

Управление ресурсами (ps, top, ps aux, pid, удалить процесс)

Лесников Юрий, seagest

1 Процессы

Определение 1.1. *Процесс — это запущенная программа.*

Каждый процесс имеет:

- Уникальный числовой идентификатор — **PID (Process ID)**
- Состояние выполнения (running, sleeping и др.)
- Ассоциированную память, файловые дескрипторы, контекст
- Владельца (пользователя)
- Родительский процесс (**PPID**)

Замечание 1.1. *После завершения процесса, его PID может быть повторно использован ядром для нового процесса.*

2 PID (Process Identifier)

Определение 2.1. *PID — целое положительное число, уникальное в момент времени.*

Особые PID:

- **PID 1** — корневой процесс (обычно **systemd** или **init**). Он не может быть убит обычными средствами и становится родителем «осиротевших» процессов.
- **PID 0** — ядро (idle/swapper), недоступен в userspace.

3 Просмотр процессов

3.1 Команда ps

Определение 3.1. *ps — утилита для получения снимка текущих процессов.*

Таблица 1: Способы вызова команды **ps** для просмотра процессов

Команда	Стиль синтаксиса	Описание
ps	—	Показывает только процессы, запущенные из текущего терминала (сессии) и ассоциированные с ним.
ps aux	BSD-стиль	Отображает все процессы в системе в удобочитаемом формате: <ul style="list-style-type: none">• a — процессы всех пользователей• u — подробный вывод с информацией о пользователе и памяти• x — включая процессы без управляющего терминала
ps axl	BSD-стиль	Вывод всех процессов в длинном (long) формате , включая дополнительные технические поля.

3.2 Разбор полей ps aux

Таблица 2: Поля вывода команды ps aux

Поле	Значение
USER	Имя пользователя, запустившего процесс
PID	Идентификатор процесса
%CPU	Процент использования CPU за всё время жизни процесса
%MEM	Процент использования оперативной памяти (RAM)
VSZ	Виртуальный размер памяти (в КБ)
RSS	<i>Resident Set Size</i> — объём физической памяти, используемой процессом (в КБ, без swp)
TTY	Терминальное устройство, к которому привязан процесс (? — фоновый/без терминала)
STAT	Состояние процесса
START	Время запуска (если > 24 ч — показывается дата)
TIME	Совокупное процессорное время, затраченное на выполнение инструкций (не wall-clock time!)
COMMAND	Команда с аргументами (может отличаться от введённой из-за расширения шаблонов в оболочке)

3.3 Состояния процесса (STAT)

- **R** — running (выполняется)
- **S** — sleeping (приостановлен, ожидает события)
- **Z** — zombie (завершён, но не "собран" родителем)
- **T** — stopped (например, **SIGSTOP**)

3.4 Команда top

Определение 3.2. *top* — интерактивный монитор процессов в реальном времени.

Основные опции запуска (top -h):

- **-b** — пакетный режим (для скриптов)
- **-d N** — интервал обновления (сек)
- **-p PID** — отслеживать только указанный процесс
- **-u user** — показывать только процессы пользователя
- **-H** — отображать потоки

Таблица 3: Интерактивные команды утилиты top

Клавиша	Действие
k	Завершить процесс (ввести PID и сигнал)
q или Esc	Выход
H	Переключить отображение потоков
c	Показать/скрыть полный путь к команде
u	Фильтр по пользователю
P / M	Сортировка по CPU / памяти
z	Цветной/монохромный режим

4 Потоки

Определение 4.1. Поток — наименьшая единица выполнения в рамках процесса.

- У потока есть **ID (Thread ID)**
- Потоки делят **память и ресурсы** родительского процесса
- В выводе **ps** потоки отображаются со знаком «дефис» в колонке **PID**

Листинг 1: Две команды для отображения потоков

```
1 ps -eLf
2 top -H
```

5 Завершение процесса

5.1 Сигналы

Замечание 5.1. Процессы завершаются посредством сигналов.

Таблица 4: Основные сигналы завершения процессов

Сигнал	Номер	Действие
SIGTERM	15	Корректное завершение (можно перехватить)
SIGKILL	9	Принудительное убийство (нельзя игнорировать)

Листинг 2: Завершение процессов

```
1 kill -9 1234
2 kill -15 1234
3 kill 1234
```

Замечание 5.2. Если не указать номер сигнала (3 строчка блока кода 2), то будет использован *SIGTERM*.

6 Приоритеты процессов (nice/renice)

Замечание 6.1. Диапазон *nice*: от **-20** (высший приоритет), до **+19** (низший приоритет).

Листинг 3: Запуск с приоритетом

```
1 nice -n 19 ./intensivescript
```

Листинг 4: Изменение приоритета у запущенного процесса

```
1 renice -n 10 1234
```

Замечание 6.2. Только *root* может назначать *nice* < 0.

7 Дополнительные механизмы управления ресурсами

7.1 cgroups (control groups)

Определение 7.1. *cgroups* — механизм ядра *Linux* для ограничения, приоритезации, изоляции и логирования ресурсов группы процессов.

- **cgroups v1**: процесс может быть в разных группах по разным контроллерам (cpu, memory)
- **cgroups v2**: процесс принадлежит только одной группе

7.2 namespaces

Определение 7.2. *Linux namespace — это абстракция над ресурсами в операционной системе.*

Замечание 7.1. *Мы можем думать об namespace, как о ящике. В этом ящике находятся системные ресурсы, которые точно зависят от типа ящика (namespace). В настоящее время существует семь типов пространств имён (namespaces): Cgroups, IPC, Network, Mount, PID, User, UTS.*