

МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Российский технологический университет»

**РТУ МИРЭА**

**ЗАЧТЕНО**

\_\_\_\_\_\_\_\_\_\_/Н.С.Ершов/

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_20\_\_\_\_г.

Дисциплина: «Обеспечение устойчивого функционирования автоматизированных систем в условиях внешних воздействий»

Практическая работа 2: «Резервное копирование»

Выполнил:

Студент 3 курса

Кутьин З.С.

Группа: БББО-05-20

Проверил:

Ершов Н.С.

**Содержание**

**Введение**

**Актуальность работы:** В работе рассматривается и изучается технология резервного копирования при помощи планировщика Linux систем cron и утилит резервного копирования даных.

**Цель работы:** Получить навыки работы с технологиями резервного копирования.

**Задачи:**

* Произвести настройку программы cron;
* Сделать резервное копирование данных в определенном диапазоне времени;

Пользоваться можно утилитой планирования cronи любыми утилитами копирования, резервирования, архивирования данных. Все основные этапы работы обозначить снимками экрана.

Материал:

1. Демон **cron** — это стандартный инструмент для периодического выполнения команд. Он запускается на этапе начальной загрузки системы и выполняется до тех пор, пока система не будет выключена.

Демон cron читает файлы конфигурации, содержащие списки командных строк и расписание их вызова. Командные строки обрабатываются интерпретатором sh, поэтому почти все, что можно делать в данном интерпретаторе вручную, разрешается перепоручать демону cron2.

Файл конфигурации демона cron называется crontab (сокращение от “crontable” — таблица демона cron). Пользовательские crontab-файлы хранятся в каталоге **/var/spool/cron**.

Несмотря на различия в конкретных реализациях, все версии демона cron стремятся минимизировать время, затрачиваемое на синтаксический анализ файлов конфигурации и выполнение вычислений. Команда crontab помогает поддерживать эффективность

работы демона cron, уведомляя его об изменениях crontab-файлов. Следовательно, вы не должны редактировать crontab-файлы сами, поскольку это может привести к тому, что демон cron не заметит ваших изменений. Если возникнет ситуация, когда cron, казалось бы, “не замечает” модификацию своих crontab-файлов, воспользуйтесь сигналом отбоя (HUP), который в большинстве систем заставляет его повторно прочитать их.

Все crontab-файлы в системе имеют одинаковый формат. Комментарии начинаются со знака решетки в первой позиции строки. Каждая строка, не являющаяся комментарием, содержит шесть полей и представляет одну команду.

**минута час день месяц день\_недели команда**

Первые пять полей (минута, час, день, месяц и день\_недели) содержат информацию о времени запуска команды. Они отделяются друг от друга пробелами, но в поле команда пробел выполняет свою обычную функцию разделителя аргументов.

|  |  |  |
| --- | --- | --- |
| Поле | Описание | Диапазон |
| минута | Минута часа | от 0 до 59 |
| час | Час дня | от 0 до 23 |
| день | День месяца | от 1 до 31 |
| месяц | Месяц года | от 1 до 12 |
| день\_недели | День недели | от 0 до 6 (0 = воскресенье) |

Каждое поле времени может содержать следующее:

• звездочку, которая обозначает любую цифру;

• целое число, трактуемое буквально;

• два разделенных дефисом целых числа, задающих диапазон значений;

• диапазон, за которым стоит косая черта и значение шага, например 1—10/2 (только для Linux);

• целые числа или диапазоны, разделенные запятыми (искомое время соответствует любому из указанных значений).

Например, строка

**45 10 \* \* 1-5**

означает “**10 часов 45 минут, с понедельника по пятницу**”. Совет: никогда не ставьте звездочку в первое поле, иначе команда будет выполняться каждую минуту.

С полями **день\_недели** и **день** сопряжена потенциальная двусмысленность, которую необходимо учитывать. День можно рассматривать как день недели и число месяца.

Если указаны оба поля, то искомому дню достаточно удовлетворять одному из этих требований, чтобы пройти отбор. Например, спецификация

**0,30 \* 13 \* 5**

означает “каждые полчаса по пятницам и каждые полчаса тринадцатого числа месяца”, но не “каждые полчаса в пятницу, 13-го”.

Поле команда содержит командную строку, выполняемую интерпретатором sh. Это может быть любая допустимая команда интерпретатора (брать ее в кавычки не нужно).

Считается, что поле команда продолжается до конца строки и может содержать пробелы и символы табуляции.

Приведем несколько примеров допустимых команд crontab-файла.

**echo The time is now 'date' > /dev/console**

**mail -s Reminder evi@anchor % Don’t forget to write your chapters.**

**cd /etc; /bin/mail -s "Password file" evi<passwd**

А теперь рассмотрим полные примеры записей.

**30 2 \* \* 1 (cd /home/joe/project; make)**

**20 1 \* \* \* find /tmp -atime +3 -type f -ехесrm -f { } ';'**

Эта строка будет активизироваться каждый день в 1:20. Соответствующая команда удаляет из каталога /tmp все файлы, к которым за последние 3 дня никто не обращался.

**55 23 \* \* 0-3,6 /staff/trent/bin/checkservers**

Эта строка запускает сценарий checkservers в 23:55 каждый день, кроме четверга и пятницы.

Команда crontabимя\_файла инсталлирует указанный crontab-файл, заменяя егопредыдущую версию, если таковая имеется. Команда crontab -e проверяет копию crontab-файла текущего пользователя, загружает ее в текстовый редактор (указанный в переменной среды EDITOR) для последующего изменения, а затем повторно записывает файл в системный каталог. Команда crontab -l отображает содержимое crontab файла, а команда crontab -r удаляет этот файл.

Пользователю root разрешается задавать аргумент имя\_пользователя, чтобы можно было просматривать или редактировать crontab-файлы других пользователей. Например, команда crontab -r jsmith удаляет crontab-файл, принадлежащий пользователю jsmith, а команда crontab -е jsmith редактирует этот файл. В системе Linux разрешается использование обоих аргументов имя\_пользователя и имя\_файла в одной команде, поэтому для устранения неоднозначности аргументу имя\_пользователя должен предшествовать флаг -u (например, crontab -u jsmithcrontab.new).

Будучи вызванными без аргументов, большинство версий команды crontab будут пытаться прочитать crontab-файл из своего стандартного входного потока. Если этот режим был активизирован случайно, не пытайтесь выйти из него, нажимая комбинацию клавиш <Ctrl+D>, так как весь crontab-файл будет удален. Необходимо нажать <Ctrl+C>. В Linux для того, чтобы команда crontab обратилась к своему стандартному входному потоку, в качестве аргумента имя\_файла ей необходимо передать дефис.

Два файла конфигурации cron.deny и cron.allow содержат информацию о том, кому из пользователей разрешено предоставлять crontab-файлы. В разных системах эти файлы расположены в разных каталогах.

Если файл cron.allow существует, то он содержит список пользователей, имеющих доступ к демону cron (по одному имени в строке). Пользователи, которые в списке отсутствуют, не имеют права выполнять команду crontab. Если файла cron.allow нет, проверяется файл cron.deny. Он также содержит список пользователей, но с противоположным назначением: доступ разрешен всем, кроме лиц, указанных в списке.

**Методы резервного копирования**

**Команда cp**

Команда cp позволяет вызвать одноименную утилиту, предназначенную для копирования файлов и директорий. При этом оригиналы файлов и директорий остаются нетронутыми, а их копии могут иметь аналогичные или отличающиеся имена и располагаться как в той же директории (при условии изменения имени копии), так и в другой директории и даже в другой файловой системе.

Базовый синтаксис команды:

**$ cp [параметры] <оригинал><копия>**

В качестве оригинала и копии могут использоваться имена и пути к файлам и директориям, причем в директорию могут копироваться срезу несколько файлов или других директорий. Утилита поддерживает множество параметров, но на практике используется лишь их малая часть. Параметр -f позволяет принудительно перезаписывать файлы, если в целевой директории уже существуют файлы с такими именами, параметр -n работает ровно противоположным образом, то есть, позволяет автоматически пропускать файлы с совпадающими именами, а параметр -u позволяет обновлять файлы, то есть, перезаписывать их только в том случае, если копируемые файлы являются более новыми. Если вы хотите принимать решения относительно перезаписи каждого из файлов, вы можете активировать интерактивный режим с помощью параметра -i. Для копирования директории со всеми ее файлами и поддиректориями следует использовать параметр -R, активирующий режим рекурсивного копирования.

Если же нужно сохранить атрибуты файла или директории, следует использовать параметр -p; при использовании этого параметра сохраняются следующие атрибуты:

Метка времени модификации

Метка времени последнего доступа

Права доступа к файлу

Идентификатор владельца файла (UID)

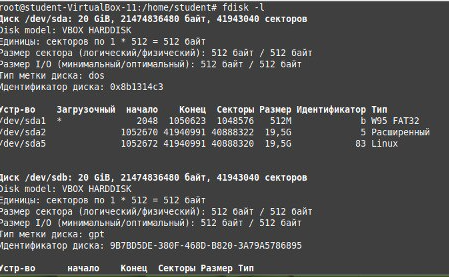
Идентификатор группы владельцев файла (GID)

Списка контроля доступа (ACL)

Расширенные атрибуты (EA)

**Комманда tar**

Одним из наиболее популярных способов создания простейшего бэкапа является архивация данных с помощью tar. Преимуществами данного способа являются возможность инкрементного бэкапа (добавления файлов в существующий архив, удаления или изменения таковых), возможность извлечения из архива отдельных файлов, а также присутствие tar практически в любой Linux-системе.



Создание архива

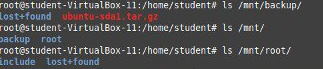
Для начала создайте точки монтирования для корневого раздела и для раздела, на котором вы собираетесь создавать бэкап, например так

**sudomkdir /mnt/{root,backup}**

Смонтируйте оба раздела. Для большей надежности можно смонтировать корневой раздел в режиме read-only (только для чтения), чтобы исключить вероятность случайного изменения данных

**sudo mount /dev/sdXY /mnt/root -o ro**

**sudo mount /dev/sdXY /mnt/backup**

****

(Вместо «sdXY» используйте ваши значения для нужных разделов. определить их можно с помощью sudofdisk -l или sudoblkid)

В случае, если вы используете отдельные разделы для /boot, /usr, /home и пр. и хотите включить их содержимое в бэкап, примонтируйте их в соответствующие папки

**sudo mount /dev/sdXY /mnt/root/usr -o ro**

**sudo mount /dev/sdXY /mnt/root/home -o ro**

При необходимости создайте на бэкап-разделе папку, в которой вы хотите разместить архив, например

**sudomkdir -p /mnt/backup/ubuntu/root**

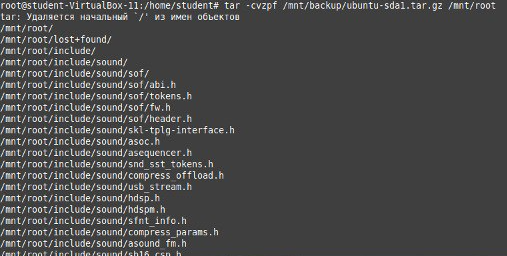
Теперь можно приступить к созданию архива. Чтобы создать архив с **gzip**-сжатием, выполните

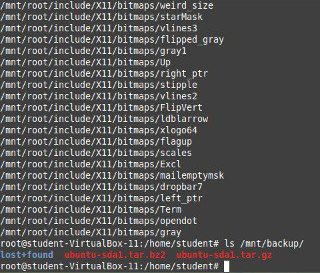
**sudo tar -cvzpf -C /mnt/root /mnt/backup/ubuntu-sda1.tar.gz .**

(Ключ -p включает сохранение владельцев и прав доступа для файлов 1))

Для **bzip2**-сжатия используйте

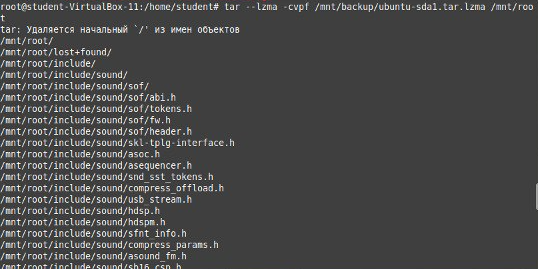
**sudotar -cvjpf /mnt/backup/ubuntu-sda1.tar.bz2 /mnt/root**

****

****

Для **lzma**-сжатия

**sudotar --lzma -cvpf /mnt/backup/ubuntu-sda1.tar.lzma /mnt/root**

****

Аналогично для lzo-сжатия - ключ --lzop вместо --lzma

Различные алгоритмы сжатия дают разный размер архива, а также отличаются по производительности.

По завершении процесса отмонтируйте все смонтированные разделы

**sudoumount /mnt/root{/boot,/var,/home,} /mnt/backup**

****

Восстановление из архива

Создайте точки монтирования для корневого раздела и раздела, на котором хранится ваш архив

**sudomkdir /mnt/{root,backup}**

Примонтируйтеразделсархивом-бэкапом

**sudo mount /dev/sdXY /mnt/backup -o ro**

Отформатируйте корневой раздел в ту же (или другую) ФС. Если вы используете отдельные разделы для /usr, /boot и пр. и архивировали их, отформатируйте их тоже

**sudomkfs.reiserfs -l "root" /dev/sdXY**

**sudomkfs.ext2 -L "boot" /dev/sdXY**

**sudomkfs.ext4 -L "home" /dev/sdXY**

(если вы восстанавливаете систему на новый жесткий диск, разметьте его с помощью fdisk/gparted и отформатируйте разделы)

Некоторые файловые системы поддерживают задание UUID при форматировании. Это дает возможность создать ФС с таким же UUID, как у старой, что позволит избежать необходимости править fstab.

Для **ext2/3/4** UUID задается с помощью ключа -U, а еще больше упростить задачу можно командой вида

**sudomkfs.ext4 -L "label" -U "$(sudoblkid -o value -s UUID /dev/sda1)" /dev/sda1**

Для **reiserfs** это будет ключ -u, для reiser4 -U, для остальных см. manmkfs.тип

Смонтируйте созданные ФС

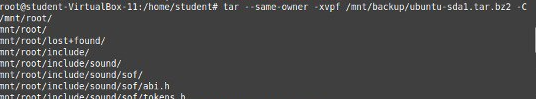
**sudomount /dev/sdXY /mnt/root**

****

**sudo mount /dev/sdXY /mnt/root/home**

Теперь можно распаковать содержимое архива на место

**sudo tar --same-owner -xvpf /mnt/backup/ubuntu-2010-10-07.tar.bz2 -C /mnt/root/**



(ключ -C задает каталог, в который нужно распаковать файлы. ключ --same-owner сохраняет владельцев файлов при распаковке)

Если вы воспользовались советом выше и создали ФС тех же типов с теми же UUID - можете смело пропустить этот пункт. В противном случае вам придется исправить fstab в соответствии с произведенными изменениями. Откройте этот файл в любимом текстовом редакторе (например, nano) и замените UUID (и тип ФС, если форматировали что-либо в другую ФС) нужных разделов

**sudonano /mnt/root/etc/fstab**

Сохраните файл, выйдите из редактора и отмонтируйте все смонтированные разделы

**sudoumount /mnt/root{/var,/boot,} /mnt/backup**

**Вывод**

В данной работе была произведена установка и настройка инструмента планирования задач cron. Были изучены и освоены методы резервного копирования информации в ОС Linux.