Redirections en C-shell

D. Bogdaniuk – P. Portejoie

Antoine Gicquel	**07/11/2016**
1A2	

Les commandes UNIX utilisent 3 fichiers standards pour leurs entrées-sorties :

- **stdin** : le fichier d'entrée standard, canal 0,
- **stdout** : le fichier de sortie standard, canal 1.
  Quand une commande s'exécute, les résultats sont normalement envoyés sur cette sortie.
- **stderr** : le fichier de sortie d'erreur standard, canal 2.
  Quand une commande produit une erreur, le message d'erreur est envoyé sur cette sortie.

Cependant, il est possible de récupérer le contenu des sorties standards dans des fichiers, ainsi que de prendre le contenu d'un fichier comme entrée standard. Ces opérations sont appelées des redirections.

- ***<fichier*** : redirection de l'entrée standard à partir de ***fichier***
- ***>fichier*** : redirection de la sortie standard vers ***fichier***
- ***>>fichier*** : redirection de la sortie standard vers ***fichier***, en ajout
- ***>&fichier*** : redirection de la sortie standard et de la sortie d'erreur vers ***fichier***
- ***>>&fichier*** : redirection de la sortie standard et de la sortie d'erreur vers ***fichier***, en ajout

De plus, il est possible de rediriger séparément les sorties par `(commande > fs) >& erfs` qui redirige respectivement la sortie standard vers le fichier `fs` et la sortie d'erreur vers `erfs`.

Souvent, le résultat d'une commande doit être passé en entrée d'une autre pour obtenir le résultat final. Une solution consiste à récupérer le résultat de la première commande dans un fichier et de l'utiliser en entrée de la suivante. Mais une solution plus souple et qui n'engendre aucune création de fichier intermédiaire consiste à placer un tube (pipe) `|` entre les deux commandes.

Le tube permet de récupérer la sortie standard pour la placer en entrée de la commande suivante.

Par exemple pour compter le nombre de fichiers d'un répertoire, sous réserve que les noms de fichiers ne comportent pas d'espace, il suffit d'utiliser un tube qui relie la sortie de la commande *`ls`* à l'entrée de la commande *`wc`* : `ls | wc -w`

TD – TP

Exercice 1

Dans chacun des cas ci-dessous, identifiez les redirections selon leur type en marquant d'un X la ou les cases appropriées (n'hésitez pas à questionner votre enseignant) :

Commande	entrée standard	sortie standard	sortie d'erreur
`ls > liste`	X	X	
`wc -w < liste`	X	X	
`wc -w < liste > res`	X	X	
`cat liste res > global`	X	X	
`(ls f1 f2 liste res > present) >& absent`	X	X	X
`(ls f1 f2 liste res) >& absent`	X	X	X
`ypcat passwd | wc -l`	X	X	

Exercice 2

Créez un répertoire exo2 ; faites-en votre répertoire courant et créez-y les fichiers suivants :

```
essai.c
essai
essai.o
script.pl
script.res
```

Pour chacune des instructions données ci-après, indiquez ce que contiennent les fichiers de sortie. A chaque fois vous donnerez une copie du résultat que vous analyserez et justifierez.

1/ `ls > f1`

```
[e1600718@ens-iutva-0388 exo2]$ ls > f1
[e1600718@ens-iutva-0388 exo2]$ cat f1
essai
essai.c
essai.o
f1
script.pl
script.res
[e1600718@ens-iutva-0388 exo2]$
```

Le resultat de la commande ls a été redigiré vers le ficher f1.

2/ `ls >> f1`

```
[e1600718@ens-iutva-0388 exo2]$ ls >> f1
[e1600718@ens-iutva-0388 exo2]$ cat f1
essai
essai.c
essai.o
f1
script.pl
script.res
essai
essai.c
essai.o
f1
script.pl
script.res
[e1600718@ens-iutva-0388 exo2]$
```

Le resultat de la commande ls a été redigiré vers le ficher f1 sans écraser son contenu.

3/ `wc -l < f1 > f2`

```
[e1600718@ens-iutva-0388 exo2]$ wc -l < f1 > f2
[e1600718@ens-iutva-0388 exo2]$ cat f2
12
[e1600718@ens-iutva-0388 exo2]$
```

le contenu du fichier f1 a été redegiré vers la comande wc -l. Le résultat de cette commande a été redigiré vers un fichier f2.

4/ `(ls  *.java > f1) >& f2`

```
[e1600718@ens-iutva-0388 exo2]$ (ls  *.java > f1) >& f2
[e1600718@ens-iutva-0388 exo2]$ cat f1
[e1600718@ens-iutva-0388 exo2]$ cat f2
ls: impossible d'accéder à '*.java': Aucun fichier ou dossier de ce type
```

Le résultat de la commande ls *.java est redirigé vers le fichier f1 en écrasant son contenu. Puis le résultat de cette même commande vers la sortie d'erreur standard et le fichier f2 en écrasant son contenu.

Donc f1 ne contient rien car il n'y a pas de fichier finissant par *java, et le fichier f2 contient ce qui a été envoyé à la stderr.

5/ `(ls -l essai vide > f1) >& f2`

```
[e1600718@ens-iutva-0388 exo2]$ (ls -l essai vide > f1) >& f2
[e1600718@ens-iutva-0388 exo2]$ cat f1
-rw-r--r--. 1 e1600718 etud 0  7 nov.  14:04 essai
[e1600718@ens-iutva-0388 exo2]$ cat f2
ls: impossible d'accéder à 'vide': Aucun fichier ou dossier de ce type
[e1600718@ens-iutva-0388 exo2]$
```

Le résultat de la commande ls -l est redirigé vers le fichier f1 en écrasant son contenu. Puis le résultat de cette même commande vers la sortie d'erreur standard et le fichier f2 en écrasant son contenu.

Donc f1 contient uniquement le résultat pour essai car vide n'existe pas et le fichier f2 contient ce qui a été envoyé à la stderr.

6/ `(ls -l essai.c > f1) >& f2`

```
[e1600718@ens-iutva-0388 exo2]$ (ls -l essai.c > f1) >& f2
[e1600718@ens-iutva-0388 exo2]$ cat f1
-rw-r--r--. 1 e1600718 etud 0  7 nov.  14:04 essai.c
[e1600718@ens-iutva-0388 exo2]$ cat f2
[e1600718@ens-iutva-0388 exo2]$
```

Même fonctionnement.

Donc f1 contient le résultat de la commande et f2 est vide car la commande ne contient aucune erreur.

7/ `(ls -l essai vide) >& f2`

```
[e1600718@ens-iutva-0388 exo2]$ cat f2
ls: impossible d'accéder à 'vide': Aucun fichier ou dossier de ce type
-rw-r--r--. 1 e1600718 etud 0  7 nov.  14:04 essai
[e1600718@ens-iutva-0388 exo2]$
```

Même fonctionnement.

Le fichier f2 contient à la fois le résultat de la commande et l'erreur de stderr.

8/ `ls -l | wc -l > f1`

```
[e1600718@ens-iutva-0388 exo2]$ ls -l | wc -l > f1
[e1600718@ens-iutva-0388 exo2]$ cat f1
8
[e1600718@ens-iutva-0388 exo2]$
```

Le contenue de ls -l est redigiré vers wc -l, donc il compte le nombre de ligne affiché par ls -l. Ce resultat est lui même redigiré vers f1 en écrasant son contenu.

Exercice 3

La commande *cat* lit les données sur l'entrée standard jusqu'à la rencontre d'une marque de fin de fichier (saisie de *ctrl-D*) et les affiche sur la sortie standard.

La commande *sort* est sensiblement identique : elle lit les données sur l'entrée standard jusqu'à la rencontre de *ctrl-D* puis elle les affiche sur la sortie standard mais en les triant .

Remarques : - *cat f1* est équivalent à *cat < f1* (les 2 notations sont permises)
- de la même façon *sort f1* est équivalent à *sort < f1*

Vous devez répondre aux questions suivantes en utilisant ces deux commandes et les redirections. Vous devrez dans chacun des cas donner une copie d'écran justificative du résultat.

Question préliminaire :

Lancez la commande *cat* seule (ie sans arguments) puis saisissez des lignes de caractères (utilisez la touche *Entrée* à la fin de chaque ligne et *Ctrl-D* pour terminer la saisie).

Vous observez que chaque ligne saisie est doublée. L'explication est simple : tout caractère entré au clavier est systématiquement envoyé sur la sortie standard pour y être affiché, puis à chaque fin de ligne (écho généré par la touche *Entrée*) la commande *cat* envoie à son tour la ligne reçue sur la sortie standard.

Faites la même chose en lançant cette fois-ci la commande *cat > fs*. Puis affichez le contenu du fichier *fs*. Qu'observez-vous ? Expliquez.

```
[e1600718@ens-iutva-0388 exo2]$ cat > fs
on écrit
du texte
[e1600718@ens-iutva-0388 exo2]$ cat fs
on écrit
du texte
[e1600718@ens-iutva-0388 exo2]$
```

Lit l'entré standard et la redirige vers le fichier fs en écrasant son contenu.

1/ Comment créer un fichier en saisissant les lignes qu'il contiendra au clavier ?

Avec cat et en redirigant avec > nom_du_fichier comme on a pu le faire pour la question préliminaire.

2/ Comment copier un fichier sans utiliser *cp* ?

```
[e1600718@ens-iutva-0388 exo2]$ cat fs > fs2
[e1600718@ens-iutva-0388 exo2]$ cat fs2
on écrit
du texte
[e1600718@ens-iutva-0388 exo2]$
```

En redirigent le résultat de cat vers un autre fichier.

3/ Comment fusionner deux fichiers existants (f1 et f2) vers un troisième qui sera crée (f3), en 2 commandes consécutives ?

```
[e1600718@ens-iutva-0388 exo2]$ cat fs > fs3
[e1600718@ens-iutva-0388 exo2]$ cat fs2 >> fs3
[e1600718@ens-iutva-0388 exo2]$ cat fs3
on écrit
du texte
on écrit
du texte
[e1600718@ens-iutva-0388 exo2]$
```

4/ Comment fusionner deux fichiers existants (f1 et f2) vers un troisième qui sera crée (f3), en une seule commande ?

```
[e1600718@ens-iutva-0388 exo2]$ cat fs > cat fs2 > fs3
[e1600718@ens-iutva-0388 exo2]$ cat fs3
on écrit
du texte
on écrit
du texte
[e1600718@ens-iutva-0388 exo2]$
```

5/ Reprenez la question 1/ (question préliminaire) cette fois-ci avec *sort*

```
[e1600718@ens-iutva-0388 exo2]$ sort > fs
on écrit
de nouveau
du texte
[e1600718@ens-iutva-0388 exo2]$ sort fs
de nouveau
du texte
on écrit
[e1600718@ens-iutva-0388 exo2]$
```

6/ En une seule commande, comment afficher, triées, des données saisies au clavier ?

À l'aide des options disponible de la commande sort (voir man sort).

```
-c      ([NDT] c = check - vérifier) Vérifie si les fichiers fournis
        sont déjà triés : s'ils ne le sont pas, afficher un message
        d'erreur, et terminer avec un code de retour valant 1.

-m      ([NDT] m = merge - melanger) Regrouper les fichiers indiqués en
        les triant. Chaque fichier d'entrée doit déjà être trié indivi-
        duellement. Il est toujours possible de trier plutôt que de
        réunir, le regroupement est fourni parce qu'il est plus rapide
        dans les cas où il fonctionne.
```

```
[e1600718@ens-iutva-0388 exo2]$ sort -m
on écrit du
texte
 encore
on écrit du
texte
 encore
[e1600718@ens-iutva-0388 exo2]$
```

7/ En une seule commande, comment trier des données d'un fichier vers un autre ?

```
[e1600718@ens-iutva-0388 exo2]$ sort -m fs > fs2
[e1600718@ens-iutva-0388 exo2]$ sort fs2
de nouveau
du texte
on écrit
[e1600718@ens-iutva-0388 exo2]$
```

8/ En une seule commande, comment fusionner deux fichiers puis afficher le résultat sous forme triée ?

```
[e1600718@ens-iutva-0388 exo2]$ sort fs fs2
de nouveau
de nouveau
du texte
du texte
on écrit
on écrit
[e1600718@ens-iutva-0388 exo2]$
```

9/ En une seule commande, comment fusionner deux fichiers dans un troisième en triant leurs données ?

```
[e1600718@ens-iutva-0388 exo2]$ sort fs
de nouveau
du texte
on écrit
[e1600718@ens-iutva-0388 exo2]$ sort fs2
de nouveau
du texte
on écrit
[e1600718@ens-iutva-0388 exo2]$ sort fs fs2 > fs3
[e1600718@ens-iutva-0388 exo2]$ sort fs3
de nouveau
de nouveau
du texte
du texte
on écrit
on écrit
[e1600718@ens-iutva-0388 exo2]$
```

Exercice 4

Petit retour sur les scripts et avertissement de sécurité

Soit le script `concat.csh` que vous trouverez sur le forum habituel.

1/ Copiez le fichier dans votre environnement et lancez son exécution (préalablement, pensez aux droits) en lui fournissant comme jeu de données : **`cle`** pour la première chaîne demandée et **`ar`** pour la seconde. Que fait-il ?

```
[e1600718@ens-iutva-0388 exo2]$ cp ~/tpsUNIX/concat.csh ./
[e1600718@ens-iutva-0388 exo2]$ chmod +x concat.csh
[e1600718@ens-iutva-0388 exo2]$ ./concat.csh
Entrez première chaîne : cle
Entrez deuxieme chaîne : ar
clear
[e1600718@ens-iutva-0388 exo2]$
```

Ce script concatène les deux chaine de caractère.

2/ Relancez-le en saisissant **`` `cle ``** pour la première chaîne demandée et **`ar` `** pour la seconde (rappel : `` ` `` est la backquote, caractère qui s'obtient par la combinaison de touches `altgr 7`). Qu'observez-vous ?

Il exécute la commande clear. Donc l'écran est nettoyé.

3/ Editez le contenu du script et observez-en l'algorithme. Que se passerait-il si l'utilisateur saisissait `` `rm -rf *` `` **(ne le faites surtout pas!)** ? Que pouvez-vous en conclure en matière de sécurité ? Proposez une solution d'ordre général (sa mise en œuvre n'est pas demandée dans le cadre de cet exercice)

Cela est très dangereux car il exécutera cette commande et donc supprimera toute les données du répertoire courant.

Ce script est très peu sécurisé car il interprète pas la backquote comme un caractère mais comme une partie de la commande du script.