Redireccionamiento - Filtros - Tuberías

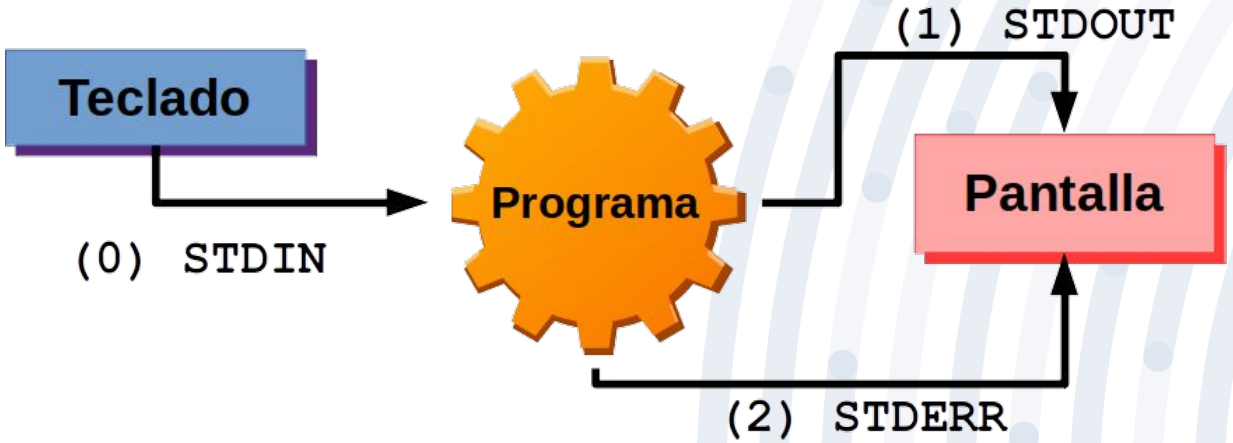
Laboratorio de Sistemas Operativos I

Redireccionamiento

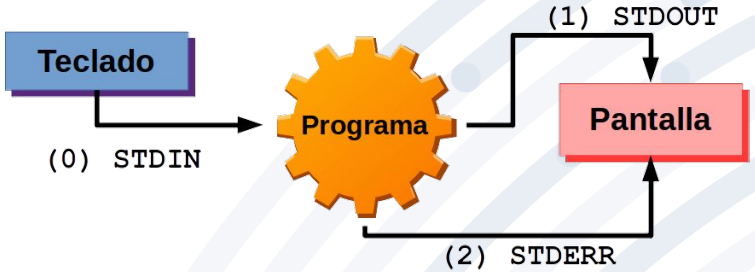
Dar una nueva dirección o un nuevo destino a algo.

Ejecución de un comando -> Proceso

Todos los procesos necesitan tener una *entrada de datos* y devuelven como resultado dos archivos, *salida y error*. Cuando el de salida está lleno, el de error está vacío y viceversa. La Shell tiene la capacidad de controlar y dirigir la entrada de datos a los programas, la salida de información y los errores.



E/S Estándar y Descriptores

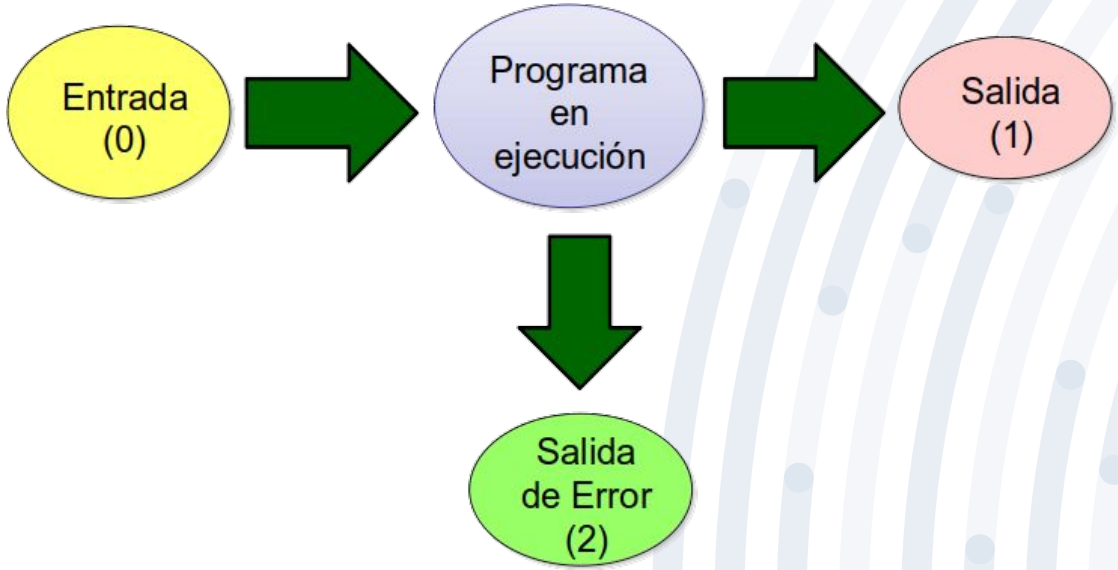


Entonces, un proceso da **3** descriptores de archivo:

- **STDIN -> Entrada estándar (0) ->** está asociado a la entrada de texto. Por defecto el teclado. Representado en la terminal como el tipo 0
- **STDOUT -> Salida estándar (1) ->** corresponde con la salida de texto normal de los programas. Por defecto la terminal de pantalla. Representado en la terminal como el tipo 1
- **STDERR -> Error estándar (2) ->** también es una salida de texto, pero se usa exclusivamente para mostrar los errores generados por los programas. Por defecto la terminal de pantalla. Representado en la terminal como el tipo 2

Ejemplo: `$ls /etc`

- Cuando se ejecuta el comando ls, su entrada estándar es la que escribimos por teclado; luego el SO ejecuta el comando y como resultado hay dos archivos. Uno vacío, pero existente.



Redireccionamiento

Como la entrada estándar y las salidas estándar y de error son archivos, pueden ser capturados por la terminal y redireccionados (trasladados) a otros archivos utilizando símbolos en la línea de comandos.

Redirección de salida estándar

- **Ejemplo:**

```
~$ls – l /etc/*.conf > archivo.txt
~$cat archivo.txt
```

- **Observaciones:**
  - Se utiliza el caracter: **>**
  - La salida del comando es guardada en el archivo indicado.
  - Si el archivo no existe, es creado.
  - Si el archivo existe, se sobrescribe su contenido, incluso si el comando tecleado no es correcto.

Redirección no destructiva

- **Ejemplo:**

```
$ls – l /etc/*.conf > archivo.txt
$date >> archivos.txt
$cat archivos.txt
```

- **Observaciones:**
  - Es posible redireccionar la salida estándar sin sobrescribir el archivo de salida (se añaden los datos al final del archivo).
  - Se utiliza el caracter: **>>**

Redirección de error estándar

- **Ejemplo:**

```
~$ ls – l /root 2> archivo.error
~$ cat archivo.error
```

- **Observaciones:**
  - Se utiliza el caracter: **2>**
  - La salida de error del comando es guardada en el archivo indicado.
  - Si el archivo no existe, es creado.
  - Si el archivo existe, se sobrescribe su contenido, incluso si el comando tecleado no es correcto.

Redirección de salida y error al mismo archivo

- **Ejemplo:**

```
~$ls – l /etc/*.conf > archivo.txt 2>&1
~$cat archivo.txt
```

- **Observaciones:**
  - Se redirige la salida y el error a un mismo archivo.
  - **2>&1** Significa que el error(2) se redirige a la salida estándar(1), y como la salida estándar se redirige al archivo, entonces ambas van al archivo.
  - También puede escribirse como:

```
~$ls – l /etc/*.conf &> archivo.txt
```

Redirección de entrada estándar

- **Ejemplo:**

```
$ wc < archivos.config
$ sort < items.txt
```

- **Observaciones:**
  - Los comandos que esperan datos desde el teclado, pueden recibir datos desde un archivo.
  - Se utiliza el caracter: **<**

Ejemplos Redireccionamiento

```
$ ls – lR > lista.dir
```

- La salida estándar de la orden ls –lR (listado de archivos y subdirectorios que contiene el directorio actual de forma recursiva) se guarda en el archivo lista.dir.

```
$ mkdir directory 2> error.txt
```

- Si al crear el directorio directory se produce algún error (por ejemplo el directorio ya existe o el usuario no tiene permisos de escritura) el mensaje del error se guardará en el archivo error.

Ejemplos Redireccionamiento

```
$cat /etc/passwd /etc/group > todo.txt  2> error.t
```

- La concatenación de los archivos passwd y group se guardará en el archivo file. Los errores que se produzcan en la ejecución de la orden se almacenarán en el archivo error.

```
$cat /etc/passwd /etc/group > todo.txt 2>&1
```

- Se puede redireccionar ambas salidas al mismo archivo utilizando 2>&1.
- El archivo "todo.txt" contendrá la salida del comando y los errores si los hubiera.

REDIRECCIONANDO LA SALIDA ESTÁNDAR O DE ERROR A NINGÚN LADO

- Si no se desea mostrar los errores de salida pero tampoco guardarlos, utilizamos entonces **/dev/null**:

```
~$cp archivo1 archivo.txt  2> /dev/null

~$cp archivo1 archivo.txt  > /dev/null
```

- /dev/null es un tipo especial de archivo que descarta toda la información que se escribe o se redirige a el. También se le conoce como "null device".

Filtros

- Un filtro es un comando que lee y escribe datos por los canales estándares de entrada y salida.
- **Modifica** o **trata** el contenido de los mismos.

Algunos comandos de filtros:

- Tarea: buscar objetivo, sintaxis y opciones

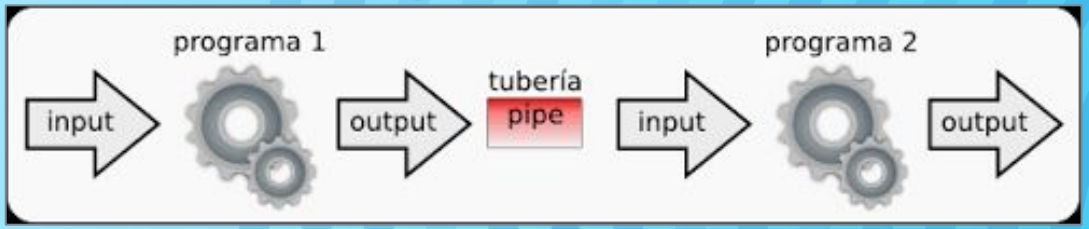
COMANDOS		
`more`	`tail`	`cut`
`less`	`grep`	`wc`
`head`	`sort`	`uniq`

Ejemplos:

- `$ls –l | sort | grep kernel > archivos.kernel`
- `$cat alumnos.txt | grep ^C > listado.letraC`
- `$head /var/log/syslog | grep WARNING`
- `$tail -25 /var/log/syslog | grep error`
- `$cat ejemplo.txt | wc -l`
- `$ps aux | wc -l`

Tuberías o Pipes

- Desde el punto de vista de un programa no hay diferencia entre leer texto guardado en un archivo o introducido mediante el teclado.
- Extendiendo esta idea, es posible **enlazar** la salida de un programa con la entrada de otro.
- Esta operación se puede realizar mediante una tubería (pipe) que une dos comandos en uno solo.



Ejemplos de Tuberías

```
$ls –l | less
```

- La tubería redirecciona la salida del comando ls – l hacia el paginador less, que permite visualizar la información pantalla a pantalla y desplazarse sobre la misma.

```
$ls –l | grep vmlinux
```

- Redirecciona la salida de la orden ls hacia la orden grep, que mostrará únicamente las líneas que contienen la palabra vmlinux.

Ejemplos de Tuberías

- Es válido usar varias tuberías a la vez

```
$ls -s /dev |  sort | less
$ls -s /dev | sort | grep “0“ | wc –l
```

- Está permitido combinar redireccionamiento con tuberías

```
$ls –l | sort > lista.archivos.txt
$cat alumnos.txt | sort > listado.alumnos
```

Guía para Redireccionamiento

operador	descripción
cmd < file	El contenido de file se utilizará como entrada de la orden cmd
cmd 1> file1 cmd > file1	La salida estándar de la orden cmd se guardará en file. Si el archivo file existe se sobrescribirá.
cmd 1>> file1 cmd >> file1	La salida estándar de la orden cmd se guardará en file. La información se escribirá al final del archivo.
cmd 2> file2	Los errores de la orden cmd se guardarán en file. Si el archivo file existe se sobrescribirá.
cmd 2>> file2	Los errores de la orden cmd se guardarán en file. La información se escribirá al final del archivo.
cmd > file 2>&1	La salida estándar y la de errores de la orden cmd se guardarán en file. Si el archivo file existe se sobrescribirá.
cmd >> file 2>&1	La salida estándar y la de errores de la orden cmd se guardarán en file. La información se escribirá al final del archivo.
cmd1 | cmd2	Redirecciona la salida del comando cmd1 hacia la entrada del comando cmd2.

¡Gracias!

¿Preguntas?