Módulo 2
Sistema de ficheros
Cerrojos

Ampliación de Sistemas Operativos. Curso 2017-2018

Cuestiones

Sistema de ficheros ext2 Estudia y ejecuta el *script shell* del fichero *script_ext2.sh*, y responde a las siguientes cuestiones:

- ¿Qué hace la primera llamada a la orden `dd`?
- ¿Qué hace la llamada a `mkfs`?
- ¿Y la orden `mount`?
- Investiga qué información, relacionada con el montaje de sistemas de ficheros, almacenan los ficheros `/etc/fstab` y `/etc/mtab`
- ¿Qué hace la segunda llamada a la orden `dd`?
- Busca información sobre el fichero de dispositivo `/dev/random`. ¿Qué obtenemos al leer de ese fichero?. Muestra el contenido del fichero *file.bin* mediante el comando `hexdump -v file.bin`.
- ¿Qué hace la tercera llamada a la orden `dd`?
- Tras esa tercera llamada, ¿hay alguna discrepancia entre el tamaño del fichero *file.bin* mostrado por `ls` y el mostrado por `du`? ¿Por qué?
- Anota todos los resultados de `ls` y `du` para las 4 versiones de *file.bin* creadas. ¿Cómo explicas esos valores? ¿Por qué el espacio en disco no sigue el mismo patrón que el tamaño lógico del fichero?

Sistema de ficheros vfat Modifica el *script* anterior para que cree un sistema de ficheros `vfat` en lugar de `ext2`. Ejecútalo de nuevo y observa los nuevos valores mostrados por `ls` y `du` para las 4 versiones de *file.bin* creadas. ¿Qué diferencias ves con los anteriores?

Sistema de ficheros ext4 (OPCIONAL) Modifica el *script* anterior para que cree un sistema de ficheros `ext4` en lugar de `ext2`. Ejecútalo de nuevo y observa los nuevos valores mostrados por `ls` y `du` para las 4 versiones de *file.bin* creadas. ¿Qué diferencias ves con los anteriores?

Cerrojos

bloquea.c Estudia, compila y ejecuta el fichero. En dos terminales diferentes, ejecuta el binario creado (es decir, lanza dos procesos independientes a partir del mismo ejecutable). ¿Qué observas?¿Se respeta el orden impuesto por el cerrojo? Compila nuevamente el código añadiendo el flag `-DNONE` a la compilación. Ejecuta nuevamente. ¿Qué ocurre ahora? ¿Por qué?

bloquea_flock.c Modifica el fichero anterior para usar la llamada `flock` en lugar de `fcntl`. Realiza las mismas pruebas que en el caso anterior y comprueba que el cerrojo funciona.

bloquea_close.c Estudia, compila y ejecuta el fichero. Ejecuta de forma similar a los apartados anteriores. ¿Sigue funcionando el cerrojo? ¿Por qué?

bloquea_flock_close.c Modifica el fichero anterior para usar la llamada `flock` en lugar de `fcntl`. Ejecuta de forma similar a los apartados anteriores. ¿Sigue funcionando el cerrojo? ¿Por qué?

simple_lock.c . Estudia, compila y ejecuta el fichero. Crea un fichero llamado *testigo* y ejecuta `echo hola >testigo` para escribir en el fichero. En un terminal, ejecuta la aplicación compilada. Sin que ésta termine, ejecuta en otro terminal una orden que acceda en lectura o escritura sobre el fichero *testigo* (como por ejemplo `cat testigo`). ¿Qué ocurre? ¿Funciona el cerrojo? ¿Por qué?

simple_lock.c (bis). Ahora, crea un nuevo sistema de ficheros con el flag `mand`. Para ello puedes usar:

```
$ mkdir dir
$ mount -t tmpfs -o mand,size=1m tmpfs ./dir
$ echo hello > dir/testigo
$ chmod g+s,g-x dir/testigo
```

Realiza ahora la misma prueba que en el apartado anterior. ¿Es efectivo ahora el cerrojo? ¿Por qué?

Entrega

La entrega consistirá en un solo fichero comprimido (`.tgz`) que contendrá:

- Todos los ficheros fuentes completos (NO debe incluirse ningún fichero objeto o ELF).
- El fichero *makefile*.
- Un fichero de texto o PDF con las respuestas a las preguntas planteadas en cada ejercicio, así como una copia de las fuentes de cada apartado completado.