Файловая система `/proc`

Файловая система `/proc` является виртуальной и в действительности она не существует на диске. Ядро создает ее в памяти компьютера. Система `/proc` предоставляет информацию о системе (изначально только о процессах - отсюда ее название). Некоторые наиболее важные файлы и каталоги рассмотрены ниже. Более подробную информацию о структуре и содержании файловой системы `/proc` можно найти в man-руководстве к `proc`.

Информация о процессах

Если вы выведете список содержимого каталога `/proc`, вы увидите много каталогов, именами которых являются номера. Эти каталоги содержат информацию о всех процессах в системе, запущенных в данный момент:

```
$ ls -d /proc/[0-9]*
/proc/1/    /proc/302/  /proc/451/  /proc/496/  /proc/556/  /proc/633/
/proc/127/  /proc/317/  /proc/452/  /proc/497/  /proc/557/  /proc/718/
/proc/2/    /proc/339/  /proc/453/  /proc/5/    /proc/558/  /proc/755/
/proc/250/  /proc/385/  /proc/454/  /proc/501/  /proc/559/  /proc/760/
/proc/260/  /proc/4/    /proc/455/  /proc/504/  /proc/565/  /proc/761/
/proc/275/  /proc/402/  /proc/463/  /proc/505/  /proc/569/  /proc/769/
/proc/290/  /proc/433/  /proc/487/  /proc/509/  /proc/594/  /proc/774/
/proc/3/    /proc/450/  /proc/491/  /proc/554/  /proc/595/
```

Посмотрите, какие процессы соответствуют вашему сеансу и зайдите в один из каталогов, имя котрого совпадает с pid одного из этих процессов.

```
$ cd /proc/127
$ ls -l
total 0-9
-r--r--r--    1 root     root            0 Dec 14 19:53 cmdline
lrwx------    1 root     root            0 Dec 14 19:53 cwd -> //
-r--------    1 root     root            0 Dec 14 19:53 environ
lrwx------    1 root     root            0 Dec 14 19:53 exe -> /usr/sbin/apmd*
dr-x------    2 root     root            0 Dec 14 19:53 fd/
pr--r--r--    1 root     root            0 Dec 14 19:53 maps|
-rw-------    1 root     root            0 Dec 14 19:53 mem
lrwx------    1 root     root            0 Dec 14 19:53 root -> //
-r--r--r--    1 root     root            0 Dec 14 19:53 stat
-r--r--r--    1 root     root            0 Dec 14 19:53 statm
-r--r--r--    1 root     root            0 Dec 14 19:53 status
$
```

Каждый из каталогов содержит одинаковые пункты: Вот краткое описание некоторых из них:

1. `cmdline`: этот (псевдо-) файл содержит полную командную строку, использованную для вызова процесса. Он не отформатирован: между программой и ее аргументами нет пробелов, а в конце строки нет разделителя строки. Чтобы просмотреть его, вы можете использовать: **perl -ple 's,\00, ,g' cmdline**.

2. `cwd`: эта символическая ссылка указывает на текущий рабочий каталог процесса (следует из имени).

3. `environ`: этот файл содержит все переменные окружения, определенные для этого процесса, в виде `ПЕРЕМЕННАЯ=значение`. Как и в `cmdline` вывод вообще не отформатирован: нет разделителей строк для отделения различных переменных, и в конце нет разделителя строки. Единственным решением для его просмотра будет: **perl -pl -e 's,\00,\n,g' environ**.

4. `exe`: эта символическая ссылка указывает на исполняемый файл, соответствующий запущенному процессу.

5. `fd`: этот подкаталог содержит список файловых дескрипторов, открытых в данный момент процессом. Смотрите ниже.

6. `maps`: когда вы выводите содержимое этого именованного канала (при помощи команды **cat**, например), вы можете увидеть части адресного пространства процесса, которые в текущий момент распределены для файла. Вот эти поля (слева направо): адресное пространство, связанное с этим распределением; разрешения, связанные с этим распределением; смещение от начала файла, где начинается распределение; старший и младший номера (в шестнадцатеричном виде) устройства, на котором находится распределенный файл; номер

inode файла; и, наконец, имя самого файла. Если устройство обозначено как 0 и отсутствует номер inode или имя файла - это анонимное распределение. Смотрите mmap(2).

7. `root`: эта символическая ссылка указывает на корневой каталог, используемый процессом. Обычно это будет `/`, однако рекомендуем вам просмотреть chroot(2).

8. `status`: этот файл содержит разнообразную информацию о процессе: имя исполняемого файла, его текущее состояние, его PID и PPID, его реальные и эффективные UID и GID, его использование памяти и другие данные. Обратите внимание, что файлы `stat` и `statm` теперь устарели. Содержавшаяся в них информация теперь хранится в `status`.

Если мы выведем список содержимого каталога `fd` для процесса 127, мы получим следующее:

```
$ ls -l fd
total 0
lrwx------    1 root     root           64 Dec 16 22:04 0 -> /dev/console
l-wx------    1 root     root           64 Dec 16 22:04 1 -> pipe:[128]
l-wx------    1 root     root           64 Dec 16 22:04 2 -> pipe:[129]
l-wx------    1 root     root           64 Dec 16 22:04 21 -> pipe:[130]
lrwx------    1 root     root           64 Dec 16 22:04 3 -> /dev/apm_bios
lr-x------    1 root     root           64 Dec 16 22:04 7 -> pipe:[130]
lrwx------    1 root     root           64 Dec 16 22:04 9 ->
/dev/console
$
```

На самом деле это список файловых дескрипторов, открытых процессом. Каждый открытый дескриптор представлен в виде символической ссылки, где имя - это номер дескриптора, который указывает на файл, открытый этим дескриптором. Обратите внимание на разрешения симлинков: это - единственное место, где они имеют смысл, поскольку они представляют собой разрешения, с которыми был открыт файл, соответствующий дескриптору.

Дополнительная информация, содержащаяся в файловой системе /proc

`/proc/cpuinfo`

Информация о процессоре, такая как тип процессора, его модель, производительность и др.

`/proc/devices`

Список драйверов устройств, встроенных в ядро.

`/proc/filesystems`

Файловые системы, встроенные в ядро.

`/proc/interrupts`

Задействованные в данный момент прерывания.

`/proc/ioports`

Задействованные в данный момент порты ввода/вывода.

`/proc/kcore`

Отображение физической памяти системы в данный момент. Размер этого файла точно такой же, как и у памяти компьютера, только он не занимает места в самой памяти, а генерируется на лету при доступе к нему программ. Однако при копировании этого файла куда-либо, он не займет места на диске.

`/proc/kmsg`

Сообщения, выдаваемые ядром. Они также перенаправляются в **syslog**.

`/proc/ksyms`

Таблица символов ядра.

`/proc/loadavg`

Ориентировочная загруженность системы. Этот файл содержит числа подобно:

```
0.13 0.14 0.05
```

Эти числа являются результатом команд uptime и подобных, показывающих среднее число процессов пытающихся запуститься в одно и то же врямя за последнюю минуту, последние пять минут и последние пятнадцать

/proc/meminfo

Информация об использовании памяти, как физической так и swap-области. Файл содержит обзор выходной информации программы free. Содержание его имеет следующий вид:

```
        total:    used:      free:     shared:     buffers:
Mem:  7528448   7344128    184320    2637824     1949696
Swap: 8024064   1474560    6549504
```

Помните что данные числа представлены в байтах!

/proc/modules

Список модулей ядра, загруженных в данный момент.

/proc/net

Информация о сетевых протоколах.

/proc/self

Символическая ссылка к каталогу процесса, пытающегося получить информацию из /proc. При попытке двух различных процессов получить какую-либо информацию в /proc, они получают ссылки на различные каталоги. Это облегчает доступ программ к собственному каталогу процесса.

/proc/stat

Различная статистическая информация о работе системы. Файл stat отображает статистику данной системы в формате ASCII. Пример:

```
cpu   5470 0 3764 193792
disk  0 0 0 0
page  11584 937
swap  255 618
intr  239978
ctxt  20932
btime 767808289
```

Значения строк:

cpu Четыре числа сообщают о количестве тиков за время работы системы в пользовательском режиме, в пользовательском режиме с низким приоритетом, в системном режиме, и с идеальной задачей. Последнее число является стократным увеличением второго значения в файле uptime.

disk Четыре компоненты dk_drive в структуре kernel_stat в данный момент незаняты.

page Количество страниц введенных и исключенных системой.

swap Количество своп-страниц введенных и исключенных системой.

intr Количество прерываний установленных при загрузке системы.

ctxt Номер подтекста выключающий систему.

btime Время в секундах отсчитываемое сначала суток.

/proc/uptime

Время, в течение которого система находится в рабочем состоянии. Файл содержит время работы системы в целом и идеализированное время затрачивоемое системой на один процесс. Оба числа представлены в виде десятичных дробей с точностью до сотых секунды. Точность до двух цифр после запятой не гарантируется на всех архитектурах, однако на всех подпрограммах Linux даются достаточно точно используя удобные 100-Гц часы. Этот файл выглядит следующим образом: 604.33 205.45 В этом случае система функционирует 604.33 секунды, а время затрачиваемое на идеальный прцесс равно 204.45 секунд.

/proc/version

Версия ядра.

Хотя многие файлы имеют обычный текстовый формат, некоторые из них имеют собственный. Существует много программ, которые не только преобразуют такие файлы в формат, доступный для чтения, но и предоставляют некоторые функции. Например, программа **free** считывает файл /proc/meminfo и преобразует значения, указанные в байтах, в килобайты (а также предоставляет некоторую дополнительную информацию).