Stdin / Stdout / Stderr

En sistemes linux hi ha tres fluxos importantíssims anomenats *stdin*, *stdout* i *stderr*, que acostumen a portar de cap els usuaris principiants i als del món windows, que no ho acaben de pair massa bé… esperem que aquestes explicacions us ajudin.

Tingueu en compte que aquest és un tema una mica complex si no s'ha treballat mai abans el concepte dels fluxos, ara el presentem però l'anireu treballant al larg de tots els EAC.

Fi de flux: ctrl+d

quan s'està escrivint un flux a l'entrada estàndard es finalitza amb ctrl+d. Aquest caràcter significa fi de flux. NO es fa mai *ctrl+c* ni *ctrl+z* (mals vicis de windows).

stdin (0)

Identifica l'entrada estàndard. Per defecte està associada al teclat. Algunes ordres consumeixen l'entrada estàndard, la processen i generen sortida per stdout.

Per defecte l'entrada estàndard és el teclat. Sovinti els alumnes confonen una cosa i l'altre, però són coses diferents. L'entrada estàndard pot procedir de un pipe, per exemple, llavors ja no és el teclat.

Exemples:

```
# comptar línies de 'entrada estàndard, s'entren valors i es finalitza amb el
caràcter de fi de flux ctrl+d.

$ wc -l

un

dos

tres

^d

3
```

l'ordre wc compta les línies de la entrada estàndard, que rep provinent del pipe.

$ cat /etc/passwd | wc -l

53

Stdout (1)

Les ordres generen sortida a *stdout* o sortida estàndard. Per defecte està associada a la pantalla, però NO és la pantalla. Es pot enviar la sortida a pipes o a fitxers (redirigint-la).

llistar l'arrel i la sortida es mostra per stdout (que està associada a la pantalla)

$ ls /

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

llistar l'arrel i enviar la sortida estàndard a un pie (no a stdout), que serà processada per wc.

$ ls / | wc -w

20

llistar l'arrel del sistema i enviar la sortida estàndard a un fitxer amb un redireccionament.

$ ls / > llistat-arrel.txt

Observeu que l'ordre *ls* allò que vol mostrar ho envia a stdout, a la sortida estàndard. En el cas general stdout està associat a la pantalla i veiem el llistat de *ls*. Però NO SEMPRE stdout va a parar a la pantalla. Podeu veure que en els altres dos exemples la sortida de *ls* s'envia en un pipe en un cas i a un fitxer en l'altre.

stderr (2)

Quan les ordres fallen generen missatges d'error a la sortida d'error anomenada *stderr*. Per defecte aquesta sortida està associada a la pantalla. Això a vegades

costa d'entendre perquè es veuen els missatges a la pantalla barrejats tant els de sortida estàndard com els de error, però són sortides diferents!.

Exemples d'ordres:

executar una ordre ls que genera un missatge d'error. Es mostra per pantalla però NO s'envia a la pantalla, s'envia a la sortida d'error.

$ ls /noninoni

ls: cannot access '/noninoni': No such file or directory

observeu el mateix cas, una ordre ls que falla, si enviem la sortida d'error al null NO veiem el missatge d'error perquè s'ha redirigit la sortida d'error a stderr.

$ ls /noninoni 2> /dev/null

idem cas anterior reenviant la sortida d'error a un fitxer de logs amb els arrors.

$ ls /noninoni 2> errors.log

És molt usual executar ordres i desar en un fitxer a part de logs els errors que es generin, d'aquesta manera no surten barrejats a la pantalla i es poden tractar posteriorment.

Generar logs:

observeu que és un error crear /tmp (ja existeix) i /tmp/noexist/dir (no existeix *noeisteix*), però si que es crea */tmp/nou*. es mostren per pantalla els dos errors.

$ mkdir /tmp /tmp/nou /tmp/noexist/dir

mkdir: cannot create directory ‘/tmp’: File exists

mkdir: cannot create directory ‘/tmp/noexist/dir’: No such file or directory

en aquest exemple enviem la sortida d’erros a un fitxer per poder-lo analitzar posteriorment.

$ mkdir /tmp /tmp/nou2 /tmp/noexist/dir 2> errors.log

$ cat errors.log

mkdir: cannot create directory ‘/tmp’: File exists

mkdir: cannot create directory ‘/tmp/noexist/dir’: No such file or directory

Separar les dues sortides

Sovint hi ha ordres que generen sortida i sortida d’errors i ens interessa separar-la posant la sortida estàndard a un lloc i la sortida d’errors a un altre. Per fer això cal separar una sortida i l’altra usant > per a stdout i 2> stderr.

ordre ls que llista l’arrel del sistema i intenta llistar un directori que no existeix. Mostra per stdout el llistat de l’arrel i per stderr el missatge d’error.

$ ls / /noexist

ls: cannot access '/noexist': No such file or directory

/:

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

En aquest exemple s’envia la sortida estàndard del llistat a un fitxer anomenat llistat.txt, i els errors es desen en un fitxer anomenat errors.log.

$ ls /   /noexist > llistat.txt 2> errors.log

$ cat llistat.txt

/:

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

$ cat errors.log

ls: cannot access '/noexist': No such file or directory

Enviar stdout i stderr junts

A vegades interessa enviar les dues sortides juntes a un fitxer, per fer-ho l’operador a utilitzar és &> que redirigeix tant stdout com stderr.

$ ls / /noexist &> sortida.txt

observeu que en el fitxer sortida.txt hi ha tant la sortida estàndard com la sortida d'error.

$ cat sortida.txt

ls: cannot access '/noexist': No such file or directory

/:

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

descartar una o ambdues sortides

Sempre es pot descartar la sortida enviant el contingut al /dev/null. Podem fer-ho tant de stdout com de stderr com de tots dos al mateix temps.

llista l'arrel i un directori inexistent. El llistat es descarta de manera que per pantalla només veiem els missatges d'error.

$ ls / /noexist > /dev/null

ls: cannot access '/noexist': No such file or directory

llista l'arrel i un directori inexistent. els errors es descarten de manera que per pantalla veiem la sortida estàndard (el llistat de l'arrel).

$ ls / /noexist 2> /dev/null

/:

bin boot dev etc home lib lib64 lost+found media mnt opt proc root run sbin srv sys tmp usr var

llistar l'arrel i un directori inexistent, tot es descarta (una ordre realment estúpida!).

```
$ ls / /noexist &> /dev/null
```

Append

Recordeu que l'operador > sobreescriu amb la sortida estàndard, per afegir (append) la sortida estàndard usem >>.

També podem fer append de la sortida d'error, en aquest cas l'operador és 2>>.