

УРОК 13. CRON, CRONTAB + TAR

ЧТО ТАКОЕ TAR	2
ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ	4
ЧТО ТАКОЕ CRON	6
CRONTAB: РАСПИСАНИЕ CRON	7
CRONTAB: СОЗДАНИЕ ЗАДАНИЯ	9



ЧТО ТАКОЕ TAR



Tar - это утилита командной строки в операционной системе Linux, которая используется для архивации файлов и директорий в один файл.

"tar" происходит от "Tape ARchive" (ленточный архив), потому что изначально она использовалась для архивации данных на магнитных лентах.

Основная функциональность tar заключается в создании архивов, добавлении файлов в архив, извлечении файлов из архива и просмотре содержимого архива. В отличие от других форматов архивов, таких как ZIP, tar не сжимает данные, он просто создает один файл, включающий в себя все выбранные файлы и структуру каталогов.

Основные ключи и функционал tar:

- -c: Создание нового архива.

Например, `tar -cf archive.tar file1.txt file2.txt`.

- -x: Извлечение файлов из архива.

Например, `tar -xf archive.tar`.

- -r: Добавление файлов в существующий архив.

Например, `tar -rf archive.tar file3.txt directory2`.

- -t: Просмотр содержимого архива.

Например, `tar -tf archive.tar`.

- -z: Сжатие данных с помощью gzip.

Например, `tar -czf archive.tar.gz directory`.

- -v: Отображение подробного вывода (verbose) во время выполнения операции.



Например, `tar -cvf archive.tar directory`.

- -f: Определение имени архива.

Например, `tar -cf archive.tar directory`.

`tar -cf archive.tar file1.txt file2.txt directory1` - эта команда создаст архив с именем `archive.tar`, который будет содержать файлы `file1.txt`, `file2.txt` и содержимое директории `directory1`.

`tar -czf archive.tar.gz directory1` -эта команда создаст архив с именем `archive.tar.gz`, который будет содержать содержимое директории `directory1`, с сжатием данных при помощи `gzip`.

В обоих случаях:

-с: Создает новый архив.

-f: Определяет имя архива.

-z: Для архивации с сжатием при помощи `gzip`.



ЗАДАНИЕ ДЛЯ ЗАКРЕПЛЕНИЯ

1. Создать скрипт, который создаст 10 файлов со случайными числами, а затем заархивирует директорию с этими файлами.

```
Unset
#!/bin/bash

# Создаем временную директорию
mkdir -p /home/tar
temp_dir="/home/tar"

# Создадим файлы
for i in {1..10}; do echo "Случайное число: $RANDOM" > "$temp_dir/$i.txt"; done

# Создаем архив из временной директории
tar -czf /tmp/RANDOM.tar.gz -C $temp_dir .

# Удаляем временную директорию
rm -r $temp_dir
```

2. Написать скрипт, который создает директорию /home/MyDir, создает 10 файлов с текущим временем внутри этих файлов, затем создает архив mydir.tar.gz из содержимого этой директории. После этого он ожидает 5 секунд и выводит список файлов в архиве в файл /root/mydir-tar.txt.

```
Unset
#!/bin/bash
set -e
mkdir -p /home/MyDir
for run in {1..10}
do
    date '+%T' > /home/MyDir/$run.txt
    sleep 0.5
done
```



```
tar -cvzf /tmp/mydir.tar.gz /home/MyDir/*  
sleep 5  
tar -tf /tmp/mydir.tar.gz > /root/mydir-tar.txt
```

- `mkdir -p /home/MyDir`: Создает директорию `/home/MyDir`, если она еще не существует.
- `for run in {1..10}`: Начинает цикл, который будет выполняться 10 раз.
- `date '+%T' > /home/MyDir/$i.txt`: В каждой итерации цикла записывает текущее время в формате HH:MM:SS в файл в директории `/home/MyDir`
- `sleep 0.5`: Приостанавливает выполнение скрипта на 0.5 секунды.
- `tar -cvzf /tmp/mydir.tar.gz /home/MyDir/*`: Создает архив с именем `mydir.tar.gz`, который содержит все файлы из директории `/home/MyDir`.
- `sleep 5`: Приостанавливает выполнение скрипта на 5 секунд.
- `tar -tf /tmp/mydir.tar.gz > /root/mydir-tar.txt`: Выводит список файлов в архиве `mydir.tar.gz` в файл `/root/mydir-tar.txt` в директории `/root`.



ЧТО ТАКОЕ CRON



Cron - это стандартный планировщик задач в Unix и Unix-подобных операционных системах, таких как Linux.

Cron появился в Unix в конце 1970-х годов и с тех пор стал одним из стандартных компонентов многих Unix-подобных систем.

Компоненты cron:

- Cron tables: Это файлы конфигурации, которые содержат информацию о том, какие задачи должны быть выполнены и когда. Обычно используются два основных файла: `/etc/crontab`, который содержит системные задачи, и файлы cron-таблиц для каждого пользователя в директории `/var/spool/cron`.
- Cron jobs: Это сами задачи, которые вы хотите автоматизировать. Каждый cron job состоит из команды или скрипта, который должен быть выполнен, и расписания, которое определяет, когда задача должна быть выполнена.
- Cron daemon: Это процесс, который работает на фоне и регулярно проверяет cron-таблицы для запуска задач в соответствии с определенным расписанием.



CRONTAB: РАСПИСАНИЕ CRON

Структура записи: Каждая запись в файле crontab содержит пять полей, разделенных пробелами или табуляцией

Минуты (0-59) Часы (0-23) День месяца (1-31) Месяц (1-12) День недели (0-7)

Значения и диапазоны:

- * - Соответствует любому значению. Например, * в поле минут будет выполнять задачу каждую минуту.
- 1,2,5 - Список значений. Например, 1,2,5 в поле часов будет выполнять задачу в 1:00, 2:00 и 5:00.
- */n - Шаг. Например, */15 в поле минут будет выполнять задачу каждые 15 минут.
- n-m - Диапазон значений. Например, 1-5 в поле часов будет выполнять задачу с 1:00 до 5:59.

Рассмотрим один из файлов с расписанием:

```
# Run the hourly jobs
SHELL=/bin/bash
PATH=/sbin:/bin:/usr/sbin:/usr/bin
MAILTO=root
01 * * * * root run-parts /etc/cron.hourly
```

01 * * * * будет означать выполнение в 01 минуту каждого часа , дня , месяца и т.д.

Примеры:

0 2 * * * - Задача будет выполняться в 2:00 каждый день.


30 8 * * 1-5 - Задача будет выполняться в 8:30 с понедельника по пятницу.

*/15 * * * * - Задача будет выполняться каждые 15 минут.

0 0 1 * * - Задача будет выполняться в полночь первого числа каждого месяца.

Один из ресурсов для удобного “чтения” crontab - <https://crontab.guru/>



 Cron Job 1

crontab guru

The quick and simple editor for cron schedule expressions by [Cronitor](#)

“At 04:05.”

next at 2024-03-12 04:05:00 random

5 4 * * *

minute	hour	day (month)	month	day (week)
*			any value	
,			value list separator	
-			range of values	
/			step values	
@yearly			(non-standard)	
@annually			(non-standard)	
@monthly			(non-standard)	
@weekly			(non-standard)	
@daily			(non-standard)	
@hourly			(non-standard)	
@reboot			(non-standard)	

Вносим в него расписание, которое вызывает у нас вопросы - и наслаждаемся человекочитаемым вариантом.



CRONTAB: СОЗДАНИЕ ЗАДАНИЯ

Поддерживаемые ключи команды crontab :

- -e: Редактирование текущих cron-заданий. Он открывает файл cron-таблицы текущего пользователя в текстовом редакторе, позволяя вам добавлять, изменять или удалять cron-задания.
- -l: Вывод текущих cron-заданий. Этот ключ отображает текущие cron-задания пользователя в стандартный вывод.
- -r: Удаление текущих cron-заданий. Этот ключ удаляет все cron-задания текущего пользователя.
- -u user: Операция от имени другого пользователя. С помощью этого ключа можно работать с cron-заданиями другого пользователя, указав его имя после ключа -u.

Создадим два файла.

Один будет файл сценария, а во второй будет попадать результат выполнения этого сценария.

Флаг -e в команде echo сообщает командной оболочке интерпретировать последовательности \n как символ новой строки.

Unset

```
touch /tmp/script2.sh /tmp/output2.log  
echo -e '#!/bin/bash\n date\n echo "it works"' > /tmp/script2.sh  
chmod +x /tmp/script2.sh
```

Попробуем создать задание, которое будет запускать скрипт с интервалом в минуту. Чтобы это сделать мы запускаем редактор crontab:

Unset

```
crontab -e
```

(пример с VI - этот редактор открывается по умолчанию)



Unset

```
***** /tmp/script2.sh >> /tmp/output2.log
```

Выходим с сохранением.

Проверить наличие задания мы можем выполнив :

Unset

```
crontab -l
```

```
***** /tmp/script.sh >> /tmp/output2.log
```

При помощи `tail -f` можем отслеживать “хвост” файла `/tmp/output2.log` и следить за изменением файла. Таким образом мы увидим, что каждую минуту скрипт будет выполняться и в файле будет появляться новая строка.