

SESIÓN N° 03

PIPES: COMUNICACIÓN ENTRE PROCESOS

I

OBJETIVOS

- ☞ Explicar, manipular y trabajar con las Pipes como mecanismo de comunicación entre procesos.

II

TEMAS A TRATAR

- ☞ Conceptos de pipe.
- ☞ Utilizar el operador "|".
- ☞ Utilizar y comprender la utilización combinada de filtros.

III

MARCO TEORICO

Entrada y Salida Estándar

Muchos comandos UNIX toman su entrada de algo conocido como entrada estándar y envían su salida a la salida estándar (a menudo abreviado como "stdin" y "stdout"). Además existe una salida especial para los mensajes de error de cada programa (stderr).

El intérprete de comandos configura el sistema de forma que la entrada estándar es el teclado y la salida la pantalla.

Veamos un ejemplo con el comando **cat**. Normalmente **cat** lee datos de los archivos cuyos nombres se pasan como argumentos en la línea de comandos y envía estos datos directamente a la salida estándar. Luego, usando el comando

```
/home/usuario/papers$ cat history-final masters-thesis
```

Uso de pipes

Ya hemos visto cómo usar **sort** como un filtro. Pero estos ejemplos suponen que tenemos los datos en un fichero en alguna parte o vamos a introducir los datos manualmente por la entrada estándar.

¿Qué pasa si los datos que queremos ordenar provienen de la salida de otro comando, como **ls**? Por ejemplo, usando la opción **-r** con **sort** ordenaremos los datos en orden inverso. Si queremos listar los ficheros en el directorio actual en orden inverso, una forma podría ser.

```
/home/usuario/papers$ ls
english-list history-final
masters-thesis
notes

/home/usuario/papers$ ls > file-list
/home/usuario/papers$ sort -r
file-list notes masters-thesis
history-final english-list

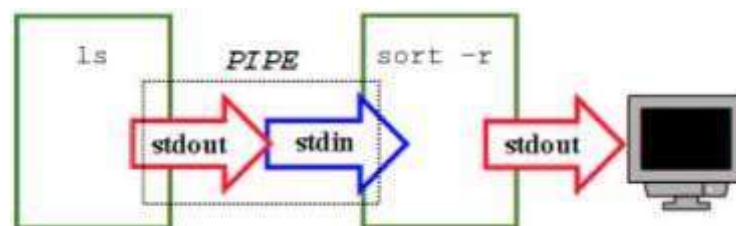
/home/usuario/papers$
```

Aquí, salvamos la salida de **ls** en un fichero, y entonces ejecutamos **sort -r** sobre ese fichero. Pero esta forma necesita crear un fichero temporal en el que salvar los datos generados por **ls**.

La solución es usar las pipes. El uso de pipes es otra característica del intérprete de comandos, que nos permite conectar una cadena de comandos en un "pipe", donde la **stdout** del primero es enviada directamente a la **stdin** del segundo y así sucesivamente. Queremos conectar la salida de **ls** con la entrada de **sort**. Para crear un pipe se usa el símbolo "|":

```
/home/usuario/papers$ ls | sort -r notes masters-thesis history-final
english-list

/home/usuario/papers$
```



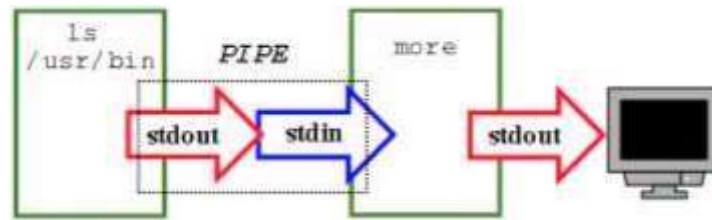
Esta forma es más corta y obviamente más fácil de escribir.

Otro ejemplo útil, usando el comando

```
/home/usuario/papers$ ls /usr/bin
```

mostrará una lista larga de los ficheros, la mayoría de los cuales pasara rápidamente ante nuestros ojos sin que podamos leerla. En lugar de esto, usemos **more** para mostrar la lista de ficheros en **/usr/bin**.

```
/home/usuario/papers$ ls /usr/bin | more
```

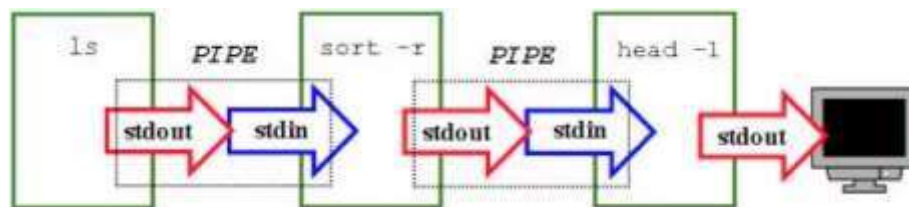


Ahora podemos ir avanzando página a página cómodamente.

Pero el potencial de los pipes no termina aquí. Podemos "entubar" más de dos comandos a la vez.

El comando **head** es un filtro que muestra la primeras líneas del canal de entrada (aquí la entrada desde una pipe). Si queremos ver el último fichero del directorio actual en orden alfabético, usaremos:

```
/home/usuario/papers$ ls | sort -r | head -1 notes
/home/usuario/papers$
```



Donde **head -1** simplemente muestra la primera línea de la entrada que recibe en este caso, el flujo de datos ordenados inversamente provenientes de **ls**.

IV

ACTIVIDADES

01. Obtener un listado en orden alfabético inverso de los primeros 3 comandos del directorio **/bin**.

Ayuda: (posible resultado: cat,bash,arch). Utilizar: **ls,sort y head**.

02. Obtener el mismo resultado que en el ejercicio anterior, pero utilizando el comando **tail** en vez de **head**.

03. Contar la cantidad de archivos en el directorio **/bin**.

Ayuda: Para contar palabras se utilizará el comando "**wc -w**".

\$

04. Buscar dentro de `/usr/doc` o `/usr/share/doc` algún archivo de texto comprimido (extensión `.gz`).

Una vez elegido el archivo, observar el contenido con el comando `zcat`. Ordenar las líneas alfabéticamente y mostrar el resultado a través del **less**. Todo usando pipes.

\$

05. Obtener (utilizando una sola línea de comandos y pipes) un archivo donde figuren todos los usuarios y los grupos definidos en el sistema. Ordenados alfabéticamente. Visualizado a través del **less**.

Ayuda: (El archivo `/etc/passwd` contiene una línea por cada usuario definido en el sistema. En el comienzo de la línea figura el nombre del usuario. Algo similar ocurre con los grupos, en el archivo `/etc/group`). Utilizar **cat** para juntar ambos archivos, ordenar las líneas alfabéticamente y mostrar el resultado.

\$

06. Obtener la cantidad de usuarios y grupos definidos en el sistema (suma de ambos).

Ayuda: Similar al ejercicio anterior, pero contando la cantidad de líneas del archivo concatenado.

\$

V

EVALUACIÓN

Prepárese para la evaluación de este laboratorio.