Министерство образования и науки Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего образования

«Пермский национальный исследовательский политехнический университет»

Электротехнический факультет

Кафедра «Информационные технологии и автоматизированные системы»

Направление 09.03.04 – «Программная инженерия»

Дисциплина: «Администрирование информационных систем»

Профиль: «Разработка программно-информационных систем»

Семестр 5

ОТЧЕТ

по лабораторной работе №5

Тема: «Знакомство с процессами»

Выполнила: студент группы РИС-22-1б

Поважный В.Е. \_\_\_\_\_\_\_\_\_\_\_

Проверил: ст. преподаватель кафедры ИТАС

Шереметьев В. Г. \_\_\_\_\_\_\_\_\_\_\_

Дата \_\_\_\_\_\_

Пермь, 2024

**Цель работы**

Познакомиться с понятием процесса. Научиться получать список имеющихся в системе процессов и управлять их состоянием.

# Теоретические сведения

**Понятие процесса**

Операционная система Linux является *многозадачной (мультизадачной)*. Это значит, что одновременно в системе может присутствовать множество процессов, каждому из которых доступно определенное количество процессорного времени. Для пользователя создается «иллюзия» одновременного выполнения процессов.

***Процесс*** – выполняемая программа с еѐ данными и контекстом.

Каждый процесс имеет уникальный в любой момент времени номер в системе – PID. Первый запускающийся в системе процесс init, имеет pid = 1.

Для описания процессов в операционной системе имеется список структур – *дескрипторов*, содержащих информацию об идентификаторе процесса, приоритете, состоянии процесса, информацию о принадлежности пользователю и группе, занимаемых процессом ресурсах и др.

Каждый процесс в системе Linux запускается каким-либо процессом. Запускающий процесс – *родительский*, новый процесс - *дочерний*. Процессы, выполняющие одну задачу, объединяются в группы, имеющие собственный идентификатор. Процесс внутри группы, идентификатор которого совпадает с идентификатором группы процессов, считается *лидером группы процессов*.

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.

# Дополнительные утилиты

Для получения информации о запущенных процессах часто используется команда **ps**. Вывод запущенной без аргументов команды содержит: информацию о процессах текущего пользователя и ассоциированных с текущим терминалом, процессорное время, занятое этим процессом, и имя исполняемого файла. Управлять форматом вывода можно с помощью дополнительных опций (Таблица 1).

# $ ps

Таблица 1. Примеры опций команды ps

|  |  |
| --- | --- |
| **Ключ** | **Описание** |
| -a | Выдать все процессы системы, включая лидеров сеансов. |
| -d | Выдать все процессы системы, исключая лидеров сеансов. |
| -e | Выдать все процессы системы. |
| -x | Выдать процессы системы, не имеющие контрольного терминала. |
| -o | Определяет формат вывода в виде списка полей, разделенных символом **«,»**. |
| -u | Выдать процессы, принадлежащие указанному пользователю. |

Например, можно получить выборочную информацию обо всех процессах в системе:

# $ ps -eo s,pid,tty,command

Альтернативным способом узнать о состоянии процессов в реальном времени является использование команды **top**. Выводом команды можно управлять с помощью специальных комбинаций клавиш. Справочную информацию можно получить, нажав клавишу «h».

Чтобы получить информацию о запущенных в системе процессах в виде дерева, можно использовать утилиту **pstree**.

# 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 командa **fg** с указанием номера задания из списка заданий.

$ **fg %2**

Если мы хотим перевести процесс в сосотояние 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

Еще одним способом управлять выполнением процессов является использование утилиты **kill**. Данная команды позволяет послать определенный сигнал процессу. Возможно завершение процесса как по имени, так и по номеру задания или по идентификатору PID процесса.

# $kill -SIGINT 124672

Получить список сигналов можно с помощью опции **-l.**

# $ kill -l

Послать сигнал нескольким процессам можно с помощью команды

# killall.

**$ kilall gedit**

Приведенная выше команда завершит все процессы текущего пользователя с именем **gedit**. По умолчанию команда отправляет сигнал **TERM** (software termination signal).

Таблица 2. Примеры команд

|  |  |
| --- | --- |
| **Команда** | **Описание** |
| ps | Выдать снимок имеющихся в системе процессов. |
| top | Выдать информацию о процессах системы в интерактивной форме. |
| pstree | Вывод дерева процессов в системе |
| jobs | Выдать список заданий в текущей сессии оболочки. |
| fg | Переместить задание из текущей сессии на передний план. |
| bg | Переместить задание из текущей сессии в фон. |
| kill | Завершить процесс, отправить сигнал процессу. |
| killall | Завершить группу процессов, отправить сигнал группе процессов. |

# Порядок выполнения лабораторной работы

1. Создайте файл **proc1**, содержащий список процессов пользователя **root**, отсортированный по идентификатору родительского процесса. Используйте команду **ps** и изученные ранее утилиты.
2. Получите информацию о процессах вашего пользователя, имеющих статус *работоспособный*.
3. Добавьте к файлу **proc1** сведения о процессе в данный момент потребляющий больший процесс ресурсов центрального процессора.
4. Запустите утилиту **top**. Изучите содержимое информационных полей, предоставляемых утилитой. Получите информацию о степени использования ресурсов системы, количестве пользователей, времени работы системы.
5. Ознакомьтесь со справочной информацией утилиты **top**. Изучите действия, выполняемые по нажатию на клавиши, **«u», «f», «R», «d»,**

**«L»**. Настройте вывод утилиты top в соответствии с заданием 3, с периодом обновления 1с.

1. В окне терминала запустите программу **firefox**. Используя команду **pstree** получите информацию о дереве процессов и их идентификаторах. Найдите поддерево для процесса **firefox**, изучите список составляющих его процессов.
2. В новом окне терминала запустите программу **firefox**. Затем два раза запустите утилиту **gedit** фоновом режиме.
3. Получите список заданий текущей сессии терминала.
4. Используя команды **fg** и **bg** и сочетания клавиш **«Ctrl + z»** и **«Ctrl + с»**

научитесь перемещать задачи из фона на передний план и наоборот.

1. Получите список сигналов для команды **kill**. Завершите запущенные процессы с помощью команды kill с помощью сигналов SIGKILL и SIGTERM.
2. Выполните команду **killall -u имя\_текущего\_пользователя**.

**Ход работы**

1. Создание файла proc1 с информацией о процессах пользователя root, отсортированных по идентификатору родительского процесса:

* ps -u root — выводит список процессов, принадлежащих пользователю root.
* --sort=ppid — сортирует процессы по идентификатору родительского процесса (PPID).

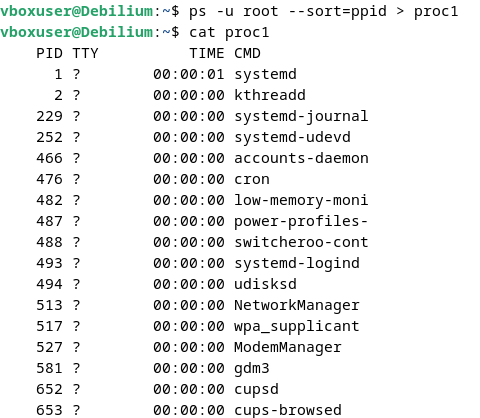


Рис. 1 – вывод списка процессов

1. Получение информации о процессах текущего пользователя, имеющих статус "работоспособный" (статус R):

* -o pid,comm,stat — указывает формат вывода: pid (идентификатор процесса), comm (имя команды), stat (статус процесса).
* grep "R" — фильтрует процессы, отображая только те, у которых статус "R" (работоспособный, running).



Рис. 2 – процессы со статусом работоспособный

1. Добавление к файлу proc1 сведений о процессе, потребляющем наибольшее количество ресурсов CPU:

* ps -eo pid,comm,%cpu — выводит список всех процессов с полями:
  + pid — идентификатор процесса,
  + comm — команда (имя процесса),
  + %cpu — процент использования процессора.
* --sort=-%cpu — сортирует список по убыванию использования процессора (процессы, потребляющие больше CPU, будут сверху).
* head -n 2 — выводит первые 2 строки (заголовок таблицы и процесс с наибольшей загрузкой CPU).

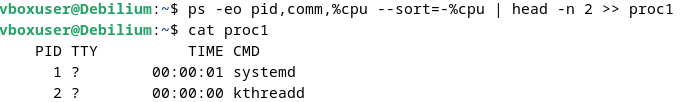




Рис. 3 – добавление информации в файл proc1

1. Запуск утилиты top для изучения информации о системе:
2. В поле верхней строки top можно увидеть:
3. Степень использования ресурсов: загрузка процессора (CPU usage), использование памяти (Memory usage).
4. Количество пользователей: информация о количестве активных сессий.
5. Время работы системы: Uptime.

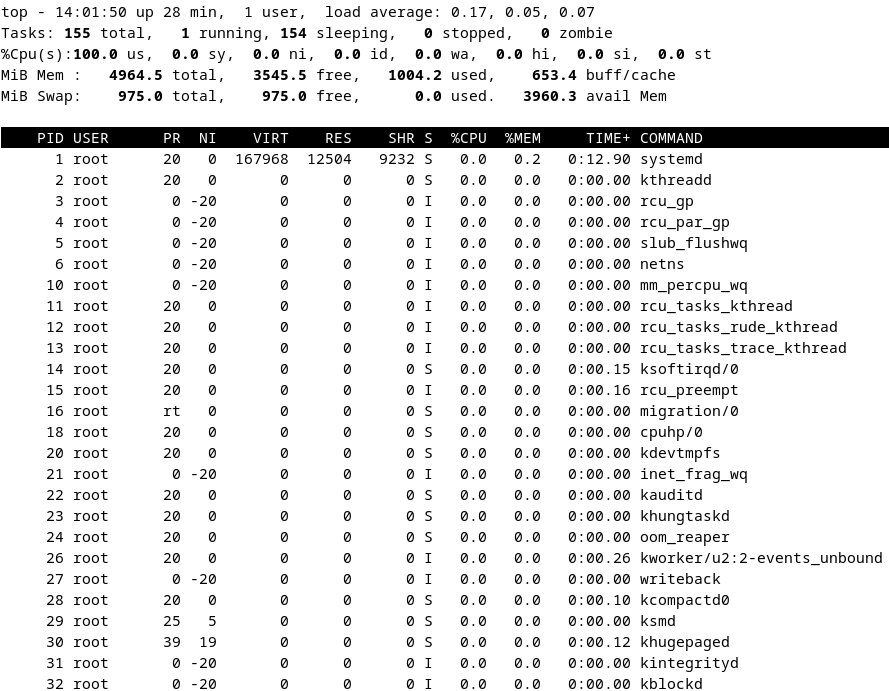


Рис. 4 – утилита top

1. Настройка вывода top и изучение клавиш:

* Нажатие клавиши u: отобразить процессы только конкретного пользователя.
* Нажатие клавиши f: настройка полей для отображения.
* Нажатие клавиши R: изменить порядок сортировки (реверсировать).
* Нажатие клавиши d: изменить интервал обновления (например, 1 секунда).
* Нажатие клавиши L: отображение и скрытие системной загрузки.

Для настройки обновления каждые 1 секунду:

**top -d 1**

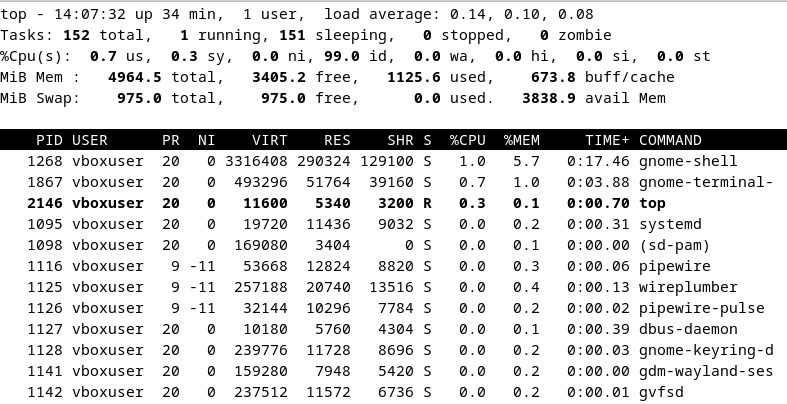


Рис. 5 – вывод информации пользователя vboxuser

1. Запуск программы firefox и получение информации о дереве процессов:

В одном окне терминала:

**firefox &**

В другом окне терминала для просмотра дерева процессов:

**pstree -p | grep firefox**

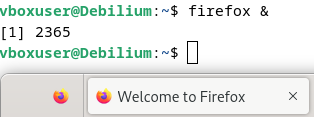


Рис. 6 – запуск браузера firefox

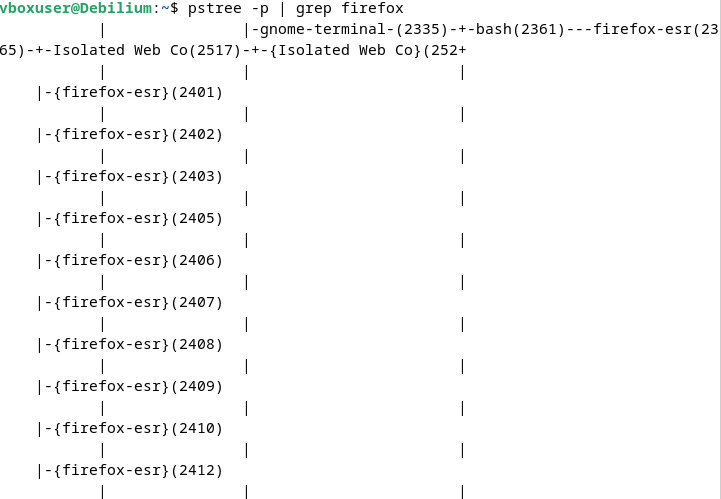


Рис. 7 – дерево процессов

1. Запуск Firefox и двух экземпляров gedit в фоновом режиме:

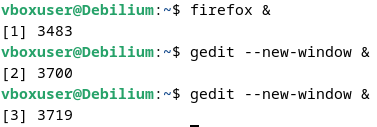


Рис. 8 – запуск нескольких процессов

1. Получение списка заданий текущей сессии терминала:

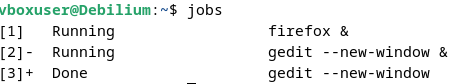


Рис. 9 – вывод списка заданий

1. Перемещение задач между фоном и передним планом:

Для перевода процесса на передний план:

**fg %<номер задания>**

Для перевода на задний план:

**bg %<номер задания>**

Использование сочетания клавиш:

* Ctrl + z: приостановить процесс и отправить его в фон.
* Ctrl + c: завершить текущий процесс.

1. Список сигналов команды **kill** и завершение процессов:

Получить список сигналов:

**Kill –l**

Завершение процесса с помощью сигналов SIGKILL и SIGTERM:

**kill -SIGKILL <pid>**

**kill -SIGTERM <pid>**

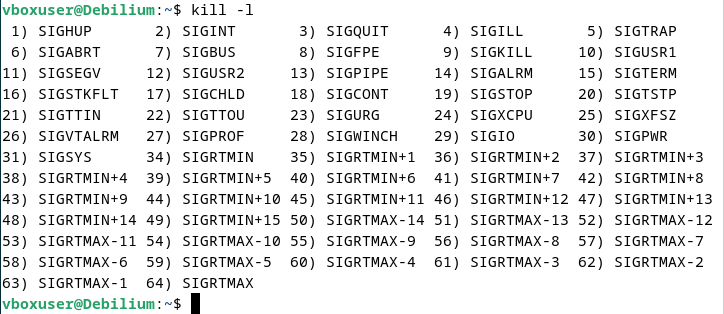
****

Рис. 10 – список сигналов.

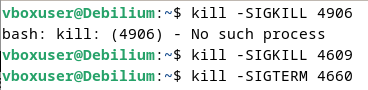


Рис. 11 – закрытие процессов libre office и firefox

1. Завершение всех процессов текущего пользователя с помощью killall:

****

Рис. 12 – killall

**Ответы на контрольные вопросы**

**1. Способы получения информации о процессах в системе:**

* **Команда ps**: отображает информацию о текущих процессах. Можно использовать различные опции для фильтрации и сортировки данных (например, ps aux, ps -ef, ps -u <user>).
* **Команда top**: показывает динамическое обновление списка процессов с информацией о загрузке CPU, памяти и другие метрики.
* **Команда htop**: улучшенная версия top с удобным интерфейсом и возможностью управления процессами (может потребоваться установка).
* **Команда pgrep**: позволяет искать процессы по имени.
* **Команда pstree**: отображает процессы в виде дерева, показывая иерархию процессов.
* **Команда vmstat**: показывает статистику по памяти, процессам и другим системным ресурсам.
* **Команда systemctl**: используется для управления службами и отображения информации о них в системах с systemd.

**2. Управление выводом утилиты top:**

* Нажатие клавиши h: отображает справочную информацию о управлении top.
* Нажатие клавиши u: позволяет отобразить процессы только конкретного пользователя.
* Нажатие клавиши f: открывает меню для настройки отображаемых полей.
* Нажатие клавиши R: изменяет порядок сортировки процессов.
* Нажатие клавиши d: позволяет изменить интервал обновления (например, на 1 секунду).
* Нажатие клавиши k: позволяет завершить процесс по его идентификатору (PID).

**3. Сигналы, отправляемые сочетаниями клавиш «Ctrl + z» и «Ctrl + c»:**

* **Ctrl + z**: приостанавливает текущий процесс и отправляет его в фоновый режим (сигнал SIGTSTP).
* **Ctrl + c**: отправляет сигнал SIGINT, который завершает выполнение текущего процесса.

**4. Какой процесс в системе запускается первым?**

Первым процессом в системе является **init** (или его современный аналог **systemd** в системах, использующих systemd). Он запускается ядром операционной системы во время загрузки и отвечает за инициализацию всех других процессов и служб.

**5. Какие типы процессов вы знаете?**

* **Фоновый процесс**: работает в фоновом режиме и не требует взаимодействия с пользователем.
* **Передний план процесс**: запущен в интерактивном режиме и может взаимодействовать с пользователем через терминал.
* **Системный процесс**: выполняется системой, часто с привилегиями администратора (например, процессы, связанные с systemd).
* **Демоны**: специальные фоновый процессы, которые запускаются при загрузке системы и работают в фоновом режиме, ожидая событий или запросов.

**6. Что такое дескриптор процесса?**

**Дескриптор процесса (PID)** — это уникальный идентификатор, присваиваемый каждому запущенному процессу в системе. Он используется операционной системой для отслеживания и управления процессами, позволяет выполнять операции над ними, такие как завершение, приостановка или изменение приоритетов. Дескрипторы процессов хранятся в таблице процессов и обычно имеют числовое значение.