Protección

Verificación de Permisos y Niveles de Privilegio

Organización del Computador II

Departamento de Computación - FCEyN UBA

12 de Noviembre de 2019

Motivación

- Necesitamos restringir de alguna manera las acciones que pueden efectuar las tareas (y procesos) de un sistema
- De otra manera no podemos garantizar que este funcionará correctamente
- Para ello tenemos los mecanismos protección, los cuales varían en efectividad según el sistema y el contexto de uso ¿Hay algún sistema 100 por ciento efectivo?
- Nosotros nos enfocaremos en restringir las lecturas, ejecuciones y escrituras no deseadas
- También buscaremos restringir el acceso a instrucciones importantes del sistema

Estructuras

Al acceder a memoria en un sistema con paginación y segmentación...

- **¿Cómo se resuelven las direcciones?**

  $$\text{Lógica} \xrightarrow[\text{Segmentación}]{} \text{Lineal} \xrightarrow[\text{Paginación}]{} \text{Física}$$

- **¿Qué estructuras son necesarias para la segmentación?**

  GDT / LDT

- **¿Y para la paginación?**

  Page Directory (PD) y Page Table (PT)

Segmentación: Selector

Veamos ahora los mecanismos de protección asociados a segmentos:

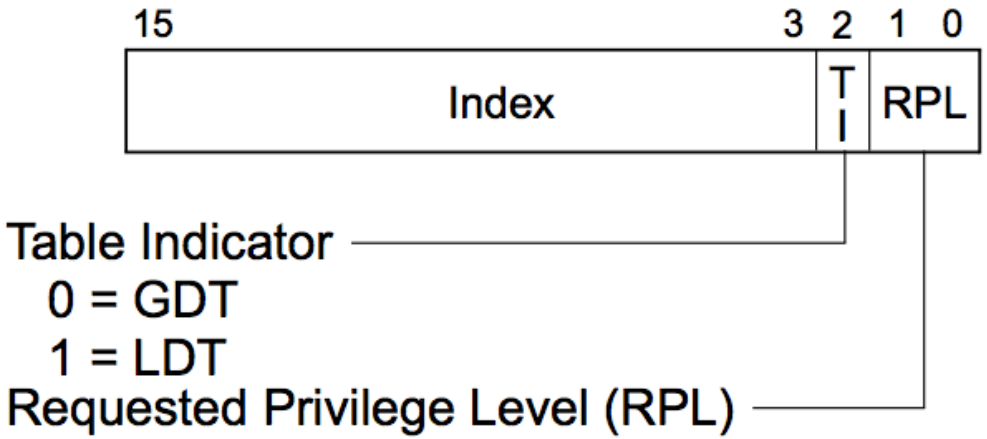
- **¿Cómo accedemos a un segmento?**

  A través de una dirección lógica

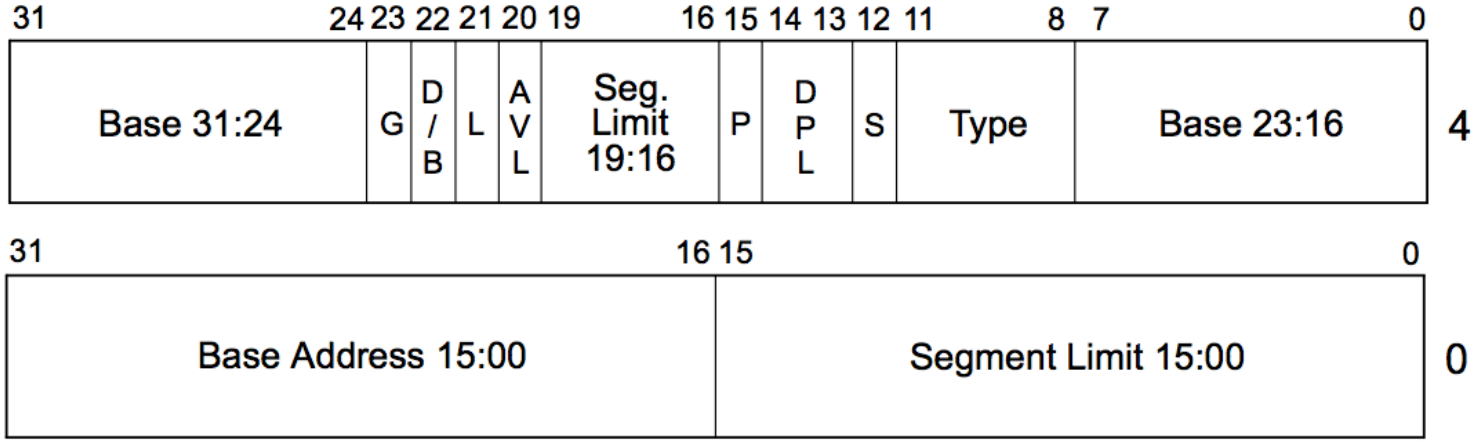
- **¿Qué forma tiene esta dirección?**

  → `(selector-segmento : offset)`

- **¿Cómo sabemos a qué segmento estamos referenciando?**



Segmentación: Descriptor GDT



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

Segmentación: Verificación de la dirección

En base a lo anterior...

- **¿Cómo sabemos si el segmento seleccionado está presente?**

  Vemos si el bit P de su respectivo descriptor está en 1

- **¿Y para saber si la operación no supera el límite del segmento?**
  - Si el bit de granularidad (G) es 0:

    offset + bytes a referenciar - 1 $\leq$ límite
  - Si no, tenemos varias alternativas:
    - offset + bytes a referenciar - 1 $\leq$ ((límite + 1) * 4kb) - 1
    - offset + bytes a referenciar - 1 $\leq$ ((límite + 1) $<<$ 12) - 1
    - offset + bytes a referenciar - 1 $\leq$ (límite $<<$ 12) + 0xFFF
- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** $\rightarrow$ `General Protection Fault (#GP)`

Segmentación: Niveles de privilegio

Definimos:

- **DPL (Descriptor Privilege Level)**:

  Es el nivel de privilegio del segmento a ser accedido (está en su descriptor)

- **CPL (Current Privilege Level)**:

  Es el DPL del segmento de código que estamos ejecutando

- **RPL (Requested Privilege Level)**:

  Es el nivel de privilegio marcado por los dos bits menos significativos del selector de segmento ¿Se puede cambiar?

- **EPL (Effective Privilege Level)**:

  Es el máximo numérico entre el CPL y el RPL

  $\rightarrow$ `EPL = Max(CPL,RPL)`

Segmentación: Verificación de niveles de privilegio

Con las definiciones anteriores:

- **¿Cómo sabemos si tenemos el nivel adecuado para acceder al segmento?**

  Depende del tipo de segmento:

  - Si el segmento es de **datos** $\rightarrow$ `EPL` $\leq$ `DPL`
  - Si es un segmento de **código** tenemos dos casos:
    - Non-conforming $\rightarrow$ `EPL` $=$ `DPL`
    - Conforming $\rightarrow$ `CPL` $\geq$ `DPL` **(Ojo que no se usa el EPL)**

  Nota: Las comparaciones $=$, $\geq$, $\leq$ son por valor numérico

- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** $\rightarrow$ `General Protection Fault (#GP)`

Segmentación: Tipos de Entradas

Type Field					Descriptor Type	Description
Decimal	11	10 E	9 W	8 A		
0	0	0	0	0	Data	Read-Only
1	0	0	0	1	Data	Read-Only, accessed
2	0	0	1	0	Data	Read/Write
3	0	0	1	1	Data	Read/Write, accessed
4	0	1	0	0	Data	Read-Only, expand-down
5	0	1	0	1	Data	Read-Only, expand-down, accessed
6	0	1	1	0	Data	Read/Write, expand-down
7	0	1	1	1	Data	Read/Write, expand-down, accessed
		C	R	A		
8	1	0	0	0	Code	Execute-Only
9	1	0	0	1	Code	Execute-Only, accessed
10	1	0	1	0	Code	Execute/Read
11	1	0	1	1	Code	Execute/Read, accessed
12	1	1	0	0	Code	Execute-Only, conforming
13	1	1	0	1	Code	Execute-Only, conforming, accessed
14	1	1	1	0	Code	Execute/Read, conforming
15	1	1	1	1	Code	Execute/Read, conforming, accessed

Segmentación: Verificación de tipos

Ahora, según la operación a efectuar...

- **¿Qué tipo de segmentos nos permiten leer, escribir o ejecutar?**

  Los que tienen el bit de sistema (S) en 1

- **¿Cuáles nos permiten realizar lecturas?**

  Los de datos y los de código con lectura habilitada

- **¿Y escrituras?**

  Los de datos con escritura habilitada

- **¿Y en el caso de las ejecuciones?**

  Los de código

- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** $\rightarrow$ `General Protection Fault (#GP)`

Paginación: PDE y PTE

- Habiendo superado la etapa anterior pasamos a los mecanismos de protección correspondientes a paginación
- En este caso atendemos al contenido de las entradas del directorio (PDE) y la tabla de páginas (PTE) correspondientes a la página a acceder:

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
Address of 4KB page frame																				Ignored			G	PAT	D	A	PCD	PWT	U/S	R/W	1	**PTE: 4KB page**
Address of page table																				Ignored				0	Ign	A	PCD	PWT	U/S	R/W	1	**PDE: page table**

Paginación: Verificación de niveles de privilegio y atributos

- **¿Cómo sabemos si la página a acceder está presente?**

  Vemos el bit de presente (P) de su PTE

- **¿Y para saber si la tabla de la PTE lo está?**

  Vemos el bit P de la PDE que la referencia ¿Puede suceder que el directorio no esté presente al tener paginación habilitada?

- **¿Qué simboliza el bit R/W de las entradas?**

  El bit Read/Write indica:
  - **0**: Sólo hay permisos lectura (Read Only)
  - **1**: Hay permisos de lectura y escritura (Read Write)

- **¿Y el bit U/S?**

  El bit User/Supervisor indica:
  - **0**: Sólo hay permisos de acceso para nivel supervisor (Supervisor)
  - **1**: Hay permisos de acceso para los niveles supervisor y usuario (User)

Nota: El nivel usuario se corresponde sólo con el nivel 3 de segmentación

Paginación: Verificación combinando PTE y PDE

Ahora, suponiendo que las entradas y la página están presentes...

- **¿Qué páginas permiten escrituras?**

  Aquellas cuyas PDE y PTE tengan R/W en 1

- **¿Y lecturas?**

  Todas las combinaciones lo permiten

- **¿Qué páginas pueden ser accedidas por nivel usuario?**

  Aquellas cuyas PDE y PTE tengan U/S en 1

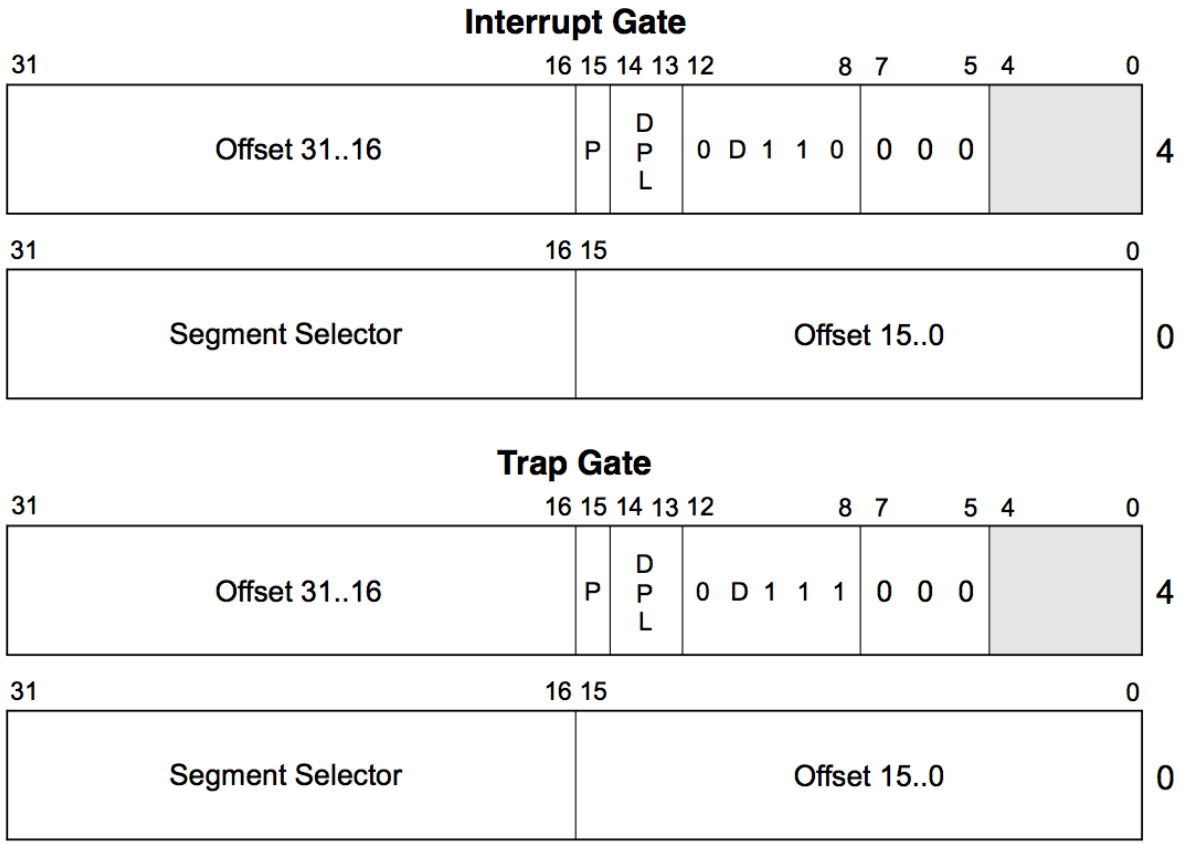
- **¿Y supervisor?**

  Todas las combinaciones lo permiten ¿Podría el nivel supervisor escribir en una página de sólo lectura?

- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** → `Page Fault (#PF)`

Interrupciones: Descriptor de IDT

Veremos ahora cuándo se puede llamar a una rutina de atención a interrupción (RAI) dada en base a cómo está descrita su interrupción:



Interrupciones: Verificación de privilegios

- **¿Cómo sabemos si la entrada de una interrupción está presente?**

  Vemos el bit P de su descriptor en la IDT

- **¿Y para saber en qué segmento se ejecuta su RAI?**

  Usamos el selector de segmento de su descriptor en la IDT

- **¿Cómo sabemos si tenemos el nivel de privilegio necesario para llamar a una interrupción (instrucción `INT`)?**

  Verificamos que `CPL` $\leq$ `DPL` siendo estos los bits de su descriptor en la IDT

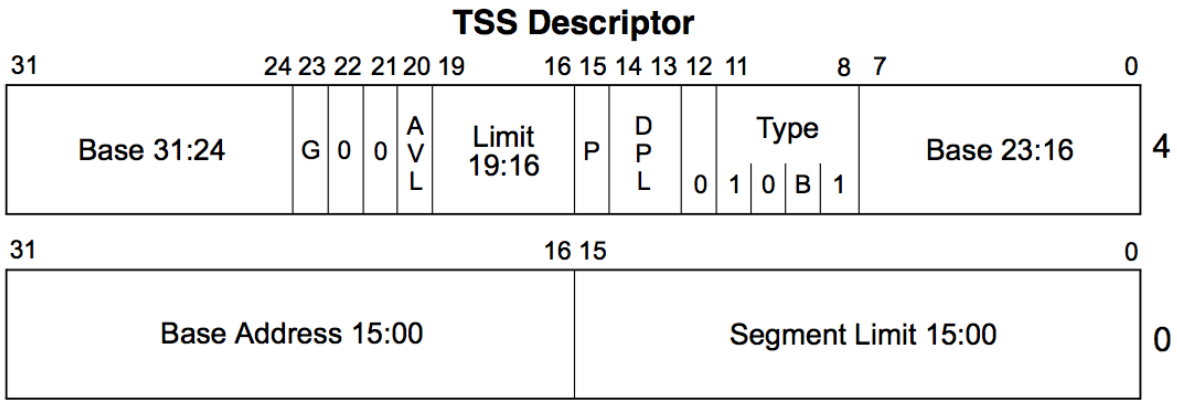
- **¿Qué sucede con las interrupciones de hardware?**

  Los bits de DPL se ignoran

- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** $\rightarrow$ `General Protection Fault (#GP)`

Intercambio de tareas: Descriptor de TSS

- Con las tareas nos enfocaremos en si podemos efectuar una conmutación a una tarea dada
- Para ello veremos los descriptores de su segmento de estado (descriptor de TSS):



AVL	Available for use by system software
B	Busy flag
BASE	Segment Base Address
DPL	Descriptor Privilege Level
G	Granularity
LIMIT	Segment Limit
P	Segment Present
TYPE	Segment Type

Intercambio de tareas: Verificación de privilegios

- **¿Cómo sabemos si la información de una tarea está presente?**

  Vemos que si el bit P del descriptor de su TSS está en 1

- **¿Y qué nivel de privilegio es necesario para saltar a ella?**

  Para realizar el salto se necesita que `CPL` $\leq$ `DPL` donde estos son los bits del descriptor de su TSS ¿Eso significa que no se puede conmutar a una tarea de nivel 0 al ejecutarse una de nivel 3?

- **¿Se puede saltar a una tarea si B es 1?**

  No, es necesario que el bit de Busy esté en 0

- **¿Qué sucede si alguna de las verificaciones anteriores no se cumple?** $\rightarrow$ `General Protection Fault (#GP)`

¿Preguntas?