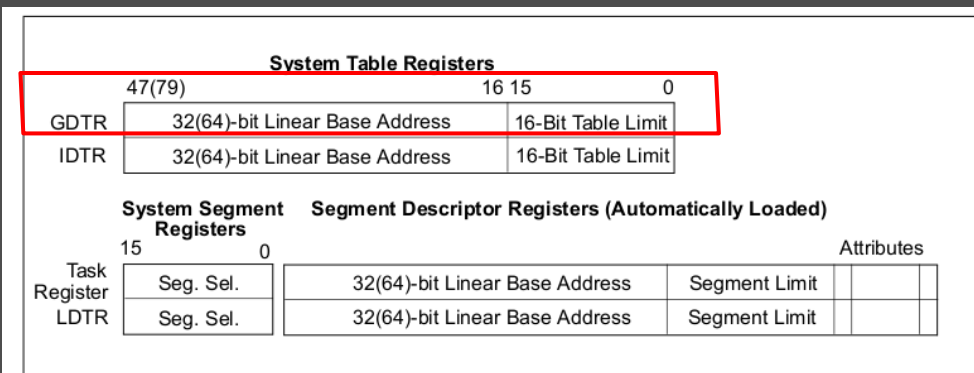


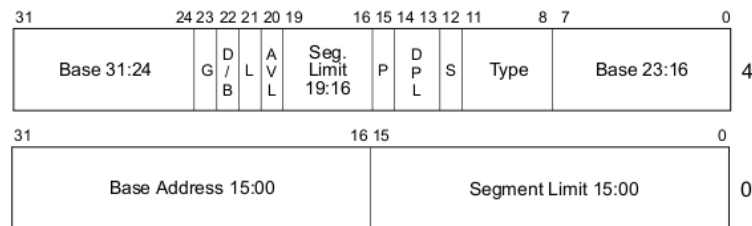
Figure 2-6. Memory Management Registers



需要处理的寄存器：

CS, ES, SS, DS, FS, GS

gdt 结构：



- L — 64-bit code segment (IA-32e mode only)
- AVL — Available for use by system software
- BASE — Segment base address
- D/B — Default operation size (0 = 16-bit segment; 1 = 32-bit segment)
- DPL — Descriptor privilege level
- G — Granularity
- LIMIT — Segment Limit
- P — Segment present
- S — Descriptor type (0 = system; 1 = code or data)
- TYPE — Segment type

Figure 3-8. Segment Descriptor

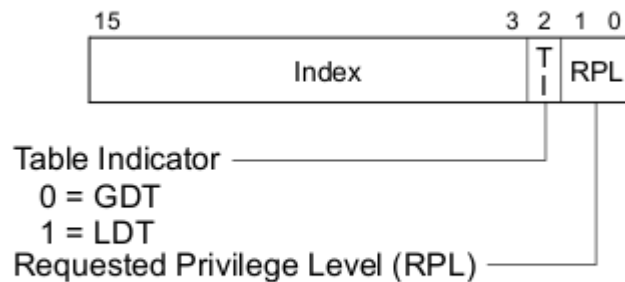


Figure 3-6. Segment Selector

Let's Code



修改 IP：各种 jump，返回，调用（近距离）

修改 CS（与 IP）：远距离 { 跳转 | 返回 | 调用 }

Example：retf - 远距离返回 (far return)

RETURN-TO-SAME-PRIVILEGE-LEVEL:

IF the return instruction pointer is not within the return code segment limit

THEN #GP(0); FI;

IF OperandSize = 32

THEN

EIP := Pop();

CS := Pop(); (* 32-bit pop, high-order 16 bits discarded *)

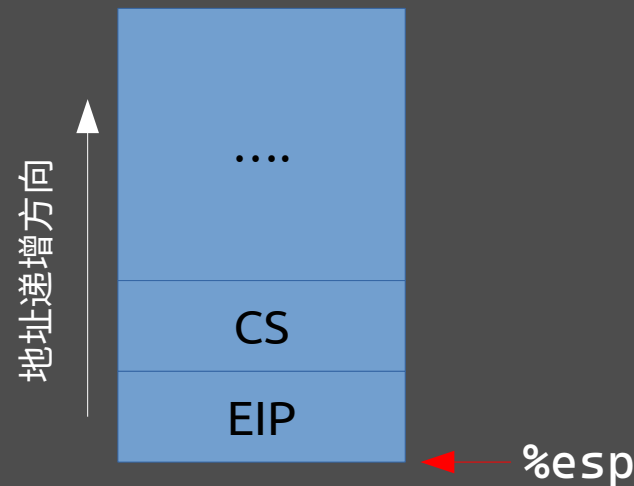
ELSE (* OperandSize = 16 *)

EIP := Pop();

EIP := EIP AND 0000FFFFH;

CS := Pop(); (* 16-bit pop *)

FI;



```
pushl <new CS value>
pushl <label>
retf
```

Source: Intel manual. pp.1748