**Слайд 2**

**1. Определение процесса в Linux:**

Процесс в Linux представляет собой исполняемую программу, которая выполняет определенную задачу. Каждый процесс имеет уникальный идентификатор (PID), который используется для идентификации и управления процессами в системе. Дополнительно, каждый процесс имеет свое состояние (running, sleeping, stopped и т. д.), приоритет выполнения и другие характеристики.

**Слайд 3**

Типы процессов

В Linux процессы могут быть двух типов:

* Процессы переднего плана — зависят от ввода пользователя. Они также называются интерактивными процессами.
* Фоновые процессы — работают независимо от пользователя. Называются неинтерактивными или автоматическими процессами.

**Слайд 4**

Процесс в Linux после его создания и до завершения может проходить через разные состояния, а именно:

* Running (выполняется) — процесс в состоянии выполнения, то есть он запущен или готов к запуску.
* Sleeping (спящий) — процесс находится в спящем состоянии, пока ожидает доступного ресурса:
* Interruptible sleep (прерывистый сон) — процесс в состоянии прерывистого сна может просыпаться для обработки сигналов.
* Uninterruptible sleep (Непрерывный сон) — процесс не будет просыпаться для обработки сигналов.
* Stopped (остановлен) — процесс переходит в это состояние, когда получает сигнал для остановки.
* Zombie — проще говоря, “мертвый” процесс, запись о котором все еще присутствует в таблице.

**Слайд 5**

**2. Запуск процесса:**

Процессы могут быть запущены из командной строки или из других процессов. Команда fork() создает точную копию текущего процесса, создавая дочерний процесс. Затем, с использованием exec(), новый исполняемый файл загружается в дочерний процесс, заменяя текущий код процесса.

Когда вызывается функция fork(), процесс создает точную копию самого себя. Эта копия является дочерним процессом, а оригинальный процесс остается родительским. Таким образом, мы теперь имеем два процесса, каждый со своим уникальным идентификатором процесса (PID).

После создания дочернего процесса часто требуется загрузить в него новый исполняемый файл. Для этого используется функция exec(). Она полностью заменяет текущий образ процесса новым, загружая указанный исполняемый файл.

**Слайд 6**

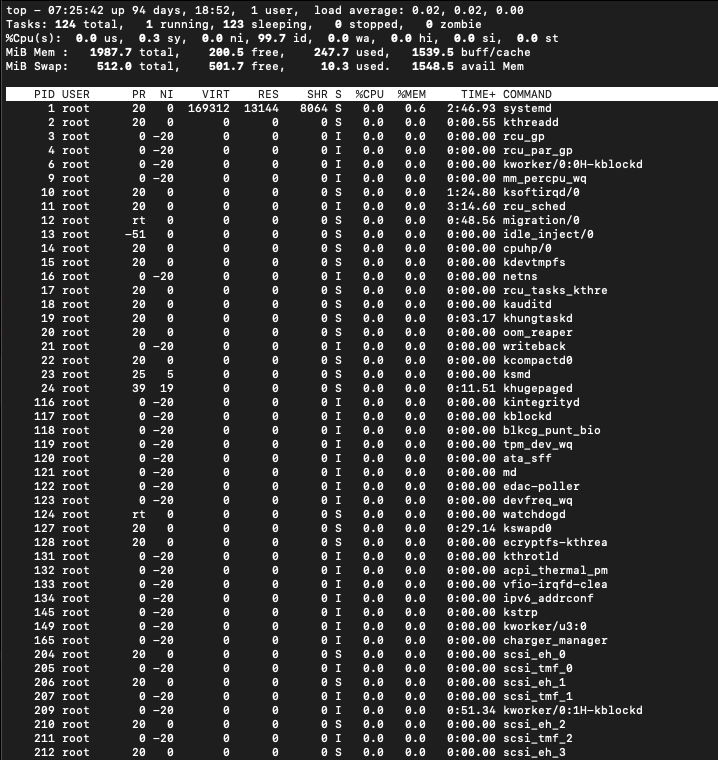
**3. Мониторинг и отладка процессов:**

В Linux доступны две команды для отслеживания запущенных процессов — top и ps.

1: Команда top

Для отслеживания запущенных процессов вы можете выполнить команду top:

$ top



Эта команда выводит список процессов, которые работают в реальном времени, а также сколько они занимают памяти и CPU. Давайте подробнее разберем ее вывод:

* PID: уникальный идентификатор процесса, присвоенный каждому процессу.
* User: имя владельца процесса.
* PR: приоритет, присвоенный процессу при планировании.
* NI: значение ‘nice’ процесса.
* VIRT: объем виртуальной памяти, которую занимает процесс.
* RES: объем физической памяти, используемой процессом.
* SHR: объем памяти, разделяемой с другими процессами.
* S: состояние процесса:
  + “D” = непрерывный сон
  + “R” = запущен
  + “S” = спящий
  + “T” = отслежен или остановлен
  + “Z” = зомби
* %CPU: процент CPU, используемого процессом.
* %МЕМ: процент RAM, которую занимает процесс.
* TIME+: общее процессорное время, потребляемое процессом.
* Command: команда для активации процесса.

Перемещаться вверх и вниз по списку можно клавишами со стрелками вверх/вниз. Для выхода из команды top нажмите q. Чтобы завершить процесс, выделите его клавишами со стрелками и нажмите ‘k’.

**Слайд 7**

2: Команда ps

ps — это сокращение от “Process Status”. Она выводит запущенные в данный момент процессы. Но, в отличие от команды top, результат не генерируется в реальном времени.

$ ps



Рассмотрим ее вывод подробнее:

* PID — идентификатор процесса.
* TTY — тип терминала.
* TIME — общее время работы процесса.
* CMD — имя команды, которая запускает процесс.

Для вывода дополнительной информации с помощью команды ps необходимо добавить параметр -u:

$ ps -u

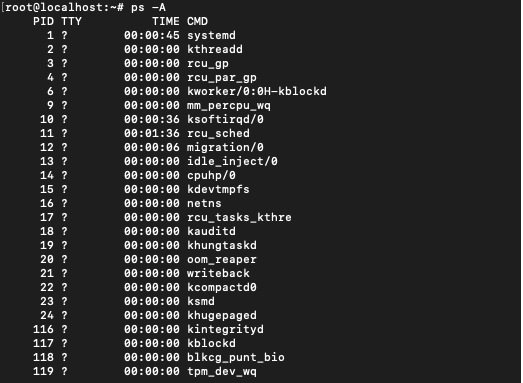


На скриншоте вы можете увидеть следующие данные:

* %CPU — количество вычислительной мощности, которую занимает процесс.
* %MEM — объем памяти, занимаемой процессом.
* STAT — состояние процесса.

Хотя команда ps выводит только запущенные в данный момент процессы, с помощью нее можно посмотреть список всех процессов с помощью команды:

$ ps -A



Эта команда выводит список даже тех процессов, которые в данный момент не запущены.

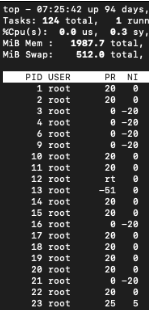
При отладке проблемных процессов часто используется инструмент gdb, предоставляющий мощные средства анализа и коррекции ошибок.

**Слайд 8**

**4. Управление приоритетом:**

Приоритет процесса определяет, насколько часто процесс получает процессорное время. Команда nice позволяет установить приоритет процесса. Отрицательные значения увеличивают приоритет, а положительные уменьшают. Значение приоритета для процесса называется ‘Niceness’, оно может варьироваться от -20 до 19. 0 — значение по умолчанию.

Четвертый столбец в выводе команды top — это столбец для значения niceness.



Чтобы запустить процесс и присвоить ему значение nice, которое отличается от значения по умолчанию, необходимо выполнить команду:

$ nice -n [value] [process name]

Для изменения значения nice уже запущенного процесса необходимо выполнить команду:

renice [value] -p 'PID'

**Слайд 9**

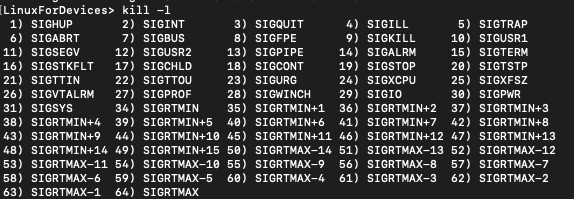
**5. Сигналы:**

Сигналы представляют собой механизм взаимодействия с процессами. Команда kill отправляет сигнал процессу, что может изменить его поведение. Например, сигнал SIGKILL немедленно завершает выполнение процесса.

Есть разные типы сигналов, которые можно отправлять. Однако самый частый — “kill -9”, который является “SIGKILL”.

Можно посмотреть все доступные сигналы с помощью следующей команды:

$ kill -L



Сигнал по умолчанию — 15, то есть SIGTERM. Это означает, что, если будет выполнена команда kill без номера, она отправляет сигнал SIGTERM.

Синтаксис для остановки процесса:

$ kill [pid]

В качестве альтернативы также можно использовать:

$ kill -9 [pid]

Эта команда отправит процессу сигнал “SIGKILL”. Ее следует выполнять в случае, если процесс игнорирует обычный запрос на остановку.

**Слайд 10**

**6. Управление фоновыми процессами и работа с заданиями:**

Добавление символа "&" в конце команды позволяет запустить процесс в фоновом режиме, позволяя пользователю продолжать работу в терминале. Команды bg и fg используются для управления фоновыми и передними процессами, что полезно при работе с несколькими заданиями одновременно.

Управление процессами в Linux можно применять и для того, чтобы в терминале можно было работать также, на заднем и переднем фоне. Напирмер, при чтении справки (**man**) возникла необходимость исполнить какую-то команду. Чтобы не закрывать **man**, можно нажать **Ctrl+Z** и приостановить **man**. При этом работа **man** не просто приостанавливается, она уходит на задний фон.

Далее можно поработать в терминале. А затем, когда возникнет необходимость, можно вернуть **man** на передний фон. Чтобы увидеть процессы на заднем фоне используется команда **jobs**:

$ jobs

[1**]**+ Остановлен man 7 signal

Разберём вывод: **[1]** — номер задания, **+** означает что это последнее приостановленное задание, а дальше идёт состояние и название процесса.

Чтобы вернуть задание на передний фон, нужно выполнить

$ fg %<номер задания>

Или можно выполнить **fg** без указания номера задания, тогда на передний план вернётся задание, которое было помечено плюсиком в выводе **jobs**.

$ fg %1

После выполнения последней команды, на передний план вернется работа **man**.

**Группа переднего фона** — это терминал и то что на нём сейчас выполняется. Все остальные группы — это **группы заднего фона**.

На заднем фоне процесс может быть не только в приостановленном состоянии. Если выполнение процесса не требует от пользователя каких-то интерактивных действий (никакого ввода или вывода), то можно заставить процесс выполнятся на заднем фоне. Для этого нужно выполнить

$ bg %<номер задания>

Например, начнём скачивание большого файла с помощью команды **wget**:

$ wget https://releases.ubuntu.com/22.04.1/ubuntu-22.04.1-desktop-amd64.iso

Затем переведём процесс скачивания на задний фон, нажмём **Ctrl+Z**. И посмотрим список заданий на заднем фоне:

$ jobs

[1**]**- Остановлен man 7 signal

[2**]**+ Остановлен wget https://releases.ubuntu.com/22.04.1/ubuntu-22.04.1-desktop-amd64.iso

Теперь, с помощью команды **bg**, запустим процесс на заднем фоне:

$ bg %2

[2**]**+ wget https://releases.ubuntu.com/22.04.1/ubuntu-22.04.1-desktop-amd64.iso &

Вывод перенаправляется в «wget-log».

**Wget** должен выводить информацию о скачивании файла, но на заднем фоне запрещено выводить информацию на терминал. Именно поэтому весь вывод автоматически перенаправляется в файл — **wget-log**.