

Операционные системы

Анализ файловой структуры UNIX. Команды для работы с файлами и каталогами

Аннагулыев Арслан

14 марта 2025

Российский университет дружбы народов, Москва, Россия

Цели и задачи работы

Ознакомление с файловой системой Linux, её структурой, именами и содержанием каталогов. Приобретение практических навыков по применению команд для работы с файлами и каталогами, по управлению процессами, по проверке использования диска и обслуживанию файловой системы.

- 1 Выполнить приимеры
- 2 Выполнить дествия по работе с каталогами и файлами
- 3 Выполнить действия с правами доступа
- 4 Получить дополнительные сведения при помощи справки по командам.

Процесс выполнения лабораторной работы

```
aannaguliev@aannaguliev:~$ touch abc1
aannaguliev@aannaguliev:~$ cp abc1 april
aannaguliev@aannaguliev:~$ cp abc1 may
aannaguliev@aannaguliev:~$ mkdir monthly
aannaguliev@aannaguliev:~$ cp april may monthly
aannaguliev@aannaguliev:~$ cp monthly/may monthly/june
aannaguliev@aannaguliev:~$ ls monthly
april  june  may
aannaguliev@aannaguliev:~$ mkdir monthly.00
aannaguliev@aannaguliev:~$ cp -r monthly monthly.00
aannaguliev@aannaguliev:~$ cp -r monthly.00 /tmp
aannaguliev@aannaguliev:~$
```

Рис. 1: Выполнение примеров

```
aannaguliev@aannaguliev:~$  
aannaguliev@aannaguliev:~$ mv april july  
aannaguliev@aannaguliev:~$ mv july monthly.00  
aannaguliev@aannaguliev:~$ ls monthly.00  
july  monthly  
aannaguliev@aannaguliev:~$ mv monthly.00 monthly.01  
aannaguliev@aannaguliev:~$ mkdir reports  
aannaguliev@aannaguliev:~$ mv monthly.01 reports  
aannaguliev@aannaguliev:~$ mv reports/monthly.01 reports/monthly  
aannaguliev@aannaguliev:~$
```

Рис. 2: Выполнение примеров

Выполнение примеров

```
aannaguliev@aannaguliev:~$  
aannaguliev@aannaguliev:~$ touch may  
aannaguliev@aannaguliev:~$ ls -l may  
-rw-r--r--. 1 aannaguliev aannaguliev 0 map 14 12:09 may  
aannaguliev@aannaguliev:~$ chmod u+x may  
aannaguliev@aannaguliev:~$ ls -l may  
-rwxr--r--. 1 aannaguliev aannaguliev 0 map 14 12:09 may  
aannaguliev@aannaguliev:~$ chmod u-x may  
aannaguliev@aannaguliev:~$ ls -l may  
-rw-r--r--. 1 aannaguliev aannaguliev 0 map 14 12:09 may  
aannaguliev@aannaguliev:~$ chmod g-r,o-r monthly  
aannaguliev@aannaguliev:~$ chmod g+w abc1  
aannaguliev@aannaguliev:~$
```

Рис. 3: Выполнение примеров

Создание директорий и копирование файлов

```
aannaguliev@aannaguliev:~$  
aannaguliev@aannaguliev:~$ cp /usr/include/linux/sysinfo.h ~  
aannaguliev@aannaguliev:~$ mv sysinfo.h equipment  
aannaguliev@aannaguliev:~$ mkdir ski.plases  
aannaguliev@aannaguliev:~$ mv equipment ski.plases/  
aannaguliev@aannaguliev:~$ mv ski.plases/equipment ski.plases/equiplist  
aannaguliev@aannaguliev:~$ touch abc1  
aannaguliev@aannaguliev:~$ cp abc1 ski.plases/equiplist2  
aannaguliev@aannaguliev:~$ cd ski.plases/  
aannaguliev@aannaguliev:~/ski.plases$ mkdir equipment  
aannaguliev@aannaguliev:~/ski.plases$ mv equiplist equipment/  
aannaguliev@aannaguliev:~/ski.plases$ mv equiplist2 equipment/  
aannaguliev@aannaguliev:~/ski.plases$ cd  
aannaguliev@aannaguliev:~$ mkdir newdir  
aannaguliev@aannaguliev:~$ mv newdir ski.plases/  
aannaguliev@aannaguliev:~$ mv ski.plases/newdir/ ski.plases/plans  
aannaguliev@aannaguliev:~$
```

Рис. 4: Работа с каталогами

Работа с командой chmod

```
aannaguliev@aannaguliev:~$ touch my_os feathers
aannaguliev@aannaguliev:~$ chmod 744 australia/
aannaguliev@aannaguliev:~$ chmod 711 play/
aannaguliev@aannaguliev:~$ chmod 544 my_os
aannaguliev@aannaguliev:~$ chmod 664 feathers
aannaguliev@aannaguliev:~$ ls -l
итого 0
-rw-rw-r--. 1 aannaguliev aannaguliev 0 map 14 12:09 abc1
drwxr--r--. 1 aannaguliev aannaguliev 0 map 14 12:10 australia
-rw-rw-r--. 1 aannaguliev aannaguliev 0 map 14 12:10 feathers
drwxr-xr-x. 1 aannaguliev aannaguliev 74 фев 22 11:44 git-extended
-rw-r--r--. 1 aannaguliev aannaguliev 0 map 14 12:09 may
drwx--x--x. 1 aannaguliev aannaguliev 24 map 14 12:07 monthly
-r-xr--r--. 1 aannaguliev aannaguliev 0 map 14 12:10 my_os
drwx--x--x. 1 aannaguliev aannaguliev 0 map 14 12:10 play
drwxr-xr-x. 1 aannaguliev aannaguliev 14 map 14 12:09 reports
drwxr-xr-x. 1 aannaguliev aannaguliev 28 map 14 12:10 ski.plases
drwxr-xr-x. 1 aannaguliev aannaguliev 10 фев 22 11:37 work
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Видео
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Документы
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Загрузки
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Изображения
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Музыка
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Общедоступные
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 'Рабочий стол'
drwxr-xr-x. 1 aannaguliev aannaguliev 0 фев 22 11:26 Шаблоны
aannaguliev@aannaguliev:~$
```

```
aannaguliev@aannaguliev:~ — less /etc/passwd

root:x:0:0:Super User:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/sbin:/usr/sbin/nologin
adm:x:3:4:adm:/var/adm:/usr/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/usr/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
mail:x:8:12:mail:/var/spool/mail:/usr/sbin/nologin
operator:x:11:0:operator:/root:/usr/sbin/nologin
games:x:12:100:games:/usr/games:/usr/sbin/nologin
ftp:x:14:50:FTP User:/var/ftp:/usr/sbin/nologin
nobody:x:65534:65534:Kernel Overflow User:/:/usr/sbin/nologin
dbus:x:81:81:System Message Bus:/:/usr/sbin/nologin
apache:x:48:48:Apache:/usr/share/httpd:/sbin/nologin
tss:x:59:59:Account used for TPM access:/:/usr/sbin/nologin
avahi:x:70:70:Avahi mDNS/DNS-SD Stack:/var/run/avahi-daemon:/sbin/nologin
geoclue:x:999:999>User for geoclue:/var/lib/geoclue:/sbin/nologin
usbmuxd:x:113:113:usbmuxd user:/:/sbin/nologin
systemd-oom:x:998:998:systemd Userspace OOM Killer:/:/usr/sbin/nologin
qemu:x:107:107:qemu user:/:/sbin/nologin
polkitd:x:114:114>User for polkitd:/:/sbin/nologin
```

Рис. 6: Файл /etc/passwd

Работа с файлами и правами доступа

```
aannaguliev@aannaguliev:~$  
aannaguliev@aannaguliev:~$ cp feathers file.old  
aannaguliev@aannaguliev:~$ mv file.old play/  
aannaguliev@aannaguliev:~$ mkdir fun  
aannaguliev@aannaguliev:~$ cp -R play/ fun/  
aannaguliev@aannaguliev:~$ mv fun/ play/games  
aannaguliev@aannaguliev:~$ chmod u-r feathers  
aannaguliev@aannaguliev:~$ cat feathers  
cat: feathers: Отказано в доступе  
aannaguliev@aannaguliev:~$ cp feathers feathers2  
cp: невозможно открыть 'feathers' для чтения: Отказано в доступе  
aannaguliev@aannaguliev:~$ chmod u+r feathers  
aannaguliev@aannaguliev:~$ chmod u-x play/  
aannaguliev@aannaguliev:~$ cd play/  
bash: cd: play/: Отказано в доступе  
aannaguliev@aannaguliev:~$ chmod +x play/  
aannaguliev@aannaguliev:~$
```

Рис. 7: Работа с файлами и правами доступа

```

MOUNT(8)                                     System Administration                                     MOUNT(8)

NAME
    mount - mount a filesystem

SYNOPSIS
    mount [-h|-V]

    mount [-l] [-t fstype]

    mount -a [-ffnrsvw] [-t fstype] [-O optlist]

    mount [-fnrsvw] [-o options] device|mountpoint

    mount [-fnrsvw] [-t fstype] [-o options] device mountpoint

    mount --bind|--rbind|--move olddir newdir

    mount --make-[shared|slave|private|unbindable|rshared|rslave|rprivate|runbindable] mountpoint

DESCRIPTION
    All files accessible in a Unix system are arranged in one big tree, the file hierarchy, rooted at /. These files can be spread out over several devices. The mount command serves to attach the filesystem found on some device to the big file tree. Conversely, the umount(8) command will detach it again. The filesystem is used to control how data is stored on the device or provided in a virtual way by network or other services.

    The standard form of the mount command is:

        mount -t type device dir

    This tells the kernel to attach the filesystem found on device (which is of type type) at the directory dir. The option -t type is optional. The mount command is usually able to detect a filesystem. The root permissions are necessary to mount a filesystem by default. See section "Non-superuser mounts" below for more details. The previous contents (if any) and owner and mode of dir become invisible, and as long as this filesystem remains mounted, the pathname dir refers to the root of the filesystem on device.

    If only the directory or the device is given, for example:

        mount /dir

    then mount looks for a mountpoint (and if not found then for a device) in the /etc/fstab file. It's possible to
    Manual page mount(8) line 1 (press h for help or q to quit)
```

```
FSCK(8)                                     System Administration                                     FSCK(8)

NAME
    fsck - check and repair a Linux filesystem

SYNOPSIS
    fsck [-lsAVRTMNP] [-r [fd]] [-C [fd]] [-t fstype] [filesystem...] [--] [fs-specific-options]

DESCRIPTION
    fsck is used to check and optionally repair one or more Linux filesystems. filesystem can be a device name (e.g., /dev/hdc1, /dev/sdb2), a mount point (e.g., /, /usr, /home), or a filesystem label or UUID specifier (e.g., UUID=8868abf6-88c5-4a83-98b8-bfc24057f7bd or LABEL=root). Normally, the fsck program will try to handle filesystems on different physical disk drives in parallel to reduce the total amount of time needed to check all of them.

    If no filesystems are specified on the command line, and the -A option is not specified, fsck will default to checking filesystems in /etc/fstab serially. This is equivalent to the -As options.

    The exit status returned by fsck is the sum of the following conditions:

    0
        No errors

    1
        Filesystem errors corrected

    2
        System should be rebooted

    4
        Filesystem errors left uncorrected

    8
        Operational error

    16
        Usage or syntax error

    32
        Checking canceled by user request

Manual page fsck(8) line 1 (press h for help or q to quit)
```

```
mkfs(8)                                     System Administration                               mkfs(8)
```

NAME

mkfs - build a Linux filesystem

SYNOPSIS

mkfs [options] [-t type] [fs-options] device [size]

DESCRIPTION

This mkfs frontend is deprecated in favour of filesystem specific mkfs.<type> utils.

mkfs is used to build a Linux filesystem on a device, usually a hard disk partition. The device argument is either the device name (e.g., /dev/hda1, /dev/sdb2), or a regular file that shall contain the filesystem. The size argument is the number of blocks to be used for the filesystem.

The exit status returned by mkfs is 0 on success and 1 on failure.

In actuality, mkfs is simply a front-end for the various filesystem builders (mkfs.fstype) available under Linux. The filesystem-specific builder is searched for via your PATH environment setting only. Please see the filesystem-specific builder manual pages for further details.

OPTIONS

-t, --type type
Specify the type of filesystem to be built. If not specified, the default filesystem type (currently ext2) is used.

fs-options
Filesystem-specific options to be passed to the real filesystem builder.

-V, --verbose
Produce verbose output, including all filesystem-specific commands that are executed. Specifying this option more than once inhibits execution of any filesystem-specific commands. This is really only useful for testing.

-h, --help
Display help text and exit.

-V, --version
Print version and exit. (Option -V will display version information only when it is the only parameter, otherwise it will work as --verbose.)

BUGS

Manual page mkfs(8) line 1 (press h for help or q to quit)

```
KILL(1)                                     User Commands                                     KILL(1)
```

NAME

kill - terminate a process

SYNOPSIS

```
kill [-signal|-s signal|-p] [-q value] [-a] [--timeout milliseconds signal] [--] pid|name...
```

```
kill -l [number] | -L
```

DESCRIPTION

The command **kill** sends the specified signal to the specified processes or process groups.

If no signal is specified, the **TERM** signal is sent. The default action for this signal is to terminate the process. This signal should be used in preference to the **KILL** signal (number 9), since a process may install a handler for the TERM signal in order to perform clean-up steps before terminating in an orderly fashion. If a process does not terminate after a **TERM** signal has been sent, then the **KILL** signal may be used; be aware that the latter signal cannot be caught, and so does not give the target process the opportunity to perform any clean-up before terminating.

Most modern shells have a builtin **kill** command, with a usage rather similar to that of the command described here. The **--all**, **--pid**, and **--queue** options, and the possibility to specify processes by command name, are local extensions.

If signal is 0, then no actual signal is sent, but error checking is still performed.

ARGUMENTS

The list of processes to be signaled can be a mixture of names and PIDs.

pid

Each pid can be expressed in one of the following ways:

- n
where n is larger than 0. The process with PID n is signaled.
- 0
All processes in the current process group are signaled.
- 1
All processes with a PID larger than 1 are signaled.

Manual page kill(1) line 1 (press h for help or q to quit)

Выводы по проделанной работе

В ходе данной работы мы ознакомились с файловой системой Linux, её структурой, именами и содержанием каталогов. Научились совершать базовые операции с файлами, управлять правами их доступа для пользователя и групп. Ознакомились с Анализом файловой системы. А также получили базовые навыки по проверке использования диска и обслуживанию файловой системы.