Отчёт по лабораторной работе №6

Дисциплина: Основы администрирования операционных систем

Верниковская Екатерина Андреевна

Содержание

# 1 Цель работы

Получить навыки управления процессами операционной системы.

# 2 Задание

1. Продемонстрировать навыки управления заданиями операционной системы
2. Продемонстрировать навыки управления процессами операционной системы
3. Выполнить задания для самостоятельной работы

# 3 Выполнение лабораторной работы

## 3.1 Управление заданиями

В консоли переходим в режим работы суперпользователя, используя *su -* (рис. 1)

Режим суперпользователя

Рис. 1: Режим суперпользователя

Вводим следующие команды: - *sleep 3600 &* - *dd if=/dev/zero of=/dev/null &* - *sleep 7200*

Так как последнюю команду мы запустили без & (т.е не в фоновом режиме), то у нас есть 2 часа, прежде чем мы снова получим контроль над оболочкой. Чтобы остановить процесс, мы вводим *ctrl+z* (рис. 2)

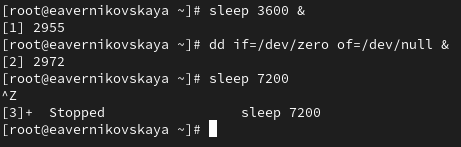


Рис. 2: Запуск нужных команд и остановка последней

Введя *jobs* мы увидим 3 задания, которые только что запустили. Первые два имеют состояние Running, а последнее задание в настоящее время находится в состоянии Stopped (так как до этого мы его остановили) (рис. 3)

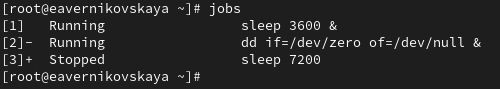


Рис. 3: Команда jobs (1)

Вводим команду *bg 3*, чтобы продолжить выполнение 3-его задания в фоновом режиме (рис. 4)

Продолжение выполнения задания 3 в фоновом режиме

Рис. 4: Продолжение выполнения задания 3 в фоновом режиме

С помощью команды *jobs* смотрим изменения в статусе заданий (рис. 5)

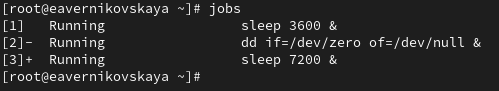


Рис. 5: Команда jobs (2)

Вводим команду *fg 1* для перемещения задания 1 на передний план. А после вводим *ctrl+c*, чтобы отменить 1-ое задание (рис. 6)

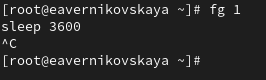


Рис. 6: Перемещение задания 1 на передний план и его отмена

Снова вводим команду *jobs*, чтобы посмотреть изменения в статусе заданий (рис. 7)

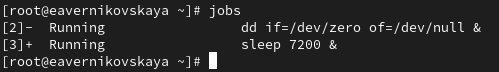


Рис. 7: Команда jobs (3)

Тоже самое делаем для отмены заданий 2 и 3. Сначала с помощью *fg* перемещаем задания на передний план, а далее отменяем их с помощью *ctrl+c* (рис. 8)

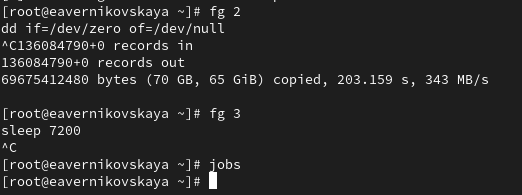


Рис. 8: Отмена задания 2 и 3

ОТкрываем второй терминал и под учётной записью своего пользователя вводим в нём: *dd if=/dev/zero of=/dev/null &*. Далее закрываем второй терминал (рис. 9), (рис. 10)

Команда dd if=/dev/zero of=/dev/null & (1)

Рис. 9: Команда dd if=/dev/zero of=/dev/null & (1)

Закрытие второго терминала

Рис. 10: Закрытие второго терминала

Снова открываем второй терминал и вводим *top* (команда, которая позволяет пользователям отслеживать процессы и использование системных ресурсов в Linux). Мы увидим что задание dd всё ещё запущено (рис. 11), (рис. 12)

Команда top

Рис. 11: Команда top

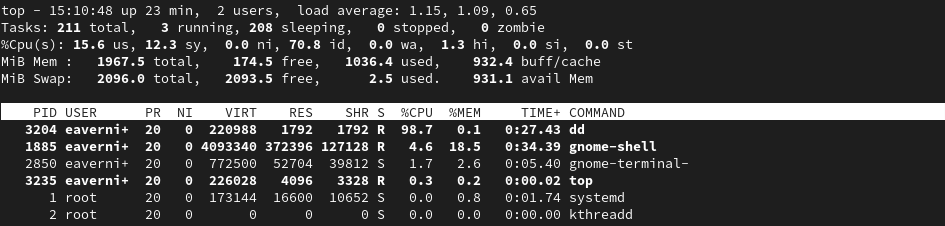


Рис. 12: Вывод команды top

Вводим *k*, чтобы убить задание dd (рис. 13), (рис. 14)

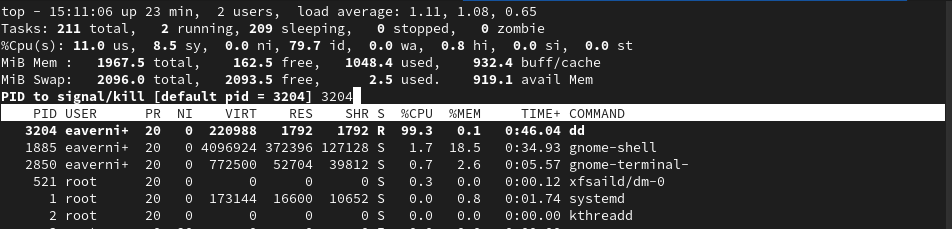


Рис. 13: Убийство задания dd

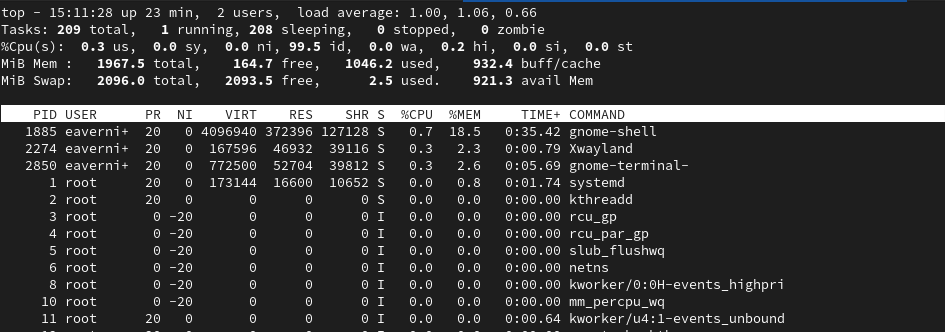


Рис. 14: Убитое задание dd

## 3.2 Управление процессами

Снова получаем права пользователя root и вводим три раза команду *dd if=/dev/zero of=/dev/null &* (рис. 15)

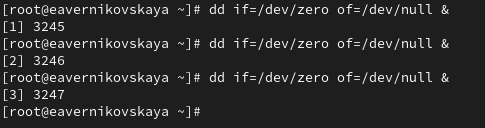


Рис. 15: Команда dd if=/dev/zero of=/dev/null & (2)

Далее вводим *ps aux | grep dd*. Эта команда показывает все строки, в которых есть буквы dd. Запущенные процессы dd идут последними (рис. 16)



Рис. 16: Команда ps aux | grep dd

Далее используем (идентификатор процесса в системе) первого процесса dd, чтобы изменить приоритет. 1-ого процесса равен 3245. Для изменения приоритета используется команда *renice -n 5*  (рис. 17)

Меняем приоритет 1-ого процесса на 5

Рис. 17: Меняем приоритет 1-ого процесса на 5

Далее вводим *ps fax | grep -B5 dd*. Параметр -B5 показывает соответствующие запросу строки, включая пять строк до этого. Поскольку ps fax показывает иерархию отношений между процессами, мы также можем увидеть оболочку, из которой были запущены все процессы dd, и её PID (рис. 18)

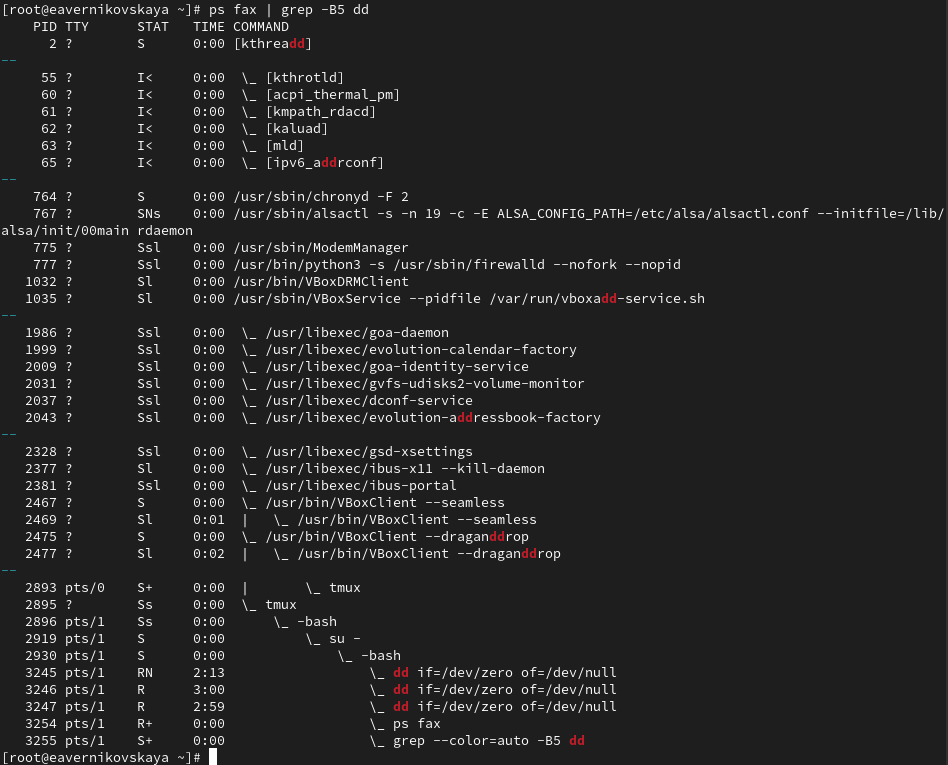


Рис. 18: Команда ps fax | grep -B5 dd

Находим PID корневой оболочки, из которой были запущены процессы dd (это 2919), после вводим *kill -9* , заменив в на значение PID оболочки (т.е на 2919). После ввода этой команды мы увидем, что наша корневая оболочка закрылась, а вместе с ней и все процессы dd. Остановка родительского процесса — простой и удобный способ остановить все его дочерние процессы (рис. 19)

Остановка родительского процесса

Рис. 19: Остановка родительского процесса

# 4 Самостоятельная работа

## 4.1 Задание 1

Запускаем команду *dd if=/dev/zero of=/dev/null* трижды как фоновое задание (рис. 20)

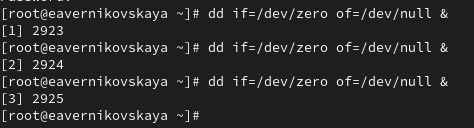


Рис. 20: Запуск нужной команды как фоновое задание

Увеличиваем приоритет одной из этих команд (например 1-ой), используя значение приоритета −5. После изменяем значение приоритета того же процесса ещё раз, но используя на этот раз значение -15.

Разница между -5 и -15:

* Приоритет -5: Увеличивает приоритет команды, делая ее более “важной” для системы. -5 запустит команду dd с более высоким приоритетом, чем обычный процесс, но не будет сильно влиять на другие приложения
* Приоритет -15: Значительно повышает приоритет, делая команду еще более “важной”. -15 запустит dd с очень высоким приоритетом. Это может привести к замедлению других процессов, так как система будет отдавать большую часть ресурсов dd

Чем ниже значение приоритета (более отрицательное), тем выше приоритет процесса (рис. 21)

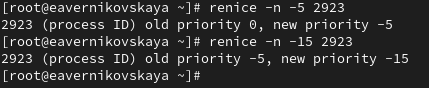


Рис. 21: Изменение приоритета 1-ого процесса

Завершаем все процессы dd, которые мы запустили командой *killall dd* (рис. 22)

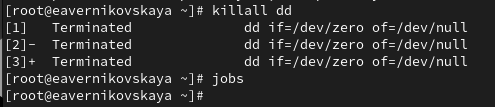


Рис. 22: Завершение всех процессов dd

## 4.2 Задание 2

Запускаем программу yes в фоновом режиме с подавлением потока вывода с помощью команды *yes > /dev/null &* (/dev/null отвечает за подавление потока вывода) (рис. 23)

Программа yes в фоновом режиме с подавлением потока вывода

Рис. 23: Программа yes в фоновом режиме с подавлением потока вывода

Далее запускаем программу yes на переднем плане с подавлением потока вывода. После приостанавливаем выполнение программы с помощью *ctrl+z* (рис. 24)

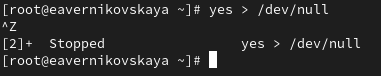


Рис. 24: Программа yes на переднем плане с подавлением потока вывода и её приостановка

Заново запускаем программу yes с теми же параметрами командой *fg 2*, затем завершаем её выполнение введя *ctrl+c* (рис. 25), (рис. 26)

Программа yes на переднем плане с подавлением потока вывода и её завершение

Рис. 25: Программа yes на переднем плане с подавлением потока вывода и её завершение

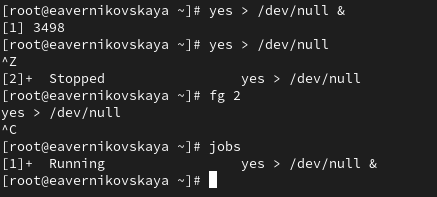


Рис. 26: Всё вместе

Запускаем программу yes на переднем плане без подавления потока вывода с помощью команды просто *yes*. После приостанавливаем выполнение программы с помощью *ctrl+z* (рис. 27), (рис. 28), (рис. 29)

Запуск программы yes на переднем плане без подавления потока вывода (1)

Рис. 27: Запуск программы yes на переднем плане без подавления потока вывода (1)

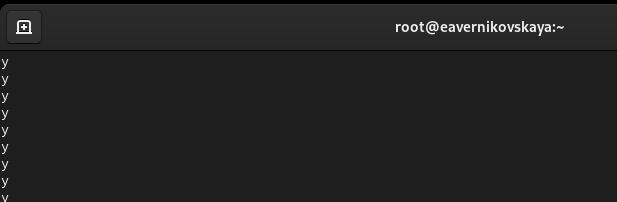


Рис. 28: Программа yes на переднем плане без подавления потока вывода

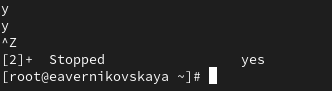


Рис. 29: Приостановка программы yes

Заново запускаем программу yes с теми же параметрами командой *fg 2*, затем завершаем её выполнение введя *ctrl+c* (рис. 30), (рис. 31)

Запуск программы yes на переднем плане без подавления потока вывода (2)

Рис. 30: Запуск программы yes на переднем плане без подавления потока вывода (2)

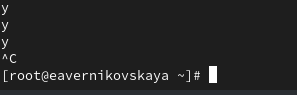


Рис. 31: Завершение программы yes

Проверяем состояния заданий, воспользовавшись командой *jobs* (рис. 32)

Команда jobs (4)

Рис. 32: Команда jobs (4)

Переводим процесс, который у нас выполняется в фоновом режиме, на передний план, и затем останавливаем его (комндой *fg1*, а затем *ctrl+z*) (рис. 33)

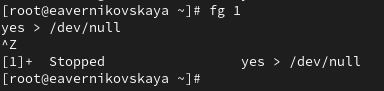


Рис. 33: Перевод процесса из фонового режима на передний план + его остановка

Переводим любой наш процесс (он у нас один) с подавлением потока вывода в фоновый режим (командой *bg1*) (рис. 34)

Перевод процесса с подавление потока в фоновый режим

Рис. 34: Перевод процесса с подавление потока в фоновый режим

Снова проверяем состояния заданий, воспользовавшись командой *jobs*. Процесс стал выполняющимся (Running) в фоновом режиме (рис. 35)

Команда jobs (5)

Рис. 35: Команда jobs (5)

Далее запускаем процесс в фоновом режиме таким образом, чтобы он продолжил свою работу даже после отключения от терминала. Это можно сделать с помощью команду *nohup*. После закрываем терминал (рис. 36)

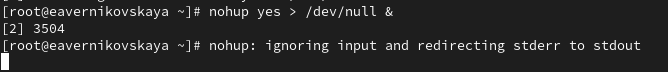


Рис. 36: Команда nohup

Заново запускаем консоль и с помощью команды top убеждаемя, что процесс продолжил свою работу (рис. 37)

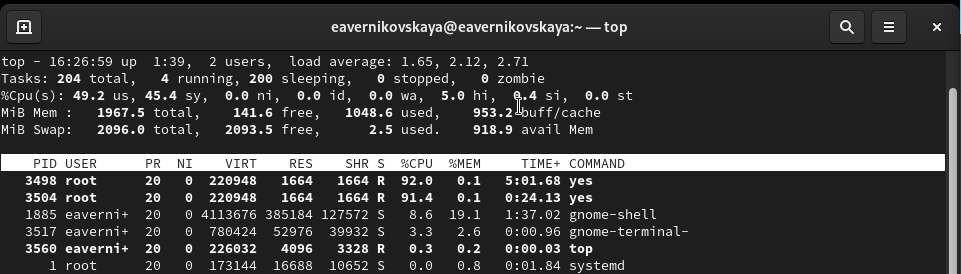


Рис. 37: Проверка с помощью top

После запускаем ещё три программы yes в фоновом режиме с подавлением потока вывода (рис. 38)

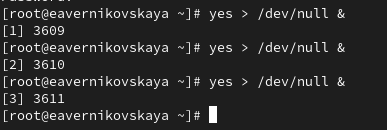


Рис. 38: Ещё 3 команды yes в фоновом режиме с подавлением потока вывода

Убиваем два процесса: для одного используем его PID (*kill -9 3609*), а для другого — его идентификатор конкретного задания (*fg2* + *ctrl+c*) (рис. 39)

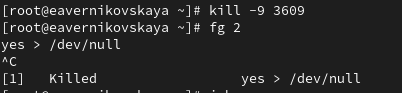


Рис. 39: Убийства двух процессов с помощью PID и идентификатора конкретного задания

Далее пробуем послать сигнал 1 (SIGHUP) процессу, запущенному с помощью nohup, и обычному процессу. В первой случае это команда *kill -1 3504*, т.к. PID процесса, запущенного с помощью nohup это 3504. Во втором это команда *kill -1 3611* (PID обычного процесса 3611) (рис. 40)

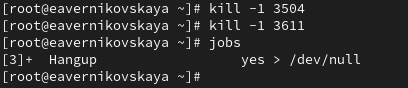


Рис. 40: Попытки послать сигналы

Запускаем ещё несколько программ yes в фоновом режиме с подавлением потока вывода. После завершаем их работу одновременно, используя команду *killall yes* (рис. 41)

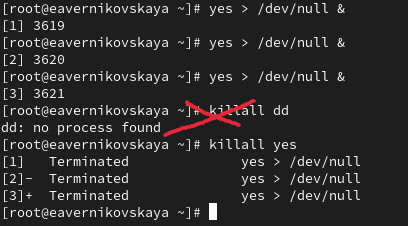


Рис. 41: Запуск нескольких команд yes и их одновременное завершение

Запускаем программу yes в фоновом режиме с подавлением потока вывода. Используя утилиту nice, запускаем программу yes с теми же параметрами и с приоритетом, большим на 5

Абсолютный приоритет определяется ядром операционной системы и показывает, с каким приоритетом выполняет процесс. Относительный приоритет является тем значением, которое может быть установленo пользователем для управления тем, как процесс будет приоритизироваться системой

Процесс 1 (PID 3662):

* Абсолютный приоритет: 80
* Относительный приоритет: 0

Процесс 1 имеет более высокий абсолютный приоритет (80), чем процесс 2 (85)

Процесс 2 (PID 3666):

* Абсолютный приоритет: 85
* Относительный приоритет: 5

Процесс 2 имеет более низкий приоритет, потому что он был понижен с помощью команды nice (рис. 42)

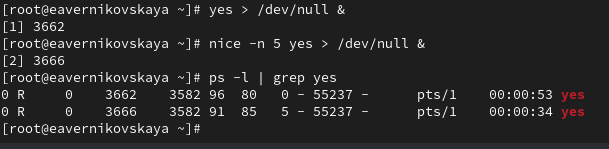


Рис. 42: Запуск программы yes в фоновом режиме с подавлением потока вывода. Запуск программы yes с теми же параметрами и с приоритетом, большим на 5. Сравнение абсолютных и относительных приоритетов

Используя утилиту renice, меняем приоритет у одного из потоков yes таким образом, чтобы у обоих потоков приоритеты были равны. В нашем случае для этого понижаем приоритет процесса 1 (PID 3662) на 5 с помощью команды *renice -n 5 3662*

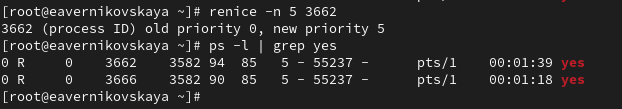


Рис. 43: Изменение приоритета

Теперь у обоих потоков приоритеты равны. Всё хорошо!

# 5 Контрольные вопросы + ответы

1. Какая команда даёт обзор всех текущих заданий оболочки?

jobs

1. Как остановить текущее задание оболочки, чтобы продолжить его выполнение в фоновом режиме?

bg номер\_задания

1. Какую комбинацию клавиш можно использовать для отмены текущего задания оболочки?

ctrl+c

1. Необходимо отменить одно из начатых заданий. Доступ к оболочке, в которой в данный момент работает пользователь, невозможен. Что можно сделать, чтобы отменить задание?

Внутри top использовать k, чтобы убить задание

1. Какая команда используется для отображения отношений между родительскими и дочерними процессами?

ps fax

1. Какая команда позволит изменить приоритет процесса с идентификатором 1234 на более высокий?

renice -n -приоритет\_процесса 1234

1. В системе в настоящее время запущено 20 процессов dd. Как проще всего остановить их все сразу?

killall dd

1. Какая команда позволяет остановить команду с именем mycommand?

Сначала узнаем PID процесса mycommand -ps aux | grep mycommand. Далее останавливаем с помощью команды kill -9 .

1. Какая команда используется в top, чтобы убить процесс?

k

1. Как запустить команду с достаточно высоким приоритетом, не рискуя, что не хватит ресурсов для других процессов?

Запустить в фоновом режиме

# 6 Выводы

В ходе выполнения лабораторной работы мы получили навыки управления процессами операционной системы

# 7 Список литературы

1. Лаборатораня работа №6 [Электронный ресурс] URL: https://esystem.rudn.ru/pluginfile.php/2400704/mod\_resource/content/4/007-process.pdf