

Презентация по лабораторной работе №10

Сячинова Ксения Ивановна НПМбд-02-21

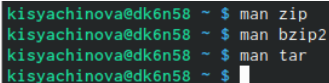
Российский Университет Дружбы Народов

Цель работы

Изучить основы программирования в оболочке ОС UNIX/Linux.
Научиться писать небольшие командные файлы.

Выполнение лабораторной работы

1. Первым делом изучим команды архивации. Для этого будем использовать команды “man zip”, “man bzip2”, “man tar”. (рис. 1), (рис. 2), (рис. 3), (рис. 4)

A terminal window with a dark background and green text. It shows four lines of commands being entered at a prompt. The first three lines are 'man zip', 'man bzip2', and 'man tar'. The fourth line shows the prompt with a cursor, indicating the command has been entered but not yet executed.

```
kisyachinova@dk6n58 ~ $ man zip  
kisyachinova@dk6n58 ~ $ man bzip2  
kisyachinova@dk6n58 ~ $ man tar  
kisyachinova@dk6n58 ~ $
```

Figure 1: Выполнение команд

```

zip(1L)
NAME
    zip - package and compress (archive) files

SYNOPSIS
    zip [-addcdDefFghjJlmmqRStuvVwxyz1065] [--longoption ...] [-b path] [-n suffixes] [-s list]

    zipcloak (see separate man page)
    zipnote (see separate man page)
    zipsplit (see separate man page)

    Note: Command line processing in zip has been changed to support long options and handle
    tently. Some old command lines that depend on command line inconsistencies may no longer

DESCRIPTION
    zip is a compression and file packaging utility for Unix, VMS, MDOS, OS/2, Windows 9x,
    Acorn RISC OS. It is analogous to a combination of the Unix commands tar\(1\) and compress\(1\)
    Katz's ZIP for MDOS systems).

    A companion program (unzip\(1L\)) unpacks zip archives. The zip and unzip\(1L\) programs can
    porting most PKZIP features up to PKZIP version 4.0, and PKZIP and PKUNZIP can work with
    tions, notably streamed archives, but recent changes in the zip file standard may facili
    is compatible with PKZIP 2.04 and also supports the zip64 extensions of PKZIP 4.5 which a
    previous 2 GB limit (4 GB in some cases). zip also now supports bzip2 compression if the
    files. Note that PKUNZIP 3.0 cannot extract files produced by PKZIP 2.04 or zip 3.0. No
    later versions) to extract them.

```

Figure 2: “man zip”

```

bzip2(1)                                General Commands Manual

NAME
  bzip2, bunzip2 - a block-sorting file compressor, v1.0.8
  bzcat - decompresses files to stdout
  bzip2recover - recovers data from damaged bzip2 files

SYNOPSIS
  bzip2 [-cdffqrv0123456789] [ filenames ... ]
  bunzip2 [-fsvz] [ filenames ... ]
  bzcat [-s] [ filenames ... ]
  bzip2recover filename

DESCRIPTION
  bzip2 compresses files using the Burrows-Wheeler block sorting text compression algo-
  rithm, which is generally considered to be significantly better than that achieved by more conventional LZW/LZ77-based com-
  pressors of the PPM family of statistical compressors.

  The command-line options are deliberately very similar to those of gzip, but the
  bzip2 expects a list of file names to accompany the command-line flags. Each file
  is compressed with the name "original_name.bz2". Each compressed file has the same modification time
  as the corresponding original, so that these properties can be correctly restored at
  a later date. In the sense that there is no mechanism for preserving original file names, permissions,
  or have serious file name length restrictions, such as MS-DOS.

  bzip2 and bunzip2 will by default not overwrite existing files. If you want this to
  be overridden, use the -f option.

  If no file names are specified, bzip2 compresses from standard input to standard output,
  producing compressed output to a terminal, as this would be entirely incomprehensible and there-
  fore not useful.

  bunzip2 (or bzip2 -d) decompresses all specified files. Files which were not created
  with bzip2 will be decompressed anyway. bzip2 attempts to guess the filename for the decompressed file from
  the filename of the compressed file.

  filename.bz2 becomes filename
  filename.bz becomes filename

```

Figure 3: “man bzip2”

```

TAR(1)                                GNU TAR Manual

NAME
    tar - an archiving utility

SYNOPSIS
    Traditional usage
    tar [A|c|d|r|t|u|x][GnSkUNOMPmsbIajZzHPlrwo] [add...]

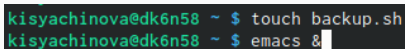
    UNIX-style usage
    tar -A [options] archive archive
    tar -c [-f archive] [options] [file...]
    tar -d [-f archive] [options] [file...]
    tar -t [-f archive] [options] [member...]
    tar -r [-f archive] [options] [file...]
    tar -u [-f archive] [options] [file...]
    tar -x [-f archive] [options] [member...]

    GNU-style usage
    tar [--catenate|--concatenate] [options] archive archive
    tar --create [--file archive] [options] [file...]
    tar [--diff|--compare] [--file archive] [options] [file...]
    tar --delete [--file archive] [options] [member...]
    tar --append [-f archive] [options] [file...]

```

Figure 4: “man tar”

После этого создадим файл, в котором будет написан скрипт, откроем его с помощью редактора “emacs” (сочетания клавиш “ctrl-x”, “ctrl-f”)(рис. 5)

A terminal window with a dark background and green text. It shows two lines of commands entered at the prompt 'kisyachinova@dk6n58 ~ \$'. The first line is 'touch backup.sh' and the second line is 'emacs &' followed by a cursor.

```
kisyachinova@dk6n58 ~ $ touch backup.sh
kisyachinova@dk6n58 ~ $ emacs & █
```

Figure 5: Создание файла

Затем, создадим скрипт, который при запуске будет делать резервную копию самого себя (т.е. файла, в котором содержится его исходный код) в другую директорию backup в нашем домашнем каталоге. При написании скрипта я буду использовать архиватор “bzip2”.(рис. 6)

```
#!/bin/bash

name="backup.sh"          # Сохранение файла со скриптом в переменную "name"
mkdir ~/backup             # Создание каталога ~/backup
bzip2 -k ${name}           # Архивирование скрипта
mv ${name}.bz2 ~/backup/   # Перемещение архивированного скрипта в каталог
echo "Выполнено"
```

Figure 6: Создание скрипта

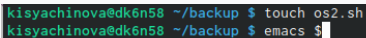
Добавим право на выполнение “chmod +x *.sh” и проверим работу скрипта “./backup.sh”. Также проверим, появился ли каталог backup/, переходим в него, просматриваем его содержимое, и просматриваем содержимое архива “bunzip2 -c backup.sh.bz2”. (рис. 7)

```
kisyachinova@dk6n58 ~ $ ls
1.txt  2.txt  backup.sh  Hello  Project  ski.plac
1.txt~ 4.txt  backup.sh~ KseniyaSyachinova.github.io  monthly  public  test.txt
2.txt  abc1   bin        lab07.sh  my_os   public_html  tap
2.txt~ australia  feathers  lab07.sh~  play    reports  tutorial
[1]+  Засепаєн enacs
kisyachinova@dk6n58 ~ $ chmod +x *.sh
kisyachinova@dk6n58 ~ $ ./backup.sh
Виконано
kisyachinova@dk6n58 ~ $ cd backup/
kisyachinova@dk6n58 ~/backup $ ls
backup.sh.bz2
kisyachinova@dk6n58 ~/backup $ bunzip2 -c backup.sh.bz2
#!/bin/bash

name="backup.sh"          # Сохранение файла со скриптом в переменную "name"
mkdir ~/backup            # Создание каталога ~/backup
bzip2 -k $(name)          # Архивирование скрипта
mv $(name).bz2 ~/backup/  # Перемещение архивированного скрипта в каталог
echo "Виконано"
kisyachinova@dk6n58 ~/backup $
```

Figure 7: Проверка

2. Создадим файл для второго скрипта и откроем его в редакторе “emacs” с помощью сочетаний клавиш.(рис. 8)



```
kisyachinova@dk6n58 ~/backup $ touch os2.sh  
kisyachinova@dk6n58 ~/backup $ emacs $
```

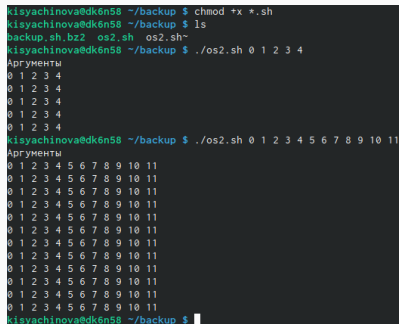
Figure 8: Создание второго скрипта

Напишем пример командного файла, который обрабатывает любое произвольное число аргументов, в том числе превышающее десять. Этот скрипт может последовательно распечатывать значения всех переданных аргументов.(рис. 9)

```
#!/bin/bash
echo "Аргументы"
for a in $@ # Цикл для прохода по введённым аргументам
do echo $a # Вывод аргумента
done
```

Figure 9: Командный файл

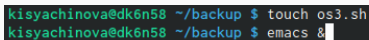
После этого проверим работу написанного скрипта. Для этого спользуем команду “./os2.sh 0 1 2 3 4 5 6 7 8 9 10 11”. Но для начала добавим право на выполнение “chmod +x *.sh”. Так как у нас файл, который обрабатывает любое произвольное число аргументов, я вводила аргументы, количество которых и меньше 10 и больше 10. Скрип работает верно. (рис. 10)



```
kisyachinova@dk6n58 ~/backup $ chmod +x *.sh
kisyachinova@dk6n58 ~/backup $ ls
backup.sh.bz2  os2.sh  os2.sh~
kisyachinova@dk6n58 ~/backup $ ./os2.sh 0 1 2 3 4
Аргументы
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
0 1 2 3 4
kisyachinova@dk6n58 ~/backup $ ./os2.sh 0 1 2 3 4 5 6 7 8 9 10 11
Аргументы
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
0 1 2 3 4 5 6 7 8 9 10 11
kisyachinova@dk6n58 ~/backup $
```

Figure 10: Проверка скрипта

3. Создаём файл для написания третьего скрипта, открываем его в “emacs”.(рис. 11)



```
kisyachinova@dk6n58 ~/backup $ touch os3.sh
kisyachinova@dk6n58 ~/backup $ emacs & █
```

Figure 11: Создание файла

Напишем командный файл, аналог команды “ls”. Он должен будет выдавать информацию о нужном каталоге и выводить информация о возможностях доступа к файлам этого каталога. (рис. 12)

```
#!/bin/bash
a="$1"
for i in ${a}/*
do
    echo "$1"

    if test -f $i
    then echo "Обычный файл"
    fi

    if test -d $i
    then echo "Каталог"
    fi

    if test -r $i
    then echo "Чтение разрешено"
    fi

    if test -w $i
    then echo "Запись разрешена"
    fi

    if test -x $i
    then echo "Выполнение разрешено"
    fi
done
```

Figure 12: Командный файл

Далее даём право на выполнение с помощью команды “`chmod +x *.sh`” и проверяем работу скрипта “`./os3.sh ~`”. (рис. 13). Работает корректно.

```
kisyachinova@dk6n58 ~/backup $ chmod +x *.sh
[1]+  Завершён      emacs
kisyachinova@dk6n58 ~/backup $ ls
backup.sh.bz2  os2.sh  os2.sh~  os3.sh  os3.sh~
kisyachinova@dk6n58 ~/backup $ ./os3.sh ~
/afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova
Обычный файл
Чтение разрешено
Запись разрешена
/afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova
Обычный файл
Чтение разрешено
Запись разрешена
/afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova
Обычный файл
Чтение разрешено
Запись разрешена
/afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova
Обычный файл
Чтение разрешено
Запись разрешена
/afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova
Обычный файл
Чтение разрешено
Запись разрешена
```

Figure 13: Проверка скрипта

4. Для выполнение третьего скрипта также создаём файл и открываем его в “emacs”. (рис. 14)

```
kisyachinova@dk6n58 ~/backup $ touch os4.sh  
kisyachinova@dk6n58 ~/backup $ emacs &
```

Figure 14: Создание файла

Напишем командный файл для вычисления количества файлов в указанной директории. Файл получает в качестве аргумента командной строки формат файла. Путь к директории также передаётся в виде аргументов командной строки. (рис. 15)

```
#!/bin/bash
b="$1"
shift
for a in $@
do
    k=0
    for i in ${b}/*.${a}
    do
        if test -f "$i"
        then
            let k=k+1
        fi
    done
    echo "$k файлов содержится в $b с расширение"
done
```

Figure 15: Командный файл

Затем даём право на выполнение с помощью команды “`chmod +x *.sh`” и проверяем работу скрипта с помощью “`./os4.sh~pdf sh txt doc`”. Для проверки создадим несколько файлов разного расширения. Видим, что скрипт работает верно. (рис. 16)

```
kisyachinova@dk6n58 ~/backup $ chmod +x *.sh
[1]+  Завершён      emacs
kisyachinova@dk6n58 ~/backup $ ls
backup.sh bz2 os2.sh os3.sh os3.sh~ os4.sh os4.sh~
kisyachinova@dk6n58 ~/backup $ touch os4.pdf os4.doc os44.doc
kisyachinova@dk6n58 ~/backup $ ./os4.sh~pdf sh txt doc
bash: ./os4.sh~pdf: Нет такого файла или каталога
kisyachinova@dk6n58 ~/backup $ ./os4.sh ~ pdf sh txt doc
0 файлов содержится в /afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova с расширением pdf
2 файлов содержится в /afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova с расширением sh
5 файлов содержится в /afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova с расширением txt
0 файлов содержится в /afs/.dk.sci.pfu.edu.ru/home/k/i/kisyachinova с расширением doc
kisyachinova@dk6n58 ~/backup $
```

Figure 16: Проверка скрипта

Выводы

В ходе выполнения данной лабораторной работы я изучила основы программирования в оболочке ОС UNIX/Linux и научилась писать разные командные файлы.