Le point de montage du pseudo système de fichiers du noyau ("/proc")

Ce répertoire représente le point de montage du pseudo système de fichiers du noyau. Ce dernier contient des fichiers permettant d'accéder aux informations sur le matériel, la configuration du noyau et sur les processus en cours d'exécution. [5] .

On retrouvera donc un répertoire par processus actif. Chacun porte le numéro du processus (PID). Il est constitué des fichiers suivants :

- cmdline contient la ligne de commande qui a créé le processus.
- status contient des informations sur l'état du processus (en attente, en exécution, propriétaire…).
- exe est un lien vers le fichier exécutable utilisé par le processus.

Dans le répertoire /proc, on retrouve des fichiers contenants des informations générales sur le système. Comme :

- uptime donnant le temps de fonctionnement du système.
- stat donnant diverses statistiques sur l'utilisation des ressources du système (CPU, mémoire…).
- meminfo donnant un récapitulatif de l'utilisation de la mémoire.
- cpuinfo donnant une description des CPU du système.

Répertoire /proc

Qu'est-ce que "/proc" ?

/proc est un système de fichier utilisé par le noyau pour envoyer des informations aux différents processus d'ou le nom **/proc**.

La modification des fichiers qu'il contient peut changer des paramètres du noyau à la volée.

A la différence des autres répertoires, **/proc** est stocké en mémoire et non sur le disque dur.

Voici, par exemple, l'intérieur du fichier **cpuinfo**. Pour le consulter rapidement :

```
cat /proc/cpuinfo
```

En voici le résultat :

```
processor : 0
vendor_id : GenuineIntel
```

```
cpu family : 15
model : 1
model name : Intel(R) Celeron(R) CPU 1.80GHz
stepping : 3
cpu MHz : 1804.811
cache size : 128 KB
fdiv_bug : no
hlt_bug : no
f00f_bug : no
coma_bug : no
fpu : yes
fpu_exception : yes
cpuid level : 2
wp : yes
flags : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat
pse36 clflush dts acpi mmx fxsr sse sse2 ss ht tm
bogomips : 3579.90
```

Les informations utiles du noyau :

Fichier	Description
/proc/cpuinfo	Concerne les informations sur le Processeur (Modèle, famille, taille du cache, etc ...).
/proc/meminfo	Concerne les informations sur la mémoire physique (Ram), l'espace du “Swap”, etc ... .
/proc/mounts	Concerne les informations sur les systèmes de fichiers déjà montés.
/proc/devices	Concerne les informations sur les périphériques disponibles.
/proc/modules	Concerne les informations sur les modules activés dans le noyau.
/proc/cmdline	Concerne les informations sur les paramètres donnés au noyau lors de la mise en route.

D'autres fichiers sont bien évidemment présents dans ce répertoire. Certains sont très sensibles comme **kcore**, à éviter d'éditer ou de modifier.

Information sur les processus en cours :

Nous pouvons nous aider du système de fichiers **/proc** pour extraire des données en cours d'exécution. Le répertoire **/proc** contient des sous-répertoires numérotés correspondant au numéro d'identification de processus (PID). Ainsi, pour chaque processus en cours d'exécution, nous avons un sous-répertoire qui porte comme nom le PID de chacun. Ces sous-répertoires contiennent des fichiers qui fournissent des détails important de l'état et l'environnement d'un processus.

Recherchons un processus en cours d'exécution :

```
ps -aef | grep mozilla
```

Résultat :

```
xxxxxxx       12807  11544   5 Apr30 ?             00:22:07
/usr/lib/mozilla-1.6/mozilla-bin
```

L'exécution de la commande nous montre un processus de Mozilla en cours avec le PID numéro 12807 . Il devrait donc exister un sous-répertoire dans **/proc** portant ce numéro.

Vérifions :

```
ls -l /proc/1280
```

Résultat :

```
total 0
-r-------- 1 standby standby 0 mai 1 02:52 auxv
-r--r--r-- 1 standby standby 0 mai 1 01:05 cmdline
lrwxrwxrwx 1 standby standby 0 mai 1 02:52 cwd -> /home/standby/
-r-------- 1 standby standby 0 mai 1 02:52 environ
lrwxrwxrwx 1 standby standby 0 mai 1 01:05 exe ->
/usr/lib/mozilla-1.6/mozilla-bin*
dr-x------ 2 standby standby 0 mai 1 02:52 fd/
-r--r--r-- 1 standby standby 0 mai 1 02:52 maps
-rw------- 1 standby standby 0 mai 1 02:52 mem
-r--r--r-- 1 standby standby 0 mai 1 02:52 mounts
lrwxrwxrwx 1 standby standby 0 mai 1 02:52 root -> //
-r--r--r-- 1 standby standby 0 mai 1 01:05 stat
-r--r--r-- 1 standby standby 0 mai 1 02:52 statm
-r--r--r-- 1 standby standby 0 mai 1 02:40 status
dr-xr-xr-x 3 standby standby 0 mai 1 02:52 task/
-r--r--r-- 1 standby standby 0 mai 1 02:52 wchan
```

Interagir avec le noyau directement grâce à /proc :

La plupart des fichiers dans **/proc** ne sont qu'en lecture seule, excepté le répertoire **/proc/sys** qui lui est en lecture/écriture ce qui implique que l'on peut modifier l'état du noyau instantanément en modifiant ces fichiers.

ATTENTION : Vous agissez directement sur le noyau, il faut donc être très délicat !

/proc/sys/kernel contient les informations sur le fonctionnement général du noyau. Ce répertoire héberge le nom de domaine et le nom d'hôte pour les ordinateurs du réseau, cette configuration se trouve dans les fichiers **domainname** et **hostname**. Ceux-ci peuvent être configurés pour modifier ces noms.

Testons et commençons par voir le hostname affecté à notre machine :

- hostname
- POSTE1 (Sans nom de domaine)
- POSTE1.domainname.com (Avec nom de domaine)

Vérifions si nous trouvons la même chose dans ce fameux répertoire. Voyons le nom de domaine :

```
$ cat /proc/sys/kernel/domainname
                      ( Sans nom de domaine )
domainname.com ( Avec nom de domaine )
```

Vérifions cette fois le hostname de la machine :

```
$ cat /proc/sys/kernel/hostname
POSTE1  ( Sans nom de domaine )
POSTE1  ( Avec nom de domaine )
```

Nous pouvons donc constater que les commande interagissent avec ce répertoire, **/proc**, plus précisément dans ce cas, **/proc/sys/kernel/** avec les fichiers **hostname** et **domainname**.

Tentons d'interagir cette fois-ci avec le noyau. Pour cela nous allons modifier le **hostname** par ligne de commande puis vérifier ce que la machine nous répondra lorsque nous lui demanderons quel est notre **hostname**.

```
$ echo "nouveau-hostname" > /proc/sys/kernel/hostname
$ hostname
nouveau-hostname    ( Sans nom de domaine )
nouveau-hostname.domainname.com    ( Avec nom de domaine )
```

Nous voyons bien que le noyau vient d'être modifieé. Bien des fichiers peuvent être modifiés mais attention à ne pas faire tout et n'importe quoi.

Un exemple assez sympathique, modifier en temps réel le noyau de façon à ne plus permettre de réponse de votre machine sur le réseau. Pour être plus clair, nous allons lui demander de ne plus répondre à la commande **PING**, cela va nous permettre de nous cacher du réseau.

Nouveau-hostname :

```
$ echo 1 > /proc/sys/net/ipv4/icmp_echo_ignore_all
$ ping POSTE1
```

Pas de réponse, si ce n'est qu'il vous montre que des packets on été envoyés mais sans retour.

Pour remettre par défaut cette valeur aussi rapidement, il suffit de remplacer la valeur 1 par 0 :

```
$ echo 0 > /proc/sys/net/ipv4/icmp_echo_ignore_all
$ ping POSTE1
64 bytes from POSTE1.domainname.com (192.168.x.x): icmp_seq=1 ttl=64
time=0.028 ms
```

La machine répond de nouveau !

En conclusion, le système de fichiers **/proc** assure une interface aux éléments internes du système Linux. Il assiste l'administrateur afin de connaître des états et de configurer des périphériques et des processus sur une machine. La compréhension et l'utilisation de ce système de fichiers sont primordiales pour bien se servir du système d'exploitation Linux. Donc ne faîtes pas de bêtises 😉 .

From:
https://chanterie37.fr/fablab37110/ - **Castel'Lab le Fablab MJC de Château-Renault**

Permanent link:
https://chanterie37.fr/fablab37110/doku.php?id=start:parcours_linux:proc

Last update: **2023/01/27 16:08**