

# /proc

- Objet : /proc
- Niveau requis :  
débutant, avisé
- Commentaires : *Le repertoire /proc.*
- Débutant, à savoir : [Utiliser GNU/Linux en ligne de commande, tout commence là !](#) 😊
- Suivi :
  - Création par  smolski le 13/06/2010
  - Testé par <...> le <...> 
- Commentaires sur le forum : [Lien vers le forum concernant ce tuto](#)<sup>1)</sup>

## Nota :

Contributeurs, les  sont là pour vous aider, supprimez-les une fois le problème corrigé ou le champ rempli !

## Introduction

C'est le répertoire contenant toutes les informations sur le fonctionnement du processeur *en direct live* !

Ces informations créent des fichiers virtuels qui vont disparaître au prochain arrêt du pc.

Le répertoire /proc ne représente pas de vrais fichiers, mais est simplement une interface entre le noyau et l'utilisateur.

Il permet d'obtenir facilement des informations sur le paramétrage du noyau et permet également de le changer.

Cet article va vous faire découvrir les subtilités de ce système de fichiers et va également tenter de remédier au manque de documentation en français sur le sujet.

Le répertoire **/proc** étant relativement peuplé, nous allons examiner ici les divers fichiers et répertoires qui l'habitent.

*Il est vivement conseillé au lecteur d'examiner les fichiers correspondants sur son système en cours de lecture.* 😊

Ceci étant dit, entrons maintenant dans le vif du sujet.

Le répertoire /proc contient un grand nombre de fichiers et de sous-répertoires.

Examinons tout d'abord les fichiers contenus dans le premier niveau de l'arborescence.

## Arborescence

Ces fichiers décrivent la configuration matérielle et l'état du système.

- apm : Informations sur l'Advanced Power Management.
- bus : Informations sur les bus.
- cmdline : Ligne de commande utilisée pour lancer le noyau (c'est-à-dire les paramètres passés

par l'intermédiaire de LILO).

- `cpuinfo` : Informations sur le(s) processeur(s).
- `devices` : Périphériques (bloc et caractères) disponibles.
- `dma` : Liste des canaux DMA utilisés.
- `filesystems` : Liste des systèmes de fichiers reconnus par le noyau.
- `interrupts` : Liste des interruptions utilisées avec les périphériques correspondants.
- `ioports` : Liste des ports d'entrée/sortie utilisés avec les périphériques correspondants.
- `kcore` : Image virtuelle de la mémoire; utilisée principalement à des fins de débogage.
- `kmsg` : Messages du noyau.
- `ksyms` : Table des symboles du noyau; utilisée pour le débogage et par l'utilitaire `insmod` pour changer à la volée les adresses dans le code des modules lors de leur installation.
- `loadavg` : Charge moyenne de la machine. Comparez les sorties des commandes `cat /proc/loadavg` et `uptime`.
- `locks` : Liste des verrous actifs (voir la page de manuel de la fonction `flock()` pour plus de détails sur le sujet).
- `meminfo` : Ce fichier donne des informations sur l'utilisation globale de la mémoire. Ces informations sont notamment reprises par la commande `free`. Comparez les sorties des commandes

```
cat /proc/meminfo
```

et

```
free
```

- `misc` : Informations diverses.
- `modules` : Liste des modules noyau en cours d'utilisation. Cette liste est notamment utilisée par [la commande `lsmod`](#).
- `mounts` : Liste des systèmes de fichiers en cours d'utilisation. Encore une fois, cette liste est très proche de la sortie [de la commande `mount`](#) lorsqu'elle est appelée sans paramètre. Bien que la commande `mount` utilise en fait le fichier `/etc/mtab`, il est possible de créer un lien symbolique<sup>2)</sup> de `/etc/mtab` vers `/proc/mounts` pour les systèmes qui tournent avec le système de fichiers racine monté en lecture seule.
- `partitions` : Liste des partitions connues par le système.
- `rtc` : Informations sur l'horloge temps réel (Real Time Clock).
- `slabinfo` : Informations sur le système d'allocation mémoire du noyau.
- `stat` : Statistiques globales cumulées depuis le dernier démarrage sur l'utilisation du système :
  - temps CPU total utilisé,
  - nombre d'interruptions,
  - nombre de défaut de pages,
  - nombre total de processus lancés,
  - etc.
- `swaps` : Liste des partitions disque utilisées par le système de mémoire virtuelle (swap).
- `uptime` : Temps en secondes depuis le démarrage du système.
- `version` : Chaîne de caractères identifiant le noyau en cours d'utilisation (date de compilation, compilateur utilisé, version du noyau, etc.).

**/proc** contient également un certain nombre de répertoires dont le nom est un nombre.

Il y en a un par processus en cours d'exécution sur le système et chacun d'entre eux contient des fichiers et répertoires qui donnent un certain nombre d'informations sur le processus en question.

Notons au passage qu'un lien nommé **self** pointe automatiquement vers le répertoire contenant les informations sur le processus courant; ainsi :

```
ls -al /proc/self
```

listera le contenu du répertoire contenant les informations sur le processus `ls` en cours.

```
ls -l /proc/2441
```

[retour de la commande](#)

```
total 0
-r--r--r--    1 root    root    0 sep 29 16:34 cmdline
lrwx-----    1 root    root    0 sep 29 16:34 cwd -> /root
-r-----    1 root    root    0 sep 29 16:34 environ
lrwx-----    1 root    root    0 sep 29 16:34 exe ->
/bin/bash
dr-x-----    2 root    root    0 sep 29 16:34 fd
pr--r--r--    1 root    root    0 sep 29 16:34 maps
-rw-----    1 root    root    0 sep 29 16:34 mem
lrwx-----    1 root    root    0 sep 29 16:34 root -> /
-r--r--r--    1 root    root    0 sep 29 16:34 stat
-r--r--r--    1 root    root    0 sep 29 16:34 statm
-r--r--r--    1 root    root    0 sep 29 16:34 status
```

Examinons les lignes une par une :

- `cmdline` : La ligne de commande qui a été utilisée pour lancer le processus.
- `cwd` : Lien vers le répertoire courant du processus.
- `environ` : Liste des variables d'environnement du processus.
- `exe` : Lien vers le fichier contenant le code exécutable du processus.
- `fd` : Répertoire contenant une entrée par fichiers et sockets ouverts par le processus.

Cette liste se présente sous la forme suivante :

```
ls -l /proc/2441/fd
```

```
total 0
lrwx-----    1 root    root    64 sep 29 16:46 0 -> /dev/pts/1
lrwx-----    1 root    root    64 sep 29 16:46 1 -> /dev/pts/1
lrwx-----    1 root    root    64 sep 29 16:46 2 -> /dev/pts/1
lrwx-----    1 root    root    64 sep 29 16:46 255 ->
/dev/pts/1
lrwx-----    1 root    root    64 sep 29 16:46 4 ->
socket:[177]
```

Les trois premiers descripteurs de fichiers (0, 1, 2) correspondent aux flux standards et seront présents pour quasiment tous les processus :

1. d'entrée (stdin),
2. de sortie (stdout)

### 3. et d'erreur (stderr).

- maps : Cartographie mémoire du processus.

```
08048000-080c0000 r-xp 00000000 03:08 29952      /bin/bash
[...]
40000000-40015000 r-xp 00000000 03:08 33989      /lib/ld-2.1.92.so
40015000-40017000 rw-p 00014000 03:08 33989      /lib/ld-2.1.92.so
40029000-4002c000 r-xp 00000000 03:08 33944      /lib/libtermcap.so.2.0.8
4002c000-4002d000 rw-p 00002000 03:08 33944      /lib/libtermcap.so.2.0.8
4002d000-4002f000 r-xp 00000000 03:08 33995      /lib/libdl-2.1.92.so
4002f000-40030000 rw-p 00001000 03:08 33995      /lib/libdl-2.1.92.so
40030000-40147000 r-xp 00000000 03:08 33991      /lib/libc-2.1.92.so
40147000-4014d000 rw-p 00116000 03:08 33991      /lib/libc-2.1.92.so
[...]
40186000-40187000 rw-p 00000000 00:00 0
bffffb000-c0000000 rwxp ffffc000 00:00 0
```

Examinons de plus près la signification de chaque ligne :

1. Les deux premiers nombres hexadécimaux correspondent à l'adresse virtuelle de la zone mémoire.
2. Le troisième champ correspond aux permissions d'accès de la zone :
  1. lecture (r),
  2. écriture (w) et
  3. exécution (x).

- mem : Mémoire utilisée par le processus.
- root : Lien vers le répertoire racine du processus. Cela permet notamment de déterminer si le processus a été lancé avec la commande chroot.
- stat : Etat du processus.
- status : Etat du processus dans un format plus lisible.
- statm Informations sur l'état de la mémoire utilisée.
- status : Ce fichier donne un grand nombre d'informations sur l'état actuel du processus. Ces informations sont notamment utilisées par des programmes comme [top](#) et [ps](#) pour déterminer l'état du processus.

```
cat /proc/2441/status
```

[retour de la commande](#)

```
Name:   bash
State:  S (sleeping)
Pid:    222
PPid:   220
Uid:    0      0      0      0
Gid:    0      0      0      0
Groups: 0
VmSize: 2360 kB
VmLck:  0 kB
```

```
VmRSS:      1392 kB
VmData:     260 kB
VmStk:      20 kB
VmExe:      480 kB
VmLib:     1416 kB
SigPnd: 0000000000000000
SigBlk: 0000000000010000
SigIgn: 8000000000384004
SigCgt: 000000004b813efb
CapInh: 0000000000000000
CapPrm: 00000000ffffefff
CapEff: 00000000ffffefff
```

Examinons une par une la signification de chaque ligne :

- Name : Le nom du processus, tel qu'il apparaîtra lorsqu'il sera listé par [la commande ps](#) (par exemple).
- State : Etat actuel du processus :
  - [S] en attente interruptible,
  - [D] en attente non-interruptible (généralement une requête d'entrée/sortie),
  - [R] en cours d'exécution,
  - [Z] Zombie,
  - [T] Stoppé ou tracé.
- Pid, PPid : Numéro du processus et de celui de son père.
- Uid, Gid : Numéro d'utilisateur réel et effectif du processus.
- Groups : Numéro du groupe de processus auquel appartient éventuellement le processus.

Les sept lignes suivantes concernent l'utilisation mémoire :

- VmSize : Quantité de mémoire totale utilisée.
- VmLck : Quantité de mémoire virtuelle verrouillée en RAM. Cela est obtenu en appelant la fonction `mlock()` et est souvent utilisé par les processus qui veulent garantir un très bon temps de réponse en évitant d'être swappé.  
Le daemon **NTP** est un des programmes utilisant cette technique.
- VmRSS : (Virtual Memory Resident Stack Size) Taille de la pile en mémoire.
- VmData : Taille du segment de données.
- VmExe : Taille utilisée par le code exécutable du processus.
- VmLib : Taille des bibliothèques dynamiques utilisées par le processus.

Les lignes restantes se réfèrent aux signaux et capacités du processus; nous ne décrivons ici que les caractéristiques principales.

- SigPnd : (Signals Pending) Signaux en attente qui doivent être délivrés au processus.
- SigIgn : (Signals Ignored) Signaux ignorés qui ne seront pas transmis au processus.

Examinons maintenant chacun des sous-répertoires plus en détail :

## /proc/ide

Ce répertoire contient des informations à propos de tous les périphériques IDE connus par le noyau. Il contient :

- un sous-répertoire par contrôleur IDE,
- un fichier nommé drivers et
- un lien par périphérique pointant vers les informations spécifiques à ce périphérique.

### drivers

```
cat /proc/ide/drivers
```

[retour de la commande](#)

```
ide-cdrom version 4.58  
ide-disk version 1.10
```

Les informations plus détaillées sont disponibles dans les sous-répertoires spécifiques à chaque contrôleur.

Ceux-ci sont nommés ide0, ide1, etc.

Parmi les fichiers contenus dans ces sous-répertoires, on note en particulier :

```
capacity : taille du disque en secteurs de 512 octets.
```

### capacity

```
cat /proc/ide/ide0/hda/capacity
```

[retour de la commande](#)

```
12692736
```

Et :

```
bc -ql
```

[retour de la commande](#)

```
12692736*512/1024  
6346368
```

On a donc affaire à un disque de 6.3 Go.

## geometry

Géométries physique et logique des disques.

```
cat /proc/ide/ide0/hda/geometry
```

[retour de la commande](#)

```
physical    12592/16/63
logical     12592/16/63
```

## media

Contient *disk* ou *cdrom* selon le cas.

## model

Contient le modèle et la marque du périphérique.

```
cat /proc/ide/ide0/hda/model
```

[retour de la commande](#)

```
IBM-DHEA-36481
```

## settings

Informations complètes sur la configuration du périphérique

## /proc/net

Le sous-répertoire **net** contient des informations sur la configuration réseau et son usage. En particulier, le contenu du fichier *dev* pourra être utilisé pour déterminer quels sont les périphériques réseaux disponibles sur la machine ainsi que le nombre de paquets reçus et envoyés par chaque interface.

Seuls les fichiers les plus importants sont mentionnés dans la liste ci-dessous :

- arp : Contenu de la table ARP du noyau.

- dev : Interfaces réseaux avec leurs statistiques d'utilisation.
- ip\_fwchains : Chaînage des règles de pare-feu.
- ip\_fwnames : Nom des chaînes de la configuration pare-feu.
- ip\_masq : Répertoire contenant les tables de "masquerading".
- netstat : Statistiques d'utilisation du réseau.
- route : Table de routage du noyau.
- rpc : Répertoire contenant les informations sur la configuration RPC (Remote Procedure Calls).
- rt\_cache : Cache des informations de routage.
- snmp : Données SNMP.
- sockstat : Statistiques sur le nombre de sockets en cours d'utilisation.
- tcp : Liste des sockets TCP en cours d'utilisation.
- udp : Liste des sockets UDP en cours d'utilisation.
- igmp : Liste des adresses multicast auxquelles appartient la machine.

**N.B. :**

Dans le cas où le support pour IPv6 est compilé dans le noyau, un certain nombre de fichiers supplémentaires avec un nom terminé par le chiffre 6 apparaîtront dans le répertoire. Ils remplissent le même rôle, mais décrivent la configuration IPv6 et non plus IPv4 comme ceux que nous venons de décrire.

## /proc/scsi

Si vous avez une carte SCSI, le répertoire `/proc/scsi` contiendra un répertoire dont le nom correspond au type de carte. Celui-ci contient un sous-répertoire par périphérique de ce type présent sur votre machine.

```
ls -l /proc/scsi/
```

[retour de la commande](#)

```
dr-xr-xr-x  2 root  root          0 Apr 28 17:47 ncr53c7xx
-rw-r-r-    1 root  root          0 Apr 28 17:47 scsi
```

## /proc/scsi/scsi

Le fichier `/proc/scsi/scsi` contient quant à lui la liste complète de tous les périphériques SCSI pris en charge par le système.

```
cat /proc/scsi/scsi
```

[retour de la commande](#)

```
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
```

```
Vendor: RICOH      Model: MP6200S      Rev: 1.20
Type:   CD-ROM    ANSI SCSI
revision: 02
```

Ici, le seul périphérique SCSI reconnu est un lecteur de CD-Rom.

## /proc/parports

Ce répertoire contient les informations sur les ports parallèles du système.

Il contient un sous-répertoire, dont le nom est le numéro du port (la numérotation débutant à zéro). Chacun de ces répertoire contient quatre fichiers :

1. `autoprobe` : Toutes les informations IEEE-1284 obtenues sur le port.
2. `devices` : Liste des pilotes de périphériques utilisant ce port. Un signe + apparaîtra en face des pilotes qui utilisent actuellement ce port.
3. `hardware` : Adresse, IRQ et canal DMA utilisés par ce port.
4. `irq` : IRQ utilisée par ce port. Une nouvelle valeur peut être écrite dans ce fichier pour reconfigurer dynamiquement le port.

## /proc/tty

Les informations à propos des terminaux virtuels disponibles et en cours d'utilisation sont disponibles dans ce répertoire et ses sous-répertoires.

- `drivers` : Liste des pilotes et leur utilisation.
- `ldiscs` : Disciplines de lignes déclarées.
- `driver/serial` : Etat des lignes séries et statistiques d'utilisation.

```
cat /proc/tty/driver/serial
```

```
serinfo:1.0 driver:5.02 revision:2000-08-09
0: uart:16550A port:3F8 irq:4 baud:9600 tx:0 rx:0
1: uart:16550A port:2F8 irq:3 baud:9600 tx:0 rx:4453 RTS|DTR
2: uart:unknown port:3E8 irq:4
[...]
```

- La première ligne correspond à un port série non utilisé;
- la seconde à un port série utilisé par une souris;
- les lignes suivantes aux ports non présents sur la machine.

## /proc/sys

Modification dynamique des paramètres du noyau

Le répertoire `/proc/sys` est une source d'information très importante, mais également un moyen de modifier les paramètres du noyau en cours d'utilisation.



Soyez cependant vigilant car une mauvaise manipulation peut conduire à une réduction des performances du système, voire à un crash.

Les fichiers contenus dans `/proc/sys` et ses sous-répertoires peuvent être utilisés pour paramétrer finement un grand nombre de choses.

Notons également que l'utilisation de certains paramètres change subtilement d'une version de noyau à une autre.

C'est le **noyau 2.2.13** qui a servi de base à cet article; ceux d'entre vous qui utilisent déjà un noyau 2.3.x ou encore un noyau 2.0.y pourront donc également noter l'apparition ou la disparition de certains fichiers.



En cas de doute, il n'y a en général malheureusement pas beaucoup de documentation sur le sujet; d'autre part, le système de fichiers proc est une cible très mouvante. La seule ressource reste bien souvent la lecture du code source du noyau, ce qui demande une bonne dose de courage.

## `/proc/sys/fs`

### `file-max`

Le fichier **file-max** détermine le nombre maximum de fichiers ouvrables simultanément par le système. Dans le cas où cette valeur soit trop faible (sur un serveur assez chargé par exemple), il est possible de la changer dynamiquement en écrivant une nouvelle valeur dans ce fichier.

```
cat /proc/sys/fs/file-max
```

[retour de la commande](#)

```
4096
```

```
echo 6000 > /proc/sys/fs/file-max
```

```
cat /proc/sys/fs/file-max
```

[retour de la commande](#)

```
6000
```

## file-nr

Le fichier **file-nr** quant à lui contient trois nombres qui correspondent respectivement :

1. au nombre de descripteurs de fichiers alloués,
2. au nombre de descripteurs de fichiers actuellement utilisés et
3. au nombre maximum de descripteurs utilisables par le système; *c'est cette dernière valeur qui peut être changée grâce au fichier **file-max**.*

Il est important de noter que ces limites sont globales au système.

Le nombre maximum de fichiers ouverts simultanément par un seul processus est limité à 1024, et il est sensiblement plus difficile de changer cette limitation.

Il faut pour cela modifier la valeur des macros NR\_OPEN dans les fichiers fs.h et limits.h, puis augmenter la valeur de la macro NR\_FILE dans fs.h (ces fichiers se trouvant dans les sources du noyau), puis recompiler celui-ci.

## inode-max

Tout comme pour les descripteurs de fichiers, le noyau alloue dynamiquement l'espace nécessaire pour stocker les i-nodes, mais ne peut pas le désallouer.

La valeur contenue dans le fichier **inode-max** est le nombre maximum de descripteurs de i-nodes. Cette valeur doit être *trois à quatre fois supérieure* à la valeur de **file-max** dans la mesure où stdin, stdout et les sockets utilisent chacun un descripteur de i-node.



Si le système se plaint d'un manque de i-nodes, c'est cette valeur que vous devrez augmenter.

Seules les trois premières valeurs de inode-state sont utilisées; elles correspondent à :

1. nr\_inodes,
2. nr\_free\_inodes et
3. preshrink.

## nr\_inodes

Nombre de i-nodes actuellement allouées.

Le nombre peut être légèrement supérieur à inode-max dans la mesure où les i-nodes sont allouées par page mémoire complète.

## nr\_free\_inodes

Nombre de i-nodes non utilisées.

## pre shrink

Cette valeur est non nulle lorsque `nr_inodes` est supérieur à `inode-max`; le système essaiera de recycler les i-nodes non utilisées au lieu d'en allouer de nouvelles.

## inode-nr

Le fichier **inode-nr** contient simplement les deux premières valeurs de `inode-state`.

## binfmt\_misc

En dehors des fichiers précédemment cités, le répertoire **sys/fs** contient également le sous-répertoire **binfmt\_misc** qui permet de contrôler les types d'exécutables reconnus par le noyau.

**binfmt\_misc** permet de déclarer de nouveaux types de fichiers exécutables au noyau sans avoir à le recompiler. Le système fonctionne en repérant les exécutables soit par nombre magique ("magic number" en anglais) ou soit par extension du nom de fichier.

Une fois que le noyau a reconnu un type d'exécutable, l'interpréteur associé sera invoqué avec le nom du fichier exécutable passé en paramètre.

Le répertoire `binfmt_misc` contient les fichiers `register` et `status`, plus un fichier par format déclaré.

La syntaxe standard pour déclarer un nouveau type d'exécutable est la suivante :

```
echo :nom:type:offset:magic:mask:interpréteur >
/proc/sys/fs/binfmt_misc/register
```

## Exemple

Un exemple valant mieux qu'un long discours, voyons comment utiliser ce système pour permettre au noyau d'exécuter directement des applications Windows (grâce à l'émulateur Wine) et des scripts Python.

```
echo ':DOSWin:M::MZ::/usr/bin/wine:' > /proc/sys/fs/binfmt_misc/register
```

```
echo ':Python:E::py::/usr/bin/python:' > /proc/sys/fs/binfmt_misc/register
```

La lettre ``M'` ou ``E'` indique si le repère pour déterminer le type du fichier sera son magic number ou l'extension du nom de fichier (les deux premiers caractères d'un exécutable Windows sont toujours MZ).

```
ls -al /proc/sys/fs/binfmt_misc/
```

[retour de la commande](#)

```
dr-xr-xr-x    2 root    root    0 May  1 16:55 .
dr-xr-xr-x    3 root    root    0 May  1 16:55 ..
-rw-r--r--    1 root    root    0 May  1 16:55 DOSWin
-rw-r--r--    1 root    root    0 May  1 16:55 Python
-w-----    1 root    root    0 May  1 16:55 register
-rw-r--r--    1 root    root    0 May  1 16:55 status
```

```
cat /proc/sys/fs/binfmt_misc/Python
```

[retour de la commande](#)

```
enabled
interpreter /usr/bin/python
extension .py
```

Essayons maintenant d'exécuter directement un script Python.

```
cat > test.py
print 'hello'
```

```
^D
```

```
chmod +x test.py
```

```
./test.py
```

```
hello
Ca marche !
```

## statuts

Le fichier **status** quant à lui permet de déterminer si la fonctionnalité binfmt\_misc est activée ou non.

```
cat /proc/sys/fs/binfmt_misc/status
```

[retour de la commande](#)

```
enabled
```

Vous pouvez de même changer cette valeur en écrivant 0 (désactiver) ou 1 (activer) dans ce fichier.

Chaque format enregistré crée un fichier dans le répertoire binfmt\_misc.

Ceux-ci ont la même fonction que le fichier status, à la différence qu'ils ne s'appliquent pas à la fonctionnalité, mais seulement au format en question.

## /proc/sys/kernel

Ce répertoire contient les paramètres qui régissent le fonctionnement général du noyau.

### acct

Ce fichier (présent si et seulement si l'accounting BSD est activé) contient trois valeurs :

1. low,
2. high et
3. frequency.

Elles permettent d'éviter l'engorgement du système de fichiers par le fichier d'historique. Si le système de fichiers où se situe l'historique contient moins de `low` pourcents de place libre, l'accounting sera suspendu jusqu'à ce que la valeur repasse au-dessus de la barre des `high` pourcents.

### frequency

Le paramètre `frequency` correspond à l'intervalle de temps (en secondes) entre chaque vérification de l'espace libre disponible. Les valeurs par défaut sont 2, 4 et 30.

Si la valeur contenue dans ce fichier est 0, la pression simultanée des touches `Ctrl+Alt+Suppr` est transmise au processus *init*.

Si cette valeur est supérieure à zéro, le noyau lancera un *reboot* immédiatement, sans démonter les systèmes de fichiers.

### N.B.

Lorsqu'un programme (comme *dosemu*) utilise le clavier en mode "raw", alors c'est lui qui intercepte la combinaison `Ctrl+Alt+Suppr` et qui décide ce qui doit être fait.

### domainname et hostname

Ces fichiers contrôlent les domaines NIS et nom d'hôte de votre machine.

- *osrelease*,
- *ostype*,
- *version*

Comme leur nom l'indique, ces fichiers contiennent respectivement le numéro de version, le nom du système et la version exacte du noyau :

```
cat /proc/sys/kernel/osrelease
```

[retour de la commande](#)

```
2.2.13
```

et :

```
cat /proc/sys/kernel/ostype
```

[retour de la commande](#)

```
Linux
```

et enfin :

```
cat /proc/sys/kernel/version
```

[retour de la commande](#)

```
#6 Sun Apr 16 23:20:59 CEST 2000
```

Si les deux premiers fichiers n'appellent pas de commentaires, attardons-nous sur le dernier :

- **#6** signifie que le noyau a été recompilé six fois à partir du même code source.
- La date qui suit est tout simplement la date à laquelle a été compilé le noyau.

## panic

La valeur contenue dans ce fichier représente le nombre de secondes que le noyau attend pour rebooter après un "kernel panic".

Si vous utilisez un watchdog logiciel, la valeur recommandée est 60.

La valeur par défaut est 0, ce qui désactive le reboot.

## printk

Les quatre valeurs contenues dans ce fichier correspondent respectivement à :

1. (console\_loglevel) : Les messages avec une priorité supérieure à cette valeur seront affichés sur la console.
2. (default\_message\_level) : Les messages sans priorité explicite seront affichés avec cette priorité.
3. (minimum\_console\_loglevel) : Valeur minimum que peut avoir console\_loglevel.
4. (default\_console\_loglevel) : Valeur par défaut de console\_loglevel.

## sg-big-buff

Ce fichier montre la taille du tampon du pilote SCSI.

Pour le moment, la seule manière de changer sa valeur consiste à changer la valeur de `SG_BIG_BUFF` dans le fichier :

```
include/scsi/sg.h
```

puis à recompiler le noyau.



Si vous utilisez un scanner, il peut être intéressant d'augmenter cette valeur. Consultez la documentation de SANE qui vous en dira plus sur le sujet.

## modprobe

L'emplacement où l'exécutable **modprobe** est situé.

Le noyau utilise ce paramètre pour charger les modules à la demande.

## /proc/sys/vm

Les fichiers de ce répertoire permettent de paramétrer le fonctionnement de la gestion de la mémoire virtuelle du noyau. Certains de ces fichiers, comme `bdflush`, ont également une influence sur l'utilisation du disque.

**bdflush** est le système qui met à jour les fichiers sur les disques en fonction des modifications effectuées sur leur copie dans le cache disque du noyau.

En effet, lorsqu'un fichier est ouvert en lecture ou en écriture, Linux en charge une copie en mémoire ce qui permet d'accélérer grandement les opérations d'écriture, car en général les mêmes opérations sont répétées sur un même fichier (par exemple : le fichier contenant une page très demandée d'un serveur Web).

La modification des paramètres qui vont être décrits n'a en général que peu d'intérêt sur une machine personnelle, mais peut considérablement améliorer les performances sur un serveur à condition d'utiliser des valeurs judicieuses.



A l'inverse, de mauvais paramètres peuvent largement dégrader les performances.

A vous d'expérimenter et de trouver les valeurs qui correspondent le mieux à votre machine.

Dans le doute, vous pouvez conserver les valeurs par défaut qui donnent généralement d'assez bons résultats.

## **bdflush**

Ce fichier contrôle le fonctionnement de la thread kernel bdflush. Il contient actuellement neuf nombres dont seuls six sont actuellement utilisés.

1. nfract : Pourcentage du tampon cache modifié avant d'activer bdflush.
2. ndirty : Nombre maximum de blocs modifiés à être écrits par cycle d'activation.
3. nrefill : Nombre de tampons libres que l'on doit essayer d'obtenir à chaque appel.
4. nref\_dirt : Réactivité de bdflush.
5. dummy : Non utilisé.
6. age\_buffer : Temps à attendre avant de synchroniser les tampons contenant un super-bloc de système de fichiers.

Les deux derniers champs ne sont pas utilisés.

## **buffermem**

Les trois valeurs que contient ce fichier contrôlent la quantité de mémoire qui doit être utilisée. Ce sont des pourcentages par rapport à la quantité totale de mémoire.

1. min\_percent : Pourcentage minimum de la mémoire qui sera utilisé pour le cache.
2. borrow\_percent : Lorsque le système tend à manquer de mémoire et que le cache disque en utilise beaucoup, alors le système de gestion de la mémoire réduira d'autant plus la taille du cache disque pour compenser.
3. max\_percent : Pourcentage maximum de la mémoire qui sera utilisé pour le cache.

## **freepages**

Ce fichier contient trois valeurs.

1. min : Lorsque le nombre de pages libres descend en dessous de ce nombre, le noyau alloue plus de mémoire.
2. low : Lorsque le nombre de pages libres descend en dessous de ce nombre, le noyau commence à swapper sur disque de manière plus agressive pour libérer de la place.
3. high : Le noyau essaye de conserver ce nombre de pages libres. Si le nombre tombe en dessous de cette quantité, le noyau commencera à swapper "en douceur" dans l'espoir de ne pas atteindre la limite "low".

## **kswapd**

**kswapd** est le nom du programme de swap, c'est-à-dire celui qui copie certaines parties de la mémoire virtuelle sur disque lorsque la mémoire physique est pleine.

Etant donné que chaque système a des besoins différents, vous pourrez avoir à changer les paramètres par défaut dans le cas d'un serveur chargé.

Le fichier kswapd contient trois valeurs :

1. tries\_base : Le nombre maximum de pages qui seront libérées à chaque appel est déduit de ce

nombre. On peut donc augmenter la vitesse du swapping en augmentant ce paramètre aux dépens d'une activité disque plus importante.

2. `tries_min` : Représente le nombre minimum de pages que `kswapd` essayera de libérer chaque fois qu'il est appelé.
3. `swap_cluster` : Ce paramètre a une grande influence sur les performances du système; il correspond au nombre de pages que `kswapd` écrira simultanément sur le disque. Il ne doit pas être trop petit pour ne pas faire appel au disque dur trop souvent, mais ne doit pas être trop élevé sous peine d'engorger la file d'attente du pilote de disque.

## overcommit\_memory

Ce fichier contient une seule valeur qui détermine la conduite à utiliser pour l'allocation de la mémoire. Si la valeur est positive, alors le système considère qu'il y a toujours assez de mémoire disponible.

Cette fonctionnalité est utile dans la mesure où les programmes réservent (grâce à la fonction `malloc`) de grandes quantités de mémoire qu'ils n'utilisent pas toujours.

Si vous laissez cette valeur à zéro, cela risque de conduire à l'échec d'un `malloc` de grande quantité de mémoire alors que le programme aurait eu assez de mémoire pour fonctionner.

Cependant, si la valeur est changée pour 1, vous pouvez provoquer un épuisement de la mémoire du système.



Sur les gros serveurs, il est fortement conseillé de laisser cette valeur à zéro à moins que vous ne connaissiez très précisément le comportement des programmes utilisés.

## /proc/sys/dev

Ce répertoire contient les paramètres spécifiques aux périphériques.

Actuellement, seuls les lecteurs de CD-Rom sont supportés, et les informations sont regroupées dans un seul fichier.

```
cat /proc/sys/dev/cdrom/info
```

[retour de la commande](#)

```
CD-ROM information, Id: cdrom.c 2.56 1999/09/09
```

```
drive name:          hdb
drive speed:         40
drive # of slots:    0
Can close tray:      1
Can open tray:       1
Can lock tray:       1
Can change speed:    1
Can select disk:     0
```

```
Can read multission: 1
Can read MCN: 1
Reports media changed: 1
Can play audio: 1
```

Chaque ligne étant suffisamment explicite, nous ne détaillerons pas plus.

## **/proc/sys/sunrpc**

Ce répertoire contient quatre fichiers qui activent ou désactivent le débogage pour les principales fonctions RPC :

1. NFS client,
2. NFS serveur,
3. RPC et
4. NLM.

La valeur par défaut est zéro (débogage désactivé) et peut être changée pour 1 (activation du débogage).

## **/proc/sys/net**

L'interface comprenant les fonctionnalités réseau du noyau est située dans ce répertoire.

Celui-ci peut contenir un grand nombre de répertoires.  
Voici les principaux d'entre eux.



Ils peuvent ou non être présents sur votre système en fonction de la configuration de votre noyau.

- core : Paramètre généraux.
- unix : Sockets unix.
- ethernet : Paramètres ethernet.
- ipv4 : TCP/IP
- ipv6 : TCP/IP version 6 (encore très peu utilisé).
- ipx : IPX
- x25 : Protocole X.25

Nous allons ici nous concentrer uniquement sur l'aspect TCP/IP, les autres types de réseaux étant beaucoup moins courants.

Bien qu'il soit également possible de changer la majorité des paramètres, il est extrêmement déconseillé de le faire :

- cela n'apportera en général pas grand chose et

- vous risquerez surtout de détériorer les performances, voire de rendre le réseau non fonctionnel.

## **/proc/sys/net/core**

Les fichiers de ce répertoire correspondent aux caractéristiques communes à tous les types de réseaux, principalement les tailles des tampons.

- `rmem_default` : La taille par défaut (en octets) des tampons utilisés pour recevoir les messages.
- `rmem_max` : Taille maximum des tampons utilisés pour recevoir les messages.
- `wmem_default`, `wmem_max` : Equivalents des deux fichiers précédents, mais cette fois pour l'envoi des messages.
- `message_burst`, `message_cost` : Ces paramètres contrôlent les messages d'avertissement écrits dans les logs du noyau par le code réseau. Ils introduisent une limite qui permet d'éviter les attaques de types DOS (Denial Of Service) : une valeur de `message_cost` plus élevée entraînera la diminution du nombre de messages générés. Le paramètre `message_burst` quant à lui limite la fréquence d'écriture des messages; par défaut, une fois toutes les cinq secondes.
- `netdev_max_backlog` : Nombre maximum de paquets stockés dans la file d'attente lorsqu'ils sont reçus plus vite que le noyau ne peut les traiter.

## **/proc/sys/net/ipv4**

IP version 4 est le protocole réseau le plus utilisé actuellement.

Il devrait cependant être remplacé par IP version 6 d'ici quelques années.

Du fait de l'importance et de la relative complexité de ce protocole, sa configuration est étalée sur plusieurs sous-répertoires.

Commençons par les fichiers qui sont situés directement dans le répertoire `/proc/sys/net/ipv4`.

### **Paramètres ICMP**

#### **`icmp_echo_ignore_all`, `icmp_echo_ignore_broadcast`**

La valeur contenue dans ces fichiers contrôle si le noyau ignore (0) ou répond (1) à toutes les demandes ICMP ECHO (utilisées principalement par [la commande ping](#)) ou seulement celles destinées aux adresses de broadcast et de multicast.



Il est très important de noter que si vous décidez d'activer la réponse aux ICMP ECHO pour les adresses broadcast et multicast, votre réseau peut être utilisé comme relai pour certains types d'attaques DDOS (Distributed Denial Of Service).

#### **`icmp_destunreach_rate`, `icmp_echo_reply_rate`, `icmp_paramprob_rate`, `icmp_timeexceed_rate`**

Définit les limites pour envoyer les différents types de paquets ICMP.

- Une valeur nulle désactive les limites.
- Une valeur positive définit le nombre maximum de paquets par centième de seconde.

## Paramètres IP

Seuls les fichiers les plus importants sont mentionnés dans cette section.

- `ip_autoconfig` : Ce fichier contient la valeur 1 si le paramétrage IP a été obtenu d'un système distant par RARP, BOOTP ou DHCP; sinon cette valeur est nulle.
- `ip_default_ttl` : TTL (Time To Live; durée de vie) des paquets reçus par les interfaces IP. Cela correspond au nombre de machines par lesquelles un même paquets peut transiter.
- `ip_forward` : Active ou désactive le "forwarding" des paquets IP pour toutes les interfaces. Cette fonctionnalité devra être activée au démarrage du système sur les machines qui servent de passerelle. Notez que le changement de cette valeur réinitialise tous les autres paramètres réseau à leur valeur par défaut.
- `ip_local_port_range` : Intervalle de ports utilisés par TCP et UDP comme ports locaux. Ce fichier contient deux nombres :  
le premier est la borne inférieure de l'intervalle,  
le deuxième est (surprise !) la borne supérieure.  
L'intervalle par défaut est 1024-4999.  
Sur un système fortement chargé, il est conseillé d'utiliser l'intervalle 32768-61000.
- `ip_masq_debug` : Comme son nom l'indique : active le débogage de l'IP masquerading.

## Paramètres TCP

- `tcp_keepalive_probes` : Nombre de sondes TCP keepalive à envoyer avant de décider qu'une connexion est coupée.
- `tcp_keepalive_time` : Périodicité avec laquelle les messages TCP keepalive doivent être envoyés. Cette durée est exprimée en secondes et la valeur par défaut est de 2 heures.

## /proc/loadavg

Ce fichier donne deux informations :

1. il fournit un aperçu de la moyenne de charge ou de l'utilisation de l'unité centrale dans le temps ;
2. il fournit aussi des données supplémentaires utilisées par la commande `uptime` et d'autres commandes.

Voici un exemple de fichier `loadavg` :

```
0.99 0.93 0.87 1/202 4027
```

Les trois premières colonnes mesurent l'utilisation de l'unité centrale en fonction des dernières périodes de 1, 5 et 10 minutes.

La quatrième colonne indique le nombre de processus en cours d'exécution et le nombre total de processus.

La dernière colonne affiche le dernier ID de processus utilisé.

# Remerciements

## Source :

- <http://okki666.free.fr/docmaster/articles/linux070.htm>

## Références :

Fichier Documentation/filesystems/proc.txt dans les sources du noyau (qui a été d'une grande aide pour la rédaction de cet article qui s'en est largement inspiré). Le code source du système de fichiers /proc dans les sources du noyau se trouve dans le répertoire fs/proc.

*Merci à :*

- **Vincent Renardias** [vincent@renardias.com](mailto:vincent@renardias.com)

Pour son article parût dans : **Linux Magazine France** n° 18 - Juin 00

<sup>1)</sup>

N'hésitez pas à y faire part de vos remarques, succès, améliorations ou échecs !

<sup>2)</sup>

Voir : [la commande ln](#)

From:

<http://debian-facile.org/> - **Documentation - Wiki**

Permanent link:

<http://debian-facile.org/doc:systeme:fhs-proc>

Last update: **03/11/2016 15:28**

