UEFI y coreboot

- ¿Qué es UEFI? ¿Cómo puedo usarlo? Mencionar además una función a la que podría llamar usando esa dinámica.

  La UEFI (Unified Extensible Firmware Interface) es una especificación que define una interfaz entre el sistema operativo y el firmware. UEFI reemplaza la antigua interfaz de la BIOS (Basic Input Output System).
  El acceso a UEFI se realiza habitualmente pulsando una determinada tecla del teclado o una combinación de ellas durante el arranque del equipo para acceder al firmware.
  Un uso básico de UEFI es ordenar el orden de arranque de las unidades de almacenamiento. Funciones más avanzadas permiten gestionar niveles de RAID para las unidades de almacenamiento o la frecuencia y voltaje del procesador y las memorias incluyendo lo que conocemos como técnicas de overclocking, «subiendo de vueltas» estos componentes por encima de su funcionamiento base.

- ¿Qué es Converged Security and Management Engine (CSME), the Intel Management Engine BIOS Extension (Intel MEBx)?

  Intel CSME es un subsistema integrado y un dispositivo PCIe (Peripheral Component Interconnect Express) diseñado para actuar como controlador de seguridad y manejabilidad en el PCH (Platform Controller Hub). Intel CSME tiene como objetivo implementar un entorno informático aislado del software principal (SW) que ejecuta la CPU, como el BIOS (sistema básico de entrada y salida), el sistema operativo (sistema operativo) y las aplicaciones.

- ¿Qúe es coreboot? ¿Qué productos lo incorporan? ¿Cuáles son las ventajas de su utilización?

  Coreboot (antes llamado LinuxBIOS) es un proyecto dirigido a reemplazar el firmware no libre de los BIOS propietarios, encontrados en la mayoría de los computadores, por un BIOS libre y ligero diseñado para realizar solamente el mínimo de tareas necesarias para cargar y correr un sistema operativo moderno de 32 bits o de 64 bits.
  Lo incorporan las computadoras portátiles Purism y las computadoras portátiles Minifree.
  Mayor rapidez en el arranque, evitar el malware o adware de los fabricantes en la BIOS, capacidad de habilitar virtualización por hardware en aquellos equipos que no lo posean o una mejor gestión de la energía en portátiles, son algunas de las ventajas que puede ofrecer Coreboot.

Linker

- ¿Qué es un linker? ¿Qué hace?

  Un linker es un programa de sistema que combina dos o más módulos o programas objetos por separado para crear un único ejecutable. Lo que hace es toma de los objetos generados en la compilación para poder recopilar todos sus datos y referencias necesarias, por ejemplo bibliotecas usadas para en un último paso crear un único programa ejecutable

- ¿Qué es la dirección que aparece en el script del linker? ¿Por qué es necesaria?

  En un dispositivo de arranque se tiene 521 bytes de código al principio y que contiene el número de arranque: 0x55AA como últimos 2 bytes. Si la BIOS encuentra 510 bytes seguidos de 0x55AA, toma los 510 bytes anteriores los mueve a la RAM (a la dirección 0x7c00) y asume que son bytes ejecutables. Este código es el llamado gestor de arranque.

- Compare la salida de objdump con hd, verifique donde fue colocado el programa dentro de la imagen.

  Salida del comando objdump:

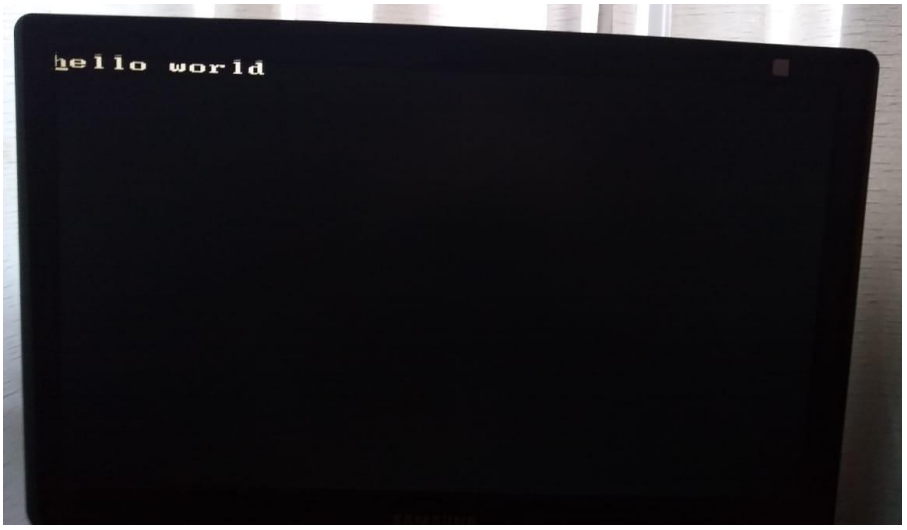
```
main.o:     file format elf64-x86-64


Disassembly of section .text:

0000000000000000 <loop-0x5>:
   0:   be 00 00 b4 0e          mov    $0xeb40000,%esi

0000000000000005 <loop>:
   5:   ac                      lods   %ds:(%rsi),%al
   6:   08 c0                   or     %al,%al
   8:   74 04                   je     e <halt>
   a:   cd 10                   int    $0x10
   c:   eb f7                   jmp    5 <loop>

000000000000000e <halt>:
   e:   f4                      hlt

000000000000000f <msg>:
   f:   68 65 6c 6c 6f          pushq  $0x6f6c6c65
  14:   20 77 6f                and    %dh,0x6f(%rdi)
  17:   72 6c                   jb     85 <msg+0x76>
  19:   64                      fs
        ...
```

  Salida del comando hd:

```
00000000  be 0f 7c b4 0e ac 08 c0  74 04 cd 10 eb f7 f4 68  |..|.....t......h|
00000010  65 6c 6c 6f 20 77 6f 72  6c 64 00 66 2e 0f 1f 84  |ello world.f....|
00000020  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
00000030  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
00000040  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
00000050  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
00000060  00 66 2e 0f 1f 84 00 00  00 00 00 66 2e 0f 1f 84  |.f.........f....|
00000070  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
00000080  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
00000090  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
000000a0  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
000000b0  00 66 2e 0f 1f 84 00 00  00 00 00 66 2e 0f 1f 84  |.f.........f....|
000000c0  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
000000d0  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
000000e0  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
000000f0  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
00000100  00 66 2e 0f 1f 84 00 00  00 00 00 66 2e 0f 1f 84  |.f.........f....|
00000110  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
00000120  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
00000130  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
00000140  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
00000150  00 66 2e 0f 1f 84 00 00  00 00 00 66 2e 0f 1f 84  |.f.........f....|
00000160  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
00000170  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
00000180  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
00000190  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
000001a0  00 66 2e 0f 1f 84 00 00  00 00 00 66 2e 0f 1f 84  |.f.........f....|
000001b0  00 00 00 00 00 66 2e 0f  1f 84 00 00 00 00 00 66  |.....f.........f|
000001c0  2e 0f 1f 84 00 00 00 00  00 66 2e 0f 1f 84 00 00  |.........f......|
000001d0  00 00 00 66 2e 0f 1f 84  00 00 00 00 00 66 2e 0f  |...f.........f..|
000001e0  1f 84 00 00 00 00 00 66  2e 0f 1f 84 00 00 00 00  |.......f........|
000001f0  00 66 2e 0f 1f 84 00 00  00 00 00 0f 1f 00 55 aa  |.f............U.|
00000200
```

El programa fue colocado al principio de la imagen.

- Grabar la imagen en un pendrive y probarla en una pc y subir una foto.



- ¿Para qué se utiliza la opción --oformat binary en el linker?

  Con la opción --oformat binary se está indiciado que se cree la “imagen binaria” para el archivo de objeto de salida.

Modo protegido

- ¿Cómo sería un programa que tenga dos descriptores de memoria diferentes, uno para cada segmento (código y datos) en espacios de memoria diferenciados?

  Se debería asignar a uno de los dos descriptores de segmento, una base que comience después de la suma de la base y el límite del descriptor anterior, para evitar que se superpongan.

- Cambiar los bits de acceso del segmento de datos para que sea de solo lectura, intentar escribir, ¿qué sucede? ¿Qué debería suceder a continuación? (revisar el teórico). Verificarlo con gdb.

- En modo protegido, ¿con qué valor se cargan los registros de segmento? ¿Por qué?

  En modo protegido, los registros de segmento se cargan con el offset del descriptor correspondiente al segmento dentro de la gdt.