## 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

Изображение выглядит как текст

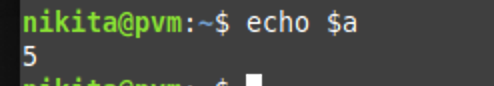
Автоматически созданное описание

1. Создаём переменную и присваиваем любое значение

Изображение выглядит как текст

Автоматически созданное описание

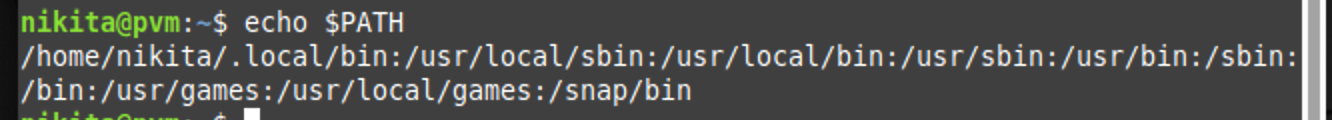
1. Отображаем её



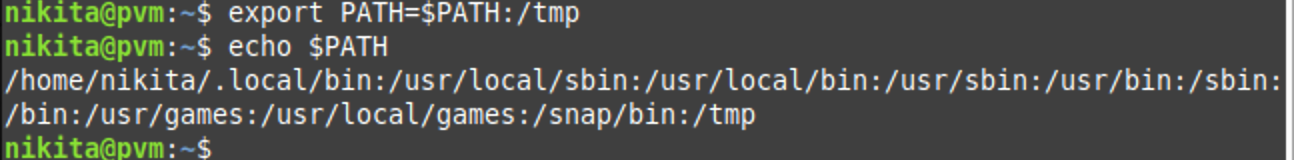
1. Проверяем, что переменная сохраняет своё значение после рестарта с помощью echo $a

Не сохраняет

1. Отображаем переменную $PATH



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

Изображение выглядит как текст

Автоматически созданное описание

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

Изображение выглядит как текст

Автоматически созданное описание

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

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

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

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

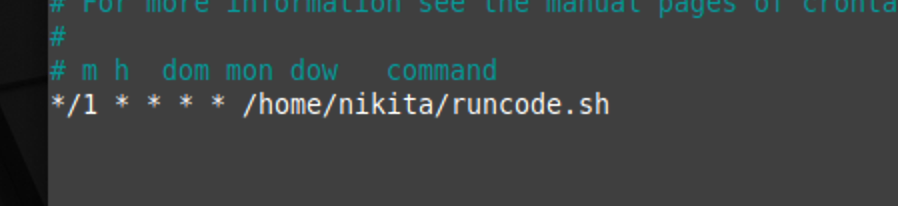
Автоматически созданное описание

1. Делаем так, чтобы в файл записывалось время создания

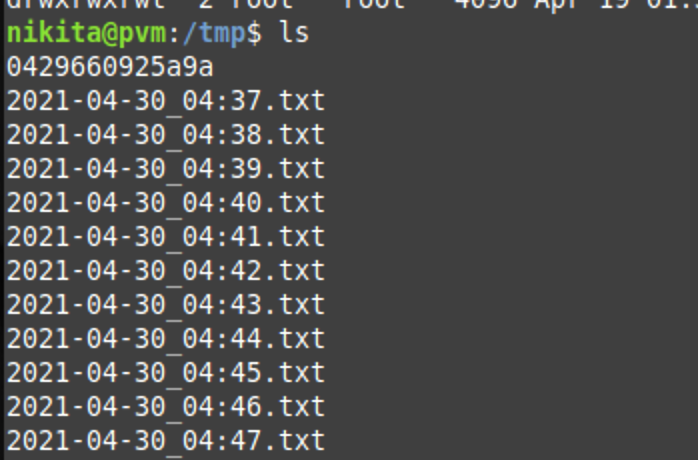
Изображение выглядит как текст

Автоматически созданное описание

1. Используя CRON заставляем скрипт исполняться ежеминутно



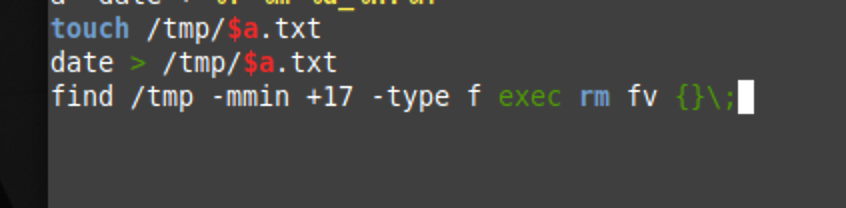
1. Через 10 минут проверяем, чтобы в директории было столько же файлов, сколько прошло минут



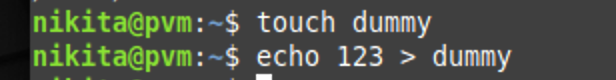
1. Обнаруживаем ошибку

Ошибки нет, так как каждый раз файлы создаются с разным временем

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



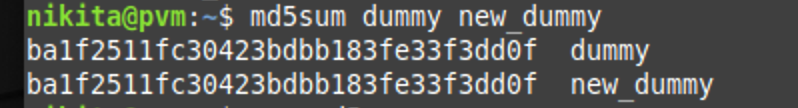
1. Создаём пустой файл и вписываем что-нибудь



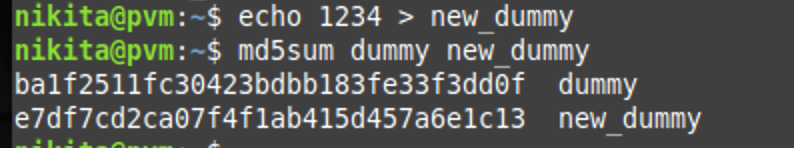
1. Копируем с сохранением всех прав в другой файл с любым названием и содержимым



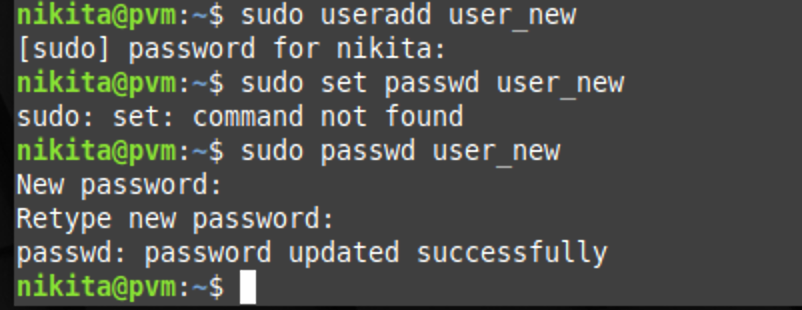
1. Сравниваем контрольные суммы



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



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.

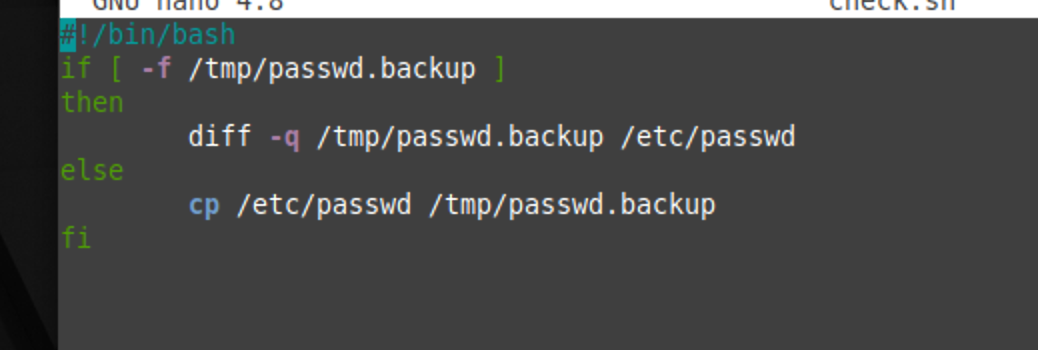
Файл shadow хранит защищенную информацию о пользователях, а также обеспечивает механизмы устаревания паролей и учетных записей.

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







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

