Debugging Système et Noyau
-
Travaux Pratique

Aurélien Cedeyn

2017-2018

Préparation

- Vous disposez d'une machine virtuelle préparer pour suivre ce TP.
  - Utilisateurs :
    - root : debug
    - user : debuguser
- Configuration de la machine virtuelle
  - Mémoire : 2048Mo
  - Système : Fedora x86_64
- Pour faciliter la récupération des fichiers de la VM
  - Installer un serveur ssh sur votre machine.
  - Copier les fichiers désirés depuis la VM vers votre machine :

```
Transfert des réponses
$ scp <fichier> <utilisateur>@<ip_de_la_machine>:<destination>
```

- Ce TP est noté, il vous est demandé de rendre ce que vous avez pu réaliser à la fin de celui-ci.
- Vous avez jusqu'au vendredi 22/12 23h59 pour envoyer le compte rendu complet du TP.
- Ces deux échéances constitueront votre note de TP.
- Les différentes réponses avec les sorties de vos commandes devront suivre le format suivant :
  - Un répertoire à vos nom et prénoms.
  - Un un fichier par question.

```
Format des réponses
$ ls aurelien.cedeyn/
1.1.txt 1.2.txt 1.3.txt 1.4.txt
2.1.txt 3.1.txt 3.2.txt 3.3.txt 3.4.txt
```

Kernel space

1 cscope

1. Placez vous dans le répertoire des sources du noyau (*/home/user/linux-4.14/*).
2. Construisez l'index des sources *make cscope* (une fois n'est pas coutume, ne pas tenir compte de erreur).
3. Lancez *export EDITOR=vim ; cscope -d -R*.
4. Pour chacune des questions suivantes, indiquez le fichier et la ligne à laquelle se trouve le symbole demandé :
   — *task_ struct*
   — *modules*
   — *vfs_ open*

2 /sys/kernel/debug/dynamic_debug

1. Activez tous les messages de debug du fichier *net/ipv4/ping.c* :
   — Parcourez la documentation et indiquez la commande à lancer pour activer ces messages (*/sys/kernel/debug/dynamic_ debug/control*).
   — Vérifiez que les messages sont bien activés via *dmesg* (indiquez la commande utilisée pour générer les messages que vous voyez).

3 /sys/kernel/debug/tracing

1. Quel est le *tracer* par défaut (*current_ tracer*) ?
2. Désactivez le tracing.
3. Configurez le tracing pour le processus nommé crazy.
4. Utilisez le *tracer* nommé *function*.
5. Visualisez le fichier *trace*.
6. Lancez une capture d'1s.
7. Visualisez le résultat.
8. **Indice :** Lire le fichier *README*.

4 perf

1. Enregistrez une trace perf pour le processus *crazy*.
   — Utilisez l'option permettant d'enregistrer le graph des functions *call-graph*.
2. Visualisez le résultat.
3. Quel sont les deux appels systèmes les plus utilisés par le processus *crazy* ?
4. Quel système de fichier accède le processus *crazy* ?

5 crash

1. Lancez *crash*.
2. Choisissez un processus (*set PID*).
3. Affichez sa structure *task_struct*.
4. Affichez la structure *mm_struct* correspondante
5. À quelle adresse se trouve :
   — Le début de la stack du processus ?
   — Le code ?
   — Les arguments ?
   — **Indice :** pour chacune des commandes indiquez le champ de la structure *mm_struct* et sa valeur.
   — **Bonus :** lisez brutalement l'adresse des argument et donnez le résultat.
6. Visualisez les processus dans l'état *UNINTERRUPTIBLE*.
7. Pour chaque processus bloqué, affichez sa pile d'appel.
8. Que pouvez vous tirer de ces informations ?