# *Управление процессами в Linux*

## Цель работы

Получить представление о процессах, как о способе управления ресурсами в Линукс. Научиться получать и анализировать информацию о процессах и управлять состоянием выполняющихся процессов.

## Задания к работе

1. Войти в систему с собственной учетной записью
2. Получить справку о команде ps
3. Командой ps вывести краткую информацию о выполняющихся процессах в текущем терминале и определить PID текущей оболочки
4. Получить подробную информацию о загруженных процессах и выяснить, какой из них использует максимальный объем памяти, а какой - максимально загружает процессор
5. Из таблицы, полученной в п.4 выяснить, какой PID имеет процесс init и от чьего имени он запущен
6. Открыть новый сеанс с собственной учетной записью в и запустить в нем текстовый редактор nano
7. Вернуться в исходный сеанс и снова просмотреть список процессов. Определить PID MC, запущенного от вашего имени
8. Командой kill снять все процессы nano
9. В tty1 выполнить команду top. Сравнить ее возможности с возможностями ps
10. Используя top или ps определить, какие процессы порождены (поле PPID) процессом init (PID=1)
11. Запустить поиск всех файлов .html от каталога /. Приостановить этот процесс (Ctrl+Z).
12. Запустить команду man bash и приостановить ее выполнение
13. Командой jobs определить номера задач, запущенных в предыдущих пунктах.

14. Командой fg продолжить выполнение man bash.
15. Принудительно (kill) завершить команду find.

## Методические указания

### *Понятие процесса*

Под процессами во всех unix-подобных системах подразумевается любая независимо выполняющаяся программа со всеми используемыми ресурсами. Процесс - одно из ключевых понятий, на которых базируется Линукс, как unix-подобная система, и оно тесно связано с такими понятиями, как учетные записи, права доступа и файловая система. Эта связь неразрывна и, вдобавок, рекурсивна:

- пользователь (реальный или виртуальный) запускает процессы, порождающие файлы;
- права доступа процесса соответствуют правам доступа запустившего его пользователя;
- порожденные процессом файлы получают права, соответствующие правам процесса;
- права пользователя определены параметрами его учетной записи.

Каждый процесс уникален. Для идентификации процесса используется числовое значение, т.н. идентификатор процесса (PID, Process Identificator). Для каждого процесса известен владелец(пользователь, запустивший процесс). Если процесс создан реальным пользователем, то он привязан к терминалу, из которого был запущен. Для виртуальных (системных) пользователей такой ассоциации не производится. Помимо собственного идентификатора, каждый процесс имеет еще и идентификатор родительского процесса (PPID, Parent PID). В каждый момент времени системе известно состояние процесса - степень его исполнения. Процесс может быть:

- выполняемым в текущий момент (R, Runned);
- находящимся в режиме ожидания (S, Suspended);
- прерванным (T, Terminated), например, при использовании клавиш Ctrl+Z;

- "зомби" (Z, Zombied) - завершившимся, но от которого родительский процесс еще не принял сигнала завершения. Спустя некоторое время "зомби" завершаются окончательно и освобождают ресурсы;
- зависшим, или в состоянии непрерывного ожидания (uninterruptible sleep). Такой процесс не реагирует на какие-либо сигналы и может быть снят только перезагрузкой системы.

Все запущенные процессы условно (в зависимости от выполняемой ими функции) можно разделить на три типа: Системные процессы являются частью ядра и всегда расположены в оперативной памяти. Они часто не имеют соответствующих им программ в виде исполняемых файлов и всегда запускаются особым образом при загрузке ядра системы. Процессы-демоны – это неинтерактивные процессы, которые выполняются в фоновом режиме. К прикладным относятся все остальные процессы, выполняющиеся в системе. Интерактивные процессы связаны с определѐнным терминалом и через него взаимодействуют с пользователем. Фоновые процессы выполняются независимо от пользователя и (псевдо)параллельно. Каждый процесс в операционной системе Linux может находиться в одном из четырѐх состояний: работоспособный, спящий (или ожидающий), остановленный и завершившийся.

Еще одной характеристикой процессов является уровень приоритета (NI, NIce value, "степень дружественности"). Уровень приоритета влияет на количество системных ресурсов, выделяемых процессу.

Основными командами для получения сведений о выполняемых процессах являются ps и top. Фрагмент вывода сведений командой ps с параметрами a (расширенный вывод), u (с указанием UID), x (в т.ч для виртуальных пользователей):

```
aag@stilo:~>                                ps                                aux
USER   PID   %CPU   %MEM   VSZ   RSS   TTY   STAT   START   TIME   COMMAND
root    1    0.0    0.0    744    284    ?    Ss    14:27    0:00    init    [5]
root    2    0.0    0.0    0    0    ?    S<    14:27    0:00    [kthreadd]
root    3    0.0    0.0    0    0    ?    S<    14:27    0:00    [migration/0]
root    4    0.0    0.0    0    0    ?    SN    14:27    0:00    [ksoftirqd/0]
root    5    0.0    0.0    0    0    ?    S<    14:27    0:00    [events/0]
root    6    0.0    0.0    0    0    ?    S<    14:27    0:00    [khelper]
root    25    0.0    0.0    0    0    ?    S<    14:27    0:00    [kblockd/0]
...
root    141    0.0    0.0    0    0    ?    S<    14:27    0:00    [kswapd0]
root    142    0.0    0.0    0    0    ?    S<    14:27    0:00    [aio/0]
root    367    0.0    0.0    0    0    ?    S<    14:27    0:00    [kpsmoused]
root    377    0.0    0.0    0    0    ?    S<    14:27    0:00    [kondemand/0]
...
root    991    0.0    0.0    0    0    ?    D<    14:28    0:01    [kjournald]
root    1682    0.0    0.0    0    0    ?    S<    14:28    0:01    [ipw2200/0]
root    1694    0.0    0.0    0    0    ?    S<    14:28    0:00    [khpsbpkt]

wwwrun   3323   0.0   2.4   104548   12804   ?   S   14:28   0:00   /usr/sbin/httpd
wwwrun   3324   0.0   2.4   104540   12816   ?   S   14:28   0:00   /usr/sbin/httpd

...
```

Дополнительная информация о команде ps доступна при указании параметра --help или в справке man.

### *Управление процессами*

Чтобы запустить программу достаточно ввести ее имя в командной строке и нажать «Enter». Однако не все команды запускают единственный процесс.

Интерактивные процессы, запущенные в терминале, занимают терминальную сессию, и оболочка не выводит пользователю строку приглашения до тех пор, пока программа не завершится.

```
$ firefox csc.sibsutis.ru
```

Работу некоторых запущенных в терминале программ можно прервать с помощью сочетания клавиш «Ctrl + c» в окне терминала. В этот момент программе посылается сигнал INT (Interrupt).

Чтобы запустить программу в фоновом режиме необходимо завершить команду символом амперсанд «&». После этого в терминал выводится информация о запущенном процессе включая номер задания терминала, и приглашения пользователю на ввод новой команды.

```
$ top &
```

Используя команду jobs мы можем получить список заданий которые запущены через терминал.

```
$ jobs
```

Чтобы вернуть запущенный в фоне процесс на передний план используется команда fg с указанием номера задания из списка заданий.

```
$ fg %2
```

Если мы хотим перевести процесс в состояние остановленный, используется сочетание клавиш «Ctrl + z». В этот момент программе посылается сигнал TSTP (Terminal Stop).

После этого мы можем либо переместить задание на передний план командой fg, либо продолжить его выполнение в фоновом режиме командой bg.

```
$ bg %2
```

Пользователь может управлять только теми процессами, владельцем которых является. Суперпользователь может управлять всеми процессами.

Управление запущенными процессами сводится к приостановке выполнения, изменению приоритета и принудительному завершению.

Приостановить выполнение активного процесса можно сочетанием клавиш Ctrl+Z. Для продолжения его работы можно использовать команду fg. Если имеется несколько приостановленных процессов, то для команды fg необходимо указать порядковый номер задания в текущей оболочке, (не путать с PID), работу которого нужно продолжить. Узнать номер задания можно командой jobs.

Изменение приоритета процесса - задача, возникающая (нечасто) при необходимости перераспределения ресурсов системы. Значения уровня приоритета (nice value) изменяется от -20 (наименьшая "дружественность", высший приоритет) до +20 (низший приоритет). Все пользовательские (и большинство системных) процессы запускаются с равным приоритетом (nice value = 0). Это значение может быть изменено двояко:

- Во-первых, при запуске программы командой nice с указанием необходимого уровня приоритета и именем запускаемой программы. Например:

```
[aag@localhost ~]$ nice -5 find / *.html
```

- Во-вторых, для ранее запущенной задачи, командой renice с указанием уровня и PID задачи. Например, так:

```
[aag@localhost ~]$ renice --7 20117
```

Пользователь имеет право понижать приоритет собственных задач. Повышать уровень приоритета любой задачи может только суперпользователь.

Гораздо чаще, чем изменение приоритета, возникает необходимость принудительного завершения (снятия) процесса. Такая ситуация возникает, например тогда, когда процесс "зависает", т.е. перестает воспринимать нажатия клавиш и не отвечает на системные события. Для снятия "зависшей" программы предназначена команда kill, которая передает ей один из сигналов завершения. Список сигналов доступен по команде kill -l, а их подробное описание - по команде man 7 signal. Здесь же отметим, что без явного указания имени (или номера), процессу будет передан сигнал SIGTERM (номер 15), предписывающий по возможности корректно, с сохранением информации, завершить работу. Примеры использования команды:

Вызов со значением сигнала по умолчанию (SIGTERM):

```
[aag@localhost ~]$ find / *.html
[aag@localhost ~]$ ps
PID TTY TIME CMD
2663 pts/1 00:00:00 bash
20712 pts/1 00:00:00 find
20762 pts/1 00:00:00 ps
[aag@localhost ~]$ kill 20712
```

Явное указание номера сигнала:

```
[aag@localhost ~]$ kill -15 20712
```

Явное указание имени сигнала (номер 9, SIGKILL, требующий немедленного завершения работы программы):

```
[aag@localhost ~]$ kill -SIGKILL 20712
```

### *Команда top*

Большей гибкостью и универсальностью по сравнению с командой ps обладает команда top. Она позволяет не только получить информацию о процессах, но и выполнять мониторинг через заданные интервалы времени. Также эта команда позволяет управлять процессами, объединяя возможности команд jobs, nice, fg и kill. Все параметры и действия команды top являются настраиваемыми. Для команды доступна как справка в формате man, так и интерактивная справка по нажатию клавиш H или ?.

## Контрольные вопросы

1. В системе зарегистрированы и работают пользователи user1 и user2. Может ли user1 завершить работу процесса, запущенного пользователем user2?
2. Может ли user1 понизить приоритет процесса, запущенного user2?

3. Может ли user1 повысить приоритет собственного процесса?

## Дополнительные задания

1. Установите программу htop и познакомьтесь с ней. Используя ее в процессе повседневной работы ответьте на несколько вопросов:
    a. Какие программы потребляют больше всего оперативной памяти?
    b. Насколько часто центральный процессор бывает загружен полностью и при выполнении каких задач?
    c. Сколько памяти потребляет система в режиме простоя?
    d. Требуется ли системе апгрейд и каких компонентов в первую очередь?