Trabajo Práctico 3 ∼ PSO

Liptak, Leandro - leandroliptak@gmail.com
Reboratti, Patricio - darthpolly@gmail.com
Silvani, Damián - dsilvani@gmail.com

July 7, 2011

Documentación

Inter-process Communication

`pipe` - Pipes

`loader` - Extensión a la carga de procesos

Implementar la llamada al sistema `fork()` resultó más sencillo que la implementación de `loader_load` o `run` en parte por la capacidad de reutilizar funcionalidades diseñadas para estas últimas. Se solicita un nuevo PID, un nuevo directorio de páginas (con el área del kernel mapeada) y se procede a crear el PCB para el proceso hijo. Luego (según la primer implementación de la syscall) se copian todas aquellas páginas presentes en el esquema de paginación del proceso padre y se mapean dichos frames con el contenido copiado y la misma dirección virtual en el esquema de paginación del hijo. La excepción a dicha copia son las páginas de kernel del área de 0 a 4MB y la página de la pila de anillo cero pues el contenido de ésta no era el adecuado para un proceso recién lanzado.

Posteriormente, se arma una pila de anillo 0 tal como lo haría `loader_load` y se mapea en la dirección esperada. Finalmente, se copian los descriptores de archivos incrementando el número de referencias en el descriptor de dispositivo asociado y se encola el proceso en el scheduler.

Una vez implementadas las funcionalidades vinculadas al manejador de memoria, fue necesario pulir el mecanismo de copia de páginas del proceso padre al hijo pues en las entradas de las tablas de páginas había mucha más información que la simple presencia de la página. Por este motivo la copia se extendió también a las entradas de las tablas de páginas transportando así la información presente en ellas al nuevo proceso. Vale aclarar que en algunos casos no se trata de una

copia directa de la entrada sino que la información presente en ella/s puede verse alterada.

Manejador de memoria

Memoria on-demand

Brindar dicha funcionalidad fue relativamente sencillo: a medida que se solicitaban páginas con `palloc()` el rango de direcciones virtuales crecía pero la solicitud del frame correspondiente se postergaba. Es así que la entrada en la tabla de páginas correspondiente a dicha página continuaba marcada como no presente pero uno de los bits restantes de dicha entrada se seteaba indicando que la página había sido previamente solicitada. Esto permitía posteriormente, frente al fallo producido a causa del acceso a dicha página poder discernir entre un intento de acceso a una dirección inválida o a una página previamente demandada. Esta lógica se incluyó en el handler del *Page Fault* y si el caso fuera el último mencionado se procede a solicitar un frame y mapearlo en dicha dirección virtual, retornando luego a la instrucción que causó la excepción.

Memoria compartida

Copy-on-write

Para implementar el mecanismo de *copy-on-write* se modificó, como se ha mencionado, aquella sección de la llamada al sistema `fork()` en que se copiaban las páginas de un proceso a otro. En lugar de esto, se procedió a crear el mapeo en el esquema de paginación del hijo, para la misma dirección virtual que el padre e indicando la página como presente, pero con acceso de sólo lectura. Tambíen la entrada del padre es marcada como de sólo lectura y adicionalmente ambas entradas se marcan indicando que han de ser copiadas ante una escritura.

Posteriormente, frente a una eventual escritura se corrobora si la página debe ser copiada o no. En caso de haber sido marcada para ser copiada, pero ser el único proceso con dicho mapeo (pues los otros ya han accedido y han hecho la copia), dicha página se otorga nuevamente con permisos de lectura/escritura sin ser copiada. En otro caso, la copia se efectúa, indicando luego que dicha página ya no necesita copia en el esquema de paginación de dicho proceso.

Tareas