

Debugging Système et Noyau

-

Travaux Pratiques

Aurélien Cedeyn

2018-2019

Préparation

- Vous disposez d'une machine virtuelle *pcocc* pour suivre ce TP.
 - Template *pcocc* : debug
 - Utilisateurs :
 - votre login
 - Configuration *pcocc*

```
_____ $HOME/.pcocc/cloud-user-data/debug _____  
#cloud-config  
  
users:  
- name : <login>  
  sudo: ['ALL=(ALL) NOPASSWD:ALL']  
  ssh-authorized-keys:  
  - <clé publique ssh>  
- name : root  
  ssh-authorized-keys:  
  - <clé publique ssh>  
_____
```

- Configuration de la machine virtuelle
 - Système : CentOS-7.5
- 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 14/12 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 (*/root/linux-3.10.0-862.14.4.el7.x86_64*).
2. Construisez l'index des sources *make cscope*
3. Lancez *export EDITOR=vim; cscope -d -R*.
4. Pour chacun des symboles suivants, indiquez le fichier et la ligne à laquelle il se trouve :
 - *cpu_hw_events*
 - *vfs_open*
 - *task_struct*

2 /sys/kernel/debug/dynamic_debug

1. Activez tous les messages de debug du fichier *net/ipv4/ping.c* :
 - Parcourez la documentation :
/root/linux-3.10.0-862.14.4.el7.x86_64/Documentation/dynamic-debug-howto.txt
 - Indiquez la commande à lancer pour activer ces messages.
 - Vérifiez que les messages sont bien activés via *dmesg* (indiquez la commande utilisée pour générer les messages que vous voyez).

— Désactivez les messages de debug.

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'ls.
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 fonctions *call-graph*.
2. Visualisez le résultat.
3. Quel sont les deux appels systèmes les plus utilisés par le processus *crazy* ?

4. À quel système de fichiers 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 arguments 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 ?