

ЛАБОРАТОРНАЯ РАБОТА №7 КОНТРОЛЬ ИСПОЛЬЗОВАНИЯ РЕСУРСОВ ОС LINUX

Цель работы – практическое знакомство с командами, используемыми для контроля использования ресурсов и виртуальной файловой системой /proc

1 КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

1.1 Команды для контроля системных ресурсов

1.1.1 Вывод информации о процессах, выполняющихся в системе

Для вывода списка всех выполняющихся на компьютере в текущий момент процессах используется команда

ps aux

Значения используемых опций: а - all – процессы всех пользователей; u – ориентированная на пользователей (отображение информации о владельце); x – процессы, не контролируемые ttys.

Полезные ключи - -e – вывод сведений обо всех процессах и о – пользовательский вывод, например

ps -eo pid, ppid, pri, stat, pgid, nice, comm.

В столбце STAT содержится информация о состоянии процесса. Наиболее важные состояния: S-спящий; R –выполняющийся; T- остановленный; Z – зомби.

Буква Z указывает, что процесс завис и его нельзя завершить. Избавиться от подобной программы можно с помощью перезагрузки системы.

Более подробную информацию представляет опция -l (long format))

Для просмотра дерева процессов используются команды:

ps axjf
pstree

Завершение выполняющегося процесса

Завершение процесса выполняется командой

kill сигнал PID

Сначала процессу посылается сигнал -15. Если это не помогает,

используется крайняя мера -посылается сигнал -9.

Отображение динамически обновляемого списка выполняющихся процессов – команда **top**

В отличие от **ps**, команда **top** представляет динамически обновляемые сведения о процессах и о том, какой объем системных ресурсов использует каждый из них (рис.1).

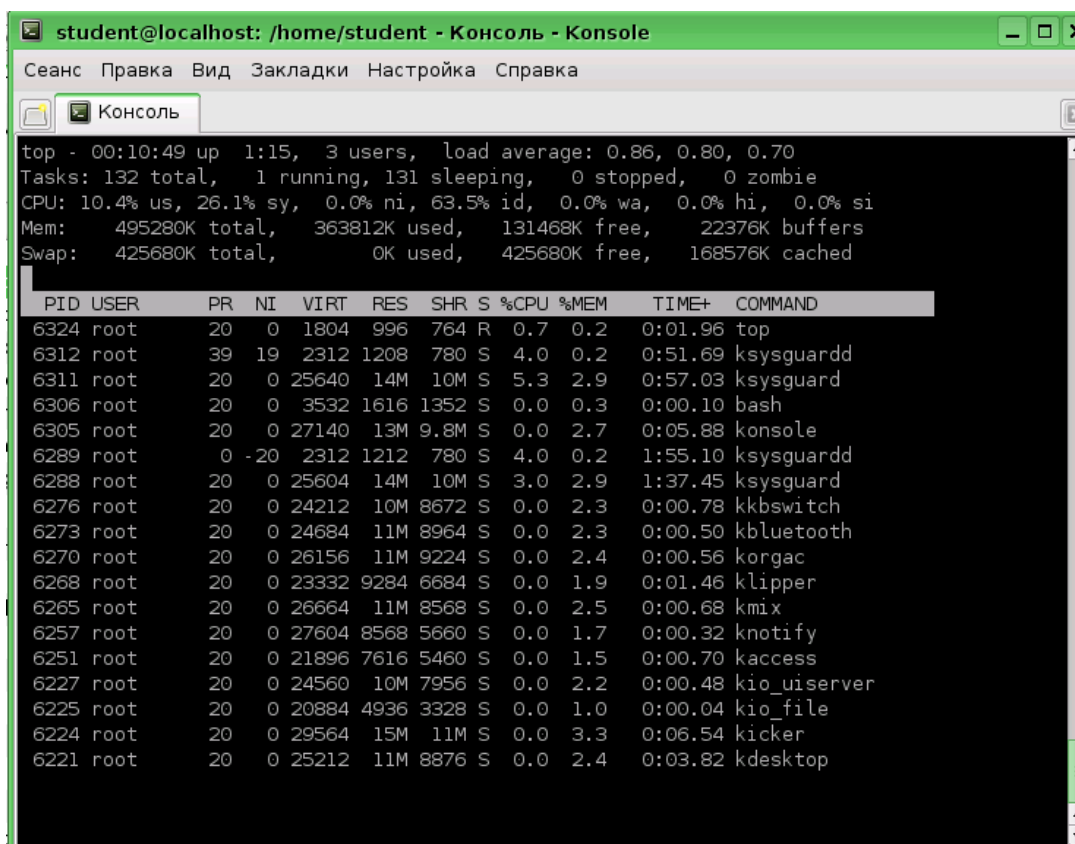
В первых пяти строках команда **top** отображает подробную информацию о системе, а затем описывает каждый выполняющийся процесс. Выходные данные сортируются по уровню загрузки ЦП данным процессом.

Для удаления процесса следует ввести символ **k**. Появится сообщение **PID to kill:**

Необходимо ввести PID процесса, программа запросит номер сигнала:

kill PID NNNN with signal [15]:

Как правило, данное значение является приемлемым, и через секунду после ввода **Enter** информация о выбранном процессе исчезнет с экрана.



```
top - 00:10:49 up 1:15, 3 users, load average: 0.86, 0.80, 0.70
Tasks: 132 total, 1 running, 131 sleeping, 0 stopped, 0 zombie
CPU: 10.4% us, 26.1% sy, 0.0% ni, 63.5% id, 0.0% wa, 0.0% hi, 0.0% si
Mem: 495280K total, 363812K used, 131468K free, 22376K buffers
Swap: 425680K total, 0K used, 425680K free, 168576K cached
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
6324	root	20	0	1804	996	764	R	0.7	0.2	0:01.96	top
6312	root	39	19	2312	1208	780	S	4.0	0.2	0:51.69	ksysguardd
6311	root	20	0	25640	14M	10M	S	5.3	2.9	0:57.03	ksysguard
6306	root	20	0	3532	1616	1352	S	0.0	0.3	0:00.10	bash
6305	root	20	0	27140	13M	9.8M	S	0.0	2.7	0:05.88	konsole
6289	root	0	-20	2312	1212	780	S	4.0	0.2	1:55.10	ksysguardd
6288	root	20	0	25604	14M	10M	S	3.0	2.9	1:37.45	ksysguard
6276	root	20	0	24212	10M	8672	S	0.0	2.3	0:00.78	kksbswitch
6273	root	20	0	24684	11M	8964	S	0.0	2.3	0:00.50	kbluetooth
6270	root	20	0	26156	11M	9224	S	0.0	2.4	0:00.56	korgac
6268	root	20	0	23332	9284	6684	S	0.0	1.9	0:01.46	klipper
6265	root	20	0	26664	11M	8568	S	0.0	2.5	0:00.68	kmix
6257	root	20	0	27604	8568	5660	S	0.0	1.7	0:00.32	knotify
6251	root	20	0	21896	7616	5460	S	0.0	1.5	0:00.70	kaccess
6227	root	20	0	24560	10M	7956	S	0.0	2.2	0:00.48	kio_userver
6225	root	20	0	20884	4936	3328	S	0.0	1.0	0:00.04	kio_file
6224	root	20	0	29564	15M	11M	S	0.0	3.3	0:06.54	kicker
6221	root	20	0	25212	11M	8876	S	0.0	2.4	0:03.82	kdesktop

Рис. 1 – результат выполнения команды **top**

В первой строке программа сообщает текущее время, время работы системы (1 час 15 мин), количество зарегистрированных (login) пользователей (3 users), общая средняя загрузка системы (load average). Общей средней загрузкой системы называется среднее число процессов, находящихся в состоянии выполнения (R) или в состоянии ожидания (D). Общая средняя загрузка измеряется каждые 1, 5 и 15 минут.

Во второй строке вывода программы `top` сообщается, что в списке процессов находятся 132 процесса, из них 131 спит (состояние готовности или ожидания), 1 выполняется (на виртуальной машине только 1 процессор), 0 процессов зомби и 0 остановленных процессов.

В третьей-пятой строках приводится информация о загрузке процессора CPU в режиме пользователя и системном режиме, использования памяти и файла подкачки.

В таблице отображается различная информация о процессе. Рассмотрим колонки PID (идентификатор процесса), USER (пользователь, запустивший процесс), S (состояние процесса) и COMMAND (команда, которая была введена для запуска процесса).

Колонка S может содержать следующие значения:

R - процесс выполняется или готов к выполнению (состояние готовности)

D - процесс в "беспробудном сне" - ожидает дискового ввода/вывода

T - процесс остановлен (stopped) или трассируется отладчиком

S - процесс в состоянии ожидания (sleeping)

Z - процесс-зомби

N – процесс с низким приоритетом, nice, $\text{pri} < 19$

< - процесс с высоким приоритетом, $\text{pri} > 19$

+ - процесс в группе фоновых процессов

l – процесс с двумя и более потоками, многопоточный

s – ведущий процесс сеанса.

Колонка PR содержит приоритет процесса – целое число от 0 до 39. Колонка NI (NICE) (фактор уступчивости процесса) содержит задаваемое значение от -19 (наименее уступчивый) до 20 (самый уступчивый, вытесняется всеми). Значение NICE прибавляется к числу 20 для получения значения приоритета

$$\text{PR} = 19 + \text{NICE}$$

Для управления командой `top` используются односимвольные команды:

- `h` – вывод справки о командах;
- `r` – `renice` – изменение приоритета, в режиме администратора (через `sudo`) можно задавать значения от -19 до 20;
- `q` – завершение работы с командой и другие (см. информацию справки).

Недостаток команды – вывод только первых $N < 26$ строк информации о процессах.

Для управления процессами с использованием графического интерфейса используется утилита Системный монитор, которая запускается из системного меню Система-Администрирование – Системный монитор (рис.2).

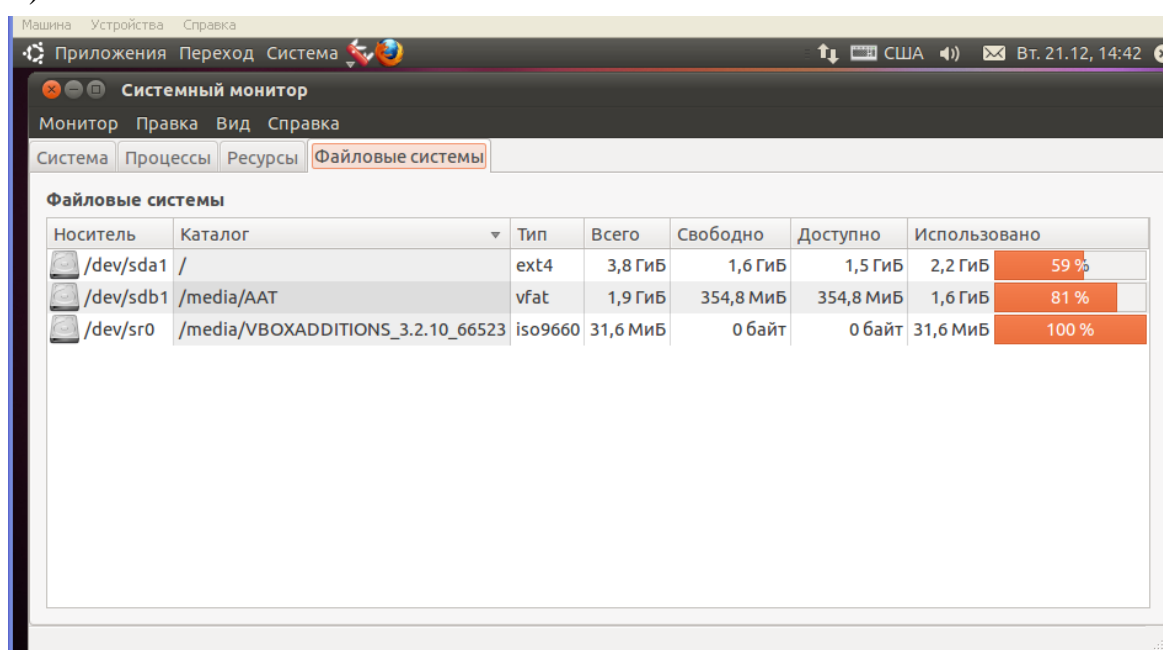


Рис. 2. Системный монитор

На закладке Процессы монитор показывает сведения о запущенных процессах, на закладке система – сведения об использовании ЦП и памяти. Достоинства монитора – использование графического интерфейса, отображение списка всех процессов.

Вывод дерева процессов

Для построения дерева процессов используются команды

`pstree`

`ps -ejH`

`ps axjt`

Получение информации о потоках

Как известно, процесс может иметь параллельно выполняющиеся потоки (threads) или облегченные процессы (LWP, Light Weight Process). Для получения информации о потоках заданного процесса используется опция – L, например `ps -fLC swriter.bin` выводит список потоков приложения writer Open Office. Процессы, использующие более одного потока – редактор звуковых файлов audacity и soffice.bin, а также демоны (службы в терминологии Windows). Как указано выше, многопоточные процессы помечено символом l в колонке состояния.

1.1.2 Получение списка открытых файлов

Команда lsof (List open files) без параметров выводит полный список открытых файлов. Пользователь-администратор получит несколько тысяч строк текста.

Для получения списка файлов, открытых конкретным пользователем, служит команда

```
lsof -u имя_пользователя
```

Получение списка пользователей конкретного файла

Необходимо ввести команду lsof с указанием имени файла. Например `lsof /bin/bash`

Отображение информации об оперативной памяти системы

Текущее состояние системной памяти позволяет получить команда **free**

По умолчанию все значения представлены в килобайтах. Значения в М позволяет получить опция –m.

1.1.3 Отображение информации об использовании дискового пространства

Команда df выводит данные об объеме доступного дискового пространства (в Кбайтах). Опция –h улучшает восприятие результатов.

Команда du дает возможность узнать объем дисковой памяти, занимаемой каталогами и файлами.

1.2 Файловая система /proc

Ядро Linux предоставляет механизм доступа к своим внутренним структурам и позволяет изменять установки ядра во время работы ОС посредством файловой системы /proc. Файловая система /proc является механизмом для ядра и его модулей, позволяющим посылать информацию процессам (отсюда и название /proc). С помощью этой виртуальной файловой системы можно работать с внутренними структурами ядра, получать полезную информацию о процессах и изменять установки (меняя параметры ядра) на лету. Файловая система /proc располагается в памяти в отличие от других файловых систем, которые располагаются на диске.

Файловая система /proc контролируется ядром. Из-за того, что она предоставляет информацию, контролируемую ядром, она располагается в памяти, контролируемой также ядром. Команда "ls -l" покажет, что большинство файлов в этой системе имеют нулевую длину, но посмотрев любой файл, Вы получите достаточно информации. Как это может быть? Все просто - файловая система /proc как любая другая файловая система регистрируется на уровне VFS (Virtual File System layer). Поэтому при запросе файлов/каталогов, файловая система /proc создает эти файлы/каталоги на основании информации, содержащейся в ядре.

В действительности многие программы собирают информацию из файлов в /proc, форматируют её своим собственным способом, а результат затем выводят на экран. Существует несколько программ, которые поступают именно так при выводе информации о процессах (top, ps и т. п.), /proc - это также хороший источник информации об аппаратном обеспечении, и по аналогии с программами, показывающими процессы, некоторые другие программы являются просто интерфейсами к информации, находящейся в /proc.

Также существует специальный подкаталог /proc/sys. Он позволяет отображать параметры ядра и изменять их в режиме реального времени.

1.2.1 Информация о процессах

Каждый из каталогов содержит одинаковые пункты, 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.

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 шестнадцатиричном виде) устройства, на котором находится распределенный файл; номер inode файла; и, наконец, имя самого файла.

7. **root**: эта символическая ссылка указывает на корневой каталог, используемый процессом. Обычно это будет /.

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.

Если вывести список содержимого каталога **fd** для процесса 127, получим примерно 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.

1.3 Информация об аппаратном обеспечении

Кроме каталогов, связанных с различными процессами, в /proc также содержится значительный объём информации об аппаратном обеспечении ПК.

Список файлов каталога /proc, полученный с помощью команды `ls -d [a-z]*` выглядит следующим образом:

acpi	fb	kmsg	slabinfo
asound	filesystems	loadavg	stat
buddyinfo	fs	locks	swaps
bus	ide	mdstat	sys
cmdline	interrupts	meminfo	sysrq-trigger
cpuinfo	iomem	misc	sysvipc
crypto	ioports	modules	tty
devices	irq	mounts	uptime
diskstats	kallsyms	net	version
dma	kcore	partitions	vmstat
driver	keys	schedstat	zonein
execdomains	key-users	self	

Например, каталог /proc/interrupts содержит список прерываний, используемых в данный момент системой, а также периферийные устройства, которые их используют.

Описание некоторых из файлов /proc:

cpuinfo: этот файл содержит, как видно из его имени, информацию о процессорах машины. Пример содержимого файла:

```
cat /proc/cpuinfo
```

```
processor          : 0
vendor_id         : GenuineIntel
cpu family        : 6
model             : 8
model name        : Pentium III (Coppermine)
stepping          : 6
cpu MHz           : 1000.119
cache size        : 256 KB
fdiv_bug          : no
hlt_bug           : no
sep_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 mmx fxsr xmm
bogomips          : 2015.85
processor          : 3
vendor_id         : GenuineIntel
cpu family        : 6
model             : 8
model name        : Pentium III (Coppermine)
stepping          : 6
cpu MHz           : 1000.119
cache size        : 256 KB
fdiv_bug          : no
hlt_bug           : no
sep_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 mmx fxsr xmm
bogomips       : 2015.29
```

modules: этот файл содержит список модулей, используемых ядром в настоящий момент, вместе со счетчиком использования каждого из модулей. Эта информация используется командой `lsmod`, которая отображает её в более удобной для чтения форме,

meminfo: этот файл содержит информацию о загрузке памяти на момент вывода его содержимого. Команда `free` выведет ту же самую информацию, но уже в более удобном для чтения формате.

bus: этот подкаталог содержит информацию обо всех периферийных устройствах, найденных на различных шинах вашего компьютера. Информация обычно не удобна для чтения, и большая её часть переформатируется внешними утилитами.

acpi: некоторые файлы и каталоги, представленные в этом каталоге, особенно интересны для ноутбуков, которые позволяют вам выбирать различные варианты энергосбережения.

1.4 Отображение и изменение параметров ядра

Назначение подкаталога `/proc/sys` - сообщать о различных параметрах ядра, и позволить изменять некоторые из них в интерактивном режиме. В противоположность всем другим файлам каталога `/proc`, некоторые файлы из этого каталога могут быть открыты для записи, но только для `root`'а.

Содержимое этих каталогов зависит от системы, а большинство файлов будет полезно только для очень специализированных приложений.

2 МЕТОДИКА ВЫПОЛНЕНИЯ

1. Вывести список всех процессов системы.
2. Вывести дерево процессов.
3. С помощью команды `top` получить список 5 процессов, потребляющих наибольшее количество процессорного времени.
4. Найти 2 процесса, имеющих более ДВУХ потоков. Использовать состояние процесса

5. Используя команду `top`, изменить приоритеты 2 процессов.
6. Получить список открытых файлов пользователя `aa`
7. Получить текущее состояние системной памяти
8. Получить справку об использовании дискового пространства.
9. Вывести информацию о каком-либо процессе, используя содержимое каталога `/proc`
10. Вывести информацию о процессоре ПК, используя содержимое каталога `/proc`
11. Вывести список модулей, используемых в настоящий момент ядром ОС.

3 ОТЧЕТ О РАБОТЕ

Готовится в письменном виде один на бригаду. Содержание отчета:

1. Результаты выполнения заданий 1- 11 (снимки экранов) и использованные команды ОС Linux.

4 КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Команды вывода списка процессов.
2. Команда получения списка потоков
3. Команда для завершения приложений.
4. Состояния процесса Linux.
5. Получение информации о потоках процесса.
6. Примеры многопоточных процессов.
7. Необходимость использования потоков.
8. Процессы – зомби: как они появляются, как их найти и что с ними делать?
9. Содержимое вывода команды `top`.
10. Как получить информацию о процессах системы, используя файловую систему `/proc`?
11. Команды для получения информации об открытых файлах
12. Получение информации о состоянии системной памяти.
13. Получение информации об использовании дискового пространства.
14. Назначение файловой системы `/proc`