3R-IN3 – Systèmes d'exploitation
TP 2

Xavier Hilaire, Clément Boin
x.hilaire@esiee.fr, clement.boin@gmail.com

22 mai 2024

1 Réponse en temps limité

La commande interne `read` du shell permet de lire une ligne à partir du terminal, mais attend indéfiniment que l'utilisateur veuille bien en saisir une. Dans cet exercice, on cherche à écrire un script qui attend la réponse de l'utilisateur en un temps limité : s'il ne saisit rien au-delà d'une limite de temps, le script doit supposer une réponse par défaut, et poursuivre son exécution.

1.1 Script question.sh, première version

Ecrivez un premier script `question.sh` qui fera les choses suivantes :
- Il affiche son PID sur sa sortie d'erreur standard
- Il affiche son unique argument, supposé être une question
- Il lit la réponse de l'utilisateur, qu'on supposera de type oui/non
- Si l'utilisateur a répondu `O` ou `o`, le script doit terminer avec le code de sortie 0, sinon avec 1.

Exemple de résultat attendu :

```
[xavier@localhost TD]$ ./question.sh "Aimez-vous la tarte au concombre (O/N) ? "
PID = 14473
Aimez-vous la tarte au concombre (O/N) ? o
[xavier@localhost TD]$ echo $?
0
[xavier@localhost TD]$ ./question.sh "Aimez-vous la tarte au concombre (O/N) ? "
PID = 14482
Aimez-vous la tarte au concombre (O/N) ? O
[xavier@localhost TD]$ echo $?
0
[xavier@localhost TD]$ ./question.sh "Aimez-vous la tarte au concombre (O/N) ? "
PID = 14493
Aimez-vous la tarte au concombre (O/N) ? pas du tout
[xavier@localhost TD]$ echo $?
1
[xavier@localhost TD]$
```

1.2 Script question.sh, deuxième version

- Modifiez le script précédent pour garantir qu'il termine au bout de 5 secondes, même si l'utilisateur ne saisit rien. Vous pourrez utiliser un autre script ou processus, et les

commandes `kill` et `sleep`, vues en cours, pour ce faire. Exemple de résultat attendu :

```
[xavier@localhost TD]$ date ; ./question.sh "Aimez-vous la tarte au concombre (O/N) ?
sam. 20 mai 2023 10:33:28 CEST
PID = 14975
Aimez-vous la tarte au concombre (O/N) ?
[xavier@localhost TD]$ date
sam. 20 mai 2023 10:33:33 CEST
[xavier@localhost TD]$
```

— Examinez le code de sortie retourné par le shell dans ce cas de figure.
— Votre script est-il sans danger dans le cas où l'utilisateur saisit quelque chose dans les temps ? S'il y a un risque, peut-on le corriger, et comment ?

1.3 Script principal

Finalement, écrivez et testez un script `principal.sh` qui appellera `question.sh`, et affichera l'un des messages suivants selon le code de retour qu'il aura reçu :

0	Je savais que vous répondriez oui
1	Je savais que vous répondriez non
autre	Vous êtes lent !

2 Scanneur ICMP

La commande `ping -c 1 -w 1 machine` permet de tester la réponse ICMP d'une `machine` spécifiée à l'aide d'un seul paquet ICMP, et en n'attendant pas plus d'une seconde. Examinez son code de sortie sur une machine ayant toutes les chances de répondre (127.0.0.1 par exemple), et peu de chances de répondre (10.0.0.15 par exemple).

Sachant cela, écrire un script `scan.sh [-p] [-o fichier] adresse/nn` qui fait les choses suivantes :

— L'ensemble `adresse/nn` est une adresse IPv4 de réseau en notation condensée CIDR. On supposera que `nn` peut valoir 32, 24, ou 16, et on ne traitera pas les autres cas – vous pourrez vous contenter d'un message « cas `nn` non traité
— Le script doit tester la réponse ICMP de l'ensemble des machines adressables par l'adresse CIDR fournie. Vous devrez a minima faire ces tests séquentiellement, mais vous pourrez faire mieux en parallèliser les tests par lots de 256. En d'autres termes, vous pouvez lancer un lot de 256 tests parallèles, puis lancer le lot suivant (si c'est nécessaire) une fois tous les tests du lot courant terminés. Conseil : écrivez et validez d'abord la version séquentielle, la version parallèle s'en déduisant presque immédiatement.
— Lorsqu'une machine répond à un ping, le script doit afficher un message de la forme « La machine xx.yy.zz.tt » a répondu a ping » et rien d'autre.
— Lorsqu'une machine ne répond pas à un ping, le script doit afficher un message de la forme « La machine xx.yy.zz.tt » n'a pas répondu a ping » et rien d'autre, sauf lorsque l'option `-p` a été utilisée
— Si l'option `-o fichier` a été utilisée, alors le script doit écrire non pas sur la sortie standard, mais dans le `fichier` spécifié plutôt qu'à l'écran.

On ne doit voir s'afficher aucun des messages écrits par `ping` à l'écran. Conseils :

— Commencez par écrire un autre scipt, ou mieux, une fonction, qui teste une adresse IP particulière en affichant ce qu'il faut et là où il faut
— Appelez ce script ou cette fonction d'abord séquentiellement, puis parallèlement
— Traitez les cas 32, 24 et 16 dans cet ordre
— Le domaine esiee.fr est de classe B, et toutes les machines sont en 147.215.0.0/16.

Mais très peu répondent au ping. Essayez principalement de tester les réponses sur le réseau en /24 sur lequel se trouve la vôtre, il est peu probable que les tests au-delà soient représentatifs.

3 Processus

Source : C. Boin

1. Lancez votre terminal
2. Avec l'éditeur « vim » ou « nano », saisir ce programme, nommé « pg_erreur », dans le répertoire courant.

```
#!/bin/bash

while (true)
do
echo Bonjour
done
```

3. Enregistrez-le.
4. Rendez-le fichier « pg_erreur » exécutable.
5. Exécutez-le.
6. Que se passe-t-il ?
7. Arrêtez le programme par Ctrl-C.
8. Relancez le programme mais en faisant suivre son nom par un « & ».
9. Arrêtez le programme par Ctrl-C. Que constatez-vous ?
10. Pourquoi ?
11. Allez sur TTY2 et connectez-vous (Ctrl Droit + F2)
12. La commande « ps » liste tous les processus actifs de la session utilisateur. Quelle commande doit-t-on taper pour voir tous les processus en cours ?
13. Recherchez le script que vous venez d'écrire. Comment le reconnaissez-vous ?
14. Quel est son identifiant (PID) ?
15. Quel est l'identifiant de son père ?
16. Lancer la commande « top ». Où est situé le programme « pg-erreur » ?
17. Quel est le processus qui consomme le plus de temps processeur ?
18. Quittez « top » en appuyant sur « q ».
19. Tuez le processus de « pg-erreur » à l'aide de la commande « kill » (ou avec « top »).
20. Vérifiez avec « top » que tout est rentré dans l'ordre. Comment le voyez-vous ?
21. Relancez à nouveau le script précédent, et interrompez-le avec la commande Ctrl-Z, que se passe-t'il ?
22. Tapez la commande « fg », que se passe-t'il ?
23. Tapez à nouveau Ctrl-Z, puis tapez la commande « bg », que se passe-t'il ? Et comment se sortir de cette situation ?
24. Expliquer brièvement le fonctionnement des commandes « nice » et « nohup »