TP1 Bash:

Exploration des processus

Introduction :

Sur un système d'exploitation basé sur le noyau Linux, les programmes s'exécutent en appelant des processus. Les processus sont représentés par des répertoires disponibles dans `/proc` nommés avec le numéro de processus. Dans un de ces répertoires on trouve le fichier `/proc/num_pid/status` qui contient des informations utiles :

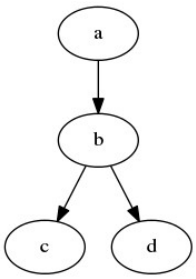
- Name : le nom du processus
- Pid : le numéro du processus
- PPid : le numéro du processus parent

1 – Prise en main de DOT :

On trace un graphique pour la syntaxe, s'appelant toto avec l'extension .dot. On utilise la commande `dot toto -Tpng -ototo.png` afin de générer une image du graphique.

```
digraph toto {
    a -> b -> c;
    b -> d;
}
```

---->



2 – Script Bash :

Nous allons représenter tous les processus exécutés sur la machine sous la forme d'un graphique orienté. Les informations à afficher pour chaque processus sont :

- le PID
- le nom du processus
- le processus père
- le ou les processus fils
- la mémoire allouée en pourcentage.

Deux méthodes sont envisageables, l'une utilise les répertoires des processus à /proc . L'autre reprend les informations renvoyées par ps . C'est la deuxième

```
#!/bin/bash
chmod a+x explorateur_processus.sh
#
#fichier .dot
tab=./explorateur_processus.dot
#
# Récupération : nom, pid, ppid, pourcentage mem utilisée, trié par pid
ps xao comm,pid,ppid,%mem,rss --sort -pid | tr -s " " > /tmp/resultats.ps
#
# Supprimer 1 ère ligne (attributs des colonnes)
sed -i 1d /tmp/resultats.ps
#
#
echo -e "digraph explorateur_processus {\n\t" > $tab
# On lit le fichier contenant le résultat de ps et on récupère les données
cat /tmp/resultats.ps |
while read line
do
  nom=$(echo $line | cut -d" " -f1)
  num_PID=$(echo $line | cut -d" " -f2)
  num_PPID=$(echo $line | cut -d" " -f3)
  mem=$(echo $line | cut -d" " -f4)
#
# Vérification existence de PID et PPID
  if [ -n $num_PID -a -n $num_PPID ]
  then
# On affiche les liens du PPID à PID. Entre []: paramètres de l'affichage
    echo -e "\t$num_PPID -> $num_PID;\n" >> $tab
    echo -e "\t$num_PID[label="\"$num_PID\\n$nom\"",shape="box",color="blue",height="$mem",width="$mem"];" >> $tab
  fi
done
#
echo '}' >> $tab
#
# Création de l'image
dot explorateur_processus.dot -Tpng -oexplorateur_processus.png
```

méthode que l'on va appliquer dans ce TP.

La variable tab contient le chemin du fichier .dot à créer.

La commande ps xao comm,pid,ppid,%mem,rss --sort -pid | tr -s « « récupère les informations dont nous avons besoin, et supprime les doublons d'espace, afin de pouvoir l'utiliser comme délimiteur plus tard.

Ensuite la commande sed -i 1d /tmp/resultats.ps sert à supprimer la première ligne des résultats de ps (qui contient les noms de colonnes).

Création du fichier .dot

On ouvre le fichier de résultats, et pour chaque ligne on boucle.
Nous allons créer le fichier .dot, nommé « explorateur_processus.dot » avec cette ligne : echo -e "digraph explorateur_processus {\n\t" > $tab
Les caractères spéciaux servent à revenir à la ligne puis mettre une tabulation.

Maintenant, on va traiter chaque ligne renvoyée par ps et remplir le fichier.dot.

On définie les variables nom num_PID num_PPID mem grâce à des cut.

```
cat /tmp/resultats.ps |
while read line
do
  nom=$(echo $line | cut -d" " -f1)
  num_PID=$(echo $line | cut -d" " -f2)
  num_PPID=$(echo $line | cut -d" " -f3)
  mem=$(echo $line | cut -d" " -f4)
#
# Vérification existence de PID et PPID
  if [ -n $num_PID -a -n $num_PPID ]
  then
# On affiche les liens du PPID à PID. Entre []: paramètres de l'affichage
    echo -e "\t$num_PPID -> $num_PID;\n" >> $tab
    echo -e "\t$num_PID[label="\"$num_PID\\n$nom\"",shape="box",color="blue",height="$mem",width="$mem"];" >> $tab
  fi
done
#
echo '}' >> $tab
#
# Création de l'image
dot explorateur_processus.dot -Tpng -oexplorateur_processus.png
```

Traitement des données

On vérifie que le PID et que le PPID existent avec une boucle if :
if [-n $num_PID -a -n $num_PPID]

Puis on défini le lien entre le processus actuel et son processus parent, on ajoute un retour à la ligne et on redirige le résultat vers le graphique.
echo -e "\t$num_PPID -> $num_PID;\n" >> $tab

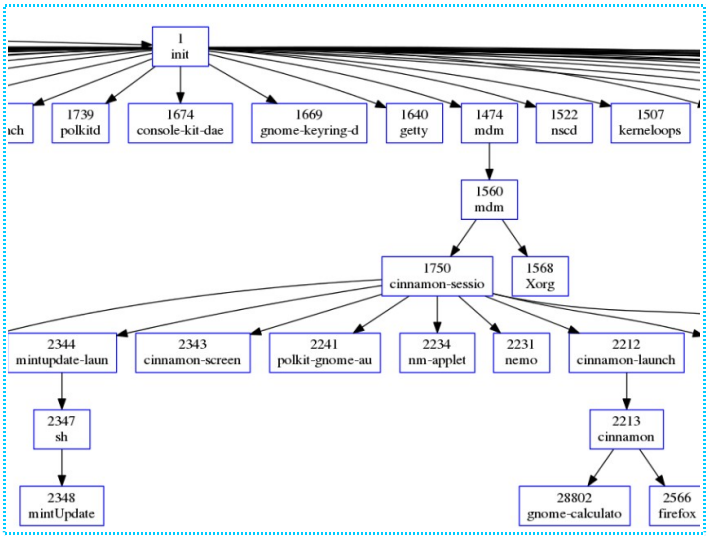
On complète avec la ligne suivante :
echo
-e"\t$num_PID[label="\"$num_PID\\n$nom\"",shape="box",color="blue",height="$mem",width="$mem"];" >> $tab

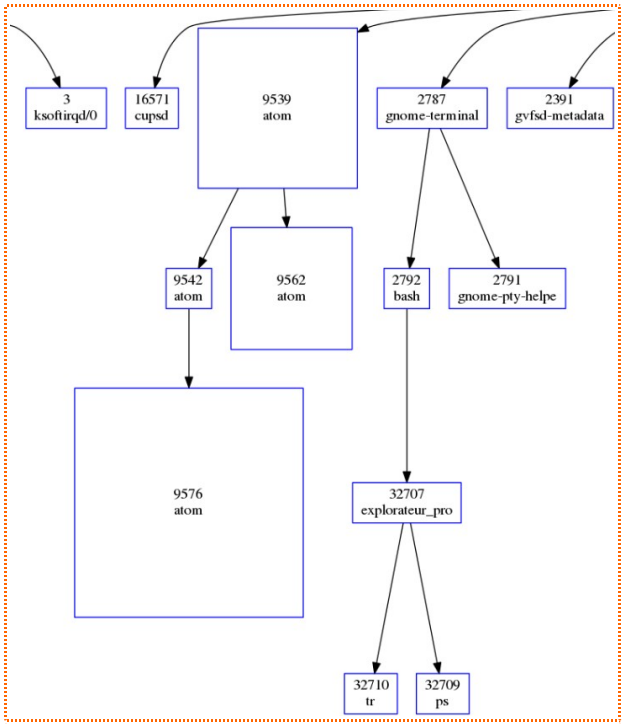
Cette instruction paramètre l'affichage de la bulle pour le processus traité (nom, forme en boite, couleur bleue, taille fonction de mémoire utilisée).

On ferme les boucles, on ferme le graphique avec un echo '}' >> $tab

La dernière étape est la génération de l'image correspondante :
dot explorateur_processus.dot -Tpng -oexplorateur_processus.png

Voici le résultat sans la mémoire allouée présentée puis avec :





Conclusion

Pour conclure, nous avons appris à représenter , et voir en détail l'ensemble des processus d'une machine sous un noyau Linux avec le DOT.