Сделать скрипт fullscript.sh , который покажет:

выведет строчку со словами:

Hello Student!

попросит ввести Ваше имя

выведет строчку:

“Welcome to terminal $name"

создаст папку test в каталоге tmp

создаст файл под названием mydate.txt в папке /tmp/test

запишет в новосозданный файл mydate.txt текущее время (не просто дату)

выведет на экран надпись “Сохраняем данные” и начнет отсчет до 10 с интервалом в полсекунды.

выведет на экран “Data saved. continue work”

дозапишет в файл mydate.txt все сведения о состоянии нашего диска в человеческом виде.

скопируем файл mydate.txt в новую подпапку mydate в папке /opt/ с переименованием файла в newmydate.txt

снова сделать интервал ожидания, но уже с отсчетом в пять раз по одной секунде и вывести на экран фразу “Well done Boss”

сделать файл исполняемым из любого места в системе.

cp /tmp/fullscript.sh /bin

localhost:~# chmod +x /bin/fullscript.sh

localhost:~# fullscript.sh

#!/bin/bash

**echo** **"hello Student"**

**echo -n** **"Enter your name: "**

read name

**echo** **"Welcome to terminal $name"**

**mkdir** /tmp/test/

date +**"%H\_%M\_%S"** > /tmp/test/mydate.txt

**echo** **"Saving Data"**

for run in {1..10}

do

**echo** **$run**

sleep 0.5

done

**echo** **"Data saved, continue work"**

df **-h** >> /tmp/test/mydate.txt

**mkdir** /opt/mydate

**cp** /tmp/test/mydate.txt /opt/mydate/newmydate.txt

for run in {1..5}

do

**echo** **$run**

sleep 1

done

**echo** **"Well done Boss"**

## **Что такое Cron и crontab?**

Если в двух словах, то Cron – это планировщик задач. Если подробнее, то это утилита, позволяющая выполнять скрипты на сервере в назначенное время с заранее определенной периодичностью.

К примеру, у вас есть скрипт, который собирает какие-либо статистические данные каждый день в 6 часов вечера. Такие скрипты называют «заданиями», а их логика описывается в специальных файлах под названием сrontab.

crontab – это таблица с расписанием запуска скриптов и программ, оформленная в специальном формате, который умеет считывать компьютер. Для каждого пользователя системы создается отдельный crontab-файл со своим расписанием. Эта встроенная в Linux утилита доступна на низком уровне в каждом дистрибутиве.

В Linux-дистрибутивах с поддержкой systemd Cron считается устаревшим решением, его заменили утилитой systemd.timer. Ее предназначение и функциональность не отличается, но фактически частота использования Cron все еще выше.

### **Для чего обычно используют Cron?**

Обычно Cron заставляют повторять вполне очевидные задачи в духе регулярного создания резервных копий данных. Но это не все.

Некоторые пользователи с помощью планировщика корректируют системное время. На многих компьютерах оно настраивается через Network Time Protocol. А так как этот протокол настраивает только время ОС, время, установленное для «железа», может отличаться. Cron позволяют регулярно корректировать время, установленное для аппаратного обеспечения, в соответствии со временем ОС.

Еще один популярный сценарий – создание оповещений, появляющихся каждое утро и рассказывающих о состоянии компьютера. В эти сообщения может входить любая полезная для пользователя информация.

Cron иногда работает даже без ведома пользователя. Эту утилиту используют такие сервисы, как Logwatch, logrotate и Rootkit Hunter. Повторяющиеся задачи они настраивают, как и пользователи, через Cron.

С помощью Cron пользователи автоматизируют самые разные задачи, сокращая вмешательство системного администратора в работу сервера.

Давайте посмотрим на практике, как это работает.

Говоря простым языком - cron - это планировщик, который запускает по расписанию какую-то программу или какой-то скрипт.

Штука очень надежная, лично сталкивался с несколькими огромными компаниями, которые используют крон для обмена финансовой информацией при помощи этого планировщика. Он запускался через определенный промежуток времени и проверял, появились ли новые файлы в системе.

Сам планировщик называется **cron**. Расписание планировщика называется **crontab**.

(заходим на сервер и будем смотреть)

**crontab -l**

показывает список задач текущего пользователя.

**touch /tmp/script2.sh /tmp/output2.txt**

**echo -e '#!/bin/bash\n date\n echo "it works"' > /tmp/script2.sh**

**chmod +x /tmp/script2.sh**

Что мы делаем? Мы файлом создаем будущий файлик скрипта и создаем будущий файл, куда этот скрипт будет что-то писать:

**touch /tmp/script2.sh /tmp/output2.txt**

Дальше применяем читерство:

**echo -e '#!/bin/bash\n date\n echo "it works"' > /tmp/script2.sh**

Все помнят команду, с помощью которой мы выводим на экран, echo, так вот у него есть ключик -е, который интерпретирует **\n \s** и другие штуки в разные конечные символы. В нашем случае, бэк **\n** будет интерпретирован как перенос строки.

По итогу мы передадим все, что у нас в одинарных кавычках:

**#!/bin/bash\n date\n echo "it works"**

с переносом строки в новый файл:

**/tmp/script2.sh**

Это еще один способ создания скриптов, одной строкой, не заходя в редактор нано или другие.

теперь, если мы на него посмотрим, мы увидим:

**cat /tmp/script2.sh**

#!/bin/bash

date

echo "it works"

[ec2-user@ip-172-31-39-17 ~]$

Этот скрипт является полноценным. Видите, у него присутствует и шебанг и прочее.

Чтобы он стал исполняемым, мы меняем ему права на выполнение:

**chmod +x /tmp/script2.sh**

единственное, что я сейчас сделаю, я его скопирую, чтобы его потом не убили перезаписью:

**cp /tmp/script2.sh /tmp/chucron.sh**

**cp /tmp/script2.sh /tmp/chumak.sh**

Теперь давайте посмотрим, что у нас тут лежит:

**ls -la /tmp**

**Видим кучу скриптов и непонятных файлов. Значит записало.**

**[ec2-user@ip-172-31-39-17 ~]$ ls -la /tmp**

**total 820**

**drwxrwxrwt 8 root root 286 Feb 22 12:46 .**

**dr-xr-xr-x 19 root root 269 Feb 21 14:40 ..**

**drwxrwxrwt 2 root root 6 Jan 11 12:04 .ICE-unix**

**drwxrwxrwt 2 root root 6 Jan 11 12:04 .Test-unix**

**drwxrwxrwt 2 root root 6 Jan 11 12:04 .X11-unix**

**drwxrwxrwt 2 root root 6 Jan 11 12:04 .XIM-unix**

**-rw------- 1 ec2-user ec2-user 12288 Feb 14 22:18 .crontab.on2Swp.swp**

**drwxrwxrwt 2 root root 6 Jan 11 12:04 .font-unix**

**-rwxrwxr-x 1 ec2-user ec2-user 36 Feb 14 21:48 cron.sh**

**-rw-r--r-- 1 ec2-user ec2-user 810732 Feb 14 21:33 output.txt**

**-rw-rw-r-- 1 ec2-user ec2-user 0 Feb 22 12:32 output2.txt**

**-rwxrwxr-x 1 ec2-user ec2-user 36 Feb 14 21:37 script.sh**

**-rwxrwxr-x 1 ec2-user ec2-user 35 Feb 22 12:34 script2.sh**

**drwx------ 3 root root 17 Jan 11 12:04**

Теперь давайте скажем нашему планировщику - а запусти ты нам этот скрипт, с интервалом в минуту.

Для того, чтобы это сделать мы запускаем редактор crontab:

**crontab -e**

**(пример с VI - этот редактор открывается по умолчанию)**

**Мы все делали в nano**

давайте удалим, все, что есть в файле. Напомню, что удалять можно построчно командой **dd**

Сохранимся и выйдем:

**ZZ**

теперь снова откроем:

**crontab -e**

и добавим странное сочетание (напомню, чтобы войти в режим редактирования нужно нажать **i**)

(можно не перезаходить, но я хочу, чтобы мы повторили команды)

набираем странное сочетание:

**\* \* \* \* \* \* (через пробел)**

\* \* \* \* \* /tmp/script2.sh >> /tmp/output2.txt

выходим с сохранением.

теперь давайте, чтобы он у нас был перед глазами:

**[ec2-user@ip-172-31-39-17 ~]$ crontab -l**

**\* \* \* \* \* /tmp/script.sh >> /tmp/output2.txt**

**Теперь задание какое у него есть:**

**/tmp/script2.sh дописать в /tmp/output2.txt**

Теперь, что же значат наши звездочки. В crontab каждая звездочка означает определенное время. А именно:

<https://crontab.guru/>

Все заходим на сайт и смотрим.

Выглядит страшно, не правда-ли? Но сверху есть подсказка и она нам может помочь.

Мы же выбрали все звездочки, как в ГТА. Но не бойтесь, полиции не будет)

Звездочки означают когда делать, а после них - что делать.

Тоесть у нас будет дозаписывать в файл слова It works каждую минуту.

Давайте убедимся:

**cat /tmp/output2.txt**

**Tue Feb 22 13:31:01 UTC 2022**

**it works**

**Tue Feb 22 13:32:01 UTC 2022**

**it works**

**Tue Feb 22 13:33:01 UTC 2022**

**it works**

**Tue Feb 22 13:34:01 UTC 2022**

**it works**

**Tue Feb 22 13:35:01 UTC 2022**

**it works**

**Tue Feb 22 13:36:01 UTC 2022**

**it works**

**Tue Feb 22 13:37:01 UTC 2022**

**it works**

**Tue Feb 22 13:38:01 UTC 2022**

**it works**

**Tue Feb 22 13:39:01 UTC 2022**

**it works**

**Tue Feb 22 13:40:01 UTC 2022**

**it works**

**Tue Feb 22 13:41:01 UTC 2022**

**it works**

ссылка на Хабр по крону:

<https://habr.com/ru/company/badoo/blog/468061/>

**Не забывайте про копирование средней кнопкой мыши! (у кого она есть)**