Операционные системы и сети

Файловая система

Павел Филонов 31/01/2021

Требования к долговременному хранению

- Хранение больших объемов данных
- Информация должна сохранятся после завершения процесса
- Несколько процессов должны иметь возможность параллельного доступа к информации

Файлы

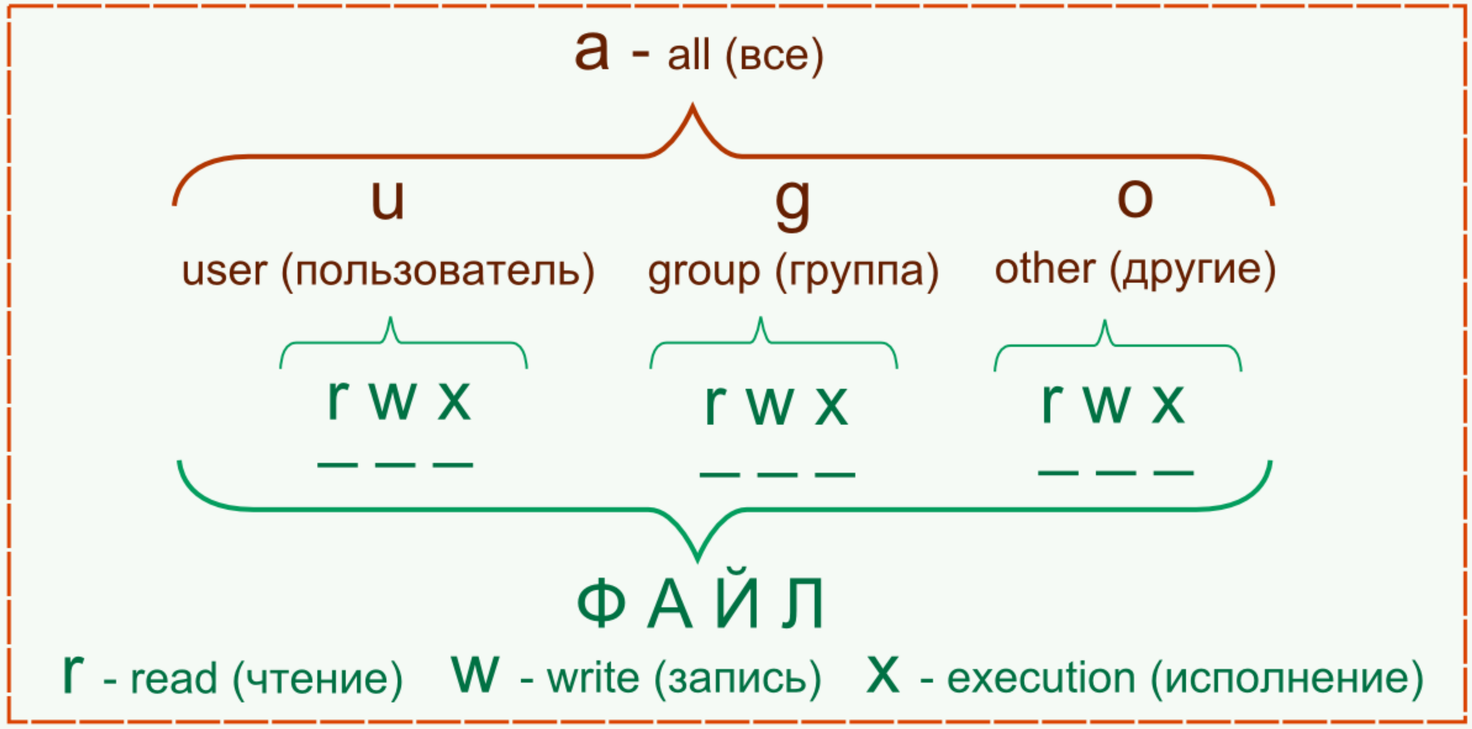
- Единица информации в файловой системе
- Процессы могут создавать и удалять файлы
- Информация в файле должна быть устойчивой (persistent). Т.е. Не нее не должно влиять завершение процесса
- Файлами управляет часть ОС, которая называется *файловой системой*

Атрибуты файлов

На примере POSIX
ls -l

Тип	Права доступа	Число ссылок	Владелец	Группа	Размер	Дата изменения	Имя
-	rwxr-xr-x	1	filonov_p	staff	49424	Jan 29 21:28	a.out*
-	rw-r--r--	1	filonov_p	staff	607	Jan 28 08:56	dc.c
-	rw-r--r--	1	filonov_p	staff	452	Jan 28 14:45	f2c.c
-	rwxr-xr-x	1	filonov_p	staff	49424	Jan 29 21:28	hello*
-	rw-r--r--	1	filonov_p	staff	251	Jan 28 14:45	hello.c
-	rw-r--r--	1	filonov_p	staff	0	Jan 29 17:08	longest_str.c
-	rw-r--r--	1	filonov_p	staff	390	Jan 28 09:00	power.c
-	rw-r--r--	1	filonov_p	staff	131	Jan 29 17:49	swap.c
-	rw-r--r--	1	filonov_p	staff	556	Jan 28 08:53	wc.c

Права доступа к файлам (rwx)



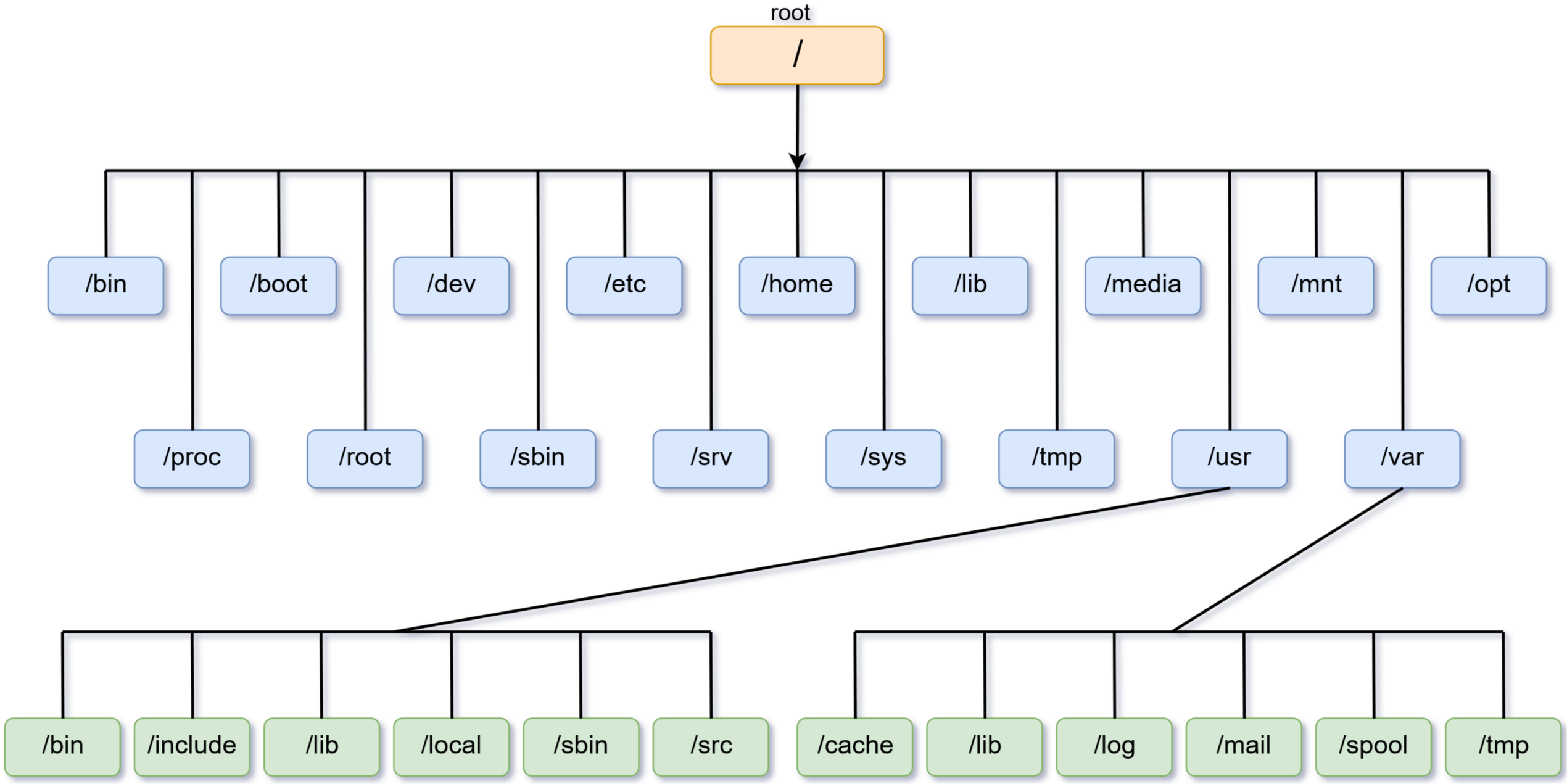
Примеры:

r w - r w - r w -	(все могут читать и изменять)
r w x - - - - - -	(полный доступ имеет владелец файла)
r w - r - - r - -	(все могут читать, владелец также изменять)
r w x r - x r - x	(все могут читать и исполнять, владелец также изменять)

Типы файлов

- - - обычный файл
- L - символическая ссылка (symbolic link)
- d - директория (хранит информацию об других файлах)
- c - символьное устройство
- b - блочное устройство
- p - канал (pipe)
- s - сокет (socket)

Структура каталогов Linux



Пути в файловой системе

- Разделитель каталогов
  - Windows - C:\Users\John\file.txt
  - UNIX - /home/John/file.txt
- . - текущий каталог
- .. - каталог верхнего уровня
- Абсолютный путь - от корневого каталога
  - /usr/bin/du
- Относительный путь - от рабочего каталога (есть в атрибутах у каждого процесса)
  - labs/os/lab1/fork.c
  - ../labs/os/lab1

Создание файлов

man creat

CREAT(2) BSD System Calls Manual CREAT(2)

NAME
creat -- create a new file

LIBRARY
Standard C Library (libc, -lc)

SYNOPSIS
#include <fcntl.h>

int
creat(const char *path, mode_t mode);

DESCRIPTION
This interface is made obsolete by: open(2).

The creat() function is the same as:

open(path, O_CREAT | O_TRUNC | O_WRONLY, mode);

SEE ALSO
open(2)

Пример. Создание файла

```c
/* Create empty file */

#include <fcntl.h>
#include <stdio.h>
#include <stdlib.h>
#include <unistd.h>

int main(int argc, char* argv[]) {
    if (argc < 2) {
        printf("Usage: %s <filepath>\n", argv[0]);
        return EXIT_FAILURE;
    }
    int fd = creat(argv[1], 0666); // mode will be affected by umask
    if (fd == -1) {
        perror("creat");
        return EXIT_FAILURE;
    }
    close(fd);
    return EXIT_SUCCESS;
}
```

Чтение из файла

man 2 read

READ(2) BSD System Calls Manual READ(2)

NAME
read -- read input

LIBRARY
Standard C Library (libc, -lc)

SYNOPSIS
```
#include <sys/types.h>
#include <sys/uio.h>
#include <unistd.h>

ssize_t
read(int fildes, void *buf, size_t nbyte);
```

DESCRIPTION
read() attempts to read nbyte bytes of data from the object referenced by the descriptor fildes into the buffer pointed to by buf.

RETURN VALUES
If successful, the number of bytes actually read is returned. Upon reading end-of-file, zero is returned. Otherwise, a -1 is returned and the global variable errno is set to indicate the error.

Пример. CRC-8

Запись в файл

man 2 write

WRITE(2) BSD System Calls Manual WRITE(2)

NAME
write -- write output

LIBRARY
Standard C Library (libc, -lc)

SYNOPSIS
#include <unistd.h>

ssize_t
write(int fildes, const void *buf, size_t nbyte);

DESCRIPTION
write() attempts to write nbyte of data to the object referenced by the descriptor fildes from the buffer pointed to by buf.

RETURN VALUES
Upon successful completion the number of bytes which were written is returned. Otherwise, a -1 is returned and the global variable errno is set to indicate the error.

Пример. Копирование файла

Прочие системные вызовы

- rename - переименование файла
- lseek - изменение текущего смещения в файле
- link - создать “жесткую” ссылку на существующий файл
- unlink - удаление ссылки на файл из директории
- ioctl - управление параметрами устройства, связанного с файловым дескриптором
- chmod - изменение прав доступа файла
- stat - чтение атрибутов файла

Копирование файловых дескрипторов

man dup

DUP(2) BSD System Calls Manual DUP(2)

NAME
dup, dup2 -- duplicate an existing file descriptor

SYNOPSIS
#include <unistd.h>

int
dup(int fildes);

int
dup2(int fildes, int fildes2);

DESCRIPTION
dup() duplicates an existing object descriptor and returns its value to the calling process (fildes2 = dup(fildes)). The argument fildes is a small non-negative integer index in the per-process descriptor table. The value must be less than the size of the table, which is returned by getdtablesize(2). The new descriptor returned by the call is the lowest numbered descriptor currently not in use by the process.

In dup2(), the value of the new descriptor fildes2 is specified. If fildes and fildes2 are equal, then dup2() just returns fildes2; no other changes are made to the existing descriptor. Otherwise, if descriptor fildes2 is already in use, it is first deallocated as if a close(2) call had been done first.

RETURN VALUES
Upon successful completion, the new file descriptor is returned. Otherwise, a value of -1 is returned and the global integer variable errno is set to indicate the error.

Пример. Перенаселение вывода

Работа с каталогом

man opendir

```
PENDIR(3)                Linux Programmer's Manual                OPENDIR(3)
NAME
       opendir - open a directory
SYNOPSIS
       #include <sys/types.h>
       #include <dirent.h>

       DIR *opendir(const char *name);

ESCRIPTION
       The opendir() function opens a directory stream corresponding to
       the directory name, and returns a pointer to the directory
       stream.  The stream is positioned at the first entry in the
       directory.

RETURN VALUE
       The opendir() function return a pointer to the
       directory stream.  On error, NULL is returned, and errno is set
       appropriately.
```

Чтение каталога

man readdir

```
EADDIR(3)                Linux Programmer's Manual                READDIR(3)
NAME
       readdir - read a directory
SYNOPSIS
       #include <dirent.h>

       struct dirent *readdir(DIR *dirp);
DESCRIPTION
       The readdir() function returns a pointer to a dirent structure
       representing the next directory entry in the directory stream
       pointed to by dirp.  It returns NULL on reaching the end of the
       directory stream or if an error occurred.

       In the glibc implementation, the dirent structure is defined as
       follows:
          struct dirent {
               ino_t          d_ino;       /* Inode number */
               off_t          d_off;       /* Not an offset; see below */
               unsigned short d_reclen;    /* Length of this record */
               unsigned char  d_type;      /* Type of file; not supported
                                              by all filesystem types */
               char           d_name[256]; /* Null-terminated filename */
           };
```

Пример. Список файлов

Пример. Атрибуты файла

Источники

- Э. Тененбаум. Операционные системы. Дизайн и разработка
- Введение в Linux и bash - https://younglinux.info/rwx
- Linux Filesystem Directory Architecture - site
- Циклический избыточный код - wikipedia.org
- Linux manuals - man7.org