Àlex Aguilera

Miguel Fernández

Práctica Mach i GNU Hurd.

RESPUESTAS:

5.

1. Cada llamada de host\_processors, asigna 0x1000.

2. Cada llamada de vm\_map, asigna 0x2000.

3. Para host\_processors: 0x1033000, 0x12e5000, 0x12e8000, 0x12eb000

y para vm\_map: 0x12e3000, 0x12e6000, 0x12e9000, 0x12ec000.

4. Se puede observar que son consecutivas y se van entrelazando, menos la primera

que no sabemos porque lo hace diferente. Hemos aumentado a 32 iteraciones para verlo

un poco mejor.

5. Buscando por internet, se encuentra lo siguiente:

#define VM\_PROT\_NONE ((vm\_prot\_t) 0x00)

#define VM\_PROT\_READ ((vm\_prot\_t) 0x01) /\* read permission \*/

#define VM\_PROT\_WRITE ((vm\_prot\_t) 0x02) /\* write permission \*/

#define VM\_PROT\_EXECUTE ((vm\_prot\_t) 0x04)

#define VM\_PROT\_DEFAULT (VM\_PROT\_READ|VM\_PROT\_WRITE)

#define VM\_PROT\_ALL (VM\_PROT\_READ|VM\_PROT\_WRITE|VM\_PROT\_EXECUTE)

#define VM\_PROT\_NO\_CHANGE ((vm\_prot\_t) 0x08)

#define VM\_PROT\_COPY ((vm\_prot\_t) 0x10)

6. Nos sale un Segmentation Fault. Para ponerlo como solo lectura, hemos

modificado la función vm\_map, poniendo "VM\_PROT\_READ".

7. Para solucionarlo con la función vm\_protect(), hemos añadido lo siguiente al

código: res = vm\_protect(mach\_task\_self(), (vm\_address\_t) p, 8192, 0, VM\_PROT\_DEFAULT);

8. Código en ex8.c. No aparece ningún error de compilación, pero al ejecutar nos sale un Segmentation Fault. Hemos estado un buen rato probando y buscando el porque del error, pero sin éxito.

9. La función mach\_setup\_thread() le da al thread una pila y la configura para que se ejecute con el PC cuando se reanude.

Mientras que, mach\_setup\_tls() le da al thread una zona TLS (thread-local storage)

10. Código en ex10.c. Si se usa el comando ps, se puede ver como se ha pedido la memoria correctamente.

12. Código en ex12.c. Se ha intentado, el código tiene sentido, pero no acaba de funcionar.