PRACTICA No. 9

COMPILACIÓN DEL KERNEL

OBJETIVO: Que el alumno aprenda a compilar el Kernel de Linux para poder actualizarlo o activar periféricos que el Kernel no reconozca.

INTRODUCCIÓN:

Actualizar, remplazar o agregar nuevo código al Kernel es usualmente un proceso simple:
1) Obtener el código fuente del Kernel
2) Hacer cualquier cambio a la configuración
3) Compilar el Kernel
4) Instalación y manejo de Modulos
5) Colocar este en el lugar apropiado sobre el sistema de archivos para que el sistema pueda correrlo en forma adecuada.

Los códigos del Kernel con nuevas características de Linux están disponibles en CD, sitios FTP, grupos de usuarios, etc.

La mayoría de las versiones del Kernel están numeradas con una versión y el nivel del parche, si tecleamos el comando:

```
#uname -r        - Nos permite ver la versión del Kernel.
2.0.18
#
```

Donde el “2" es la característica mayor, “0" es la característica menor de la versión del Kernel, y “18" es el numero de parche.

Los parches algunas veces están numerados en forma diferente, y no requerimos el código entero del Kernel para instalarlo. Nosotros solo requeriremos el código del parche. En la mayoría de los casos, el parche se sobrepone en la sección del Kernel ya existente del código fuente y una simple recompilación es todo lo que necesitamos para instalar el parche, en su mayoría los parches son modificaciones pequeñas del Kernel.

Debemos tener una versión del compilador gcc o g++ (compiladores GNU) actualizada, porque ocasionalmente las nuevas características del Kernel no serán soportadas por versiones antiguas de gcc o g++.

DESARROLLO:

1. Configuración del Kernel.
Tecleamos:

 # cd /usr/src - En este directorio se encuentra el código fuente del Kernel.
 # ls -l - Checamos que se encuentre el directorio Linux-2.x.x
 # cd Linux-2.x.x - Nos colocamos en el directorio donde se encuentra el código fuente y es donde vamos a trabajar.

Tenemos que checar que el archivo `/usr/scr/Linux-2.x.x/Makefile` contenga la linea ROOT_DEV = CURRENT. El cual es el dispositivo que es usado por la raíz del sistema de archivos cuando booteamos Linux.

Tecleamos:

 # vi Makefile - Con el editor vi checaremos el archivo.

a. Con la tecla flecha ↓ nos movemos a través del archivo.
b. Localizar la linea ROOT_DEV=CURRENT

Esc
:q! - Salimos del archivo sin hacer modificación alguna.
#

El proceso de compilación empieza cuando tecleamos el comando.

 # make config - Aparecerán una gran cantidad de modificaciones que podemos hacerle al Kernel.

También tenemos otras opciones:

#make menuconfig -Aparece las mismas opciones pero con un menú mas amigable.

#make xconfig -Si tenemos sesión gráfica.

Como por ejemplo:

Las respuestas que daremos con las respuestas serán “y” (sí) ó “n” de (no). Otros dispositivos tipicos tienen opción de “m” (modulo), ya que estos no se podrán dentro del kernel, pero como serán un modulo cargable. También si tenemos dudas esta la opción “?”.

Estas opciones son para agregar, remover drivers, protocolos, filesystems, etc. veremos menus

principales:

- Emulador matematico en el kernel.
- Discos y CDROM IDE soportados.
- Soprte para RED (TCP/IP, PPP, SLIP, Firewall)
- System V IPC
- Tipo de Procesador (386, 486, Pentium, Ppro)
- Soprte para SCSI (CDROM, Discos, Cintas, etc.)
- Dispositivo de red soportado (Tarjeta de red, PLIP)
- Sistema de Archivos (minix, etxfs, etx2fs, msdos, umsdos, proc, iso9660, hpfs, nfs, etc.)
- Dispositivos tipo caracter (impresoras, busmouse, PS/2 mouse, etc.)
- Tarjeta de sonido
- Otras configuraciones

```
#
# Using defaults found in arch/i386/defconfig
#
*
* Code maturity level options
*
Prompt for development and/or incomplete code/drivers
(CONFIG_EXPERIMENTAL) [N/y/?]

*
* Loadable module support
*
Enable loadable module support (CONFIG_MODULES) [Y/n/?]
Set version information on all symbols for modules
(CONFIG_MODVERSIONS) [Y/n/?]
Kernel daemon support (e.g. autoload of modules) (CONFIG_KERNELD)
[Y/n/?] ?
```

Tenemos que encontrar tres lineas que son las importantes para desactivar el puerto paralelo lp1. Teclearemos una ”n” para desactivar esa opción.

Nota: Si tecleamos ? aparecerá en la pantalla información de ayuda referente a esa opción. Para movernos con rapidez para localizar estas opciones basta con teclear enter (↵) después de cada petición.

```
PLIP (parallel port) support (CONFIG_PLIP)[M/n/y/?] N
```

```
IOMEGA Parallel Port ZIP drive SCSI support
(CONFIG_SCSI_PPA)[M/n/y/?] N
Parallel printer support (CONFIG_PRINTER)[M/n/y/?] N
```

Con esto se finaliza la parte de configurar el kernel.

2. Limpieza y Dependencias.

Tecleamos los comandos:

```
# make dep ; make clean
```

'make dep' - Asegura que todas las dependencias, tales como los archivos include, estén en su lugar.
'make clean' - Este comando remueve todos los archivos objeto y algunas otras cosas de la versión anterior.

3. Compilación del Kernel.

Tecleamos el comando:

 # make zImage - Crea una imagen compactada del nuevo Kernel.

Podemos crear los kernel: zImage, zlilo, bzImage, zdisk. Estos kernel están comprimidos y se descomprimirán automaticamente cuando se ejecuten.

Esta es la parte mas tardada del proceso y depende de la maquina que ocupemos. En una maquina pentium a 75 MHz toma aproximadamente 40 minutos y una Pentium a 233 Mhz con 32 Mb se aproximadamente 9 minutos.

4. Instalación del nuevo Kernel.

```
# pwd
/usr/src/linux
# cd arch/i386/boot          - Movernos al directorio boot.
# cp zImage /boot/zImage     - Copia el nuevo kernel al directorio de arranque.
# cd /etc                    - Nos cambiamos de directorio para modificar el archivo de
                               arranque LILO (Linux Loader).

# vi lilo.conf               - Editamos el archivo lilo.conf con el editor vi.
```

a. Tecleamos **`i`** - nos permite modificar el archivo.

El archivo es el siguiente.

```
map=/boot/map
install=/boot/boot.b
prompt
timeout= 50
image=/boot/vmlinuz
        label=linux
        root=/dev/hda2
        read_only
```

b. Agregar las siguientes lineas al final del archivo.

```
image=/boot/zImage
label=linux2
root=/dev/hda2
read_only
```

El archivo `lilo.conf` nos queda de la siguiente forma:

```
map=/boot/map
install=/boot/boot.b
prompt
timeout= 50
image=/boot/vmlinuz
        label=linux
        root=/dev/hda2
        read_only
image=/boot/zImage
        label=linux2
        root=/dev/hda2
        read_only
```

c. Tecleamos:

`Esc`

`:wq` - Salimos del editor vi guardando los cambios hechos a el archivo.

`# lilo` - Actualiza el archivo lilo para tener una nueva forma de arranque.

Con estas modificaciones tenemos dos kernels para arrancar el sistema Linux. El anterior con el puerto paralelo lp1 activado (linux) y el nuevo con el puerto paralelo lp1 desactivado (linux2).

4. Comprobación de la practica.

Salir del sistema con el comando:

 # shutdown -h 0 - Nos salimos de forma total de Linux.

Apagar la computadora y volverla a encender.

Cuando aparezca

LILO

Tecleamos:

TAB (⇋) - Aparecerán los nombres de los dos kernels con los que se puede activar el sistema Linux.

LILO: linux2 - Arrancara con el nuevo Kernel.

Con el comando **'dmesg'** checamos que no aparezca el puerto lp1.

 # dmesg | more - Nos permite ver el despliegue del comando por partes.

Checamos que no se encuentre la linea siguiente:

lp1 at 0x0378, (polling)

Si no se encuentra, la practica tuvo éxito, si se encuentra tenemos que checar que punto nos salio mal, y en el peor de los casos volver a empezar.

Volver a repetir el punto 4 pero en vez de cargar **'linux2'**. Arrancar con el Kernel anterior **'linux'** y teclear el comando **'dmesg'** para ver que se encuentre la linea **lp1 at 0x0378, (polling).**

MODULOS

Al cargar los modulos en el kernel puede "salvar" memoria y una fácil configuración. Estos modulos incluye filesystems, drivers para tarjeta ethernet, drivers para impresora y mas. Hay varios programas que tenemos para manejar los modulos como:

insmod, rmmod, ksyms, lsmod, genksyms, modprobe y depmod.

El comando *insmod* inserta un modulo dentro del kernel corriendo. Estos modulos tienen extensión .o, por ejemplo:

```
#insmod drv_hola.o
```

Con ello cargamos es driver, ademas si queremos ver los modulos cargados será con el comando *lsmod*:

```
#lsmod
Module:   #pages:   Used by:
drv_hola       1
```

Ahora si quiero remover el modulo usaremos el comando *rmmod*:

```
#rmmod drv_hola
```

Después de configurar el kernel ya sea cuando se reinicia el sistema o acabando la opción de "*make config*" (o similares) dentro del directorio */usr/src/linux*, teclear:

```
#make modules          -Preparar los modulos que se ejecutarán
```

Después de compilar todos los modulos y estos se encontrará en el archivo /usr/src/linux/modules, los modulos que se ejecutarán. Ahora para que estos se ejecuten cada vez que se reinicie la computadora se ejecutará

```
#make modules_install
```

lo cual instalarán los modulos en el directorio /lib/modules/x.y.z donde x.y.z es la versión del kernel.

CONCLUSIONES: