Unidad 6: Tuberías y redirecciones

Conceptos básicos

Antes de pasar explicarles en que consisten las tuberías y su importancia (la parte divertida), debemos tener claros tres conceptos fundamentales en Linux: entrada estándar, salida estándar y error estándar.

La entrada estándar: representa los datos que son necesarios para el correcto funcionamiento de una aplicación. Un ejemplo de ellos puede ser un archivo con datos estructurados o información ingresada desde la terminal. En la terminal se representa como el tipo 0.

La salida estándar: es el medio que utiliza un aplicación para mostrar información de sus procesos y/o resultados, estos pueden ser simples mensajes, avisos referentes al progreso o archivos con datos estructurados como resolución de un proceso (un reporte, por ejemplo). En la terminal se representa como el tipo 1.

El error estándar: es la manera en que las aplicaciones nos informan sobre los problemas que pueden ocurrir al momento de su ejecución. Se representa como el tipo 2 en la terminal.

Todos los tipos son representados como archivos físicos en el sistema, en Linux, todo es un archivo.

Redirecciones

Ahora bien, ¿qué es una redirección?

Las redirecciones consisten en trasladar información de un tipo a otro (los tipos mencionados anteriormente), por ejemplo, de error estándar a la salida estándar o de la salida estándar a la entrada estándar. **A través de la terminal, logramos eso haciendo uso del símbolo >.**

Redireccionar salida y error estándar

Por ejemplo, para hacer la redirección de la salida de un comando y enviarla a un archivo; nos basta con ejecutar:

ls -la ~ > (nombre del archivo)

Sin embargo, si realizamos la ejecución de esa forma, el contenido de nuestro archivo sera remplazado, cada vez, por la salida del comando. Si lo que quisiéramos es que se adicione dicha salida en el archivo, entonces la ejecución seria de la siguiente manera:

ls -la ~ >> (nombre del archivo)

Lo que resulta interesante es que, **podemos redireccionar las salidas, los errores y las entradas estándar.** Es aqui donde toman sentido los números que les mencionaba al

inicio. Por ejemplo, para forzar que un programa nos muestre en pantalla los errores que se generen durante una ejecución, redireccionamos el error estándar hacia la salida estándar durante su ejecución:

aplicación 2 >> &1

Donde 2, representa al error estándar y &1 representa la salida estándar.

También podemos hacer un descarte del error estándar en cierto proceso, algo común en la administración de sistemas. Para esto ejecutamos:

aplicación 2 > /dev/null

Inclusive descartar la salida estándar:

aplicación > /dev/null

Ya que en Linux, **el archivo /dev/null, es un archivo especial a donde se envía la información para que sea descartada**.

Redireccionar la entrada

De la misma manera en que redireccionamos salidas y errores estándar, podemos hacerlo con **las entradas estándar desde un archivo y para ello utilizamos el operador <**.

Esto resulta útil en comandos o programas donde se introducen los argumentos por teclado, del tal forma que podemos sustituirlos por un archivo, por ejemplo:

```
echo "Hola mundo" > saludo
cat < saludo
Hola mundo
```

Tuberías

Luego de comprender el funcionamiento de las redirecciones, el concepto de tuberías sera bastante sencillo. Entre los principios dela filosofía de Unix, tenemos el hecho de tener aplicaciones pequeñas que se encarguen de realizar tareas muy puntuales y que en conjunto realicen tareas complejas. Siguiendo este principio, **tiene que existir una manera para que un conjunto de aplicaciones pueda interactuar entre si**. Es aqui donde surgen las llamadas tuberías.

Las tuberías son un tipo especial de redirección que permiten enviar la salida estándar de un comando como entrada estándar de otro. La forma de representarlo es **con el símbolo | (pipe)**. Su principal útilidad es que nos ofrece la posibilidad de concatenar comandos, enriqueciendo la programación.

Un ejemplo sencillo y muy útil es ver los procesos que están ejecutando en el sistema con **ps** y redireccionar su salida a **sort** para ordenarlos por PID:

ps -a | sort

Unidad 6: Órdenes útiles

sort

Ordenar líneas de los archivos de entrada a partir de criterios de ordenación. Los espacios en blanco son tomados por defecto como separadores de campo.

Su sintaxis es de la forma:

```
sort [opciones] [archivo]
```

Alguno de sus opciones son:

-b Ignora espacios en blanco precedentes.

-d Ordena ignorando todos los caracteres salvo caracteres letras, números y espacios.

-f considera iguales las mayúsculas y minúsculas.

-n ordena por valor numérico.

-r invertirá el orden.

-k3 Ordenar por el tercer campo, o el número que se indique (cada campo está separado por un delimitador, como por ejemplo, el espacio)

-o salida.txt Escribir el resultado en salida.txt.

-R desordena al azar.

Ejemplos de uso, para lo cual se tomara como ejemplo el archivo pepe.txt cuyo contenido es:

```
# cat pepe.txt
     15 windows
     16 acer
      2 actualizacion
      1 actualizada
      3 admiral
      2 www
      1 agregar
     10 aires
      1 al
      1 algunos
```

```
      1 empleos
     60 en
      2 equipo
     45 archivos
      3 word
```

sort pepe.txt Ordena el archivo pepe.txt:

```
# sort pepe.txt
     10 aires
     15 windows
     16 acer
      1 actualizada
      1 agregar
      1 al
      1 algunos
      1 empleos
      2 actualizacion
      2 equipo
      2 www
      3 admiral
      3 word
     45 archivos
     60 en
```

sort -k 1 pepe.txt Ordena el archivo usando el primer campo como clave de ordenación. Tiene el mismos efecto que sort pepe.txt:

```
# sort -k 1  pepe.txt
     10 aires
     15 windows
     16 acer
      1 actualizada
      1 agregar
      1 al
      1 algunos
      1 empleos
      2 actualizacion
      2 equipo
      2 www
      3 admiral
      3 word
     45 archivos
     60 en
```

sort -k 2 pepe.txt Ordena el archivo usando el segundo campo como clave de ordenación:

```
# sort -k 2  pepe.txt
     16 acer
      2 actualizacion
```

```
      1 actualizada
      3 admiral
      1 agregar
     10 aires
      1 al
      1 algunos
     45 archivos
      1 empleos
     60 en
      2 equipo
     15 windows
      3 word
      2 www
```

sort -n pepe.txt Ordena el archivo por valor numérico:

```
# sort -n pepe.txt
      1 actualizada
      1 agregar
      1 al
      1 algunos
      1 empleos
      2 actualizacion
      2 equipo
      2 www
      3 admiral
      3 word
     10 aires
     15 windows
     16 acer
     45 archivos
     60 en
```

sort -nr pepe.txt Ordena el archivo por valor numérico de mayor a menor (invierte el orden):

```
# sort -nr pepe.txt
     60 en
     45 archivos
     16 acer
     15 windows
     10 aires
      3 word
      3 admiral
      2 www
      2 equipo
      2 actualizacion
      1 empleos
      1 algunos
      1 al
      1 agregar
      1 actualizada
```

cut

Trocea una cadena.

Opciones:

-d indica el delimitador, por defecto es el espacio.

-f2 columna con la que nos quedamos.

Ejemplo:

Trocear por el delimitador espacio y quedarse con la segunda columna:

```
cut -d " " -f2
```

grep

Imprime las líneas donde encuentra el patrón.

Opciones:

-c imprime el número de coincidencias.

-w solo busca palabras completas.

-v invierte la búsqueda.

-i no diferencia mayúsculas y minúsculas.

tr

Sustituye caracteres o elimina caracteres repetidos.

Ejemplos:

- Sustituir tabuladores por espacio: tr “\t” “ ”

- Eliminar espacios repetidos: tr -s “ “