
Cuestiones de Sistemas Operativos

Facultad de Informática, UCM

Módulo 5: Gestión de Memoria

1.-Un sistema de paginación pura tiene un tamaño de página de 512 palabras, una memoria virtual de 512 páginas numeradas de 0 a 511, y una memoria física de 10 marcos de páginas numerados de 0 a 9. El contenido actual de la memoria física es el siguiente:

| | |
|------|----------------------|
| 0 | — |
| 512 | — |
| 1024 | — |
| 1536 | Pág. 34 |
| 2048 | Pág. 9 |
| 2560 | — |
| 3072 | Tabla de pág. de P |
| 3584 | Pág. 65 |
| 4096 | — |
| 4608 | Pág. 10 |

- a) Mostrar el contenido actual de la tabla de páginas (TP) del proceso P
- b) Mostrar el contenido de la misma tabla de páginas después de cargar la página 49 en la posición 0 y de sustituir la página 34 por la página 12
- c) ¿Qué direcciones físicas referencian las direcciones virtuales 4608, 5119, 5120 y 33300?
- d) ¿Qué ocurre cuando se referencia la dirección virtual 33000?
- e) Si la página cargada en el marco de página 9 es un procedimiento y otro proceso Q desea compartirlo, dónde debe aparecer en la TP de Q ? (Indicar la entrada afectada de la TP)

Solución:

(a) Contenido actual de la TP

P denota el bit de presencia o validez de la página

| Entradas | Contenido |
|----------|------------------------------|
| 0-8 | $P=0 \rightarrow$ no válidas |
| 9 | $P=1$, marco = 4 |
| 10 | $P=1$, marco = 9 |
| 11-33 | $P=0 \rightarrow$ no válidas |
| 34 | $P=1$, marco = 3 |
| 35-64 | $P=0 \rightarrow$ no válidas |
| 65 | $P=1$, marco = 7 |
| 66-511 | $P=0 \rightarrow$ no válidas |

(b) Contenido de la TP tras cargar P49 en marco 0 y sustituir P34 por P12

| Entradas | Contenido |
|----------|------------------|
| 0-8 | P=0 → no válidas |
| 9 | P=1, marco = 4 |
| 10 | P=1, marco = 9 |
| 11 | P=0 → no válida |
| 12 | P=1, marco = 3 |
| 13-48 | P=0 → no válidas |
| 49 | P=1, marco = 0 |
| 50-64 | P=0 → no válidas |
| 65 | P=1, marco = 7 |
| 66-511 | P=0 → no válidas |

(c) dir. virtual → dir. física

$$\text{DirFísica} = \text{NumMarco} * \text{TamPágina} + \text{Desplazamiento}$$

donde:

$\text{NumMarco} \equiv$ Acceso a entrada NumPágina de la tabla de páginas

$$\text{Offset} = \text{DirVirtual} \bmod \text{TamPágina}$$

$$\text{Desplazamiento} = \left\lfloor \frac{\text{DirVirtual}}{\text{TamPágina}} \right\rfloor$$

- 4608 → Pág. 9, Marco 4, Offset 0 = 2048
- 5119 → Pág. 9, Marco 4, Offset 511 = 2559
- 5120 → Pág. 10, Marco 9, Offset 0 = 4608
- 33300 → Pág. 65, Marco 7, Offset 20 = 3604

(d) Dirección física 33000 → Pág. 64, Offset 232. Como esa página no está en MP → Fallo de página.

(e) Queremos que página cargada en marco 9 sea una región compartida de código (ej. biblioteca dinámica) entre dos procesos P y Q .

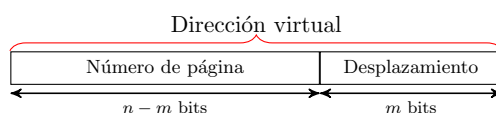
- Si el código es independiente de posición (PIC), éste podría ubicarse en cualquier entrada de la TP de Q .
- En otro caso debería usarse el mismo rango de direcciones virtuales en ambos procesos → misma entrada de la TP en ambos (la 10).
 - El proceso que creó la pagina (P) es el que determina la página “virtual” que se usa

2.-Considerar los cuatro sistemas siguientes:

| | A | B | C | D |
|--------------------------------|-----|-----|------|------|
| Tamaño de página (en palabras) | 512 | 512 | 1024 | 1024 |
| Tamaño de palabra (en bits) | 16 | 32 | 16 | 32 |

Para cada sistema determinar el número de entradas de la tabla de páginas (TMP). Suponer que sólo existe una TMP para cada sistema y que cada dirección virtual ocupa una palabra (16 o 32b).

Solución:



- $T_{palabra(bits)} \equiv n$
- $T_{página(palabras)} \equiv 2^k$
- $T_{página(bytes)} \equiv 2^m = T_{página(palabras)} \cdot \frac{T_{palabra(bits)}}{8} = 2^k \cdot \frac{n}{8} = 2^{k-3} \cdot n$
 $\Rightarrow m = \log_2 (2^{k-3} \cdot n)$

| | n | k | m | Número de entradas TP (2^{n-m}) |
|---|-----|-----|-----|-------------------------------------|
| A | 16 | 9 | 10 | $2^6 = 64$ |
| B | 32 | 9 | 11 | $2^{21} = 2M$ |
| C | 16 | 10 | 11 | $2^5 = 32$ |
| D | 32 | 10 | 12 | $2^{20} = 1M$ |

3.-En un sistema de paginación por demanda se obtiene que, con cierta carga de trabajo, la CPU se emplea un 15% del tiempo y el disco de swap está ocupado el 92% del tiempo. ¿Cuál de estas acciones aumentaría más la utilización de la CPU?

- Ampliar la memoria principal
- Aumentar el grado de multiprogramación
- Cambiar el disco de swap por otro de más capacidad
- Cambiar la CPU por otra más rápida

Razone la respuesta.

Solución:

El hecho de que el disco de swap está en uso un 92% del tiempo quiere decir que el SO está explotando el swapping en gran medida. Esto se debe a que la memoria principal no puede albergar todas las páginas que los procesos activos necesitan en este momento. Por lo tanto, la acción que más aumentaría la utilización de la CPU es ampliar la memoria principal—opción a)—, para que cupieran más páginas de los procesos activos y no se tuviera que utilizar tanto el dispositivo de swap.

4.-Un proceso en UNIX ejecuta el siguiente código en un sistema con memoria virtual:

```
1 #include <stdlib.h>
2 #include <stdio.h>
3
4
5 #define CONSTANT 10
6
7 int num1 = CONSTANT;
8 int num2;
9
10 char string[] = "hello";
11
12 void* print_something(void* arg){
13     char* msg=(char*) arg;
14     printf("Message: %s\n",msg);
15     sleep(20);
16     return NULL;
17 }
18
19 int main(int argc, char *argv[]) {
20     pthread_t thread;
21     int *i=NULL;
22
23     if ( (i=(int*)malloc((sizeof(int)))) == NULL )
24         return(-1);
25     num2=argc;
26
27     for(*i=0;*i<CONSTANT;*i=*i+1) {
28         fprintf(stdout, "%s: %d, argc: %d\n", string, num1--, num2);
29     }
30
31     pthread_create(&thread,NULL,print_something,string);
32
33     pthread_join(thread,NULL);
34
35     return(0);
36 }
```

- a) Para las siguientes variables/macros indique cuáles ocupan espacio en el ejecutable. Indique también la región del mapa de memoria en la que se encuentran.

| Variable/Macro | Ocupa espacio en ejecutable (Sí/No) | Región del mapa de memoria |
|----------------|-------------------------------------|----------------------------|
| CONSTANT | | |
| num1 | | |
| num2 | | |
| i | | |
| *i | | |

- b) Indique de qué regiones estará constituido el mapa de memoria del proceso cuando el hilo principal del programa se encuentre bloqueado en la llamada a `pthread_join()`.

Solución:

| Variable/Macro | Ocupa espacio en ejecutable (Sí/No) | Región del mapa de memoria |
|----------------|-------------------------------------|----------------------------|
| CONSTANT | No | N/A |
| num1 | Sí | Datos inicializados |
| num2 | No | Datos no inicializados |
| i | No | Pila |
| *i | No | Heap |

Regiones MM: Código, Datos inicializados, Datos no inicializados, Pila proceso, Pila thread, Heap, Datos y código librerías dinámicas (libc y libpthread)

5.-En un sistema con memoria virtual paginada indicar qué sucede y cuáles de las acciones son realizadas por el sistema operativo (especificando a qué estructuras de datos accede y cómo las modifica) en los siguientes casos: a) un proceso intenta escribir en una página de sólo lectura; b) un proceso intenta acceder a una dirección virtual de su espacio de direcciones correspondiente a una página que no está en memoria.

Solución:

- a) La MMU consulta la tabla de páginas (o el TLB), ve un acceso no permitido y notifica al SO, mediante una excepción, del acceso no permitido. El SO mata el proceso.
- b) La MMU consulta la tabla de páginas (o el TLB) y notifica al SO de un fallo de página. El SO (1) bloquea el proceso (ya no está listo para ejecutar), (2) busca en la partición/fichero de intercambio los datos necesarios, los carga en un marco vacío de memoria principal (posible reemplazo y escritura en disco), (3) actualiza la TP (y TLB) y (4) pone el proceso en el estado *listo*. Cuando el proceso se selecciona de nuevo para ejecutar, la MMU busca el dato correspondiente consultando el TLB (o TP) y la memoria principal.

6.-Cuando el mapa de memoria de un proceso se construye a partir de un fichero ejecutable ELF (p.ej., al invocar *exec()*) se crean, entre otras, las siguientes regiones: Código, Datos con valor inicial, Datos sin valor inicial, Pila. Para cada una de ellas indique: de dónde se obtiene el contenido inicial (si es que tienen contenido inicial) y si su tamaño es fijo o variable.

Solución:

Código: fichero ejecutable (.text), tamaño fijo

Datos con valor inicial: fichero ejecutable (.data), tamaño fijo

Datos sin valor inicial: fichero ejecutable (solo tamaño - .bss), tamaño fijo

Pila: -, variable

7.-Estudie el siguiente código que se ejecutará en un sistema Linux e indique si cada una de las afirmaciones posteriores son ciertas o falsas JUSTIFICANDO SU respuesta para cada una de ellas. Asuma que, tras alojarse el marco de pila de la función *main()*, el sistema sólo ha asignado una página para pila, una para código y una para datos globales (tamaño de página de 4 KB).

```
1 #include <stdio.h>
2
3 int len;
4
5 int getNumberIter() {
6     return 200;
7 }
8
9 int main() {
10     int i;
11     int M[128];
12     int x,y;
13
14     x = M[0];
15     y =0;
16     len = getNumberIter();
17     for (i=1; i < len; i++) {
18         y = x + M[i];
19         M[i] = y;
20     }
21     return y;
22 }
```

- a) El código compilará correctamente pero siempre dará una excepción por Violación de

segmento al ejecutarlo porque accedemos a elementos de M no reservados.

- b)* El código compilará correctamente. En ejecución, es posible pero no seguro que haya excepción por Violación de segmento. Si no se produce la excepción se corromperán datos de la pila pero la ejecución continuará.
- c)* El código compilará correctamente. En ejecución, se corromperán datos de la pila y, quizás, de otras regiones de memoria (como el heap, código...)
- d)* Podemos asegurar que se producirá una excepción si la función `getNumberIter()` devuelve un número mayor de 4096.

Solución:

- a)* Falso. Ese código no da violación de segmento, porque no se sale de la página de 4KB que tiene asignada la pila
- b)* Verdadero. Si se sale de la página, da violación. Si no, no. Pero se modificarán posiciones anteriores de la pila, que harán que el comportamiento sea impredecible.
- c)* Falso. Tal y como hemos visto que se forma el mapa de memoria, la región de pila está al final del rango de direcciones lógicas de espacio de usuario. Por lo tanto, salirse de ese array nunca modificará otras regiones.
- d)* Verdadero. Al pasarnos de los 4KB (y asumiendo que sólo teníamos una página como se indica en el enunciado), nos saldremos de la página de pila, pasaremos a zona del kernel y habrá violación de segmento.