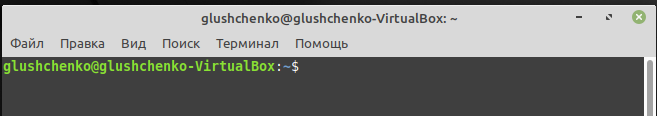
## Продвинутая работа с bash

### Цель работы

Продемонстрировать основные навыки работы в командной строке Linux с использованием возможностей интерпретатора bash.

### Задания для выполнения

1. Запускаем командную строку



1. Создаём переменную и присваиваем любое значение

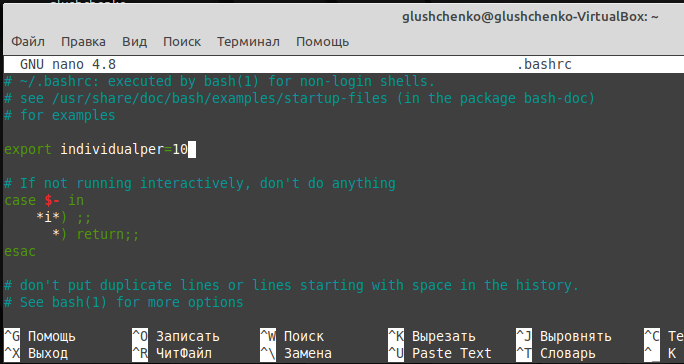


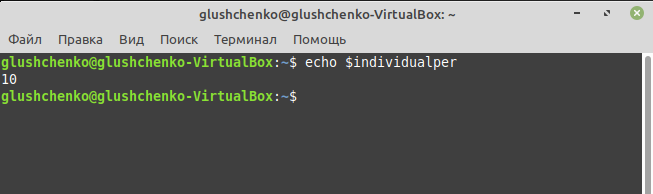
1. Отображаем её





1. Проверяем, что переменная сохраняет своё значение после рестарта с помощью echo $a





1. Отображаем переменную $PATH

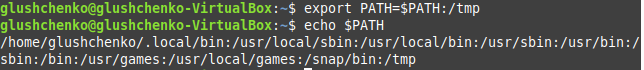


1. Разбираемся с тем как она работает

**$PATH** — это переменная среды, используемая для указания оболочке, где искать исполняемые файлы. **$PATH** обеспечивает большую гибкость и безопасность для систем [**Linux**](https://blog.sedicomm.com/2018/01/21/kak-ustanovit-golang-yazyk-programmirovaniya-go-v-linux/), и, безусловно, можно сказать, что это одна из самых важных переменных среды.

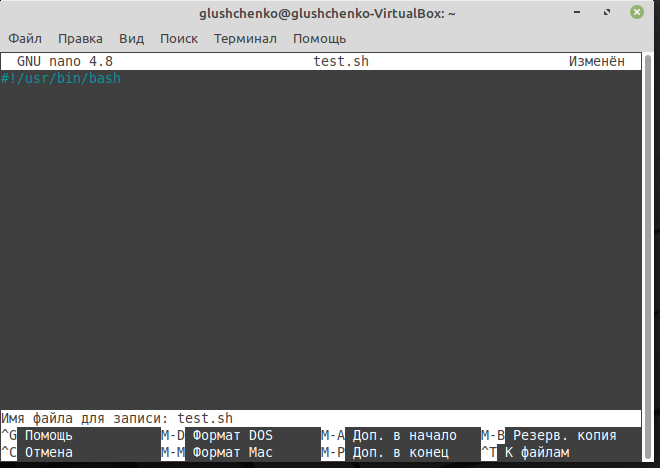
**По команде echo $PATH** отображается список каталогов, разделенных двоеточиями. На самом деле хорошей практикой является создание локальной папки **bin** для пользователя, где пользователи могут размещать исполняемые файлы. Каждый пользователь будет иметь свою отдельную папку для хранения своего содержимого. Это также является хорошей мерой для обеспечения безопасности вашей системы.

1. Делаем так, чтобы добавился ещё один путь — /tmp

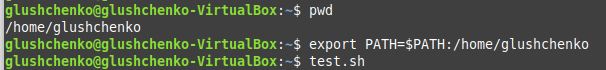


1. Пишем пустой скрипт с расширением .sh и запускаем его. Скрипт - это простой текстовый файл, в первой строке которого содержится шебанг:

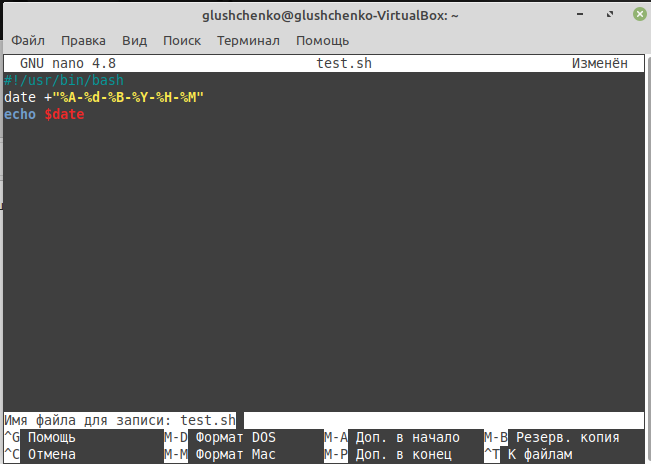
#!/usr/bin/bash



1. Делаем так, чтобы скрипт можно было запускать без указания пути

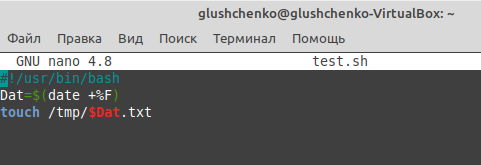


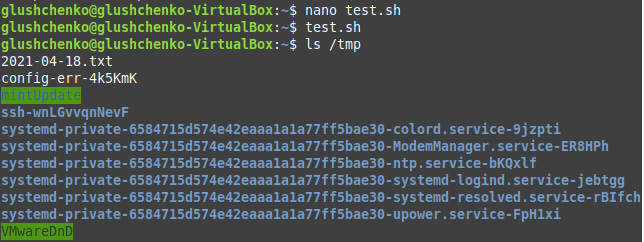
1. Делаем так, чтобы скрипт инициализировал DATE и задавал значение времени, затем выводил на экран



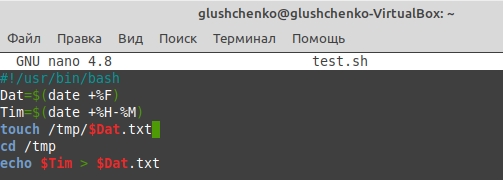
День недели/число/ месяц в родительном падеже/ год/час/минуты

1. Делаем так, чтобы вместо вывода на экран создавался файл в директории подобный /tmp/2017-08-17.txt





1. Делаем так, чтобы в файл записывалось время создания

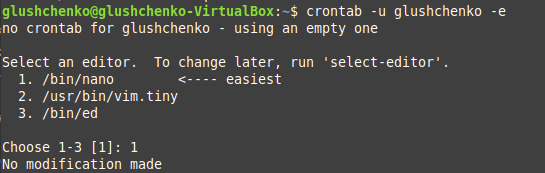


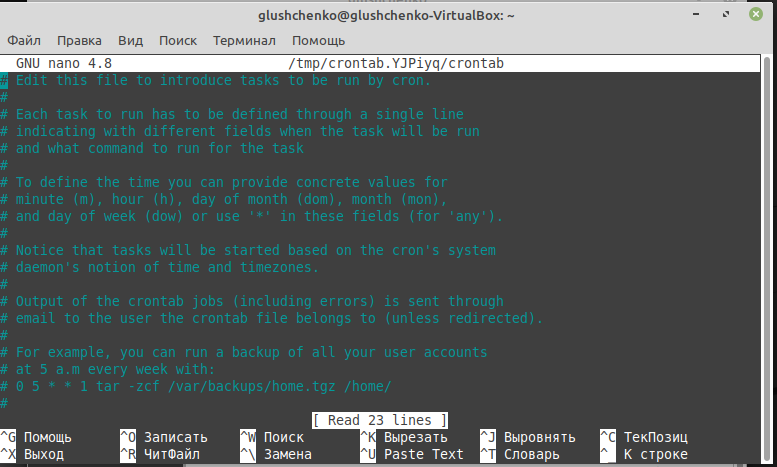
1. Используя CRON заставляем скрипт исполняться ежеминутно

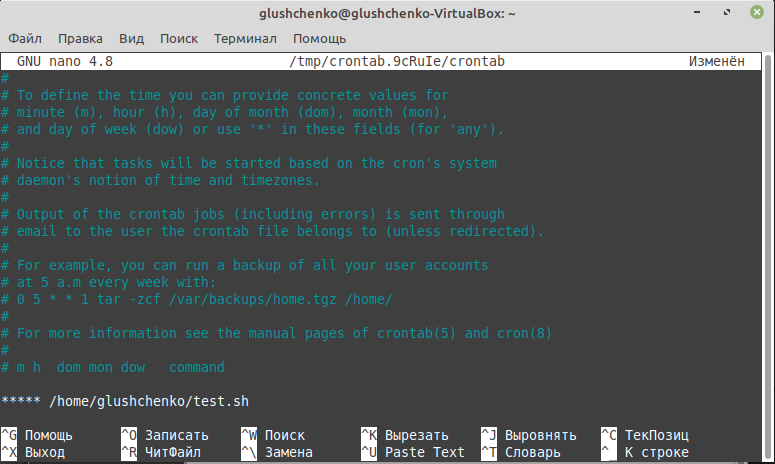
Cron - это сервис, как и большинство других сервисов Linux, он запускается при старте системы и работает в фоновом режиме. Его основная задача выполнять нужные процессы в нужное время. Существует несколько конфигурационных файлов, из которых он берет информацию о том что и когда нужно выполнять. Сервис открывает файл /etc/crontab, в котором указаны все нужные данные.

**минута час день месяц день\_недели /путь/к/исполняемому/файлу**

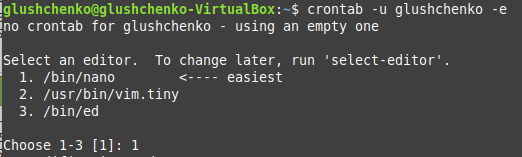
Через команду crontab -u $username -e получаем список cronjobs, работающих под управлением пользователя

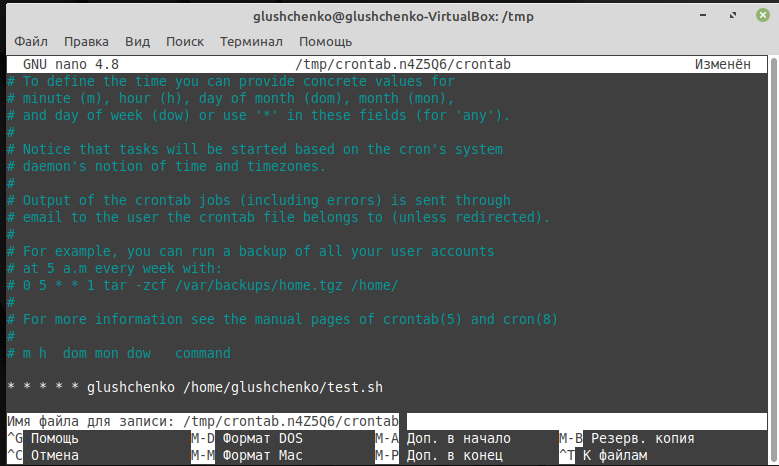
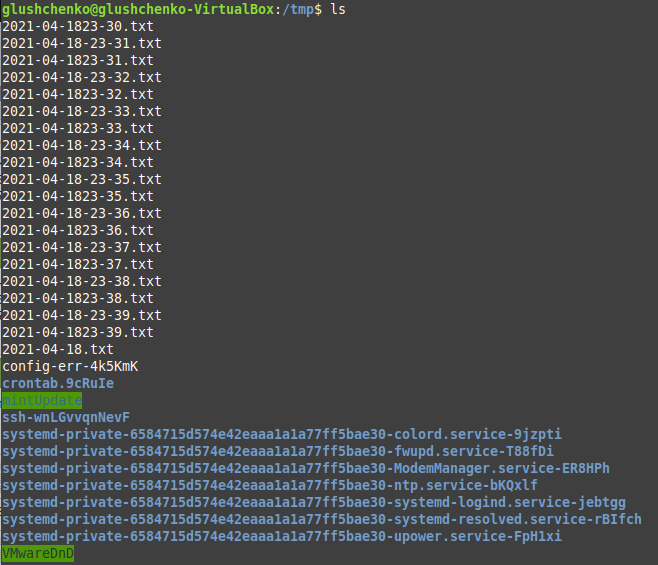




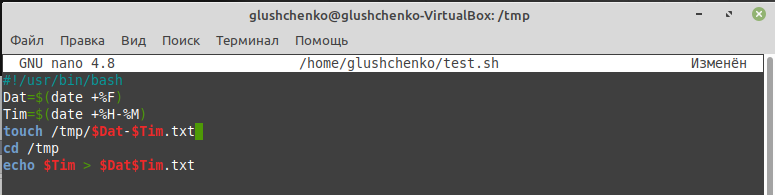
1. Через 10 минут проверяем, чтобы в директории было столько же файлов, сколько прошло минут



1. Обнаруживаем ошибку

Я предотвратил ошибку перезаписывания одного файла с разным временем и переписал скрипт так, чтобы каждый раз создавался новый файл.



Так как цикличное выполнение начинается на следующую минуту, то за 10 минут создается 9 файлов.

Остановить выполнение:



1. Используя find делаем так, чтобы директория проверялась раз в 10 минут и удаляла файлы, созданные больше, чем 17 минут назад

Find - это одна из наиболее важных и часто используемых утилит системы Linux. Это команда для поиска файлов и каталогов на основе специальных условий. Ее можно использовать в различных обстоятельствах, например, для поиска файлов по разрешениям, владельцам, группам, типу, размеру и другим подобным критериям.

Команда find имеет такой синтаксис:

**find [папка] [параметры] критерий шаблон [действие]**

**Папка**- каталог в котором будем искать

**Параметры** - дополнительные параметры, например, глубина поиска, и т д

**Критерий** - по какому критерию будем искать: имя, дата создания, права, владелец и т д.

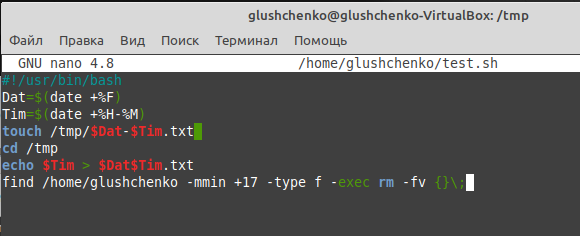
**Шаблон** - непосредственно значение по которому будем отбирать файлы.

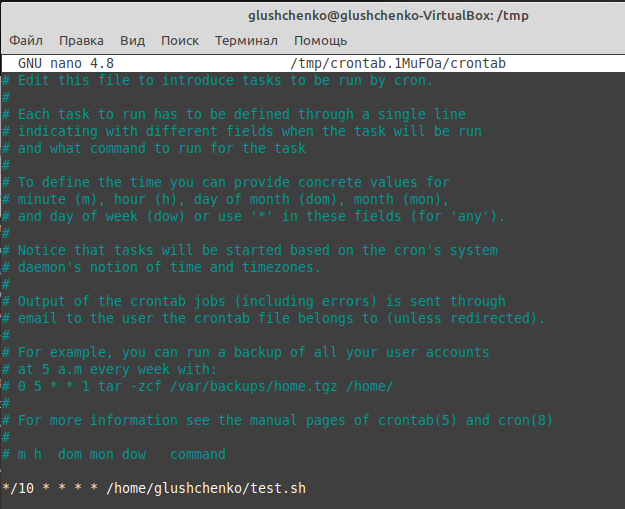
-type f – искать только файлы

Бывают и такие случаи, что программы "сходят с ума" и заполняют директории тысячами мелких файлов, при этом вы не сможете просто использовать команду rm \* по той причине, что командная оболочка не в состоянии заменить символ \* на имена всех этих файлов, зато в состоянии удалить эти файлы по очереди:

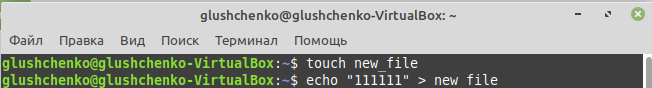
find \* -exec rm {} \;

-f - Игнорировать несуществующие файлы и аргументы. Никогда не выдавать запросы на подтверждение удаления.

-v Выводить информацию об удаляемых файлах. 



1. Создаём пустой файл и вписываем что-нибудь



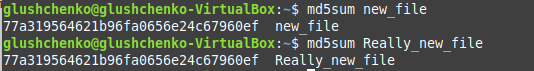
1. Копируем с сохранением всех прав в другой файл с любым названием и содержимым



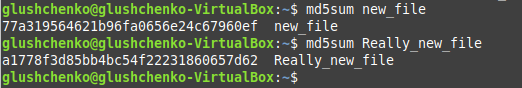
1. Сравниваем контрольные суммы

Контрольная сумма - это цифра или строка, которая вычисляется путем суммирования всех цифр нужных данных. Ее можно использовать в дальнейшем для обнаружения ошибок в проверяемых данных при хранении или передаче. Тогда контрольная сумма пересчитывается еще раз и полученное значение сверяется с предыдущим.

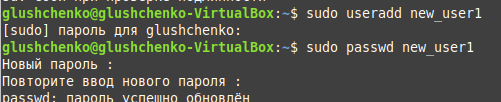
Контрольные суммы Linux с вычисляемые по алгоритму MD5 (Message Digest 5) могут быть использованы для проверки целостности строк или файлов. MD5 сумма - это 128 битная строка, которая состоит из букв и цифр. Суть алгоритма MD5 в том, что для конкретного файла или строки будет генерироваться 128 битный хэш, и он будет одинаковым на всех машинах, если файлы идентичны. Трудно найти два разных файла, которые бы выдали одинаковые хэши.



1. Меняем содержимое любого из файлов и смотрим разницу в контрольных суммах

  
Контрольные суммы изменились

1. Создаём пользователя и задаём ему пароль



1. Разбираемся с /etc/passwd, /etc/shadow, /etc/group

При использовании теневых паролей в /etc/passwd и /etc/group вместо самого пароля устанавливается символ 'x', что и является указанием на хранение пароля в /etc/shadow или /etc/gshadow.

Файл shadow хранит защищенную информацию о пользователях, а также обеспечивает механизмы устаревания паролей и учетных записей.

Учитывая ежедневно увеличивающиеся требования к безопасности, в Linux есть возможность использовать скрытые пароли. Файлы /etc/passwd и /etc/group доступны для чтения всем пользователям, что является довольно большой брешью в безопасности системы. Именно поэтому в современных версиях Linux предпочтительнее использовать скрытые пароли. Такие пароли располагаются в файлах /etc/shadow и /etc/gshadow, для паролей пользователей и групп соответственно.

Все группы, созданные в системе, находятся в файле /etc/group.

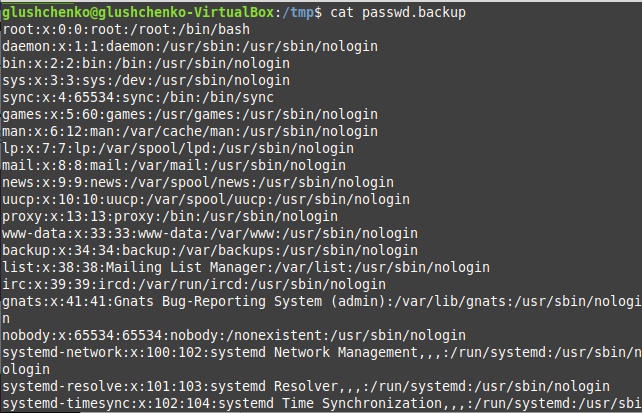
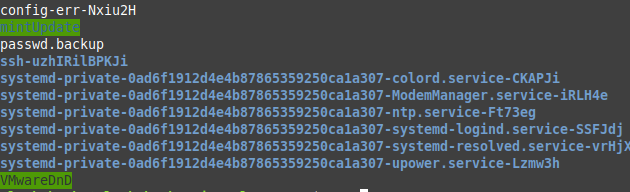
/etc/passwd (от англ. password — пароль) — файл, содержащий в текстовом формате список пользовательских учётных записей

1. Делаем так, чтобы, если нет файла /tmp/passwd.backup, /etc/passwd копировался в /tmp/passwd.backup, а если /tmp/passwd.backup есть, то сравнивался с /etc/passwd

Проверка на существование файла -f

diff -q - выводить только отличия файлов;





1. Если различаются, то пусть копируется, если нет, пусть завершает работу скрипта

diff -s - выводить только совпадающие части (чтобы выводилось на экран совпадают файлы или нет)

