

Операционная система Linux:

Процессы, управление процессами

Предисловие

На этом занятии мы поговорим о:

- процессах в ОС Linux;
- атрибутах процессов;
- потоках;
- сигналах.

По итогу занятия вы сможете получить информацию о запущенных в системе процессах и освоите методы взаимодействия процессов между собой.

План занятия

1. [Предисловие](#)
2. [Процессы](#)
3. [Атрибуты процесса](#)
4. [Сигналы](#)
5. [Потоки](#)
6. [Итоги](#)
7. [Домашнее задание](#)



Процессы

Определение процесса

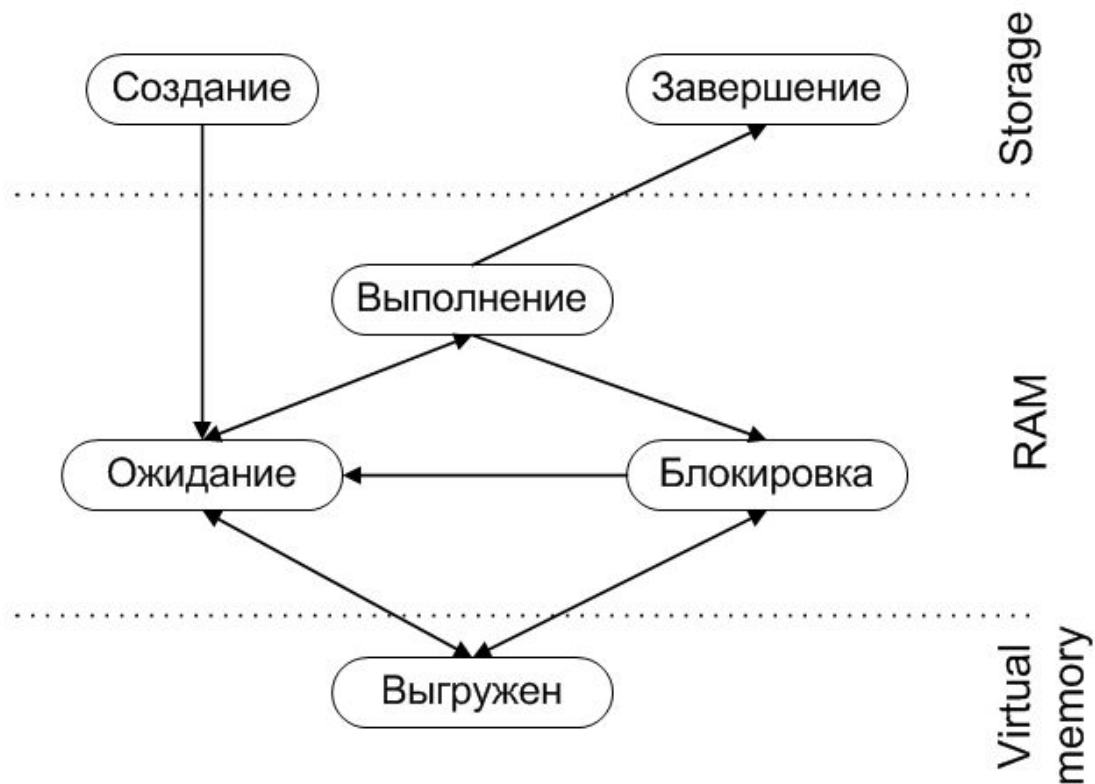
- Программа во время выполнения
или
- Сущность, представляющая понятие активности/работы с точки зрения операционной системы

Основные ресурсы процесса

- память;
- процессорное время;
- счетчик команд;
- оборудование (устройства ввода-вывода).

Жизненный цикл процесса

Переход процессов ОС из одного состояния в другое



Утилита ps

Просмотр запущенных
процессов

Пример:

user@user:~\$ ps -a

PID	TTY	TIME	CMD
1298	tty1	00:00:10	Xorg
1307	tty1	00:00:00	gnome-session-b
1455	tty1	00:00:28	gnome-shell
1498	tty1	00:00:00	ibus-daemon
1502	tty1	00:00:00	ibus-dconf
1505	tty1	00:00:00	ibus-x11
1589	tty1	00:00:00	gsd-power
1590	tty1	00:00:00	gsd-print-notif
1593	tty1	00:00:00	gsd-rfkill
1594	tty1	00:00:00	gsd-screensaver
1596	tty1	00:00:00	gsd-sharing
1602	tty1	00:00:00	gsd-sound
1603	tty1	00:00:00	gsd-xsettings
1609	tty1	00:00:00	gsd-smartcard
1612	tty1	00:00:00	gsd-wacom
1621	tty1	00:00:00	gsd-a11y-settin
1624	tty1	00:00:00	gsd-clipboard
1626	tty1	00:00:00	gsd-color
1629	tty1	00:00:00	gsd-datetime
1631	tty1	00:00:00	gsd-housekeepin
1632	tty1	00:00:00	gsd-keyboard
1634	tty1	00:00:00	gsd-media-keys
1638	tty1	00:00:00	gsd-mouse

Поля вывода ps

- **PID** — идентификатор процесса;
- **TTY** — устройство (консоль), на котором запущен процесс;
- **STAT** — статус процесса;
- **TIME** — количество времени CPU использованное процессом;
- **COMMAND** — команда запуска;
- **START** - время запуска.

Статус процесса ps

- **R** — выполняется;
- **D** — uninterruptable sleep (ожидает ввод-вывод);
- **S** — interruptable sleep;
- **I** — idle (бездействует > 20 секунд);
- **T** — приостановлен;
- **Z** — зомби;
- **W** — выгружен на диск (swap file);
- **<** — имеет повышенный приоритет;
- **N** — имеет пониженный приоритет;
- **L** — страницы заблокированы в ядре;
- **s** — лидер сеанса (например, консоль);

Основные параметры ps

- **-e, -A** — все процессы;
- **-t** — только процессы данной консоли;
- **-N** — инверсия (ps **-N -t** - все процессы, кроме данной консоли);
- **-p <PID>,<PID>** — просмотр процессов с заданным PID;
- **-C <строка>** — просмотр процессов с заданной командой;
- **-U <username>** — просмотр процессов заданного пользователя;
- **-G <group>** — просмотр процессов заданной группы.

Утилита top

top —динамически отображает статистику системы

```
top - 11:56:19 up 40 min,  1 user,  load average: 0,13, 0,14, 0,16
Tasks: 200 total,   1 running, 160 sleeping,   0 stopped,   0 zombie
%Cpu(s):  1,1 us,  0,4 sy,  0,0 ni, 92,5 id,  6,0 wa,  0,0 hi,  0,0 si,  0,0 st
KiB Mem : 4030896 total, 1195092 free, 1187084 used, 1648720 buff/cache
KiB Swap: 1003516 total, 1003516 free,      0 used. 2546196 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
1079	root	20	0	813388	337576	12732	S	7,6	8,4	3:30.37	Suricata-Main
1526	user	20	0	2958040	233668	116324	S	0,7	5,8	0:33.02	gnome-shell
163	root	20	0	0	0	0	I	0,3	0,0	0:01.00	kworker/0:3-eve
913	mongodb	20	0	1021836	73008	34748	S	0,3	1,8	0:05.96	mongod
1363	user	20	0	469364	110076	59156	S	0,3	2,7	0:13.42	Xorg
1	root	20	0	160260	9524	6700	S	0,0	0,2	0:02.50	systemd
2	root	20	0	0	0	0	S	0,0	0,0	0:00.00	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_gp
6	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	kworker/0:0H-kb
7	root	20	0	0	0	0	I	0,0	0,0	0:00.00	kworker/0:1-eve
9	root	0	-20	0	0	0	I	0,0	0,0	0:00.00	mm_percpu_wq
10	root	20	0	0	0	0	S	0,0	0,0	0:00.16	ksoftirqd/0
11	root	20	0	0	0	0	I	0,0	0,0	0:00.62	rcu_sched
12	root	rt	0	0	0	0	S	0,0	0,0	0:00.00	migration/0
13	root	-51	0	0	0	0	S	0,0	0,0	0:00.00	idle_inject/0
14	root	20	0	0	0	0	S	0,0	0,0	0:00.00	cpuhp/0

Утилита pidstat

pidstat — мониторинг выбранного процесса в реальном времени

Установка:

```
sudo apt install sysstat
```

```
user@user-VirtualBox:~$ pidstat -p 1583 1
Linux 5.4.0-54-generic (user-VirtualBox)      25.11.2020      _x86_64_      (1 CPU)

19:30:13      UID      PID      %usr  %system  %guest   %wait   %CPU   CPU   Command
19:30:14      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:15      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:16      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:17      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:18      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:19      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:20      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:21      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:22      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
19:30:23      1000      1583      0,00    0,00    0,00    0,00    0,00    0   gsd-keyboard
```

Каталог /proc

/proc — специальный каталог Linux (отличается от Unix), содержащий системную статистику, информацию о запущенных процессах и параметрах ядра. **/proc** является специальной файловой системой, которую ядро создает в памяти.

Примеры:

user@user:~\$ `cat /proc/cpuinfo` — информация о CPU

user@user:~\$ `cat /proc/version` — версия Linux

user@user:~\$ `cat /proc/stat` — системная статистика

Файлы каталога /proc

- **cgroup** — группа управления процесса;
- **fd** — дескрипторы открытых файлов;
- **cmdline** — командная строка;
- **environ** — переменные окружения;
- **stat** — информация о состоянии процесса (ps);
- **statm** — информация об использовании памяти;
- **root** — ссылка на корневой каталог процесса.

Утилита **strace**

strace — отображает каждый системный вызов, выполненный процессом и каждый полученный сигнал.

Примеры:

```
user@user:~$ sudo strace -p 3359
```

```
user@user:~$ sudo strace -p 3359 -P /home/user/
```

```
user@user:~$ sudo strace -p 3359 -e trace=file
```




Атрибуты процесса

Основные атрибуты

- таблица распределения памяти;
- текущее состояние;
- приоритет;
- информация о ресурсах (CPU, память и т.д.);
- информация об открытых файлах (портах, сокетах);
- маска сигнала;
- имя владельца;
- PID.

PID и PPID

PID (Process IDentifier) — уникальный идентификатор процесса, назначаемый ядром. Присваивается по мере создания процессов, но может отличаться для разных пространств имен пользователей.

PPID (Parent Process IDentifier) — идентификатор родительского процесса. Если родительский процесс завершился, то PPID = 1 (systemd, init).



UID и EUID

UID (User IDentifier) — идентификатор пользователя, **создавшего** данный процесса.

EUID (Effective User IDentifier) — **текущий** идентификатор пользователя процесса.

Приоритет

Приоритет (nice) — число, показывающее, какая часть процессорного времени **может быть** выделена (ядром) для процесса. Т.е. это пользовательский приоритет, который отличается от приоритета **планировщика**. По умолчанию, наследуется от родительского процесса.

Диапазон значений: -20...19

Запуск процесса с указанным приоритетом:

```
user@user:~$ sudo nice -n 13 nano
```

Изменение приоритета у запущенного процесса:

```
user@user:~$ sudo renice 5 13311
```

Фоновое выполнение

Фоновый процесс — процесс, не блокирующий консоль.

Рассмотрим на следующем примере:

```
user@user:~$ pidstat -p 1 -t 1
```

```
user@user:~$ pidstat -p 1 -t 1 &
```

```
user@user:~$ pidstat -p 1 -t 1 > text &
```

Перенаправление вывода

`ls > file` — вывод в новый файл

`ls >> file` — дополнение к существующему файлу

`ls 1>file 2>error` — запись потоков вывода и ошибок в разные файлы

`ls 1>file 2>&1` — запись потоков вывода и ошибок в один и тот же файл

Каналы

Канал (труба, конвейер, pipe) — средство межпроцессного взаимодействия, позволяющее перенаправлять поток вывода одной команды (процесса), на поток ввода другой.

Примеры:

Команда 1 | Команда 2 | Команда 3 | ... | Команда N

`ps aux | grep root` — показать все процессы пользователя root

`ps aux | grep root | wc -l` — подсчитаем количество таких процессов

`echo "test" | cat` — что делает эта команда?



Сигналы

Источники сигналов

Сигнал (signal) — уведомление процесса о каком-либо событии.

Источники сигналов:

- терминал (Ctrl-C, Ctrl-Z);
- пользователь;
- ядро;
- другой процесс.

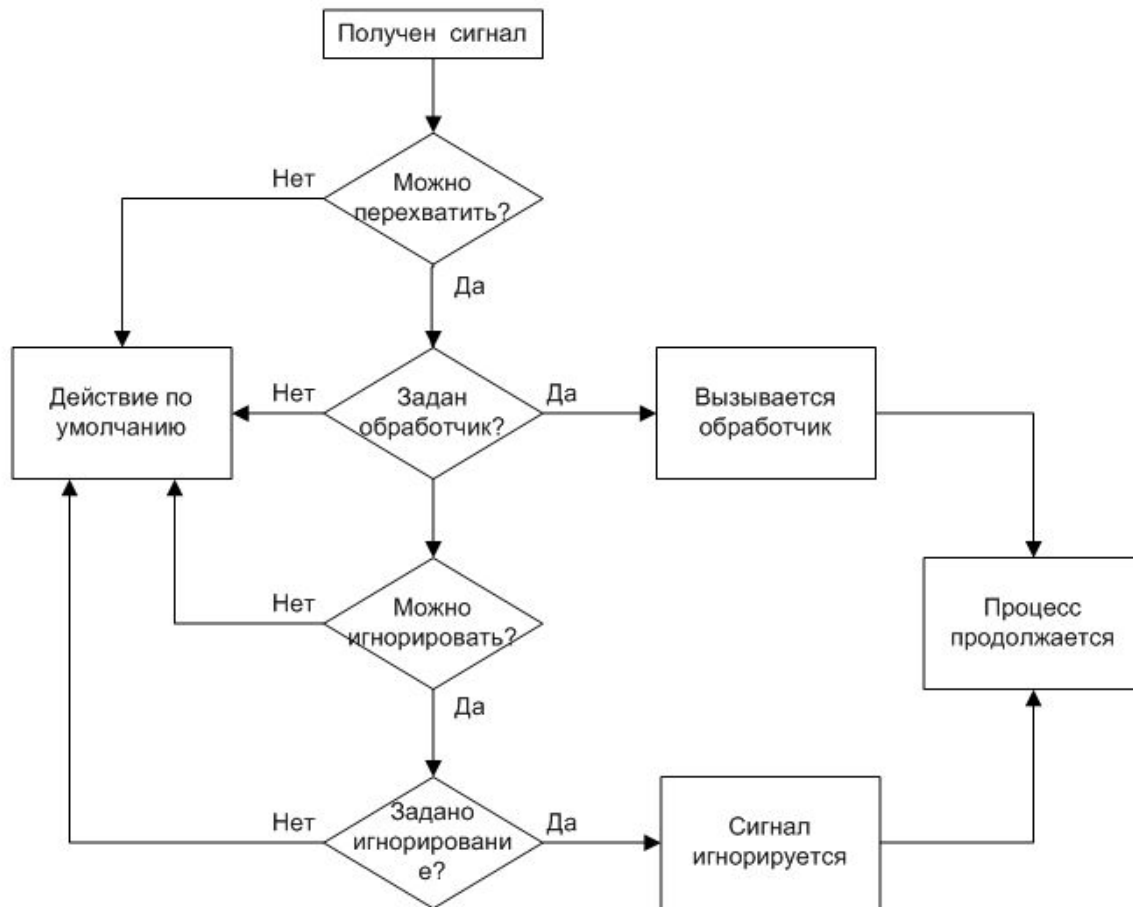
Список сигналов

user@user:~\$ kill -l

```
user@user-VirtualBox:~$ kill -l
```

1) SIGHUP	2) SIGINT	3) SIGQUIT	4) SIGILL	5) SIGTRAP
6) SIGABRT	7) SIGBUS	8) SIGFPE	9) SIGKILL	10) SIGUSR1
11) SIGSEGV	12) SIGUSR2	13) SIGPIPE	14) SIGALRM	15) SIGTERM
16) SIGSTKFLT	17) SIGCHLD	18) SIGCONT	19) SIGSTOP	20) SIGTSTP
21) SIGTTIN	22) SIGTTOU	23) SIGURG	24) SIGXCPU	25) SIGXFSZ
26) SIGVTALRM	27) SIGPROF	28) SIGWINCH	29) SIGIO	30) SIGPWR
31) SIGSYS	34) SIGRTMIN	35) SIGRTMIN+1	36) SIGRTMIN+2	37) SIGRTMIN+3
38) SIGRTMIN+4	39) SIGRTMIN+5	40) SIGRTMIN+6	41) SIGRTMIN+7	42) SIGRTMIN+8
43) SIGRTMIN+9	44) SIGRTMIN+10	45) SIGRTMIN+11	46) SIGRTMIN+12	47) SIGRTMIN+13
48) SIGRTMIN+14	49) SIGRTMIN+15	50) SIGRTMAX-14	51) SIGRTMAX-13	52) SIGRTMAX-12
53) SIGRTMAX-11	54) SIGRTMAX-10	55) SIGRTMAX-9	56) SIGRTMAX-8	57) SIGRTMAX-7
58) SIGRTMAX-6	59) SIGRTMAX-5	60) SIGRTMAX-4	61) SIGRTMAX-3	62) SIGRTMAX-2
63) SIGRTMAX-1	64) SIGRTMAX			

Обработка сигнала



Утилита kill

```
user@user:~$ sudo kill <pid>
```

```
user@user:~$ sudo kill -<имя_сигнала> <pid>
```

```
user@user:~$ sudo kill -s <имя_сигнала> <pid>
```

```
user@user:~$ sudo killall nano
```

```
user@user:~$ sudo pkill -u user2
```

Часто используемые сигналы:

- KILL — завершает процесс на уровне ядра. Не блокируется;
- INT — запрос на завершение текущей операции (CTRL-C);
- TERM — запрос на завершение программы.



Потоки

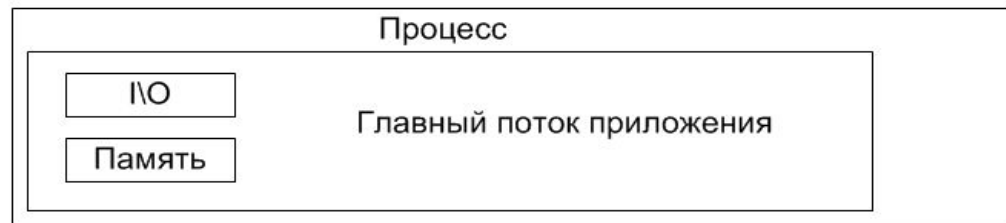
Определение потоков

Потоки (нити, threads) — программно выделенные части одного процесса, использующие его ресурсы совместно.

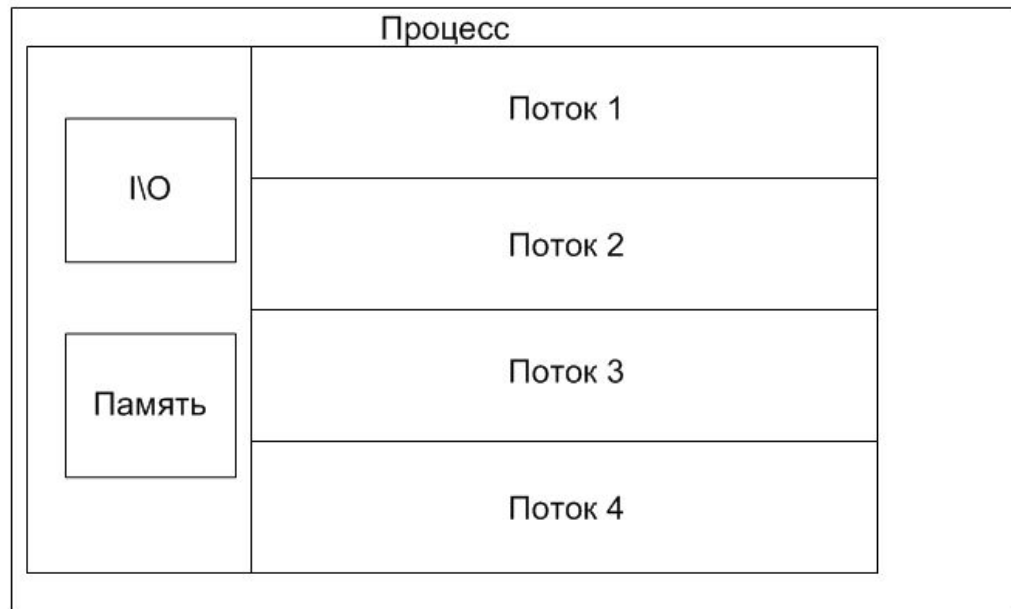
- ➔ Создание потока происходит проще и быстрее, чем процесса
- ➔ Обмен данными между потоками проще, чем между процессами

Одно- и многопоточное приложение

*Однопоточное
приложение*



*Многопоточное
приложение*



Просмотр потоков

TID (Thread ID) —
идентификатор потока

ps m — просмотр потоков:

- PID — процесс
- «дефис» — поток процесса

ps -eLf -просмотр потоков:

- LWP — id потока (TID)
- NLWP — количество потоков

```
user@user-VirtualBox:~$ ps m
```

PID	TTY	STAT	TIME	COMMAND
1252	tty1	-	0:00	/usr/lib/gdm3/gdm-x-s
-	-	Ssl+	0:00	-
-	-	Ssl+	0:00	-
-	-	Ssl+	0:00	-
1254	tty1	-	0:00	/usr/lib/xorg/Xorg vt
-	-	Sl+	0:00	-
-	-	Sl+	0:00	-
1263	tty1	-	0:00	/usr/lib/gnome-sessio
-	-	Sl+	0:00	-
-	-	Sl+	0:00	-
-	-	Sl+	0:00	-
-	-	Sl+	0:00	-
-	-	Sl+	0:00	-
1411	tty1	-	0:05	/usr/bin/gnome-shell

LWP и tasks

Изначально в Linux потоки (threads) создавались «**внутри**» процесса **пользователя** и не были видны ядру ОС.

Сейчас потоки представляются как LWP (light weight process) и **видны ядру как отдельные процессы**, но разделяющие одно адресное пространство.

При этом на уровне ядра процесс будет называться задачей (task).

Многопоточный процесс будет содержать несколько задач.



Итоги

Итоги

Сегодня мы рассмотрели процессы в ОС Linux:

- что такое процессы и потоки в Linux;
- как использовать утилиты для работы с потоками;
- сигналы и работу с ними;
- работу с процессами из командной оболочки.