

Лабораторная работа №3. Команды Linux для работы с процессами

Цель: изучение команд Linux для работы с процессами.

ЗАДАНИЕ

1. Изучите главу 4 теоретического курса «Команды Linux для работы с процессами».
2. Войдите в Linux и запустите программу Terminal.
3. С помощью команды `man` изучите встроенную справку для команд `jobs`, `bg`, `fg`, `ps`, `pstree`, `top`, `at`, `nohup`, `nice`, `renice`, `kill`, `killall`. Обратите внимание на опции команд, не рассмотренные в теоретическом курсе.
4. Выполните рассмотренные команды с различными значениями управляющих ключей и входных данных, учитывая типичные сценарии использования команд. При исследовании команд управления процессами можно использовать две встроенные программы Linux – `hexeyes` и `yes`. Программа `yes` выдает на стандартный вывод последовательность строк, состоящих из одной буквы «у» и может использоваться как отладочная программа для выдачи потока данных. Запускается командой `yes`. Программа `hexeyes` демонстрирует возможности графических программ XWindow. Она выводит графическое окно с изображением глаз, которые следят за перемещениями мыши. Программа `hexeyes` не использует возможности терминала. Данную программу можно запустить командой `$hexeyes –center red`.

```
guzovskiy@debian:~$ help bg
bg: bg [задание ...]
    Перевод заданий в фоновый режим.

    Переводит указанные задания в фоновый режим, как если бы
    они были запущены с «&». Если задание не указано, используются данные
    текущего задания.

    Состояние выхода:
    Возвращает успех, если не включено управление заданиями или не произошла
    ошибка.

guzovskiy@debian:~$ help fg
fg: fg [задание]
    Перевод задания в интерактивный режим.

    Переводит указанное задание в интерактивный режим, делая
    его текущим заданием. Если задание не указано, используется
    текущее задание.

    Состояние выхода:
    Возвращает состояние команды, переведённой в интерактивный режим, или ошибку,
    если возникла ошибка.
```

```
guzovskiy@debian:~$ help jobs
jobs: jobs [-lnprs] [задание ...] или jobs -x команда [аргументы]
      Display status of jobs.

Lists the active jobs.  JOBSPEC restricts output to that job.
Without options, the status of all active jobs is displayed.

Options:
  -l      lists process IDs in addition to the normal information
  -n      lists only processes that have changed status since the last
          notification
  -p      lists process IDs only
  -r      restrict output to running jobs
  -s      restrict output to stopped jobs

If -x is supplied, COMMAND is run after all job specifications that
appear in ARGS have been replaced with the process ID of that job's
process group leader.
```

Рисунок 1 - Пример использования команд

```
guzovskiy@debian:~$ yes > /dev/null
^Z
[3]+  Остановлен    yes > /dev/null
guzovskiy@debian:~$ jobs
```

Рисунок 2 - Просмотр фоновых заданий

```
[3]+  Остановлен    yes > /dev/null
guzovskiy@debian:~$ bg %1
[1] yes > /dev/null &
```

Рисунок 3 - Перевод задания из фона на передний план

```
guzovskiy@debian:~$ yes > /dev/null
^Z
[3]+  Остановлен    yes > /dev/null
guzovskiy@debian:~$ jobs
[1]  Остановлен    yes > /dev/null
[2]-  Остановлен    yes > /dev/null
[3]+  Остановлен    yes > /dev/null
guzovskiy@debian:~$ bg %1
[1] yes > /dev/null &
```

Рисунок 4 - Приостановка и возобновление

```

guzovskiy@debian:~$ ps
  PID TTY          TIME CMD
 1846 pts/0    00:00:00 bash
  8341 pts/0    00:00:34 yes
  8343 pts/0    00:00:11 yes
  8344 pts/0    00:00:09 yes
  8345 pts/0    00:00:00 ps
guzovskiy@debian:~$ ps aux | grep yes
guzovsk+  8341 35.0  0.0   5356   556 pts/0    R   22:25   0:46 yes
guzovsk+  8343 16.0  0.0   5356   500 pts/0    T   22:26   0:11 yes
guzovsk+  8344 17.9  0.0   5356   560 pts/0    T   22:26   0:09 yes
guzovsk+  8347  0.0  0.0   6264   636 pts/0    S+  22:27   0:00 grep yes

```

Рисунок 5 - Просмотр процессов

```

guzovskiy@debian:~$ pstree -p
systemd(1)─ModemManager(458)─{ModemManager}(471)
              │               └─{ModemManager}(477)
              └─NetworkManager(405)─{NetworkManager}(457)
                  │               └─{NetworkManager}(460)
                  └─accounts-daemon(400)─{accounts-daemon}(406)
                      │               └─{accounts-daemon}(448)
                  └─avahi-daemon(402)─avahi-daemon(429)
                  └─colord(825)─{colord}(848)
                      │       └─{colord}(850)
                  └─cron(403)
                  └─cups-browsed(7534)─{cups-browsed}(7538)
                      │               └─{cups-browsed}(7539)
                  └─cupsd(7533)
                  └─dbus-daemon(404)
                  └─fwupd(8195)─{fwupd}(8196)
                      │       └─{fwupd}(8197)

```

Рисунок 6 - Древовидное представление

```

guzovskiy@debian:~$ top
top - 22:28:09 up 2 days, 21:02,  1 user,  load average: 0,66, 0,24, 0,09
Tasks: 157 total,   5 running, 150 sleeping,   2 stopped,   0 zombie
%Cpu(s): 17,2 us, 82,8 sy,  0,0 ni,  0,0 id,  0,0 wa,  0,0 hi,  0,0 si,  0,0 st
MiB Mem : 1971,1 total,  433,5 free,  687,3 used,  850,4 buff/cache
MiB Swap:  975,0 total,  975,0 free,   0,0 used. 1108,5 avail Mem

```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
8341	guzovsk+	20	0	5356	556	496	R	96,6	0,0	1:07.21	yes
8349	guzovsk+	20	0	10148	3676	3196	R	3,4	0,2	0:00.01	top
1	root	20	0	164136	10364	7684	S	0,0	0,5	0:05.33	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.07	kthreadd
3	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_gp
4	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	rcu_par+
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker+
8	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mm_perc+
9	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_tas+
10	root	20	0	0	0	0	S	0,0	0,0	0:00.00	rcu_tas+
11	root	20	0	0	0	0	S	0,0	0,0	0:02.15	ksoftir+
12	root	20	0	0	0	0	T	0,0	0,0	0:01.85	rcu_sch+

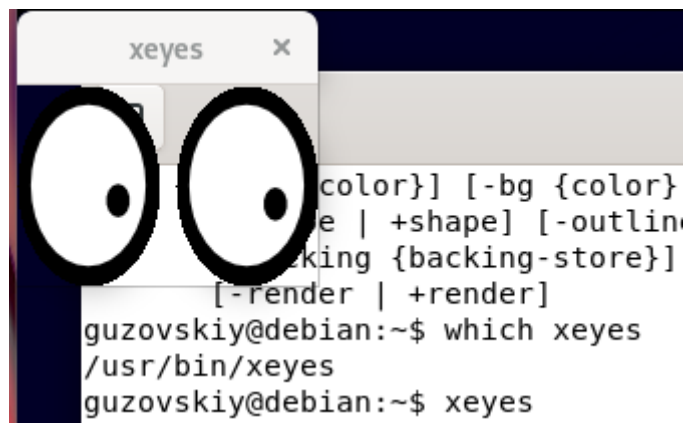


Рисунок 8 - Пример работы с xeyes

Контрольные вопросы

1. Понятие процесса

Процесс — это выполняемая программа, загруженная в оперативную память. Он представляет собой экземпляр программы, включающий её код, используемые ресурсы (файлы, память, дескрипторы), текущее состояние процессора (регистры, счётчик команд) и уникальный идентификатор (PID).

Различают следующие типы процессов:

- Родительский — запускает другие процессы;
- Дочерний — создаётся с помощью системного вызова;
- Сестринский — имеет общего родителя с другими процессами;
- Процесс `init` или `systemd` — первый процесс в системе, запускаемый ядром, и предок всех остальных (PID = 1).

2. Атрибуты и состояния процессов

К основным характеристикам процесса относятся:

- PID — идентификатор процесса;
- PPID — идентификатор родительского процесса;
- UID/GID — пользователь и группа, от имени которых выполняется процесс;
- PRI — приоритет выполнения;
- NI — значение, влияющее на приоритет;
- STAT — текущее состояние;
- TTY — терминал, из которого был запущен процесс;
- TIME — использованное процессорное время;
- CMD — имя выполняемой команды.

Состояния процесса:

- R — активное выполнение;
- S — ожидание события;
- D — непрерываемое ожидание, например при вводе-выводе;
- T — остановлен вручную;
- Z — завершён, но не обработан родительским процессом (зомби).

3. Передний и фоновый режимы

Процессы могут запускаться в переднем или фоновом режиме. В первом случае они занимают терминал до завершения, во втором — работают параллельно, не мешая пользователю выполнять другие команды. Существуют команды для управления такими задачами: отображение списка, перевод в фон или передний план.

4. Информация о процессах

Для получения сведений о запущенных процессах используется команда, позволяющая узнать их идентификаторы, состояние, владельца, потребление ресурсов и другие параметры. Существуют различные режимы отображения, включая вывод всех процессов системы и расширенную информацию.

5. Мониторинг процессов в реальном времени

Существует интерактивная утилита, отображающая процессы в реальном времени. Она показывает загрузку процессора, использование памяти и системных ресурсов, а также список процессов, отсортированных по степени активности. В интерактивном режиме доступны действия по управлению процессами, изменению приоритета и завершению работы.

6. Планирование задач

Для запуска задач в определённое время используется команда, позволяющая выполнить действие один раз в будущем, даже если пользователь выйдет из системы. Существуют команды для просмотра очереди заданий и их удаления. Также применяется утилита, обеспечивающая продолжение работы процесса после выхода пользователя, защищая его от завершения сессии и перенаправляя вывод в отдельный файл.

7. Приоритеты процессов

Относительный приоритет определяет, насколько часто процесс получает доступ к процессору. Чем выше значение, тем ниже приоритет. Существуют команды для запуска процесса с заданным приоритетом и для изменения приоритета уже работающего процесса по его идентификатору.

8. Сигналы и управление процессами

Сигналы — это способ взаимодействия между процессами, позволяющий управлять их выполнением: останавливать, возобновлять или завершать. Существуют команды для отправки сигнала конкретному процессу по его идентификатору, а также для завершения всех процессов с указанным именем. Главное различие между

ними — одна работает с PID, другая — с именем процесса.