**Tutorial 23**

En este tutorial se aprenderá a como introducir nuestro sistema operativo a emplear nuevas formas de acceder a rutinas del sistema y proporcionar un ambiente para usuario utilizando un Ring 1-3 y dejamos atrás el Ring 0.Para empezar, debemos recapitular el uso de la tabla GDT y crear dos nuevas tablas GDT, una para el código de programa del nivel de usuario y otra para la memoria o data de acceso de estos programas a nivel de usuario.

Para hacer el cambio a un Ring diferente de ejecución, se puede recurrir a utilizar la llamada IRET que simplemente busca los punteros de SS, ESP, EFLAGS, CS y EIP. Una vez que se hayan colocado en stack y puesto apropiadamente los registros de las nuevas entradas del GDT, se puede llamar a IRET para que pueda saltar a este nuevo nivel de ejecución. Para poder cambiar entre modo kernel y modo usuario, los sistemas operativos pueden hacer uso del TSS (Task State Segment), en el cual prácticamente se guarda el estado de los registros, de los stack segments y stack pointers de cada nivel de Ring (0, 1 y 2 en nuestro caso). Su estructura es la siguiente:

#ifdef \_MSC\_VER

#pragma pack (push, 1)

#endif

struct tss\_entry {

uint32\_t prevTss;

uint32\_t esp0;

uint32\_t ss0;

uint32\_t esp1;

uint32\_t ss1;

uint32\_t esp2;

uint32\_t ss2;

uint32\_t cr3;

uint32\_t eip;

uint32\_t eflags;

uint32\_t eax;

uint32\_t ecx;

uint32\_t edx;

uint32\_t ebx;

uint32\_t esp;

uint32\_t ebp;

uint32\_t esi;

uint32\_t edi;

uint32\_t es;

uint32\_t cs;

uint32\_t ss;

uint32\_t ds;

uint32\_t fs;

uint32\_t gs;

uint32\_t ldt;

uint16\_t trap;

uint16\_t iomap;

};

#ifdef \_MSC\_VER

#pragma pack (pop, 1)

#endif

Ahora, para poder utilizar rutinas y servicios de nuestro kernel, es necesario poder implementar llamadas al sistema en nuestro sistema operativo. Para esto, debemos instalar un ISR o Interrupt Service Routine que se encargue de ejecutar una acción cada vez que hagamos un llamado a una interrupción especifica. Para el caso de este tutorial, se ha fijado la interrupción 0x80, la misma utilizada en Linux. Cada vez que nuestro código en modo usuario requiera leer un archivo o imprimir en pantalla, se hara una llamada al sistema usando la interrupción 0x80, seguido por el comando que queramos ejecutar y pasar los demás parámetros necesarios para poder ejecutar el comando deseado.