Unix et Programmation Shell

10 janvier 2015

Vous répondrez aux questions en utilisant deux lignes en moyenne.

Q 1. Commenter

```
$ echo {a..h}{1..8} | wc -w
64
```

Q 2. Citer trois paramètres spéciaux correspondant à l'expression régulière : `\$[^1-9]`

Q 3. Citer 5 sections du manuel **unix**.

Q 4. Citer deux pionniers du système **unix**.

Q 5. Que signifie GUI ?

Q 6. Citer 5 types de fichiers présents dans un système de fichier.

Q 7. Décrire deux mécanismes fondamentaux du système UNIX.

Q 8. Donner un exemple de binaire **suid** ?

Q 9. Commenter.

```
$ touch x xy xyz
$ find . -name x*
find: les chemins doivent preceder l
   'expression : xy
```

Q 10. Quelle différence entre `fopen` et `open` ?

Q 11. Commenter

```
$ find /bin -name 's*' -type l
/bin/sh
```

Q 12. Que donne

```
find /etc -perm 000
```

Q 13. Citer trois commandes externes homonymes d'une commande interne.

Q 14. Que fait la commande :

```
$ find -name '*' | xargs rm -f
```

Q 15. Expliquer

```
$ touch ls
$ PATH=.
$ ls
bash: ./ls: Permission non accordee
```

Q 16. Expliquer

```
$ unalias x
```

Q 17. Ecrire un script `tri.sh` pour trier les arguments de la ligne de commande en mode numérique.

Q 18. Proposer une explication

```
$ find -name x*
./x.log
$ touch xxx
$ find -name x* 2>/dev/null;
$ echo $?
1
```

Q 19. Quel est le résultat de

```
$ echo {1..3} \
       | awk -F2 '{print $2$1}'
```

```
execve("/bin/bash...
open("/etc/ld.so.cache", O_RDONLY)      = 3
open("/dev/tty", O_RDWR|O_NONBLOCK|O_LARGEFILE) = 3
open("/proc/meminfo", O_RDONLY)         = 3
read(3, "MemTotal:         508520 kB\nMemF"..., 1024) = 1024
clone(Process 2261 attached
[pid  2260] clone(Process 2262 attached
[pid  2262] dup2(3, 0)                  = 0
[pid  2262] open("/tmp/dst", O_WRONLY|O_CREAT|O_TRUNC|O_LARGEFILE, 0666)
[pid  2262] dup2(3, 1)                  = 1
[pid  2262] execve("/usr/bin/wc", ["wc", "-l"], [/* 52 vars */] <unfinis
[pid  2261] dup2(4, 1)                  = 1
[pid  2261] open("/tmp/src", O_RDONLY|O_LARGEFILE) = -1 ENOENT (No such
[pid  2261] read(3, "# Locale name alias data base.\n#"..., 4096) = 2512
[pid  2261] read(3, "", 4096)           = 0
[pid  2261] write(2, "bash...
) = 52
[pid  2262] <... execve resumed> )      = 0
[pid  2262] open("/etc/ld.so.cache", O_RDONLY) = 3
[pid  2262] read(3, "# Locale name alias data base.\n#"..., 4096) = 2512
[pid  2262] read(3, "", 4096)           = 0
[pid  2262] read(0, "", 16384)          = 0
[pid  2262] write(1, "0\n", 2)          = 2
--- {si_signo=SIGCHLD, si_code=CLD_EXITED, si_pid=2261, si_status=1, si_
```

FIG. 1 – traces d'une commande.

Q 20. Que fait la commande

```
$ sed -irE 's/[ ]+$//' fichier
```

Q 21. Que fait la commande

```
$ grep -vE '^$' src > dst
```

Q 22. Que fait la commande

```
$ grep -rlE '([0-9]+\.){3}[0-9]+' \
  --include=r*.*f /tmp 2>/dev/null
```

Q 23. Expliquer

```
$  x=$$; while [ $x !=  0 ];do;  \
ps --no-headers -p$x -opid,cmd;  \
x=$(ps --no-headers -p$x -oppid);\
done

2679 /bin/bash
2678 vim upsh-x-ct-13-1.tex
2571 bash
2049 gnome-terminal
   1 /sbin/init
```

Q 24. Expliquer

```
$ mkfifo /tmp/fifo
$ tr 123 456 < /tmp/fifo &
$ echo 1 2 3 4 5 6  > /tmp/fifo
4 5 6 4 5 6
```

Q 25. Le script ci-dessous a été utilisé pour obtenir les traces de la figure (1).

```
#!/bin/bash
strace -ff -e trace=open,read,write,dup2,clone,execve \
   bash -c "$*" |& sed 's/bash.*/bash.../' \
    | grep -vE '(share|lib|stack|^Proc|ELF)' | cut -c1-72 > traces.txt
```

- Combien de processus ont été lancés ?
- Quelle commande a été tracée ?