МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение   
высшего образования

«КРЫМСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ им. В. И. ВЕРНАДСКОГО»

ФИЗИКО-ТЕХНИЧЕСКИЙ ИНСТИТУТ

Кафедра компьютерной инженерии и моделирования

**ОТЧЕТ ПО ПРАКТИЧЕСКОМУ ЗАДАНИЮ №2**

**«Изучение базовых возможностей OC Linux.»**

Практическая работа

по дисциплине «Современные технологии программирования»

студента 1 курса группы ПИ-б-о-231(2)

Покидько Максима Сергеевича

направления подготовки 09.03.04 «Программная инженерия»

|  |  |  |
| --- | --- | --- |
|  |  |  |

Симферополь, 2024

**Цель:**

Освоить основные возможности консоли OC семейства Linux. Изучить базовый набор команд для работы с OC Ubuntu. Познакомиться с фундаментальными концепциями, используемыми в ОС семейства Linux.

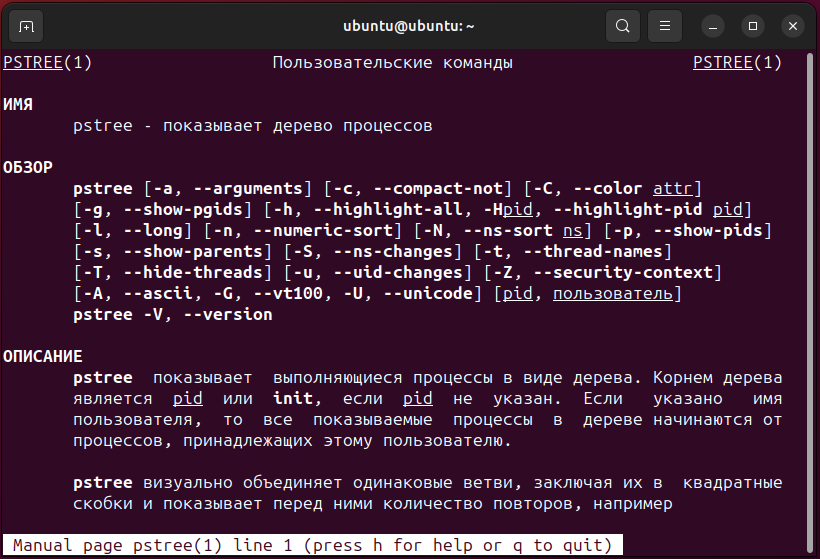
**Ход выполнения задания.**

**1. Справочная система Linux.**

**Команда man**

1. При помощи команды man получил справку по команде man.



2. Теперь с помощью того же man выведу справку по команде pstree на русском языке

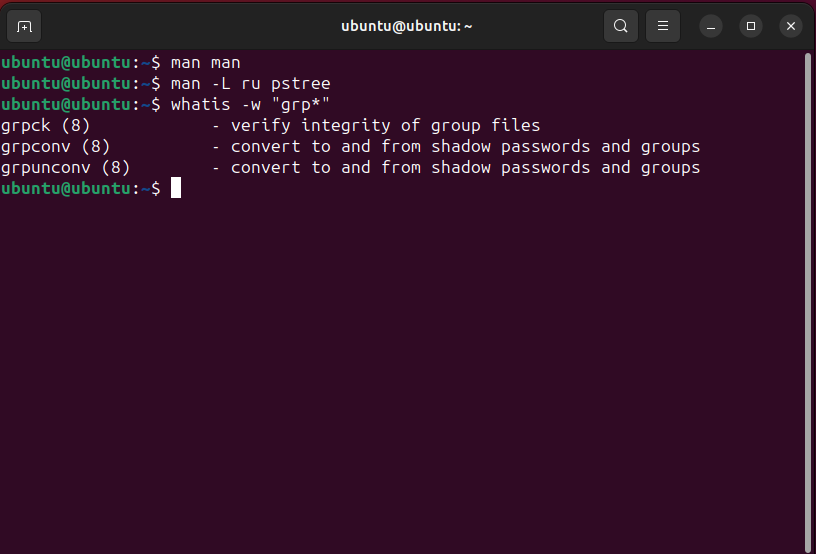
3. Нашёл информацию об этой же команде на сайте <https://manpages.ubuntu.com/>.



## Команды apropos и whatis

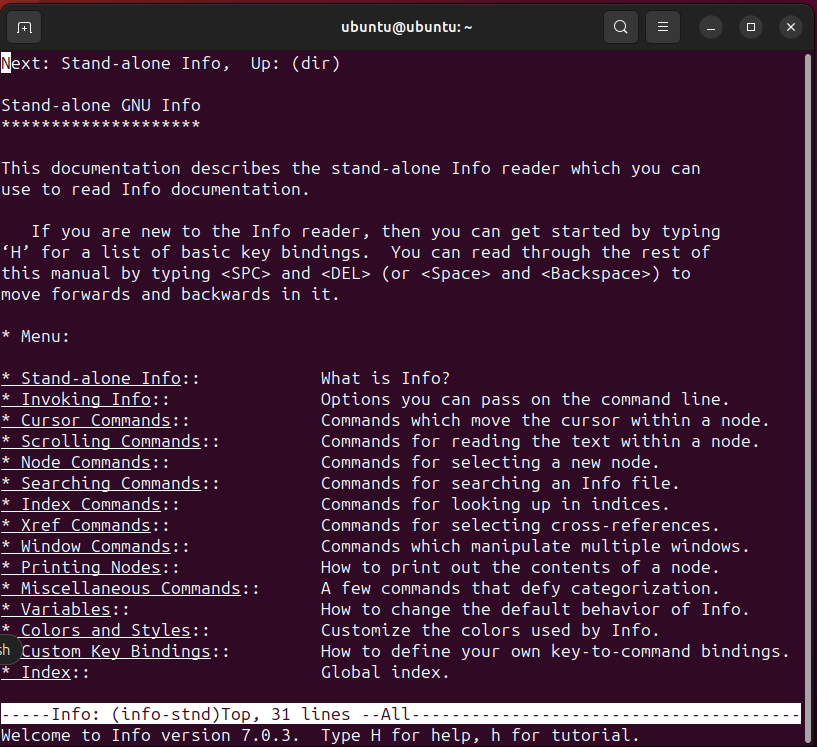
4. При использовании команды apropos столкнулся с проблемой. Он не мог найти ничего. После того, как выполнил update и upgrade, зарабштотало.

5. С помощью команды whatis нашёл команды, название которых содержит grp.

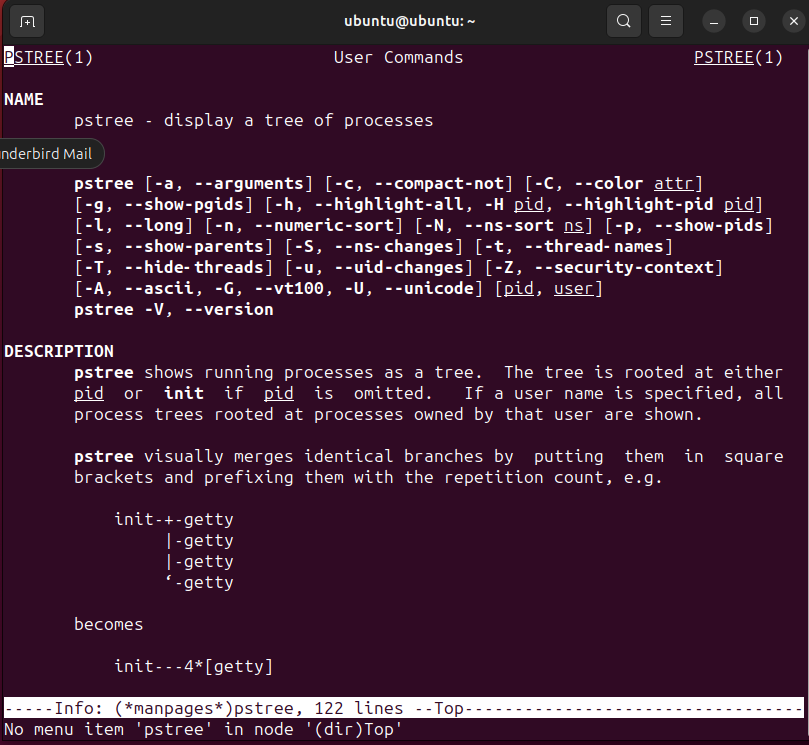


## GNU info

6. Изучил справку по команде info при помощи команды info.

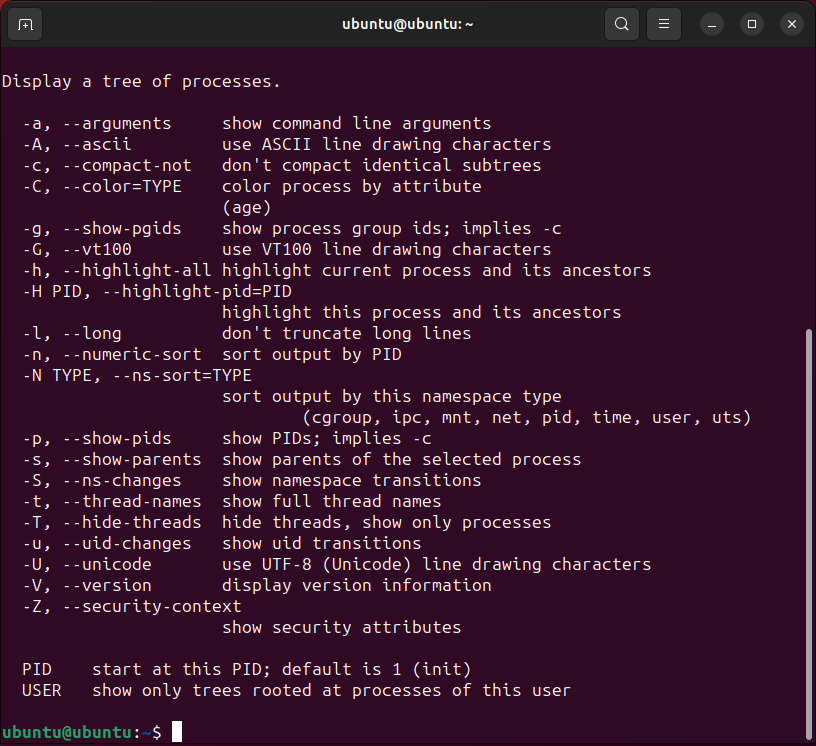


7. Теперь изучу справку для той же самой команды pstree.



## Ключи --help -h

8. Изучу справочную информацию по команде pstree с помощью ключа --help

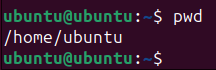


9. В целом, все три метода получения информации очень похожи между собой. Мне показалось, что info и man предоставляют очень подробную информацию, в то время как --help выглядит компактнее и удобнее, когда нужно бегло получить информацию.

**2. Файлы и утилиты для работы с ними.**

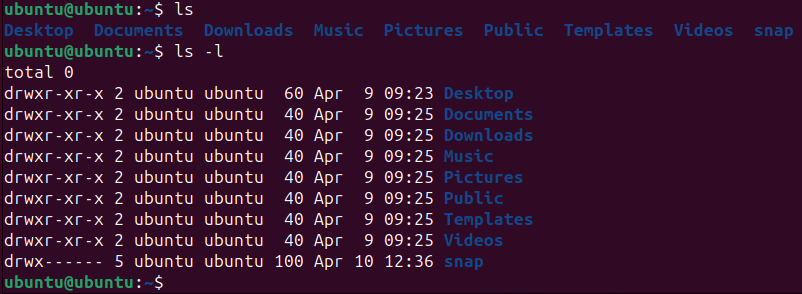
## Утилита pwd

1. С помощью команды pwd получил путь к своему текущему каталогу.

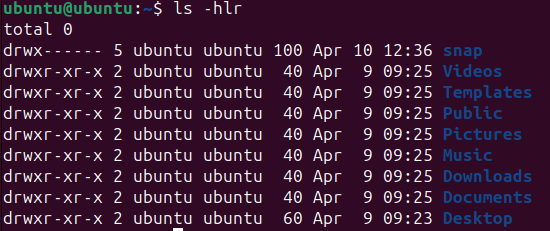


## Утилита ls

2. Выведу содержимое моего текущего каталога командами ls, ls –a и ls –l



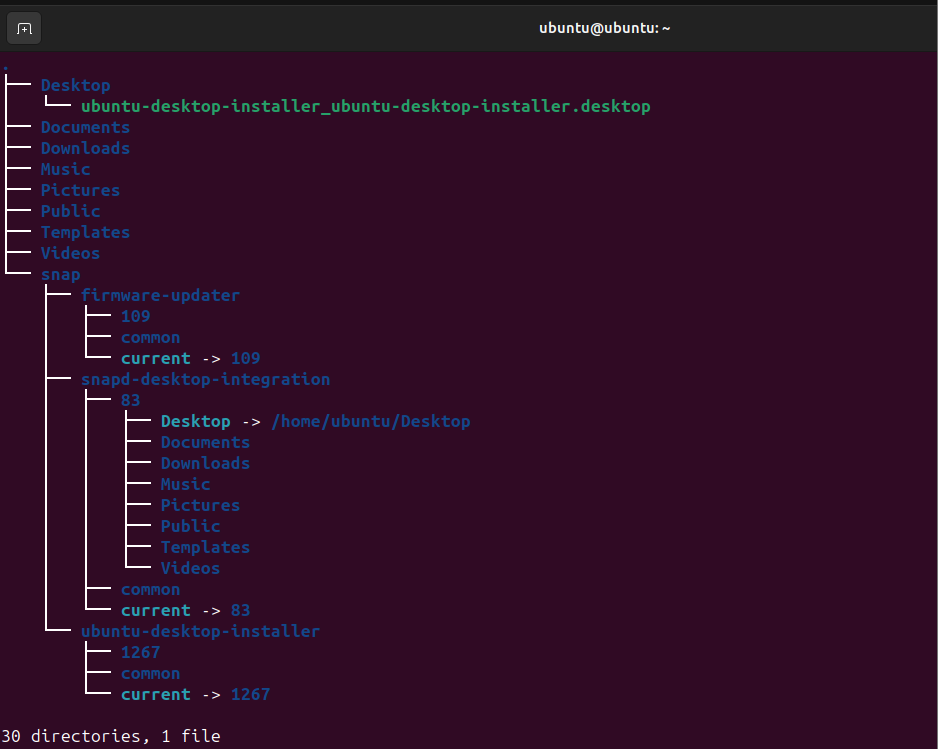
3. Выведу содержимое каталога, отсортированное по размеру файлов при помощи ls -lSr. Порядок сортировки: по возрастанию размера.



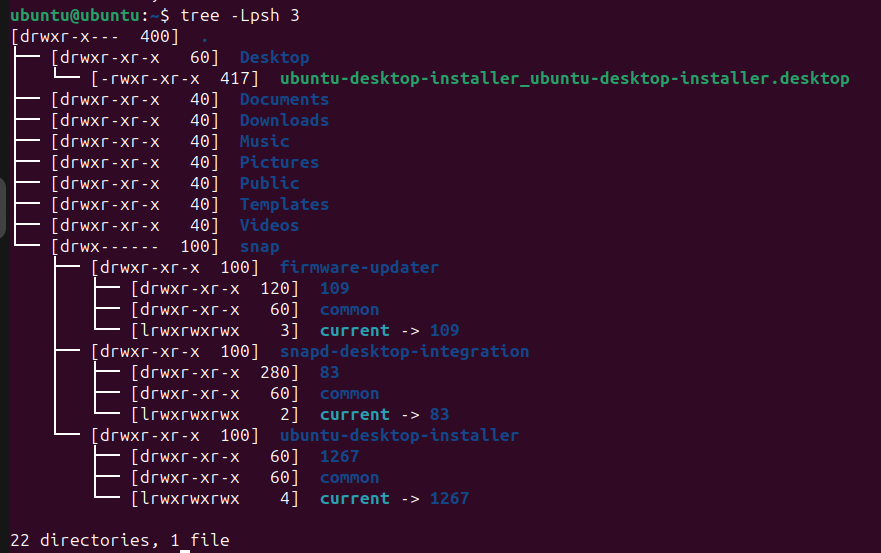
На VB у меня нет файлов, поэтому всё одного размера.

## Утилита tree

4. Посмотрю содержимое текущего каталога утилитой tree.



5. Посмотрю содержимое родительского каталога и добавлю ключи для просмотра всех файлов, их прав и размеров: –L 3 для содержимого родительского каталога, -p для просмотра прав файлов, -s для размера, с помощью –h удобнее смотреть.

c

## Команда cd

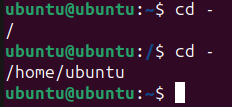
6. Перейду в корневой каталог



7. Перейду по пути /home/ubuntu



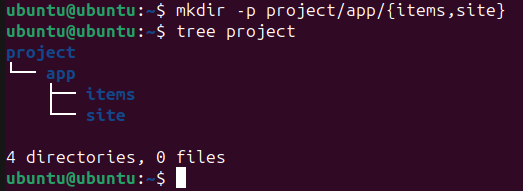
8. Вернусь в корневой, а затем в домашний с помощью -.



## Утилита mkdir

9. В домашнем каталоге создам каталог project.

10. С помощью mkdir –p создам иерархию каталогов project/app/items и project/app/site.



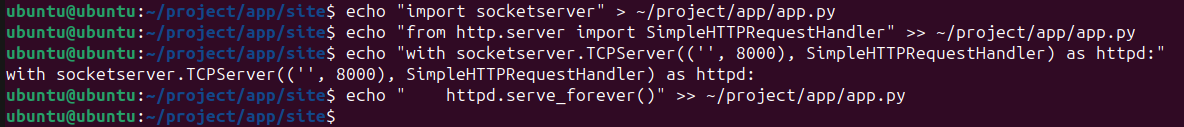
## Команды touch и echo

11. Используя команду touch, создам в каталоге site файл emty.html, находясь при этом в домашнем каталоге.



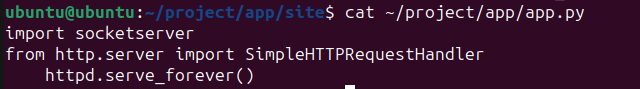
12. Теперь перейду в в ~/projects/app/site.

13. Находясь в нём, создам в каталоге app файл app.py, используя команду echo.



## Утилиты cat и tac

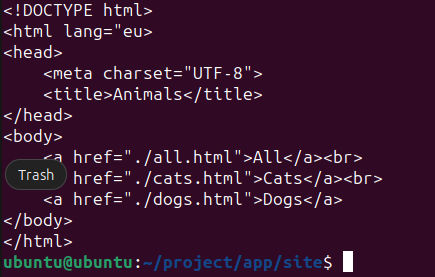
14. С помощью команды cat просмотрю содержимое только что созданного файла.



15. С помощью команды cat создам многострочный файл index.html в каталоге app.

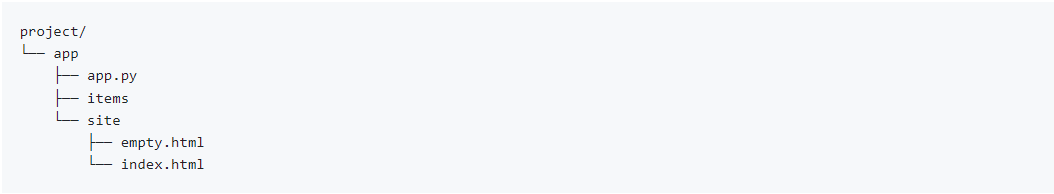


16. Проверю содержимое этого файла.

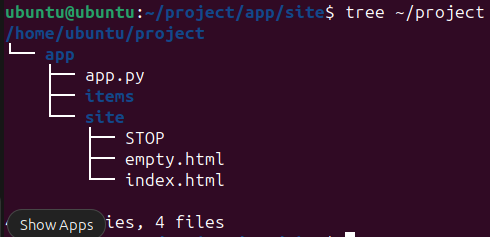


17. Теперь проверю структуру каталога project командой tree.

Нужный вид:



Мой результат:



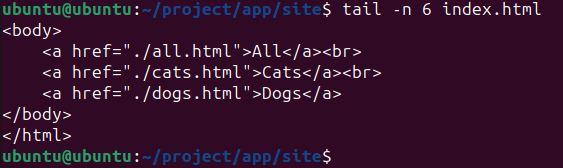
Всё сходится.

## Утилиты head и tail

18. С помощью команды head просмотрю первые 6 строк файла index.html.



19. Теперь с помощью команды tail просмотрю последние 6 строк файла index.html.

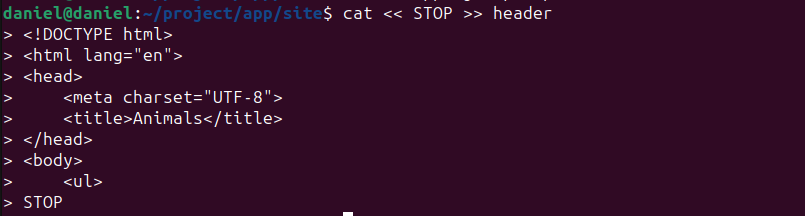


## Утилита find

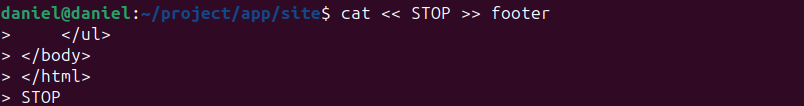
20. Создам в каталоге ~/project каталог templates



21. В нём создам файл header с нужным мне содержанием.



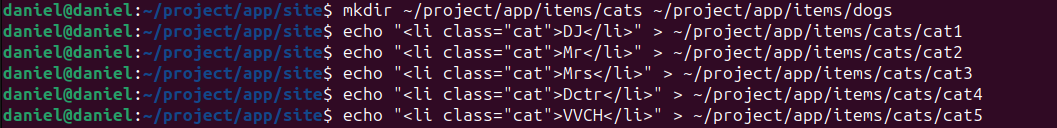
22. Там же создам файл footer.



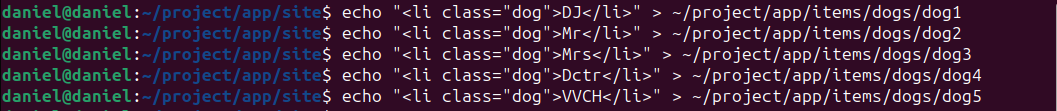
23. В каталоге ~/project/app/items создам два новых каталога, с именами: cats и dogs



24. Теперь в каталоге cats создам 5 файлов с названием cat{Номер} содержащих текст, соответствующий шаблону.



25. Далее создам 5 файлов с названием dog{Номер} в каталоге dogs, содержащих текст, соответствующий шаблону.



26. Теперь находясь в site, найду с помощью find все файлы (но не папки) с именами начинающимися на cat и для каждого найденного файла выполнила команду cat имя\_файла >> cats.



27. Затем выполню аналогичные действия с dogs.



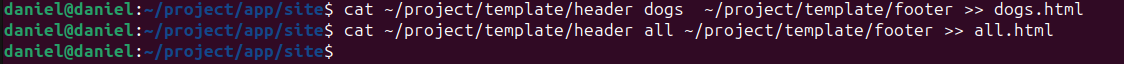
28. Далее напишу команду find таким образом, чтобы она в каталоге items нашла все файлы (но не папки) и для каждого найденного файла выполнила команду cat имя\_файла >> all.



29. С помощью cat склею файлы header, cats и footer в файл cats.html.



30. Повторю эти действия для dogs и all.

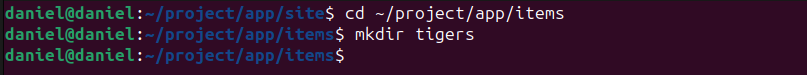


Теперь структура выглядит так:



## Команды cp и mv

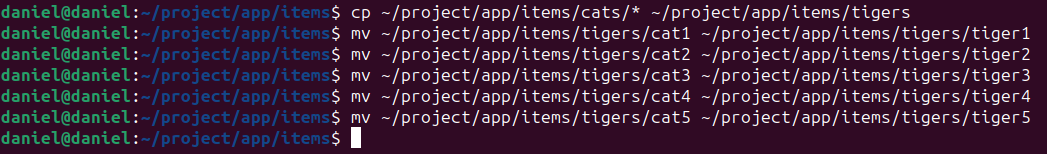
31. Перейду в каталог ~/project/app/items и создам в нем каталог с именем tigers



32. Скопирую все файлы из каталога cats в tigers



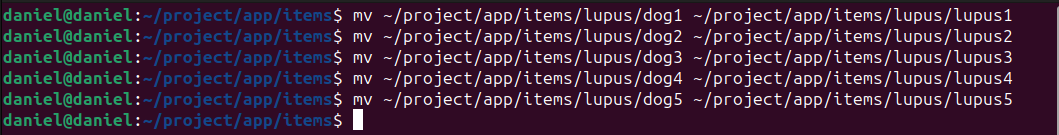
33. Переименую файлы в каталоге tigers, таким образом, чтобы они начинались не на cat, а на tiger.



34. Скопирую каталог dogs вместе с файлами таким образом, чтобы копия называлась lupus.



35. Переименую файлы в каталоге lupus, таким образом, чтобы они начинались не на dog, а на lupus.

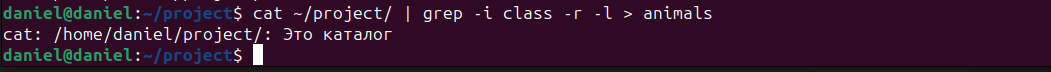


## Утилита grep

36. Перейду в каталог ~/project/.



37. Используя grep, найду все файлы, в тексте которых присутствует слово class. Добавлю ключ -l, чтоб в вывод показало только имена файлов. Вывод направлю в файл animals;



38. С помощью файла animals, связки cat и grep найду все файлы, не заканчивающиеся на .html и запишу результат в файл trash.



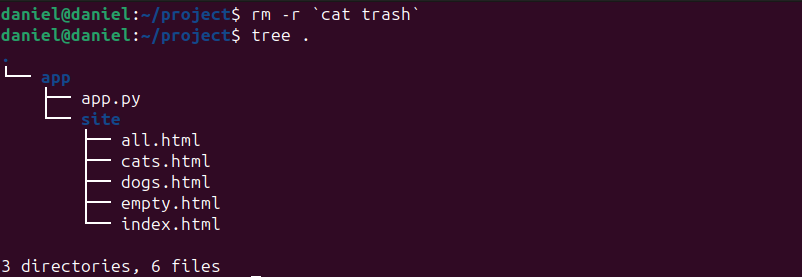
## Команда rm и rmdir

39. Добавлю в файл trash каталоги app/items и templates, а также файлы animals и trash.

40. На всякий случай создам копию каталога project



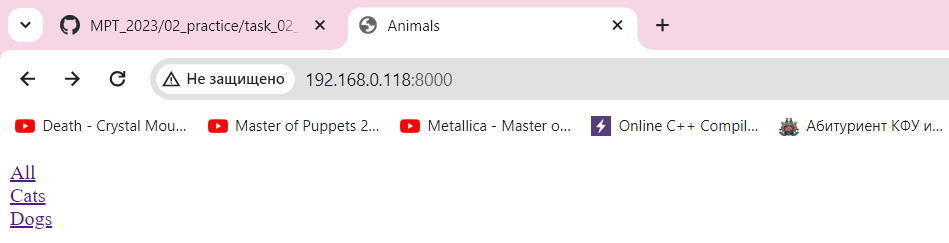
41. Воспользуюсь командой rm, чтобы удалить все файлы и папки из списка.



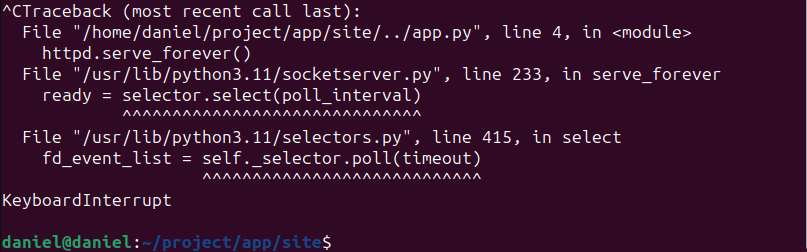
42. Находясь в каталоге project/app/site запущу скрипт app.py командой python3 ../app.py.



43. Далее в браузере хостовой машины введу в поисковой строке ip-адрес моей виртуальной машины и через двоеточие порт 8000. В результате открылась страница с тремя ссылками: All, Cats, Dogs, переходя по которым отображаются списки кличек животных, которые я добавил ранее.



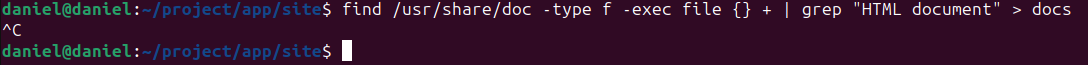
44. Остановлю приложение в терминале сочитанием Ctrl+C.



## Утилита file

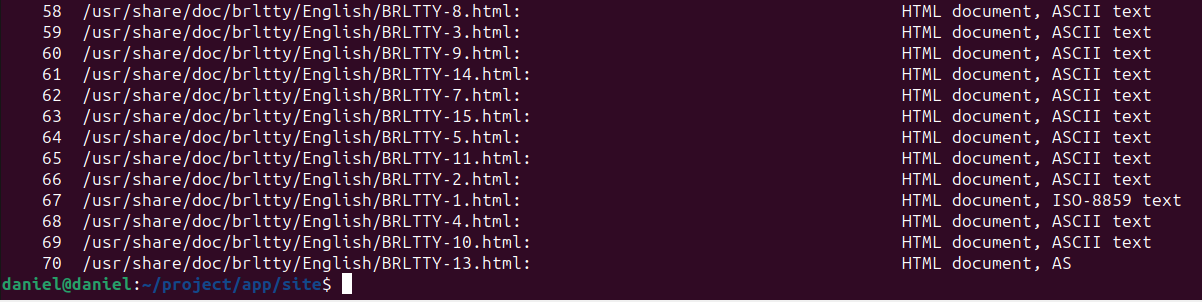
45. Используя find найду все файлы в директории /usr/share/doc и к каждому найденному файлу применю утилиту file.

46. Результат передам утилите grep которая должна найти в выводе текст HTML document. То что получится сохраню в файл docs



47. С помощью nl определю количество строк в файле.





**Утилиты less и more**

48. Пропишу команду cat docs | less.

49. В поисковом окне найду через / слово examples.

В результате подсветились все совпадения – всего их получилось три.

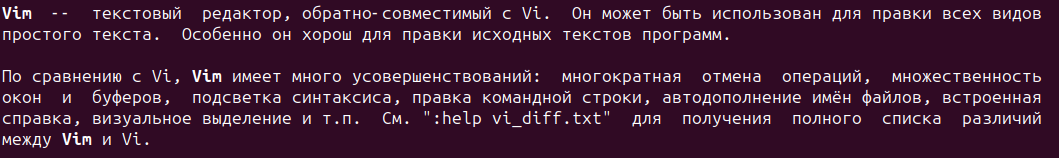




## Текстовые редакторы

50. Изучим nano и vimtutor.

Nano — это консольный текстовый редактор. Он прост в освоении, поэтому подходит начинающим пользователям. Редактор относится к немодальным: вводить и редактировать текст можно сразу после открытия файла.



**3. Пользователи и группы.**

**Утилиты whoami, id, groups**

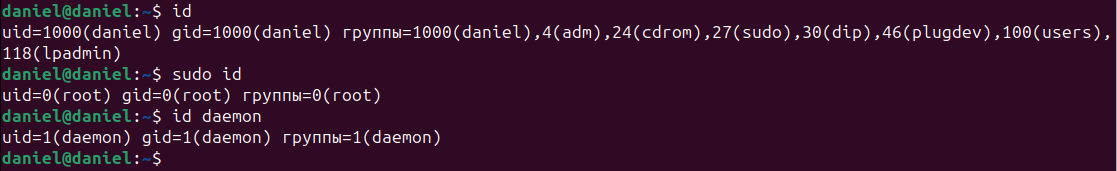
1. Выполню команды whoami и sudo whoami.



2. Выполню команду groups, а затем sudo groups и groups daemon.

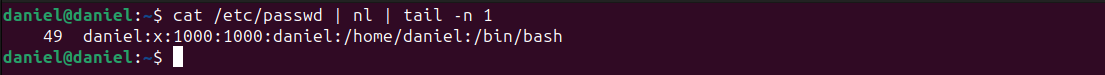


3. Выполню команды id, затем sudo id и id daemon.

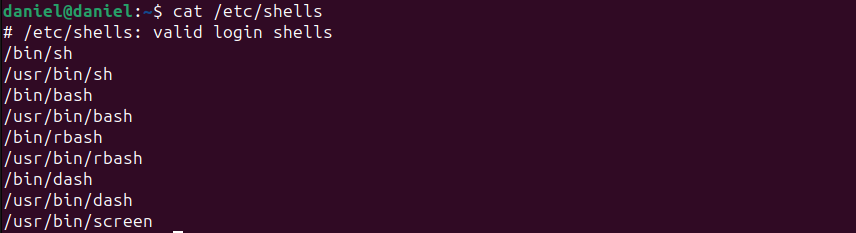


## Как получить список пользователей

4. Используя связку cat и nl определите количество пользователей в файле /etc/passwd



5. Посмотрю содержимое файла /etc/shells. Это список всех командных оболочек присутствующих в системе, которые могут быть установлены как оболочки по умолчанию для пользователей. Запомню путь к оболочке screen;



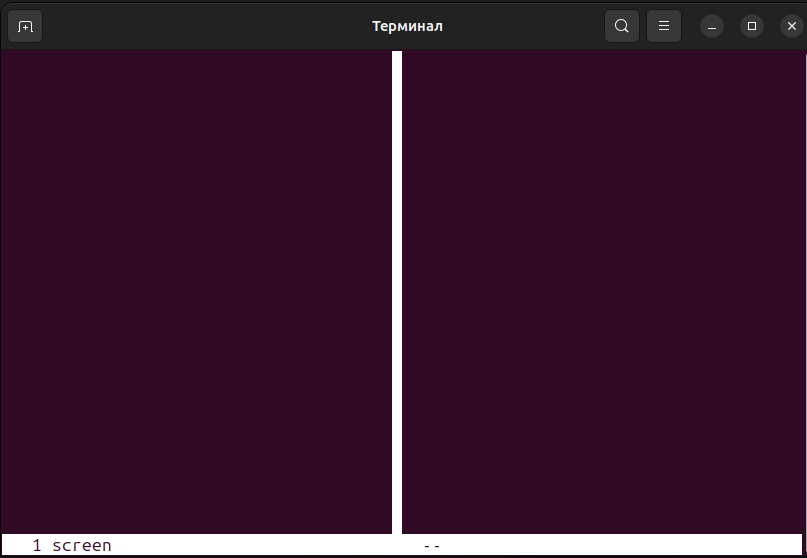
6. Открою файл /etc/passwd в текстовом редакторе (через sudo) и заменю у своего пользователя оболочку на screen



7. Перезапустил систему.

8. Выполнил последовательность действий:

Нажмите сочетание клавиш Ctrl + a и затем | (экран разделился пополам);



Нажал сочетание клавиш Ctrl + a и затем Tab (курсор перешёл на второй экран);

Нажал сочетание клавиш Ctrl + a и затем с (на втором экране началась сессия);

Выполнил команду /bin/bash (на втором экране запустилась оболочка bash);



9.

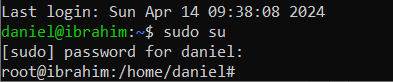
Введу команду chsh и укажу старую оболочку (обычно это bash). Команда chsh сама внесёт изменение в /etc/passwd, но при этом гарантированно его не повредит и проконтролирует, что введённая мной оболочка действительно доступна в системе;



10. Переподключился к серверу.

## А где пароль?

11. Подключился к серверу вторым окном и перешёл в нём под root (sudo su).



12. Поменял конфиг ssh-сервера таким образом, чтобы разрешить вход пользователям с пустыми паролями (PermitEmptyPasswords yes) и перезапустил службу.



13. Сделаю резервную копию файла /etc/shadow.



14. Открыл файл /etc/shadow в текстовом редакторе и удалил у своего пользователя пароль.

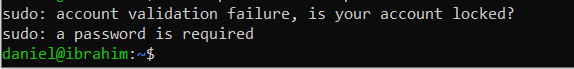


15. В первом окне разлогинился и снова подключился. Пароль больше вводить не нужно;

16. В первом окне удалил файл /etc/shadow. sudo не попросил пароль.



17. В первом окне попробовал восстановить файл /etc/shadow из резервной копии. Теперь sudo ругается на отсутствие пароля.



18. Восстановил исходный /etc/shadow через второе окно с помощью команды restore –if /etc/shadow.bak

## Как получить список групп

19. Открою файл /etc/group в текстовом редакторе и добавлю себя в группу games.



20. Проверю список групп своего пользователя. На данный момент новой группы нет.



21. Перелогинюсь и проверю список своих групп снова.

 группа есть.

## Пароль группы

22. Открою файл /etc/gshadow в текстовом редакторе и у группы mail заменю пароль на $1$MPE2cemT$mjAVzNRCcNfuvWsylLAwS1.



23. Введу команду newgrp mail и на запрос пароля введу: 123.



24. Проверю список групп своего пользователя. Обращу внимание на основную группу.



25. Введу команду exit и снова проверю список своих групп. Mail пропал.

## Утилиты для управления пользователями и группами

### **Утилита passwd**

26. Сделал резервную копию файла /etc/shadow.

27. В первом окне, для своего пользователя выполню команду: sudo passwd –lock.

28. К паролю моего пользователя приписался символ “!”, что означает, что пароль заблокирован.

29. В первом окне перелогинюсь. Теперь я не могу войти ни под каким паролем.

30. Во втором окне сниму блокировку со своего пользователя;

31. В первом окне попробую залогиниться.

32. Выполню в первом окне для своего пользователя команду: sudo passwd –expire.

33. Перелогинюсь. Теперь мой пароль установлен как требующий смены при входе.

34. Попробую ввести тот же пароль, что и был, а затем какой-нибудь простой пароль вроде 123. В обоих случаях отказ.

35. Во втором окне открою файл /etc/pam.d/common-password.

36. Изучу man по pam\_unix .

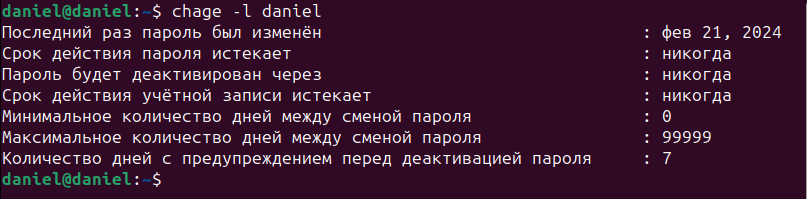
37. Изменю параметры pam\_unix.so таким образом, чтобы убрать проверки сложности, установить тип хеширования md5 и минимальную длину пароля равную 1 символ. 38.

38. В первом окне, попробую перелогиниться и в качестве нового пароля установлю любой 1 символ;

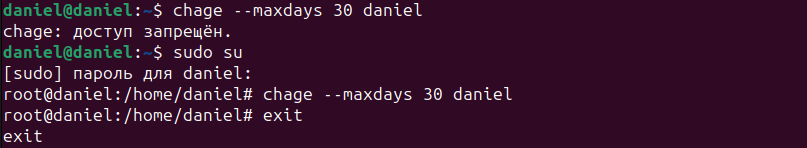
39. Используя утилиту passwd, установлю пароль.

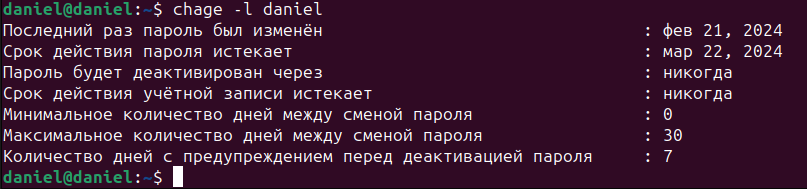
### **Утилита chage**

40. Посмотрю информацию о сроках действия пароля для своего пользователя;



41. Установлю период устаревания пароля 30 дней и снова посмотрю информацию о сроках.

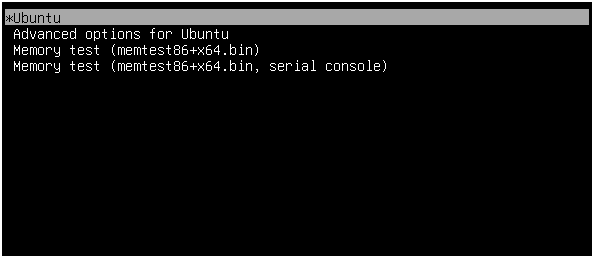




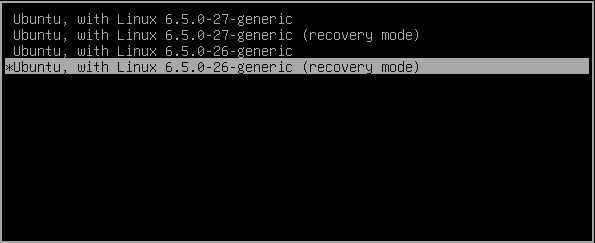
p.s. без рута отказывало в доступе

42. В файле /etc/shadow заменю свой пароль на $1$V/N2w9n3$0LQa/OywnCLPSlEcd5.w8.

43. Перезагружу виртуальную машину, и в самом начале загрузки зажму клавишу Shift. В результате я попаду в меню загрузчика.



44. Выберу пункт "Advanced options for Ubuntu" и затем пункт "Ubuntu, with Linux <версия ядра> (recovery mode)".



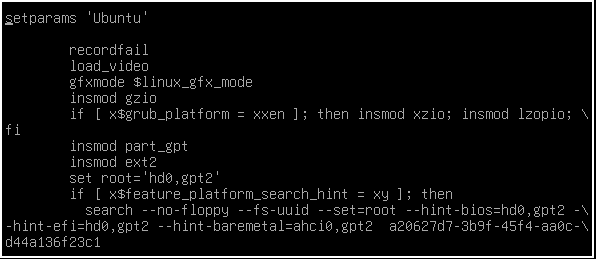
45. После загрузки я зашёл под root. Установлю старый пароль для своего пользователя (не для root) при помощи passwd.



46. Введу exit и система продолжит загружаться дальше. Теперь я снова могу войти под своим паролем.

47. Теперь, снова отредактирую файл /etc/shadow, но в этот раз установлю указанный выше пароль для root, а у своего пользователя добавлю восклицательный знак перед паролем.

48. Перезагружу виртуальную машину, и в самом начале загрузки зажму клавишу Shift, а затем, в меню загрузчика, клавишу e, чтобы отредактировать загрузочный скрипт;



49. Промотаю скрипт до строки, которая должна выглядеть примерно так:

linux /vmlinuz-6.5.0-21-generic root=/dev/mapper/ubuntu--\vg-ubuntu--lv ro

и допишу в конец init=/bin/sh. Это приведёт к тому, что вместо процесса инициализации (первый процесс в системе) будет запущена командная оболочка.

50. Нажму Ctrl + x или F10, чтобы продолжить загрузку.

51. Изменить пароль сейчас не получится, т.к. файловая система смонтирована в режиме только для чтения. Чтобы перемонтировать её в режиме для записи введу:



52. Чтобы вернуть root пароль в виде !, а свой пароль разблокировать введу:

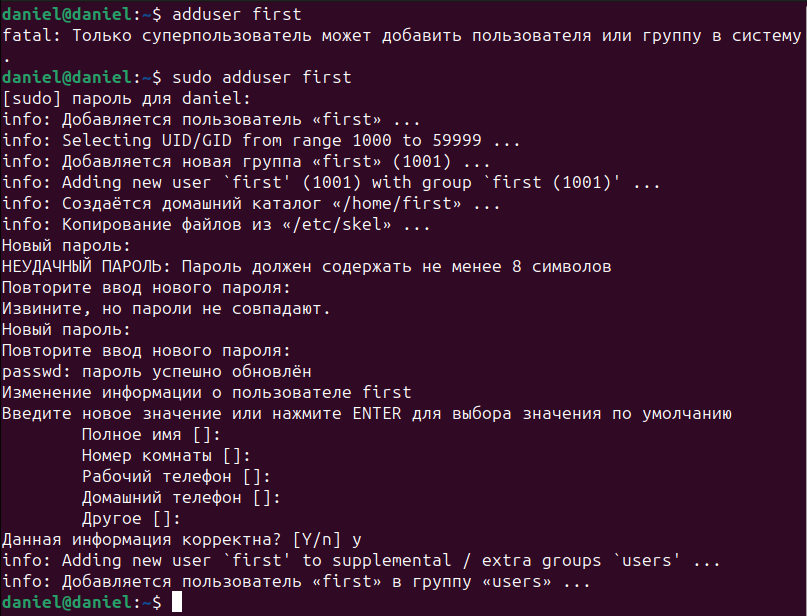


53. Далее можно выключить машину и заново запустить или можно продолжить загрузку запустив процесс инициализации, который мы ранее заменили своим:



### **Утилиты adduser и useradd**

54. Используя утилиту adduser, создам пользователя с логином first и любым паролем.



55. Используя утилиту useradd, создам пользователя с логином second.



56. Открою файлы /etc/passwd, /etc/shadow и каталог /home/ и сравню результат создания пользователей.

57. Используя утилиту useradd создам псевдоним для пользователя first с логином alf:

укажу ключи --non-unique, чтобы разрешить создавать не уникальных пользователей и --uid (после этого ключа нужно указать UID пользователя first);

укажу ключ --create-home, чтобы создать отдельный домашний каталог для нового пользователя;

укажу ключ --shell /usr/bin/tmux, чтобы указать командную оболочку, которую нужно запустить при входе пользователя;

после создания пользователя установлю для него пароль при помощи команды passwd.

58. Открою новое окно и залогинюсь под пользователем first:

посмотрю вывод утилиты id;

посмотрю полный путь к моей домашней директории;

разлогинюсь.

59. Теперь залогиньтесь под пользователем alf:

обратите внимание на логин в строке приглашения, а также, что оболочка по прежнему bash, а не tmux;

посмотрите вывод утилиты id;

посмотрите полный путь к вашей домашней директории;

посмотрите, кто владеет текущей директорией;

попробуйте перейти в /home/first;

разлогиньтесь.

60. Используя утилиту useradd создам псевдоним для пользователя root с логином alro.

61. Залогинюсь под alro. Как видно, несмотря на то, что для root пароль заблокирован, можно войти с его правами через alro.

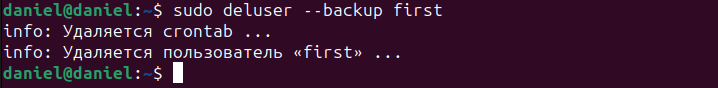
62. Разлогинюсь.

### **Утилиты deluser и userdel**

63. Используя утилиту userdel (без дополнительных флагов) удалите пользователей second, alf и alro.

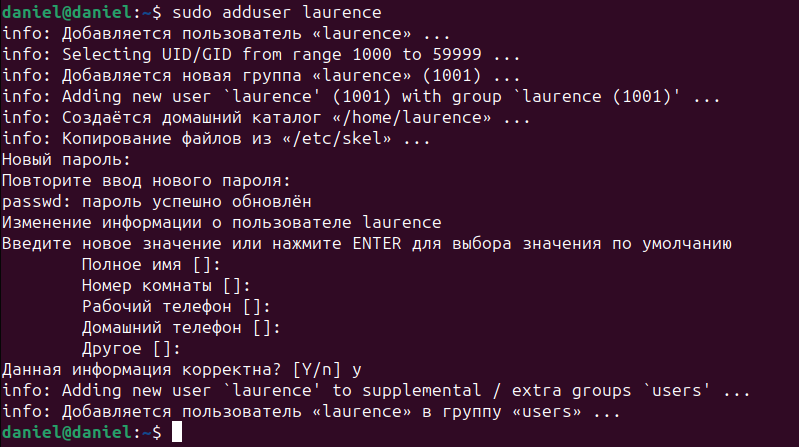


64. Используя утилиту deluser удалите пользователя first вместе с его домашней директорией и всеми файлами которые ему принадлежат (домашний каталог удалённого пользователя alf тоже принадлежит пользователю first). Дополнительно укажите опцию для создания бэкапа.

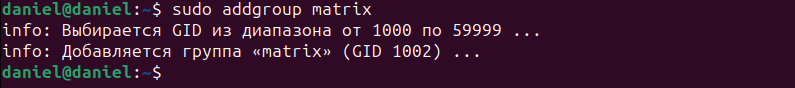


### **Утилита groupmod**

65. Создам пользователя с логином laurence при помощи утилиты adduser.



66. Создам группу matrix.



67. При помощи usermod добавлю к вторичным группам пользователя laurence группу matrix.



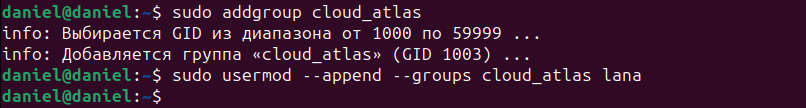
68. Затем установлю для пользователя laurence значение 1999 в качестве UID.



69. Поменяю логин пользователя laurence на lana



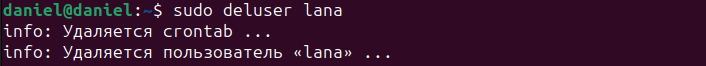
70. К группам пользователя lana добавлю группу cloud\_atlas.



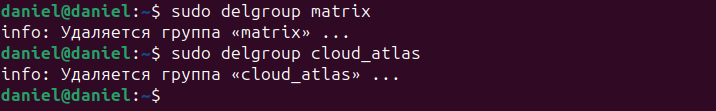
71. Изменю GID группы cloud\_atlas на 2012.



72. Удалю пользователя lana.

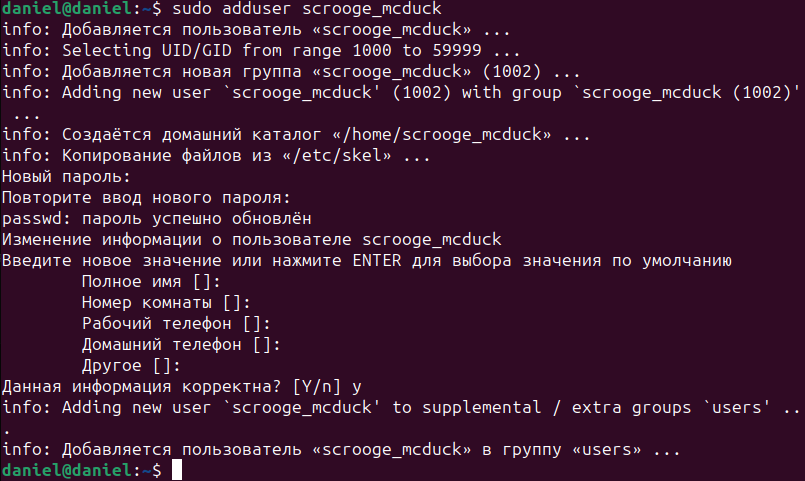


73. Удалю группы matrix и cloud\_atlas.



### **Утилита su**

74. Создам нового пользователя scrooge\_mcduck;



75. Выполню команду mkdir money от имени пользователя scrooge\_mcduck;



76. Посмотрю кому принадлежит папка money.

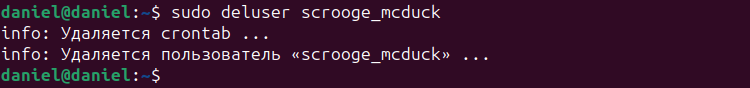
77. Перейду под пользователя scrooge\_mcduck с командной оболочкой /usr/bin/tmux.



78. Удалю папку money.

79. Завершу сеанс пользователя scrooge\_mcduck;

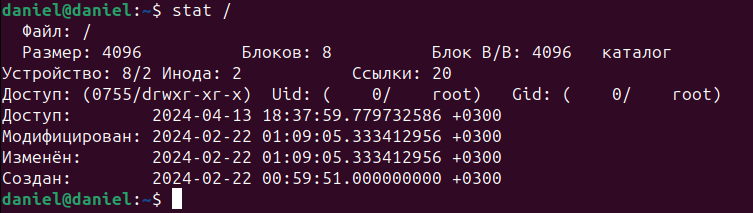
80. Удалю пользователя scrooge\_mcduck.



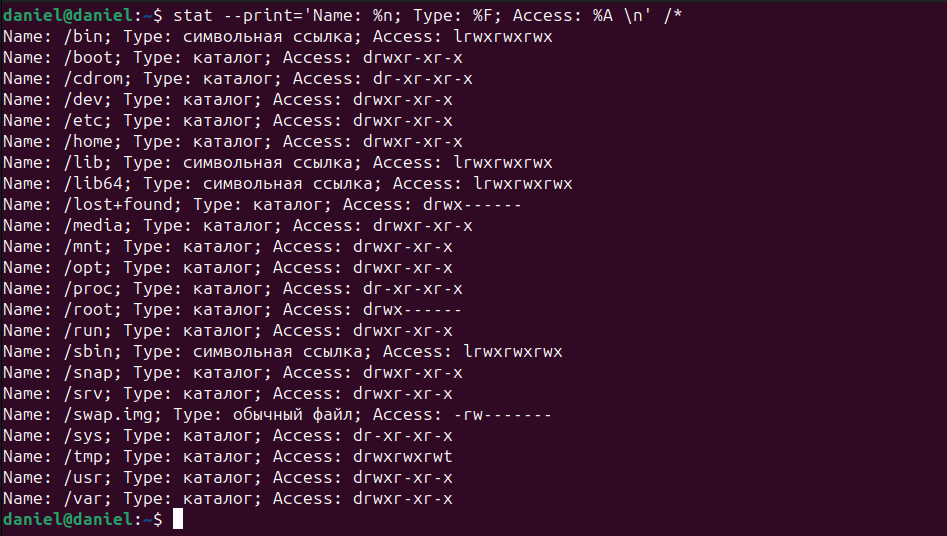
# **4. Файлы и управление правами доступа**

**Стандартные атрибуты**

1. Воспользуюсь утилитой stat и посмотрю информацию о каталоге /. Номер inode у корневого каталога равен 2.

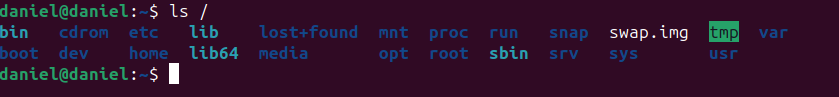


2. Использую утилиту stat с параметром --print, чтобы отобразить информацию о содержимом каталога /. Составлю такую форматную строку, чтобы данные выводились в соответствии с примером:



3. Использую утилиту ls и посмотрю информацию о каталоге /. Содержимое каталога не выводится.

4. Использую утилиту ls, чтобы отобразить информацию о содержимом каталога /. Укажу соответствующие ключи, чтобы НЕ отображались . и .., логины владельца и название группы.

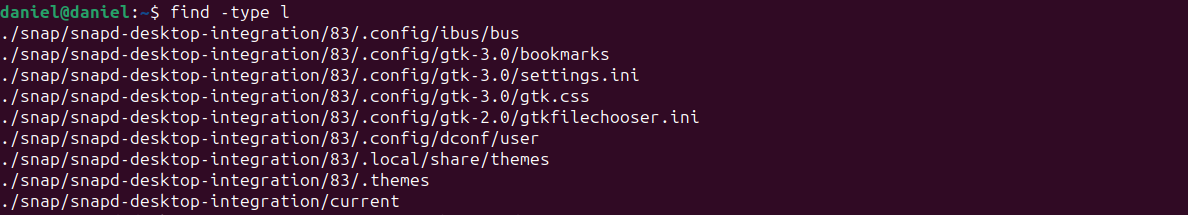


### **Типы файлов**

5. Воспользуюсь утилитой find, чтобы найти файлы каждого типа описанного выше;



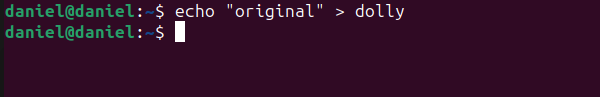




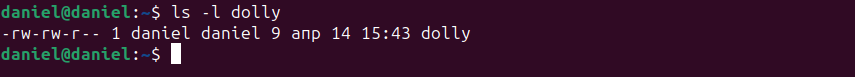




6. В домашнем каталоге создам файл dolly с тексом "original".



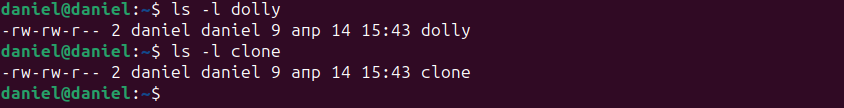
7. При помощи команды ls -l посмотрю количество жёстких ссылок на dolly.



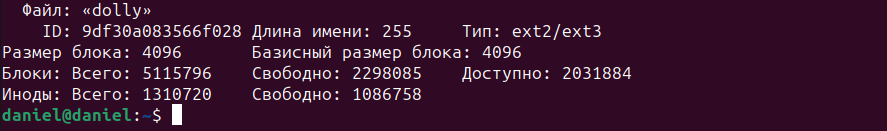
8. Использую команду ln чтобы создать жёсткую ссылку на dolly под названием clone;

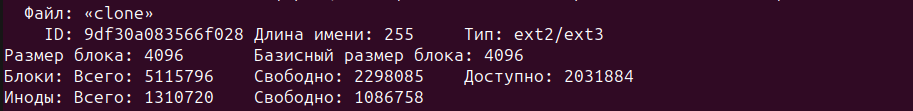


9. Посмотрю количество жёстких ссылок на dolly и на clone.



10. При помощи команды stat посмотрю время последней модификации (modify) и номер inode файлов dolly и clone.

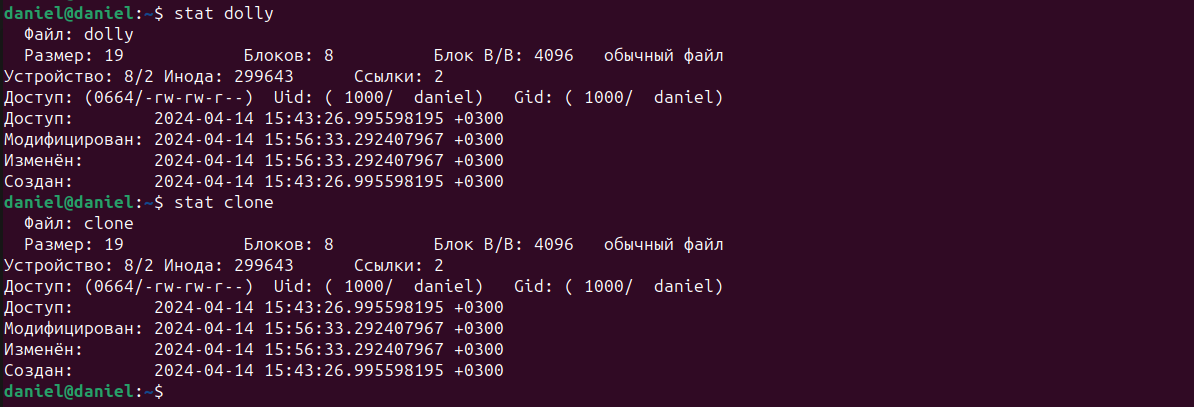




11. Добавил в файл clone любую строку.



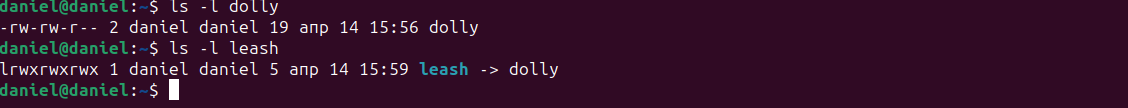
12. Посмотрел время последней модификации файла dolly и файла clone. Как видно, оно совпадает, хотя я работал только с файлом clone.



13. Использую команду ln чтобы создать символьную ссылку на dolly под названием leash.



14. При помощи команды ls -l посмотрите количество жёстких ссылок на dolly и на типа файла leash.



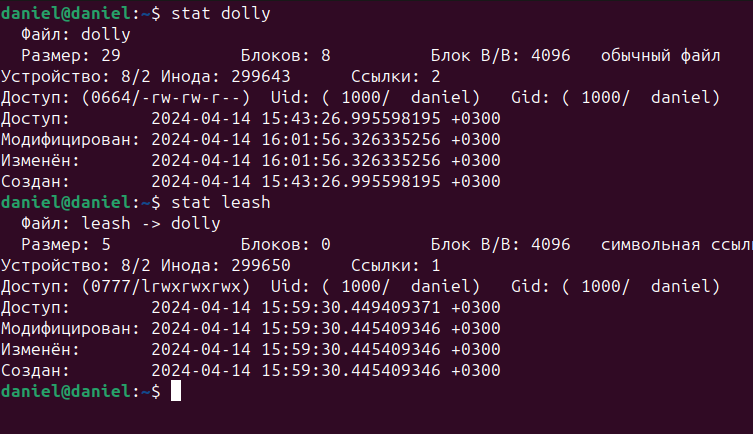
15. При помощи команды stat посмотрел время последней модификации (modify) и номер inode файлов dolly и leash.



16. Добавьте в файл leash любую строку.



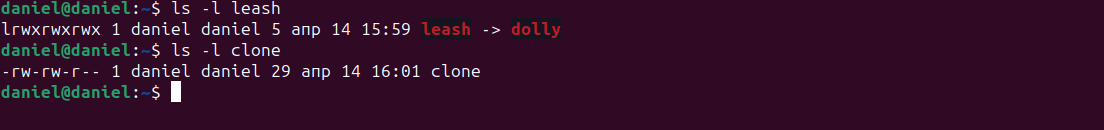
17. Посмотрите время последней модификации файла dolly и файла leash. Как видно, время модификации leash не изменилось вообще, а у dolly изменилось.



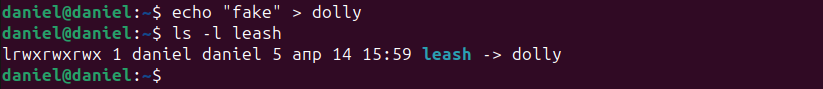
18. Удалил файл dolly.



19. Выведу содержимое файлов clone и leash. Как видно, теперь leash указывает "в никуда".



20. Создайте файл dolly с текстом "fake" и снова выведите содержимое файла leash.



21. Удалил файлы dolly, clone, leash.



**Временные метки**

22. Создам каталог data и в нём три файла с интервалом в 1 минуту или больше: one, two, three;

23. Воспользуюсь утилитой find и ключом фильтрующим по времени последней модификации (modify), чтобы найти в каталоге data сначала только файл three, затем two, three и затем one, two, three;

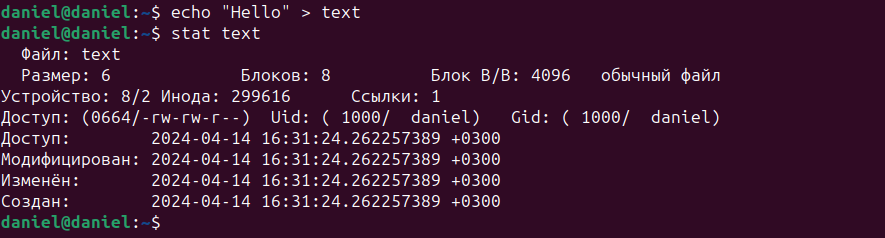
24. Воспользуюсь утилитой touch, чтобы установить для файлов one, two, three дату последнего доступа для чтения (access) в значения: 8 декабря 2228 года, 6 января 2230 года, 22 марта 2233 года соответственно (в качестве часов и минут укажите нули);

25. Создам файл four.

26. Воспользуюсь утилитой find, чтобы найти и удалить все файлы в каталоге data к которым был получен доступ на чтение (access) менее 0 минут назад (т.е. one, two, three).

### **Размер файла**

27-28. Создам файл text содержащий текст: "Hello" и изучу 2-ю строку в выводе утилиты stat для этого файла. Как видно размер файла в байтах (Size) равен - 6, a в блоках (Blocks) - 8:



Размер в блоках объясняется тем, что тут указаны блоки выделенные на диске, а не в файловой системе. Если принять, что 1 блок в файловой системе равен 4096 байт, а размер блока на диске 512 байт, тогда получаем, 8 блоков на диске - это 1 блок в файловой системе;

Почему же размер в байтах равен 6, хотя символов в файле всего 5?

29. Посмотрю содержимое файла text побайтово при помощи команды:

cat text | hexdump –c

 - переход строки.

**Права доступа**

30. Воспользуюсь утилитой find, чтобы найти во всей системе файлы принадлежащие моему пользователю и имеющие разрешение для владельца, как минимум на чтение. Результат сохраню в файл readable и воспользуюсь утилитой wc, чтобы определить сколько в нём получилось строк.

31. Воспользуюсь утилитой find, чтобы найти во всей системе файлы принадлежащие основной группе вашего пользователя и имеющие разрешение на исполнение для любой категории пользователей. Результат сохраню в файл executable и воспользуюсь утилитой wc, чтобы определить сколько в нём получилось строк;

32. Воспользуюсь утилитой find, чтобы найти во всей системе файлы НЕ принадлежащие моему пользователю и имеющие разрешения в точности равные 755. Результат сохраню в файл 755 и воспользуюсь утилитой wc, чтобы определить сколько в нём получилось строк.

#### **Изменение прав доступа**

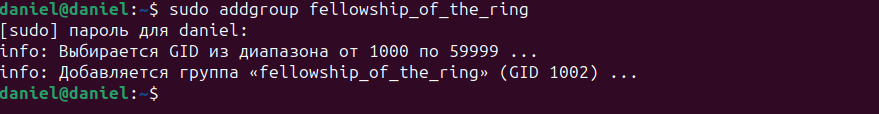
33. В домашнем каталоге создам файл с именем my\_precious и установлю для него следующие права доступа: владелец может всё, группа и остальные - ничего.



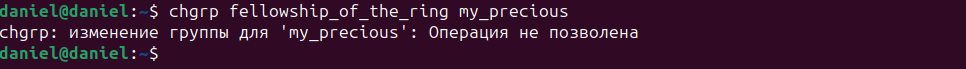


#### **Смена пользователя-владельца**

34. Создам группу fellowship\_of\_the\_ring.



35. При помощи команды chgrp попробую сменить группу владеющую файлом my\_precious на fellowship\_of\_the\_ring. Т.к. я не состою в этой группе, мне должны отказать в доступе.



36. Добавлю себя в группу fellowship\_of\_the\_ring и повторно попробуйте сменить группу владеющую файлом my\_precious.

37. При помощи команды chown изменю владельца файла my\_precious на root и группу-владельца на root;

38. Удалю файл my\_precious.

#### **Действие прав доступа на файлы и каталоги**

39. Использую утилиту find, чтобы найти во всей системе файлы, для которых установлен SUID бит. Результат выведу в файл и посчитаю сколько таких файлов.



40. Перейду в каталог /tmp и скопирую /bin/bash в /tmp/bash. Для копии владельца и группу на свои.

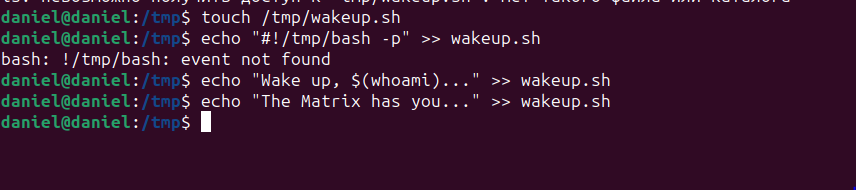


41. В каталоге /tmp создам файл wakeup.sh содержащий:

#!/tmp/bash -p

echo "Wake up, $(whoami)..."

echo "The Matrix has you..."



Первая строка в скрипте - это шебанг. Он показывает какой программе следует передать этот скрипт для интерпретации;

42. Установлю права владения файлом wakeup.sh rwx---r-x;

43. Создам пользователей: neo и morpheus;

44. Перейду под пользователя neo и запущу скрипт командой ./wakeup.sh.

45. Вернусь под своего пользователя и добавьте SUID бит к правам /tmp/bash;

46. Перейду под пользователя morpheus и запущу скрипт командой ./wakeup.sh.

##### **SetGID**

47. Находясь в /tmp создам каталог matrix и установлю ему права доступа 770 и перейду в него.

48. Создам группу share;

49. Добавлю пользователей neo и morpheus в группу share (как их дополнительную группу).

50. Изменю группу для каталога /tmp/matrix на share;

51. Из-под пользователя neo создам в каталоге matrix файл "message" с любым текстом.

52. Под пользователем morpheus попробую дописать в "message" ещё строку текста.

53. Проверю права доступа к файлу. Несмотря на то, что каталог и оба пользователя состоят в группе share файлы по-прежнему создаются с основной группой пользователя.

54. Вернусь под своего обычного пользователя и добавлю каталогу matrix SGID бит.

55. Из-под пользователя neo удалю старый файл "message" и создам его заново с любым текстом. Поверю, что теперь группа-владелец файла - это share.

56. Под пользователем morpheus попробую дописать в "message" ещё строку текста.

##### **StikyBit**

57. Установлю на каталог matrix права 777. Теперь никаких спец битов не установлено.

58. Из-под пользователя morpheus создам в каталоге matrix файл "morpheus\_file".

59. Зайду под пользователя neo и попробую удалить "morpheus\_file". Файл удалён, т.к. права каталога matrix это разрешают, хотя файл и не принадлежит neo.

60. Добавлю к правам каталога matrix StikyBit.

61. Из-под пользователя morpheus создам в каталоге matrix файл "morpheus\_file".

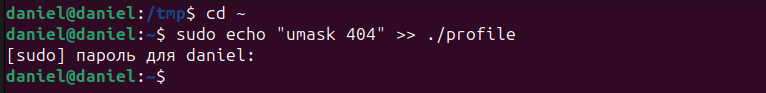
62. Зайду под пользователя neo и снова попробую удалить "morpheus\_file". На этот раз система не сделала это.

63. Пока я зaлогинен как neo, создам 2 файла: "spoon" и "neo\_file".

64. Удалю "spoon".

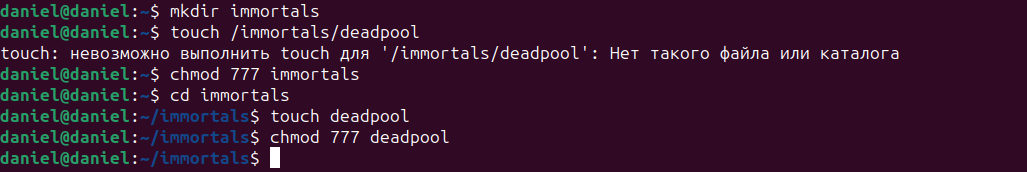
65. Вернусь под своего обычного пользователя и удалю файлы "morpheus\_file" и "neo\_file".

66. Установлю значение маски таким образом, чтобы все новые каталоги создавались с правами 404.



## INode флаги

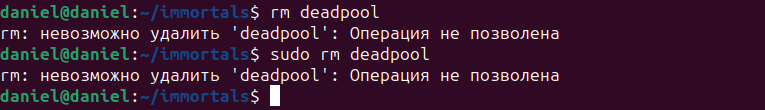
67. Создам каталог immortals и в нём файл deadpool. Обоим назначьте права 777.



68. Установлю для файла deadpool флаг i.

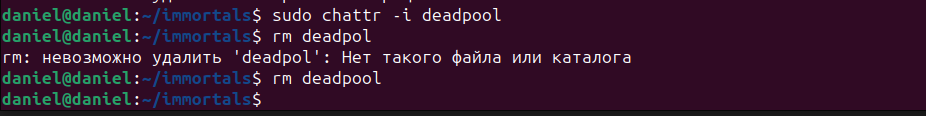


69. Выполню команду удаления файла deadpool от имени своего пользователя, а затем от root.



70. Удалю флаг i;

71. Удалю файл.



## Расширенные атрибуты

72. Создам файл trophy с содержимым: "After completing the laboratory work you will get a cake";

73. Добавлю к этому файлу расширенный атрибут user.top\_secret со значением "The сake is a lie";

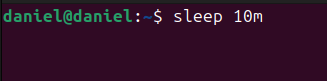
74. Проверю, что атрибут установлен;

75. Скопирую файл trophy в эту же папку под именем trophy2 и проверю список его атрибутов.

**5. Процессы**

**Список процессов в системе (утилиты ps, top, htop)**

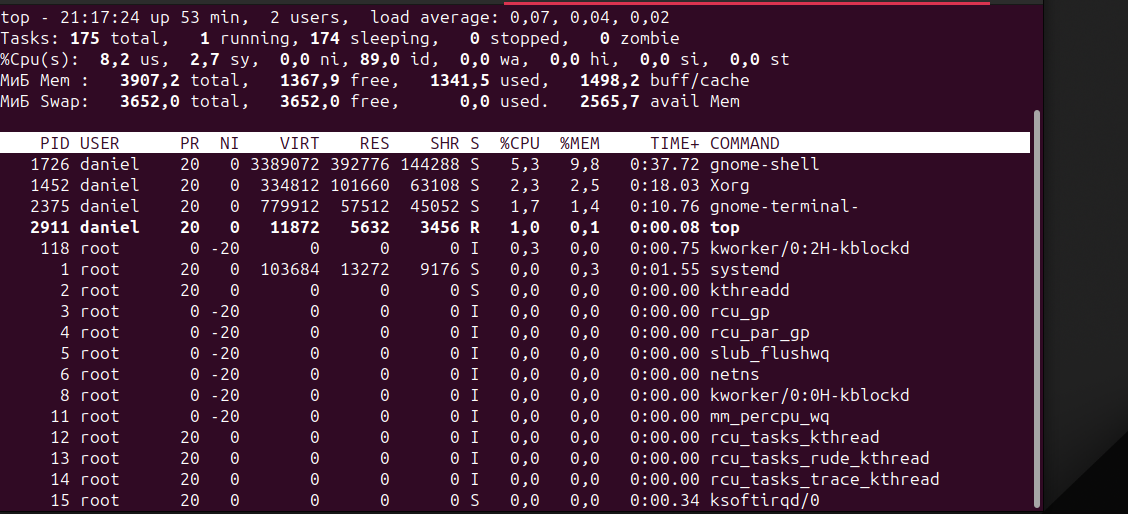
1. Запущу команду sleep 10m. Это запустит процесс, который захватит терминал и уснёт на 10 минут.



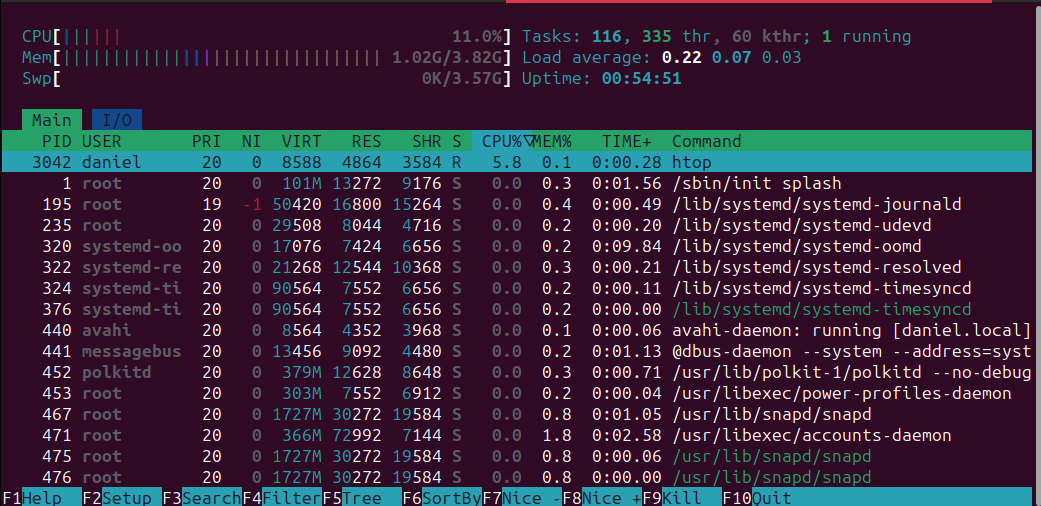
2. Открою новый терминал и воспользуюсь утилитой ps, чтобы определите PID процесса созданного командой sleep.



3. Теперь воспользуюсь утилитой top, чтобы найти тот же самый процесс и его PID.

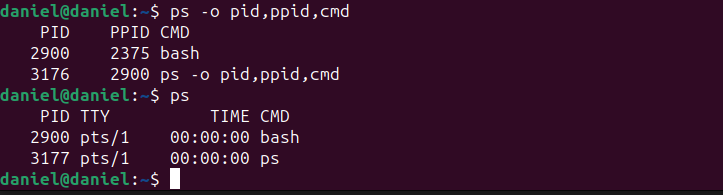


4. Для тех же целей воспользуюсь утилитой htop.

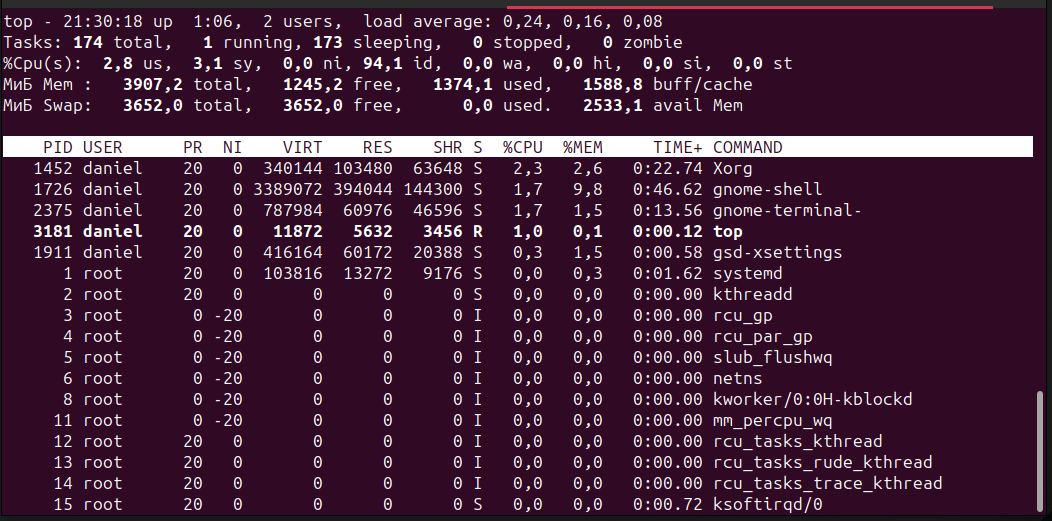


**Откуда берутся процессы**

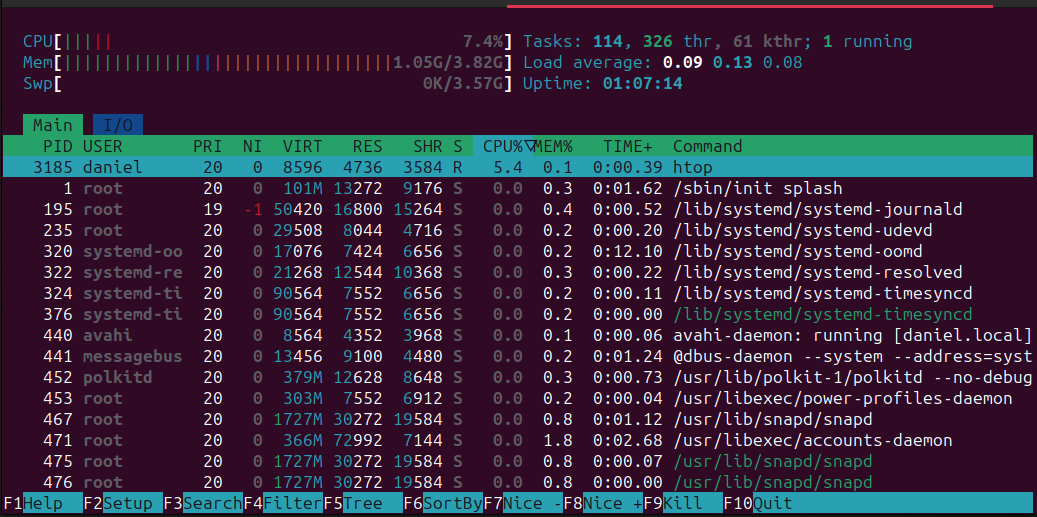
5. Настрою утилиту ps таким образом, чтобы она дополнительно выводила PPID. Затем запущу её без параметров и найду в списке процессов тот процесс, который является родительским для ps.



6. Запущу утилиту top и найду предка процесса PID которого определил на прошлом шаге.



7. Запущу утилиту htop и найду предка процесса.



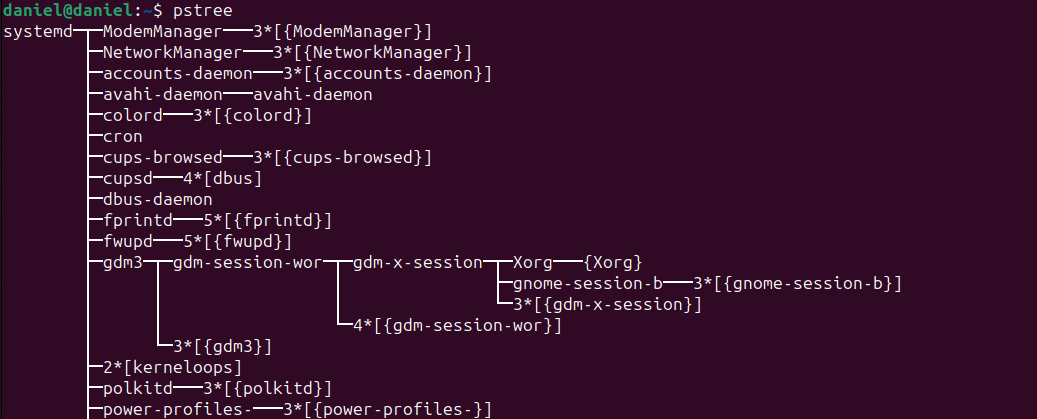
**Первые процессы в системе**

8. Посмотрю информацию о файле /sbin/init. Как видно это всего лишь символьная ссылка на настоящий init-процесс;

9. Воспользуюсь связкой ps и grep, чтобы найти среди всех процессов PID systemd;

10. Воспользуюсь связкой ps и grep, чтобы найти среди всех процессов PID kthreadd;

11. Воспользуюсь утилитой pstree и построю дерево всех процессов начиная от PID = 0.



12. В окне с ssh подключением запущу команду watch -n 0.5 "pstree -p", затем перейду в окно виртуальной машины и нажму сочетание клавиш Alt+ 2, затем Alt+ 3 и Alt+ 4.

13. Введу логин и пароль во втором терминале.

14. Теперь запущу процесс top. Как видно bash теперь стал родителем для top.

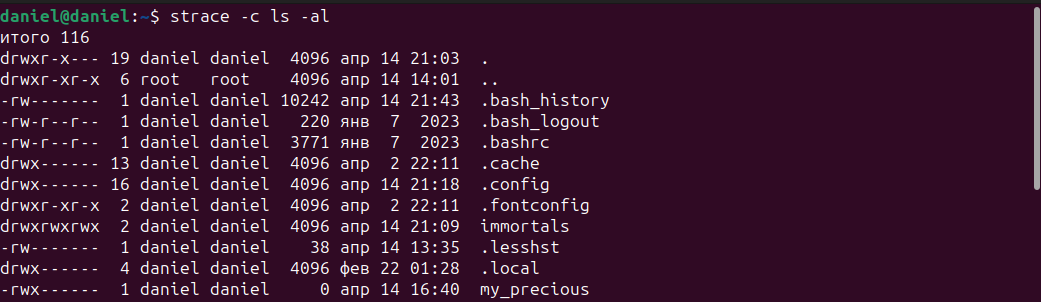
15. Разлогинюсь из второго терминала и остановлю watch;

16. Определю PID корневого sshd процесса, т.е. непосредственного потомка systemd.

17. Перезапущу watch следующим образом watch -n 0.5 pstree -p PID\_вашего\_sshd, а затем подключусь по ssh к серверу вторым окном.

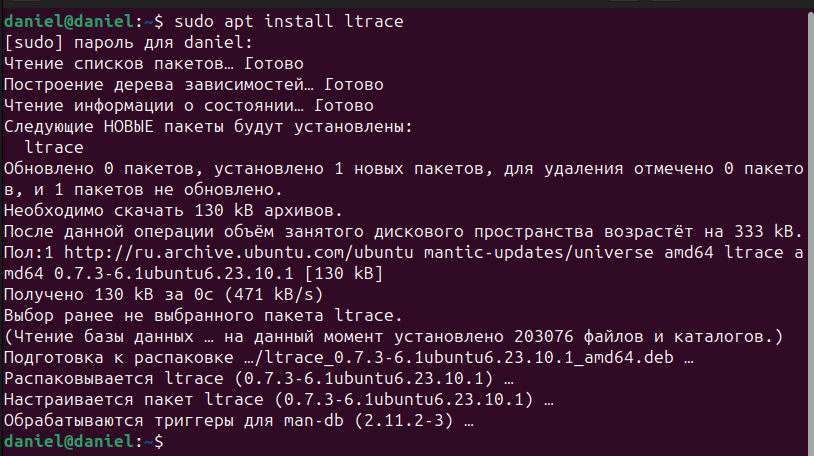
**Системные вызовы**

18. Введу команду strace -c ls -al.



19. Посмотрю конкретный системны вызова execve при помощи команды ` strace -e trace=execve ls -ls;

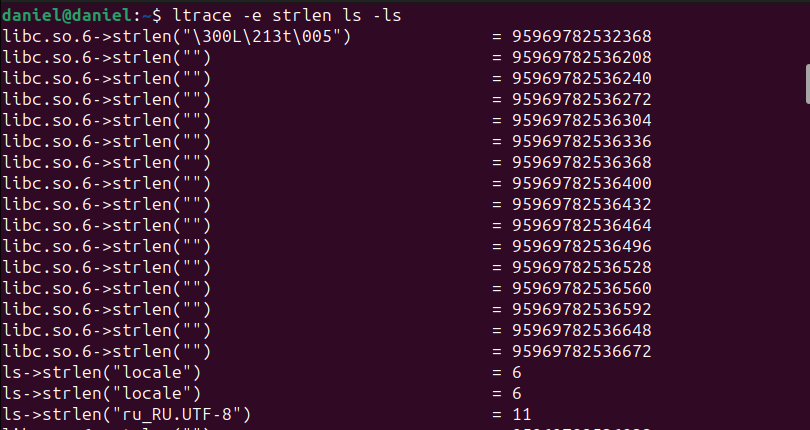
20. Установлю утилиту ltrace.



21. Введу команду ltrace -c ls -al.



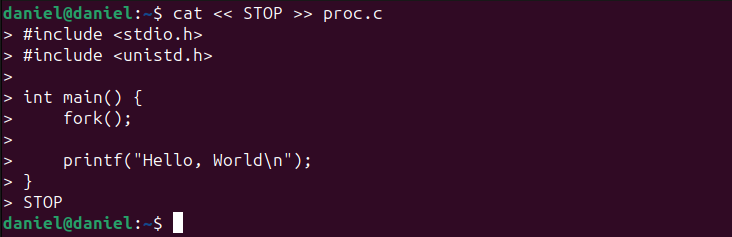
22. Отфильтрую только вызов функции strlen командой ltrace -e strlen ls –ls.



**Жизненный цикл процесса**

23. Установлю компилятор языка C: sudo apt install gcc;

24. В домашней директории создам файл proc.c.



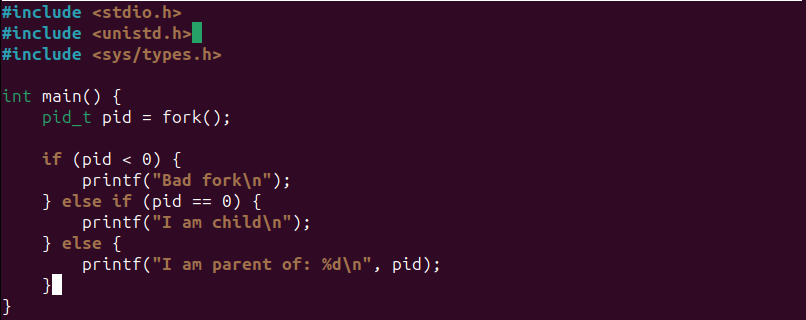
25. Скомпилирую исполняемый файл командой: gcc -o proc proc.c. В результате будет создан файл proc.



26. Запущу программу командой ./proc.



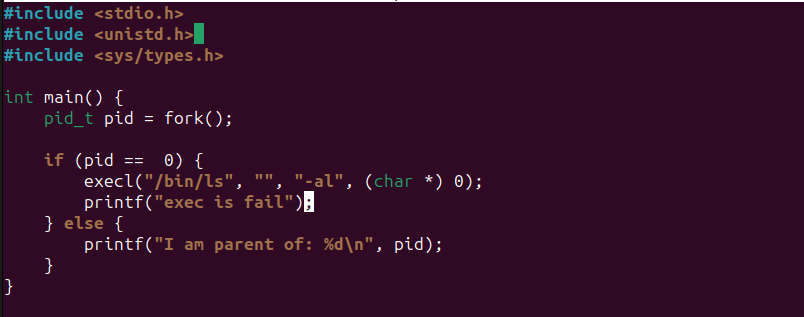
27. Изменю код в файле proc.c.



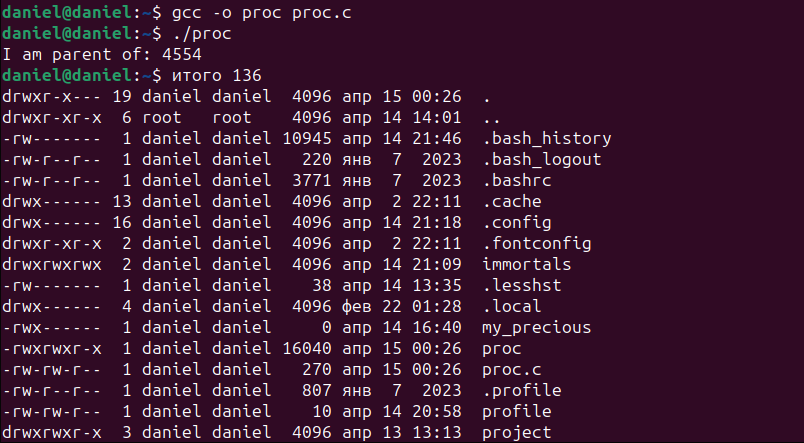
28. Скомпилирую и запущу программу.



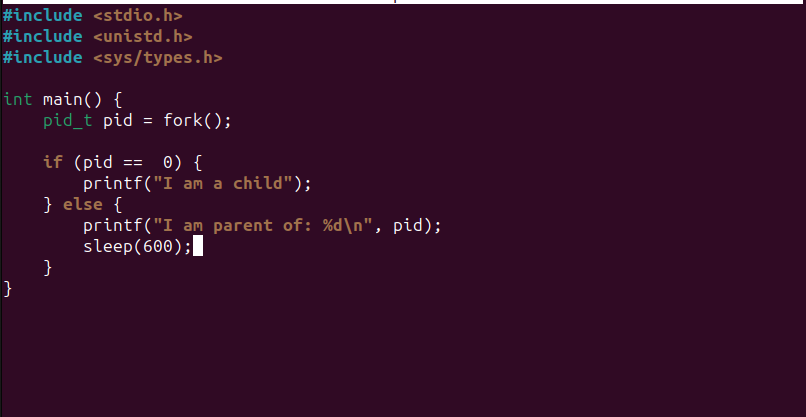
29. Снова изменю код в файле proc.c.



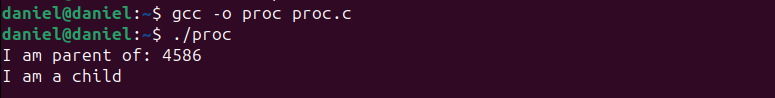
30. Скомпилирую и запущу программу.



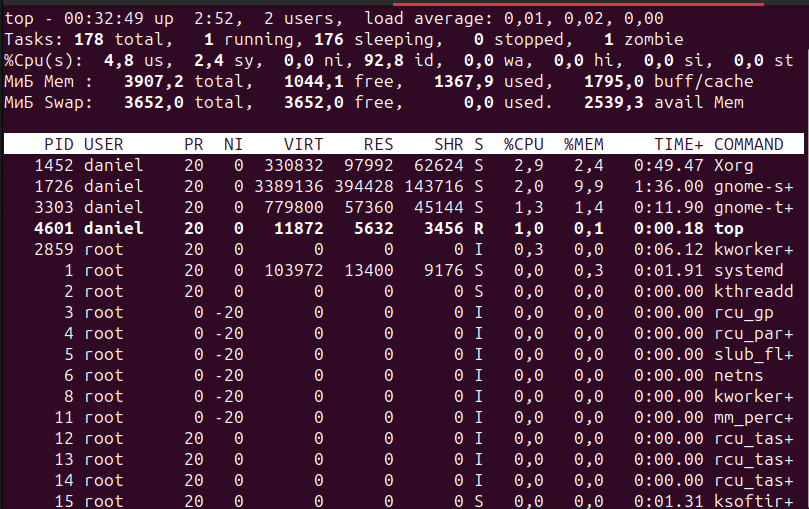
31. Снова изменю код в файле proc.c.



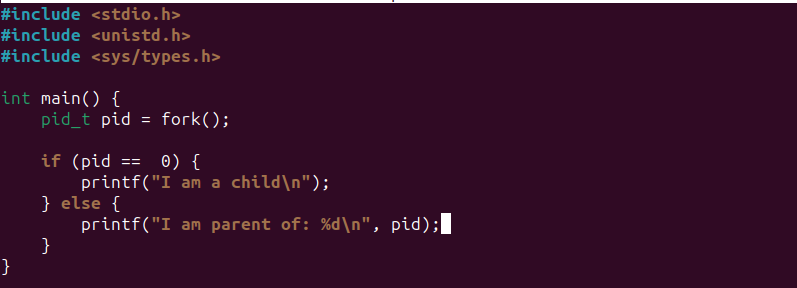
32. Скомпилирую и запущу программу.



33. Пока родительски процесс спит открою новый терминал и запустите программу top.



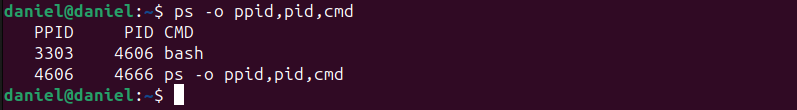
34. Снова изменю код в файле proc.c.



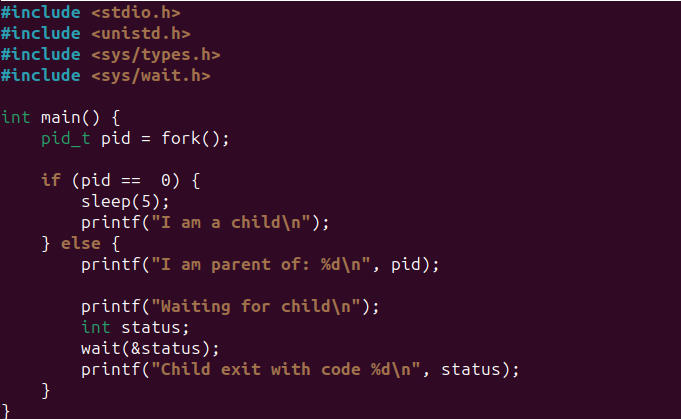
35. Скомпилирую и запущу программу.



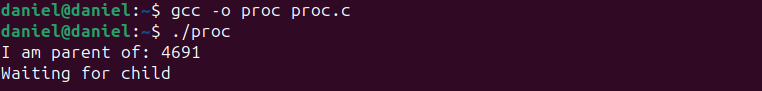
36. Запущу утилиту ps и настрою её на вывод PPID, PID и команды.



37. Снова изменю код в файле proc.c.



38. Скомпилирую и запущу программу.

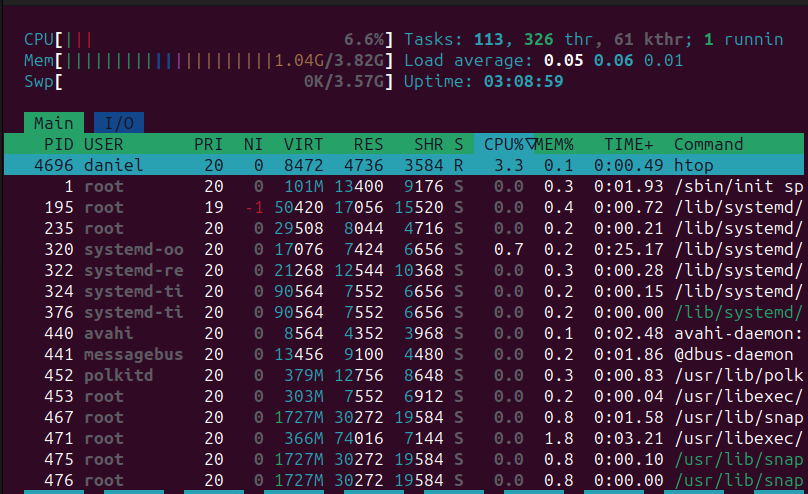


В результате, родительский процесс будет "висеть" пока дочерний не завершится.



**Процесс во время работы**

39. Запущу утилиту htop и посмотрю на её состояние в столбце S.



Т.к. утилита сейчас активно мониторит ресурсы и обновляет экран, ей требуется процессор и поэтому она находится в состоянии R. Большая часть остальных процессов находится в спящем состоянии.

40. Найду столбец NI (NICE), для утилиты htop сейчас там выставлено стандартное значение 0. Изменю его клавишами F8 и F7. Значение увеличивается, но не уменьшается.

41. Не закрывая htop перейду в новое окно и установлю для неё значение NICE равное -20 при помощи утилиты renice.

42. Теперь запущу команду sleep 10m, установив NICE в -10.

43. Отсортирую вывод утилиты htop по значению NICE от меньшего к большему. Значения, установленные ранее, применились.

44. Запущу команду sleep 10m затем нажму Ctrl + Z. Это привело к остановке процесса.

45. Запущу утилиту top и тоже остановлю её.

46. Запущу утилиту htop и отсортирую процессы по столбцу S. Обе утилиты находятся в состоянии сна. Остановлю htop.

47. Запущу команду sleep 10m, остановлю её.

48. Наберу команду jobs -l. Я получил список процессов находящихся в фоне с пометками, какие из них остановлены, а какие запущены.

49. Определю номер (первый столбец) процесса top в списке и переведу его из фона на передний план командой fg <номер>. Завершу top.

50. Определю номер первого процесса sleep и выполню команду bg <номер.

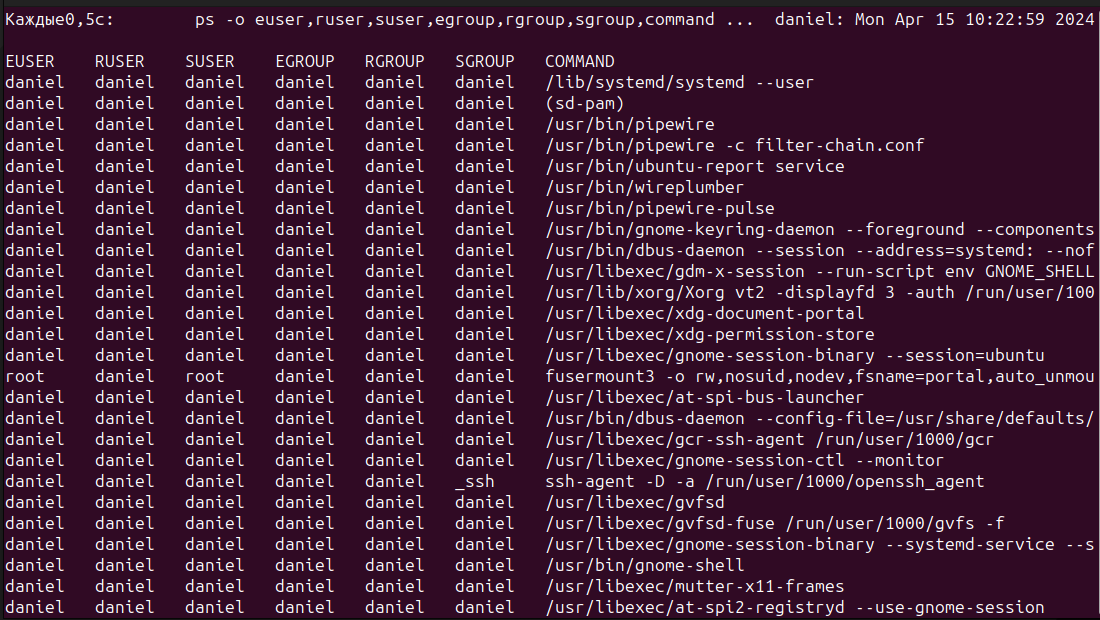
51. Посмотрю список фоновых процессов командой jobs –l.

52. Завершу все фоновые процессы.

**Процессы и права доступа**

53. Запущу команду:

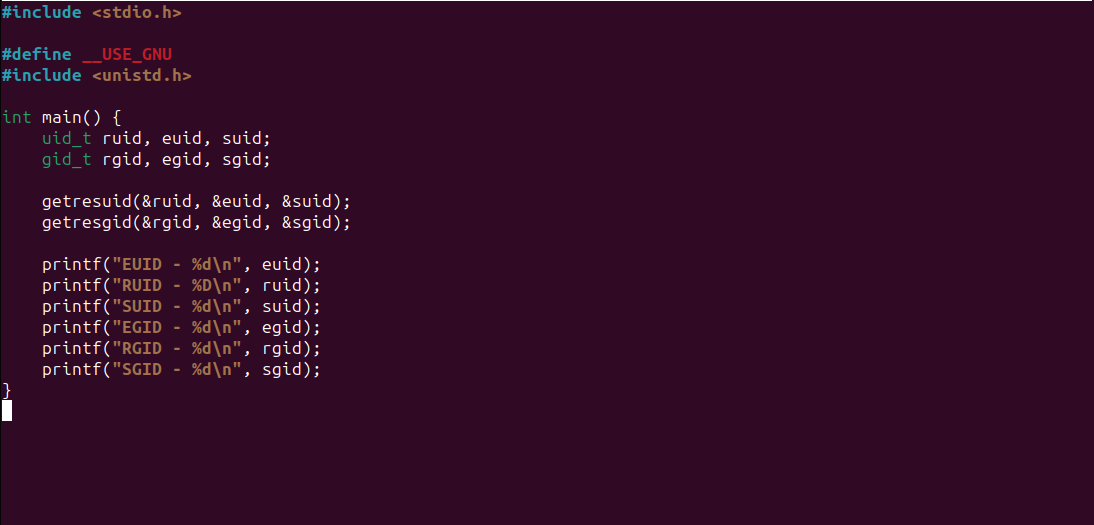
watch -n 0.5 ps -o euser,ruser,suser,egroup,rgroup,sgroup,command -U ваш\_логин



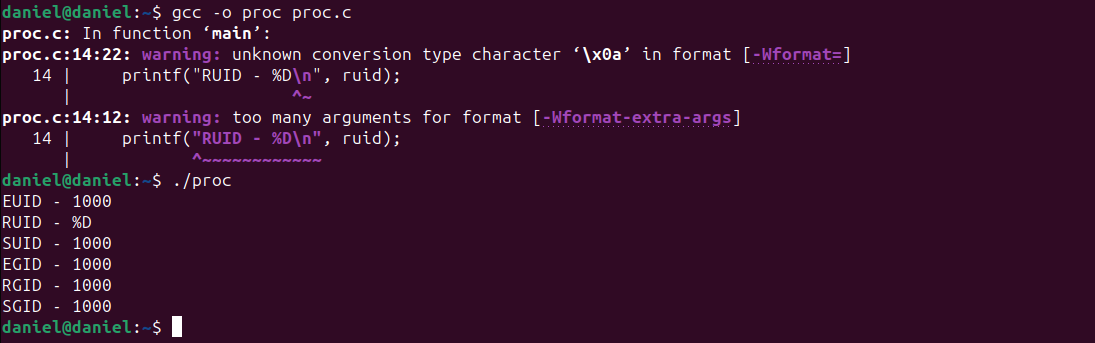
54. В другом терминале залогинюсь под своим основным пользователем и запущу утилиту passwd от имени своего пользователя.

55. Посмотрю вывод ps.

56. Снова изменю код proc.c:



57. Скомпилирую и запущу программу.



58. Снова изменю код proc.c:

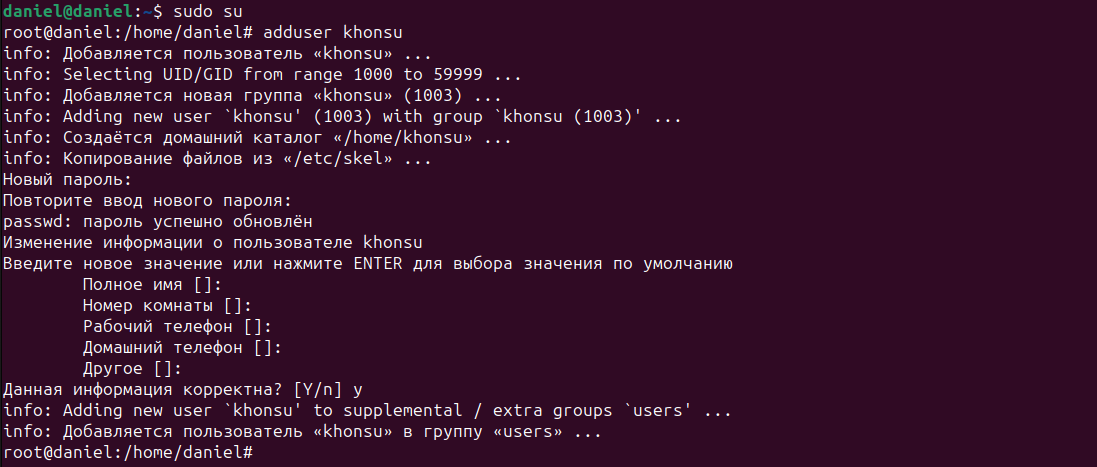


59. Скомпилирую и запущу программу.



**Утилита sudo**

60. Добавлю нового пользователя khonsu.



61. Открою второе окно и залогинюсь под ним, а в первом окне перейду под root.



62. В каталог /etc/sudoers.d добавлю файл 01\_khonsu содержащий:

khonsu <ip вашего сервера> = /usr/bin/ls

63. Проверю, что khonsu может выполнять команду sudo ls и sudo ls -al;

64. Изменю файл 01\_khonsu на:

khonsu <ip вашего сервера> = /usr/bin/ls -al

65. Изменю файл 01\_khonsu на:

khonsu <ip вашего сервера> = EXEC: NOPASSWD: /usr/bin/ls -al

66. Введу команду:

khonsu <ip вашего сервера> = CWD=/home/khonsu NOPASSWD: /usr/bin/ls -al

67. Создам пользователя marc\_spector.



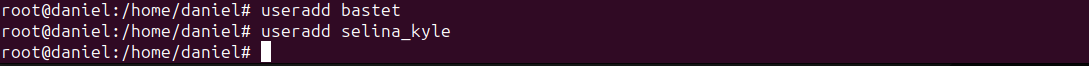
68. Добавлю в файл 01\_khonsu ещё одну запись:

khonsu <ip вашего сервера> = (marc\_spector:root) NOPASSWD: /usr/bin/touch

69. Под пользователем khonsu перейду в каталог /tmp и введу команду: sudo -u marc\_spector touch moon. Проверю владельца файла.

70. Допишу в конец последнего правила разрешение выполнять команду /usr/bin/rm от имени пользователя marc\_spector, но обязательно с паролем.

71. Создам пользователей bastet и selina\_kyle.



72. Замените содержимое файла 01\_khonsu на:

User\_Alias GODS = khonsu, bastet

Runas\_Alias HUMANS = marc\_spector, selina\_kyle

GODS ALL = (HUMANS) NOPASSWD: ALL

**Процессы и командная оболочка**

73. Выполню команду: sleep 10m & три раза.

74. Затем команду: ps -o pid,sid,pgid,comm -u мой\_логин.

75. Выполню команду: ps -o pid,sid,pgid,comm -u мой\_логин | less

**Межпроцессное взаимодействие**

76. Выполню команду: sleep 20m & три раза. На экране должны отобразится PID-ы процессов.

77. Отправлю всем трём сигнал SIGSTOP.

78. Отправлю последнему запушённому сигнал SIGKILL, а первому.

79. Определю PID своей командной оболочки (echo $$) и пошлю ей сигнал SIGHUP;

80. Зайду обратно на сервер и проверю список фоновых процессов.

81. Запущу команду python3 -m http.server 8000 & и проверю в браузере, что сервер работает.

82. Разлогинюсь и снова проверю вывод браузера. Сервер перестал работать.

83. Зайду на сервер и проверю все процессы своего пользователя.

84. Отправлю зависшему процессу сигнал SIGKILL и перезапущу сервер командой nohup python3 -m http.server 8000 &

85. Разлогинюсь и проверю, что сервер по-прежнему работает.