

حالت حفاظت شده

قسمت دوم

در این قسمت درباره تنظیم کردن حالت حفاظت شده بصورت low level بحث خواهیم کرد. این آموزش فقط به شما طریقه تنظیم حالت حفاظت شده را میگوید و خیلی از مباحث پیشرفته در این مقاله بحث نمیشود!

جداول توصیفگر (Descriptor Tables)

ما با Descriptor Table ها شروع میکنیم.

آنها چکار میکنند؟

آنها اطلاعات آدرس دهی از حالت واقعی به حالت مجازی را در خود نگه داری میکنند. هر مدخل (entry) در Descriptor Table فرمت مشخص خود را دارا میباشد. هرکدام طول، آدرس فیزیکی و بعضی از آنها خواص را نگه داری میکنند. خواص همچنین میتوانند بصورت فقط خواندنی/اجرا شدنی تنظیم شوند. Privilege mode : برنامه باید در Ring 3 پایین ترین سطح privilege mode اجرا شوند که کمترین حق دسترسی به اجزای پایه سیستم را بصورت مستقیم داشته باشد ولی سیستم عامل باید در Ring 0 اجرا شود که بالاترین سطح بوده و تمامی حقوق دسترسی به آن تعلق میگیرد.

تنظیم جداول

قبل از اینکه به حالت محافظت شده سوئیچ کنید شما باید جداول زیر را تنظیم کنید:
(در واقع شما فقط باید جداول IDT و GDT را تنظیم کنید و LDT را در اوان ورود به حالت محافظت شده خالی بگذارید)
خب حال من به شما تنظیم GDT را نشان خواهیم داد. ما میتوانیم به IDT کلک زده و بعداً آنرا تنظیم کنیم!
در مورد LDT هم همانطور که گفتیم این جدول، جدول ضروری برای سوئیچ کردن به حالت محافظت شده نیست و میتوان از آن چشم پوشی کرد.
خب در قدم اول ما باید جایی در حافظه برای IDT پیدا کنیم و آنرا به 0 تنظیم کنیم. و بعداً از طریق خط فرمان آنرا اصلاح کنیم. IDT (Interrupt Description Table) استفاده نخواهد شد! (البته منظور در این مقاله هست).
شروع کار:

LIDT [QWORD PTR IDT_BASE_ADDR] ; Load Interrupt Descriptor Table

IDT_BASE_ADDR میتواند مانند زیر تنظیم شود:

**IDT_BASE_ADDR dw 256*8 ; Size of IDT
 dd 6000h ; Physical address in Memory**

اولین مقدار (dw 256*8) اندازه آن میباشد. مقدار بعدی (6000h) جایگاه آن در حافظه میباشد.

**GDT_BASE_ADDR dw (8*8192) - 1 ; Size of GDT
 dd 6000h + 256*8 ; Physical address in Memory**

ما GDT را دقیقاً بعد از IDT در حافظه قرار خواهیم داد.

LGDT [QWORD PTR GDT_BASE_ADDR] ; Load Global Descriptor Table

قبل از اینکه دستور بالا را اجرا کنید باید دو دستور قبلی که منجر به مقیم شدن GDT و IDT در حافظه میشود را اجرا کنید.

در قدم بعدی من به شما طریقه ساخت GDT را نشان خواهیم داد(که با انجام اینکار شما خواهید توانست GDT را در آدرس فیزیکی حافظه (RAM) قبل از تنظیم قرار دهید این عمل هنگامی رخ میدهد که شما میخواهید به حالت حفاظت شده پرش کنید).

اولین descriptor تهی (null) میباشد. descriptor ها معمولاً ۸ بایت طول دارند همچنین descriptorها از آدرس 8h شروع میشوند این بخاطر قالب انتخابگرها (selector) میباشد:

The bit layout:

ssss stpp

s = انتخابگر (Selector) از 0 شروع میشود (descriptor تهی که استفاده نمیشود) سپس به 1 میرود.

1,2,3,4,5,... مدلهایی در GDT میباشند.

اما هنگامی که آنها به بیتهای بالاتر آدرس دهی میشوند مقادیر واقعی اینها هستند:

8h,10h,18h,20h,28h,etc.

t به معنای شماره جدول میباشد. 0=GDT, 1=LDT. خوب با این اوصاف آنها خود میدانند که به کدام جدول باید رجوع کنند.

pp به معنای privilege level میباشد از 0 تا 3 اما در flat model فقط 0 و 3 استفاده میشوند(حالا فهمیدید چرا ویندوز دو Ring بیشتر ندارد؟)

شما از حالت multisegmented برای داشتن حالتی 1,2 استفاده کنید. با استفاده از این مدل شما قابلیت بسیار انعطاف پذیر و پیشرفته‌ای برای کار با segmentها را بدست می‌آورید ولی این مدل بسیار پیچیده و سردرگم میباشد.

اکثر سیستمهای عامل از حالت falt استفاده میکنند با استفاده از این مدل شما از تمامی قابلیت‌های CPU استفاده نخواهید کرد ولی مزیتی که این روش به روش multisegmented دارد راحتی پیاده سازی آن میباشد.

چینش GDT بصورت زیر میباشد:

اولین Word = پایین ترین Word از طول segment

دومین Word = پایین ترین Word از آدرس فیزیکی

Byte بعدی = پایین ترین byte از بالاترین word که اشاره به آدرس فیزیکی دارد.

Byte بعدی = خاصیتها = bit در دسترس، DPL، bit حاضر، bit سیستم و

Byte بعدی = پایین ترین nibble (نیم بایت) که بالاترین قسمت طول segment میباشد. بالاترین nibble دارای bit

گرانولیته میباشد، اندازه پشته و در دسترس بود (Available)

آخرین Byte = بالاترین Byte از آدرس فیزیکی پایه.

(برای توضیحات بهتر به مستندات شرکت اینتل مراجعه کنید)

```

dq 0 ; NULL Descriptor
; 8h Selector
dw 0FFFFh ; Segment Length
dw 0h ; Low Word Base Address
db 8 ; Low Byte of High Word of Base Address
db PRESENT_BIT or DPL0 or SYSTEM_BIT or READ_EXE_BIT or CODE_BIT
db 0Fh or GRAN_PAGE_BIT or DEFAULT_SIZE32 ; High Nibble of Segment Length
db 0 ; High Byte of Base Address
    
```

و حالا اولین segment را تنظیم میکنیم و آن 8h selector میباشد. و ما به 8h:0 پرش خواهیم کرد هنگامیکه به

Pmode پرش میکنیم در زیر من برای راحتی کار شما مقادیر پیش فرض را تنظیم کرده‌ام این مقادیر به به آدرس 8000h

در حافظه اشاره دارند جایی که کد ۳۲ بیتی pmode اجرا خواهد شد.

; Equate Constants

```

DPL0 EQU 00h
DPL1 EQU 20h
    
```

```

DPL2      EQU 40h
DPL3      EQU 60h
SYSTEM_BIT EQU 10h
NO_SYSTEM_BIT EQU 0h
PRESENT_BIT EQU 80h
NO_PRESENT_BIT EQU 0h
GRAN_PAGE_BIT EQU 80h
GRAN_BYTE_BIT EQU 0h
AVL_BIT    EQU 10h
UNAVL_BIT  EQU 0
DEFAULT_SIZE32 EQU 40h
DEFAULT_SIZE16 EQU 0
DATA_BIT    EQU 0
CODE_BIT    EQU 8
CONFORM_BIT EQU 4
NO_CONFORM_BIT EQU 0
READ_EXE_BIT EQU 2
EXE_ONLY_BIT EQU 0
ACCESSED_BIT EQU 1
CLR_ACCESSED_BIT EQU 0

```

سوئیچ کردن به حالت pmode :

سوئیچ کردن بسیار ساده میباشد:

```

MOV EAX, CR0
OR AL, 1
MOV CR0, EAX

```

همین!!!

بگذارید یک مطلب را برای شما روشن کنم اگر شما از کد بالا برای ساخت یک bootloader برای سیستم عامل جدید خود استفاده میکنید هیچ مشکلی ندارید ولی اگر از این کد در DOS استفاده کنید ممکن است به مشکل برخورد کنید به دو دلیل :

۱- ویندوز (منظور ویندوزهای قبل از ۹۵ میباشد)

۲- نرم افزارهای مدیریت حافظه

اگر نرم افزار مدیریت حافظه‌ای در حال اجرا باشد شما نمیتوانید به حالت pmode بروید. برای امتحان DOS که میخواهیم برروی آن نرم افزار گسترش دهنده حالت محافظت شده خود را اجرا کنیم باید کدی مانند زیر را در آن بگنجانیم:

```

MOV EAX, CR0
TEST AL, 1
JNZ SHORT ERROR_MEMORY_MANAGER
OR AL, 1
MOV CR0, EAX

```

این کد قادر خواهد بود که نرم افزارهای مدیریت حافظه ای که در داس قرار دارند را پیدا کند. اگر بیت PM تنظیم شده باشد یک نرم افزار مدیریت حافظه وجود دارد در ویندوز ۹۵ همچنین DOS در ماشین مجازی قرار دارد و بیت PM تنظیم نخواهد شد! خب چگونه میتوانیم این نرم افزارها را تشخیص بدهیم هنگامی که در ویندوز ۹۵ هستیم ؟ مانند زیر:

```

MOV EAX, CR0
TEST AL, 1
JNZ SHORT ERROR_MEMORY_MANAGER
OR AL, 1
MOV CR0, EAX
MOV EAX, CR0
TEST AL, 1
JZ SHORT ERROR_WINDOWS95

```

شما میتوانید از این کلک استفاده کنید :

چون در ویندوز ۹۵ ما نمیتوانیم بیت PM را تنظیم کنیم. اول آنرا تنظیم کرده و آن را چک میکنیم اگر این بیت تنظیم نشد به این معنا میباشد که ما در ویندوز ۹۵ هستیم و دیگر نمیتوانیم کاری انجام دهیم.

شما بعد از اینکه Pmode را تنظیم کردید نیاز به پاک کردن صف prefetch خواهید داشت:

البته بعضیها این کد را قبل از تنظیم pmode انجام میدهند:

```
JMP $+2  
NOP  
NOP
```

اما من کدهایی که از دستورات بالا هم استفاده نکرده‌اند را هم دیده‌ام. شما میتوانید بعد از تنظیم pmode هرگونه که میخواهید کدنویسی کنید.

در آخر شما باید یک selector در GDT خود داشته باشید که قبلاً بصورت ۳۲ بیتی کامپایل شده باشد و در حافظه در جایی مشخص لود شده باشد که بتوانید بوسیله یک selector دیگر به آن اشاره کنید.

خب کد زیر باعث یک پرش دور میشود که بوسیله دستورات اسمبلی معمول انجام نمیشود! (بحثهای بسیاری در این مورد هست بسیاری معتقدند که اینکار باید به همین صورت انجام شود و بسیار دیگر بر این اصل قائل نیستند من فقط در اینجا این دستورالعملها را می‌آورم).

db 67h	; 32 Bit Memory Address
db 66h	; 32 Bit Instruction
db 0Eah	; Far Jump Opcode
dd 0h	; Memory Offset in Selectory
dw 8h	; Selector

آخرین نکته :

وقتی که ثباتهای selector را قبل از ورود به حالت محافظت شده تنظیم کنیم و بعد از آن به حالت pmode پرش کنیم باعث از بین رفتن مقادیر ثباتها میشویم و دوباره باید آنها را تنظیم کنیم خب بهترین راه حل اینست که اول به حالت pmode پرش کرده و بعد ثباتهای selector را تنظیم کنیم.

پایان قسمت دوم

Translated by NETSPC
Persian OS group(_LOVE_CODER_,MASTER,NETSPC)
Please contact us at

os@persiasecure.com
http://groups.google.com/group/Persian_OS
<http://www.persiasecure.com/OS>

Released in 2006 April