Projet UNIX

O. Delgrange

Bloc complémentaire au Master d'Informatique
Bloc complémentaire au grade de Master Ingénieur Civil IG
Systèmes d'Exploitation
2018-2019

Généralités

— Les scripts doivent être rendus au plus tard le **vendredi 17 mai 2019** sur la plateforme d'E-learning (Moodle).
— Les scripts UNIX demandés peuvent être développés sur n'importe quel UNIX, mais ils doivent **impérativement** être réalisés dans le Bash. De plus, il doivent fonctionner sur la machine utilisée lors des travaux pratiques (`tpos.umons.ac.be`), c'est sur cette machine qu'ils seront testés.
Vous pouvez y transférer des fichiers en utilisant le protocole SFTP. Pour cela, des clients gratuits existent pour toutes les plate-formes (Filezilla, gftp, ...).
— Si des commandes non vues au cours et non précisées dans ce document sont utilisées, il convient de préciser leur rôle et leur syntaxe.
— Les solutions doivent absolument être accompagnées de **textes explicatifs** décrivant vos méthodes de résolution.
— D'autres scripts peuvent être créés si nécessaire.
Il est impératif que les scripts soient fournis sous la forme de fichiers textes standards et **non sous la forme de fichiers en format propriétaire tels que `.doc`, `.rtf`,...**
En ce qui concerne le texte explicatif, le fournir soit en format texte, soit le convertir en .pdf.
— Le travail est **strictement personnel**, il compte pour $\frac{1}{3}$ des points du cours « Systèmes d'Exploitation ». Veillez donc à **ne pas divulguer trop d'informations sur votre travail à vos camarades, vous pourriez être sanctionné !**

De la même manière, il vous est interdit de faire réaliser les scripts par quelqu'un d'autre.

Vous pourriez être soumis à des interrogations pointues sur votre manière de résoudre le problème, pour s'assurer que ces règles ont été respectées.

Application client/serveur de lancement paramétré de commandes

Développer, en BASH, une application client-serveur de lancement paramétré d'une commande externe. Le client demande à l'utilisateur quelle commande externe exécuter, recherche la commande, demande quelles options doivent être associées à la commande (lancement en arrière plan, redirection des résultats vers un fichier, lancement moins prioritaire, ...) avant de donner l'ordre au serveur (via les communications par tube) d'exécuter la commande.

Cahier des charges

Fonctionnement du script client

Le script client commence par demander à l'utilisateur le nom de la commande à lancer. L'utilisateur répond en donnant une expression régulière correspondant au nom de la commande.

Les noms des commandes externes à rechercher sont ceux satisfaisant l'expression régulière, qui correspondent à des fichiers accessibles en exécution, et qui de plus vérifient une des conditions suivantes :

— commande présente directement dans un des répertoires de `$PATH`. Par exemple, `/usr/exec/mkobj` pour l'expression `.*kob.*` si `/usr/exec` est présent dans `$PATH`

— commande présente dans la descendance d'un des répertoires de `$PATH`, descendance qui doit contenir explicitement un répertoire `bin`. Par exemple, `/usr/exec/ver3/bin/sciprog/mkenv` pour l'expression `^mk.*` si `/usr/exec` est présent dans `$PATH`.

Si une seule commande vérifie ces conditions, c'est elle qui est automatiquement choisie, autrement un menu, comportant toutes les commandes satisfaisant les conditions, est présenté à l'utilisateur qui choisit celle qui l'intéresse.

Ensuite, le script demande les paramètres à passer à commande, si la commande doit être lancée avec une priorité moindre, si elle doit être lancée en arrière plan, et enfin si la sortie de la commande doit être redirigée, et le cas échéant vers quel fichier.

Si la commande doit être lancée en arrière plan, il faut systématiquement demander vers quel fichier doit être redirigée la sortie de la commande.

La commande est ensuite soumise au serveur (par l'intermédiaire d'un tube de communication) qui l'exécutera conformément aux choix de l'utilisateur.

Si la commande est lancée en arrière plan, le client a terminé son travail car la commande s'exécutera indépendamment de lui. Dans le cas contraire, les résultats de la commande doivent être communiqués du serveur au client pour qu'il les affiche (que la redirection ait été demandée ou non). Le client doit donc attendre que les résultats soient disponibles avant de se terminer.

Toutes les entrées de l'utilisateur doivent être vérifiées. Par exemple, est-ce que le serveur tourne au moment où le client est sollicité, est-ce que le fichier de redirection existe déjà, veut-on l'écraser, est-ce que le fichier de redirection peut-être créé dans son répertoire, est-ce que la réponse de l'utilisateur est valide,. . . ? Lors de la fin du client, tous les fichiers, répertoires et tubes temporaires doivent être correctement effacés.

Fonctionnement du script serveur

Le serveur sera lancé par un script. Une fois lancé, il se place dans une boucle dont chaque étape consiste à recevoir une demande d'un client, la traiter et lui communiquer éventuellement les résultats. Contrairement au client, il n'est absolument pas interactif : il communique avec le client pour recevoir une commande à traiter et lui pour fournir éventuellement les résultats une fois la commande exécutée. Il peut donc être lancé en arrière plan.

Le serveur sera terminé par l'invocation d'un script `stopserv` qui lui donnera l'ordre de se terminer. Tous les fichiers, répertoires et tubes temporaires, seront alors correctement effacés.

Il convient que `startserv` et `stopserv` vérifient que, si le serveur est déjà démarré, il ne peut plus l'être à nouveau ; pour stopper le serveur, il faut qu'il ait été démarré, . . .

Exemple :

```
$ echo $PATH
.:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:
/snap/bin:/home/olivier/util
$ ./stopserv.sh
Erreur : Le serveur ne tourne pas.
$ ./startserv.sh &
[4] 25225
$ Démarrage du serveur, proc 25225

$ ./client.sh
```

```
Introduire une expression pour la commande à exécuter : ^cat$
Commande : /bin/cat
Introduisez les parametres de la commande : /etc/fstab
Vous souhaitez executer la commande : /bin/cat /etc/fstab
Souhaitez-vous que la commande soit ralentie lors de l'exécution ? o
Souhaitez-vous que la commande soit lancée en arrière plan ? n
Souhaitez-vous que la sortie standard soit redirigée ? o
Nom de fichier pour la redirection des résultats : /tmp/res.txt

# /etc/fstab: static file system information.
#
# Use 'blkid' to print the universally unique identifier for a
# device; this may be used with UUID= as a more robust way to name devices
# that works even if disks are added and removed. See fstab(5).
#
# <file system> <mount point>   <type>  <options>       <dump>  <pass>
# / was on /dev/nvme0n1p6 during installation
UUID=31049d3a-3bbe-4ead-84dc-77d401790454 /               ext4    errors=remount-ro 0       1
# /boot/efi was on /dev/nvme0n1p1 during installation
UUID=F2BB-2E14  /boot/efi       vfat    umask=0077      0       1
# /home was on /dev/nvme0n1p8 during installation
UUID=afd2f407-b280-42ec-9fb8-edb1c66096f9 /home           ext4    defaults        0       2
# /mnt/Windows was on /dev/nvme0n1p3 during installation
UUID=DA28E42328E3FFFD /mnt/Windows    ntfs    defaults,umask=007,gid=46 0       0
# swap was on /dev/nvme0n1p7 during installation
UUID=390832de-db4e-4fcb-b13b-4cff64c11ba3 none            swap    sw              0       0
$ ls -l /tmp/res.txt
-rw------- 1 olivier olivier 1005 mar 19 12:15 /tmp/res.txt

$ ./client.sh
Introduire une expression pour la commande à exécuter : .*ls$
 1) /bin/ls                          15) /usr/bin/mailutils
 2) ./bin/nls                        16) /usr/bin/md5sum.textutils
 3) /bin/ntfsls                      17) /usr/bin/messages.mailutils
 4) ./bin/rep/myls                   18) /usr/bin/movemail.mailutils
 5) /home/olivier/util/bin/nls       19) /usr/bin/mtools
 6) /home/olivier/util/bin/rep/myls  20) /usr/bin/pdf-redact-tools
 7) /usr/bin/bib2gls                 21) /usr/bin/readmsg.mailutils
 8) /usr/bin/dotlock.mailutils       22) /usr/bin/smbcacls
 9) /usr/bin/dpkg-gensymbols         23) /usr/bin/subptools
10) /usr/bin/frm.mailutils           24) /usr/bin/systemd-cgls
11) /usr/bin/from.mailutils          25) /usr/bin/tdbbackup.tdbtools
12) /usr/bin/gvfs-ls                 26) /usr/bin/traceroute6.iputils
13) /usr/bin/hp-levels               27) /usr/bin/zipdetails
14) /usr/bin/mail.mailutils
Faire votre choix : 1
Commande : /bin/ls
Introduisez les parametres de la commande : /home/olivier/util
Vous souhaitez executer la commande : /bin/ls /home/olivier/util
Souhaitez-vous que la commande soit ralentie lors de l'exécution ? n
Souhaitez-vous que la commande soit lancée en arrière plan ? o
Nom de fichier pour la redirection des résultats : /tmp/res.txt
Fichier /tmp/res.txt existant, désirez vous l'écraser ? o
$ cat /tmp/res.txt
bin
calcul
move
mp32wav
```

```
myX
Normalize
ods
reconstruct
removepoint
rtlrestart
stdrename
supprchaine
synosync
tb
webget

$ ps
  PID TTY          TIME CMD
 5677 pts/1    00:00:00 bash
 5687 pts/1    00:00:51 thunderbird
 9758 pts/1    00:01:44 latexila
 9863 pts/1    00:00:48 evince
14229 pts/1    00:00:22 geany
25225 pts/1    00:00:00 startserv.sh
25921 pts/1    00:00:00 ps
$ ./stopserv.sh
$
[4]+  Processus arrêté      ./startserv.sh
$ ps
  PID TTY          TIME CMD
 5677 pts/1    00:00:00 bash
 5687 pts/1    00:00:51 thunderbird
 9758 pts/1    00:01:44 latexila
 9863 pts/1    00:00:48 evince
14229 pts/1    00:00:22 geany
25931 pts/1    00:00:00 ps
```

Indications :

— La commande `find` possède un critère de sélection des références sur base des expressions régulières : `find ... -regextype egrep -regex` *`expr.reg...`* Cette option peut être utilisée. Attention cependant que l'expression régulière s'applique à toute la référence du fichier (et pas seulement au nom final du fichier dans le répertoire). L'expression régulière fournie au client doit donc être modifiée pour prendre cela en compte.

— Il convient d'être prudent sur la façon d'utiliser les tubes. Ainsi, il faut éviter que des informations soient perdues par les ouvertures/fermetures multiples des tubes et que des informations soumises par un client entrent en interférence avec les informations d'un autre client lancé quasiment simultanément.

Bon travail !