Файловые системы

- Файловая система — формат хранения данных на внешнем устройстве
- Файловая система — компонента ядра процессора, обеспечивающая работу с файлами
- Файл — именованная последовательность данных

Организация файлов

- Файл как последовательность байт
- Файл как последовательность записей одинакового размера (VAX VMS)
- Файл как последовательность записей переменного размера

Организация файлов

- Однопоточные файлы
- Многопоточные файлы
  - MacOS: data fork/resource fork
  - NTFS: alternate data streams
    - CreateFile("data.txt:hidden", …)
  - Могут использоваться для сохранения метаинформации о файле

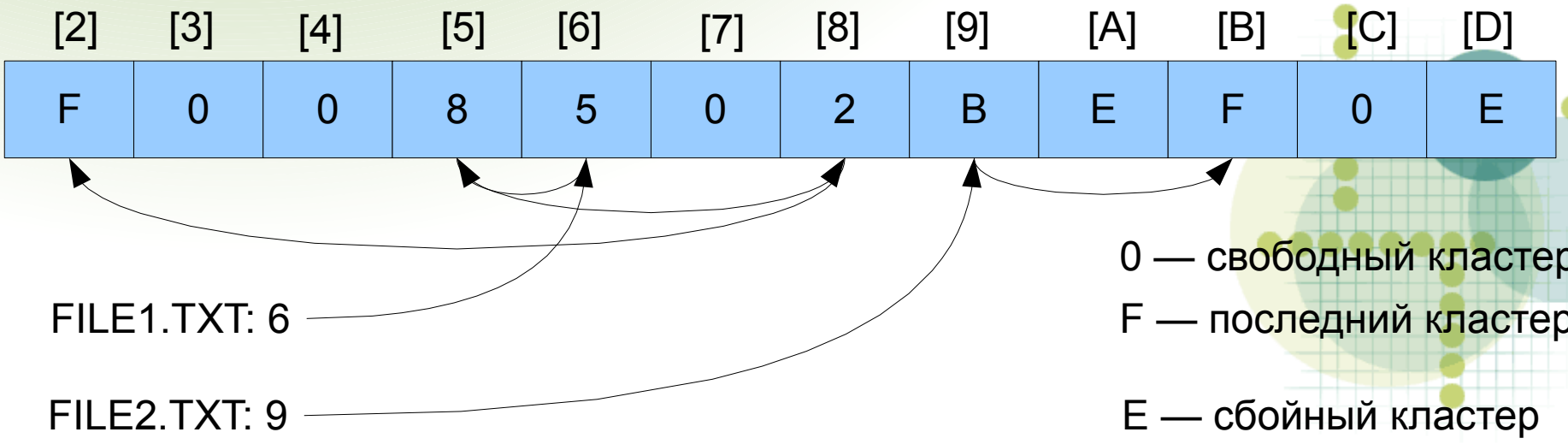
- Будем рассматривать модель однопоточных файлов — последовательностей байт

Файловая система RT-11

- Одноуровневая иерархия файлов
- Файлы хранятся в непрерывных областях области данных диска
- Имена файлов — 6 + 3 заглавные латинские буквы и цифры (кодируется в 6 байтах)
- Записи о файлах в каталоге диска располагаются в порядке размещения файлов в области данных диска, специальные записи для «дыр»

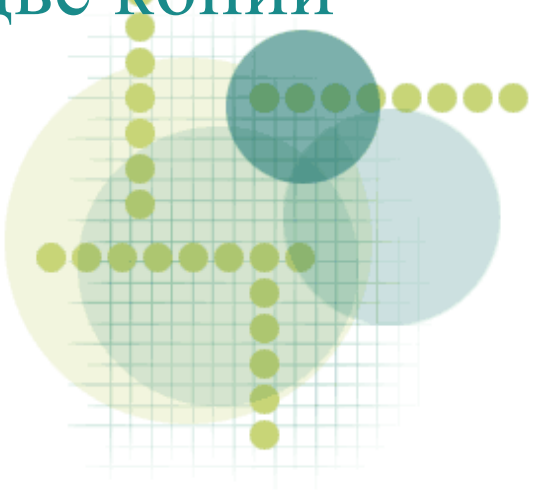
FAT

- Иерархическая файловая система
- Имена файлов: 8 + 3 (символы занимают один байт)
- Таблица размещения файлов:

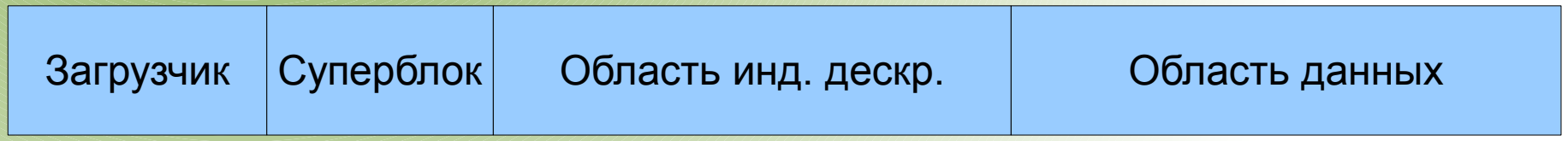


FAT

- Метаинформация о файле хранится в записях каталога
- FAT16: Максимальный размер кластера — 32 KiB (при размере блока 512 байт — 64 блока на кластер)
- Для надежности на диске хранится две копии FAT
- Для эффективной работы в памяти приходится держать FAT целиком
- Фрагментация файлов

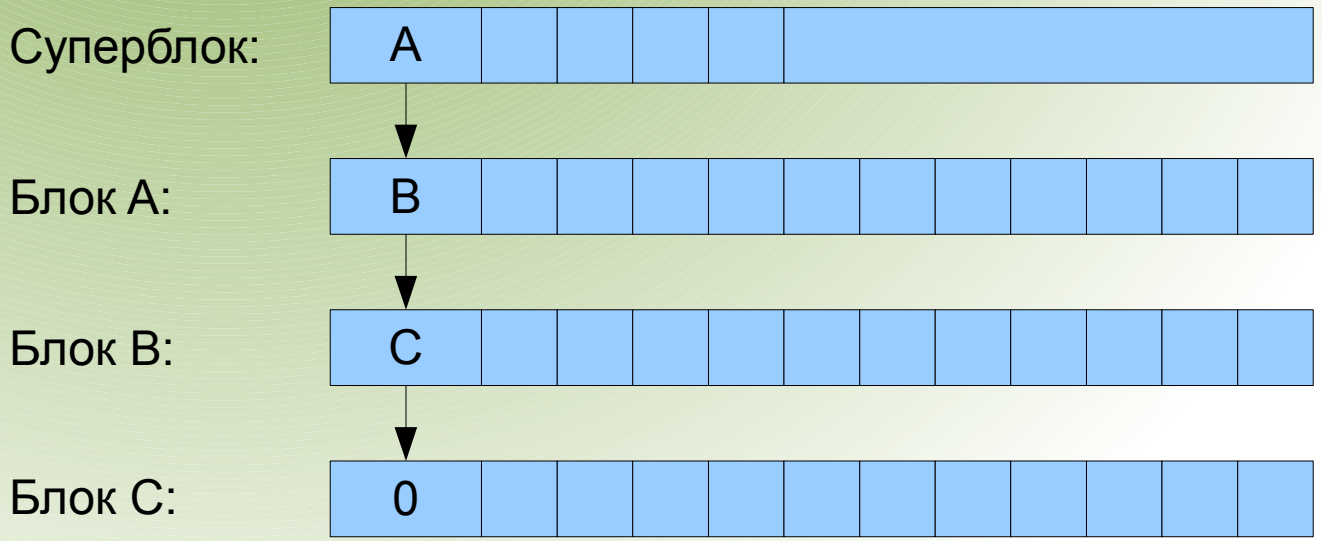


UNIX System V FS (s5fs)



- Суперблок хранит информацию о файловой системе:
  - Размер файловой системы в блоках
  - Размер области индексных дескрипторов (inode) в блоках
  - Число свободных блоков и инд. дескр.
  - Номер первого свободного инд. дескр.
  - Список свободных блоков данных (частично)
- Загружается в память при монтировании ФС

Список свободных блоков



- При удалении блока он добавляется в начало списка
- При выделении блока он берется из начала списка

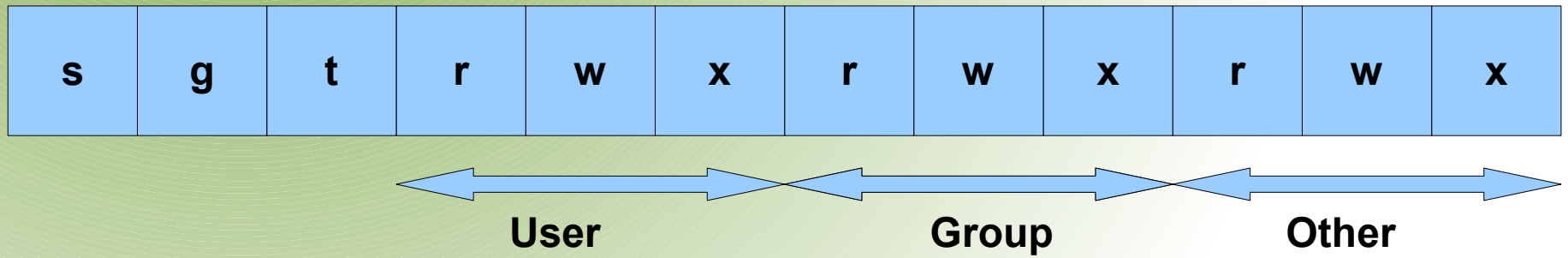
Индексный дескриптор (inode)

Поле	Размер	Описание
di_mode	2	Права доступа и тип файла
di_nlinks	2	Число ссылок на этот и. д.
di_uid	2	Идентификатор пользовател
di_gid	2	Идентификатор группы
di_size	4	Размер файла
di_addr	39	Массив адресов блоков данных
di_gen	1	Поколение
di_atime	4	Время посл. доступа к файлу
di_mtime	4	Время модификации файла
di_ctime	4	Время создания файла

Размер — 64 байта

Индексный дескриптор в памяти содержит дополнительные поля!

Права доступа



Бит	Влияние на файлы	Влияние на каталоги
x	Право выполнения	Право открытия файлов в этом каталоге
t	Не используется в н. вр.	Удалять файлы может только их владелец
g	Процесс, запускаемый из этого файла, может изменить эффективный ид. группы	Файлы и каталоги, создаваемые в этом каталоге, наследуют группу этого каталога

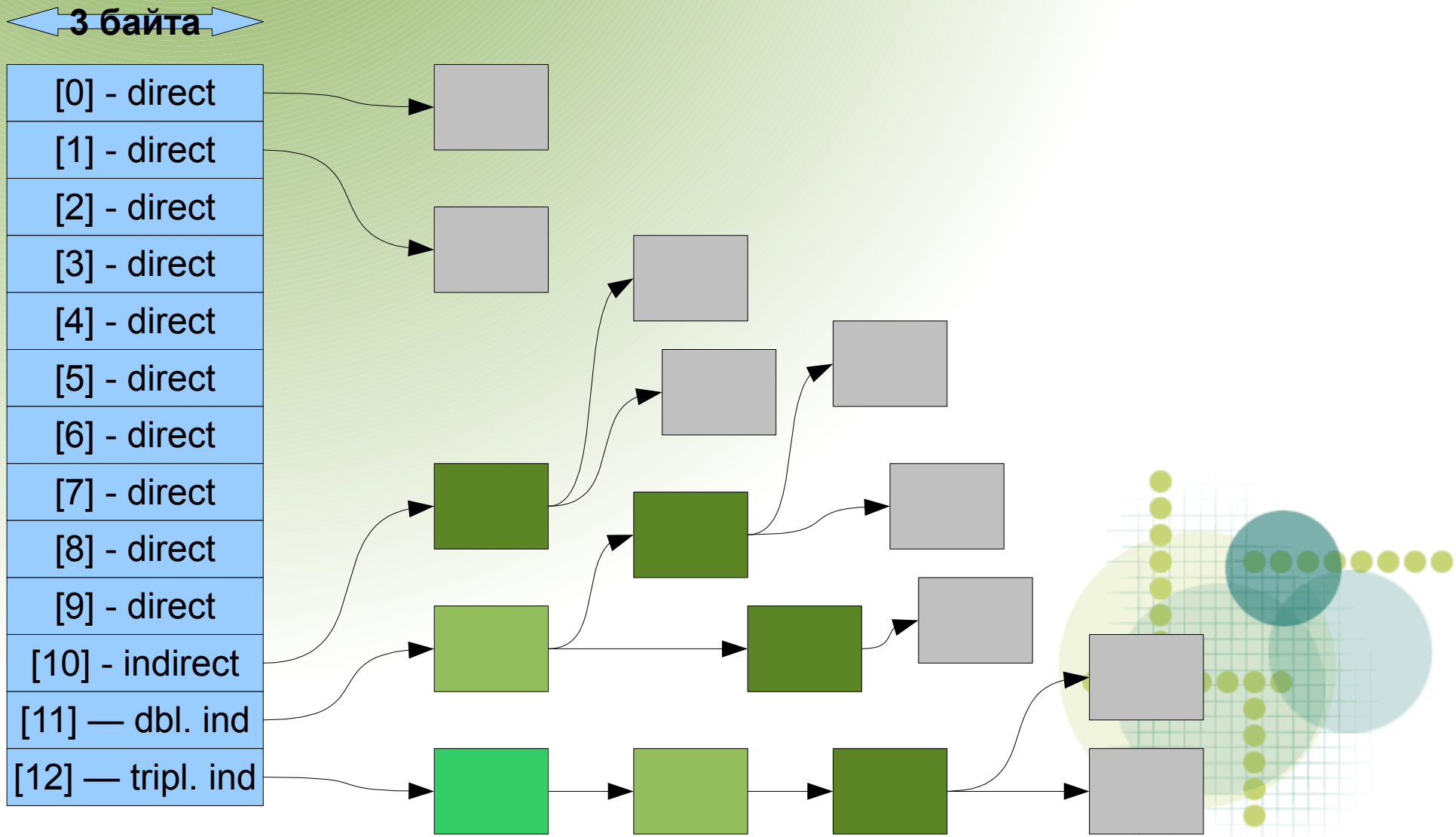
Типы файлов

- Регулярный файл
- Символическая ссылка — в наследниках s5fs
- Каталог
- UNIX-сокет
- FIFO
- Посимвольное устройство
- Блочное устройство

Время в UNIX

- Время (timestamp) является 32-битным значением
- Число секунд, прошедших от 01.01.1970
- Временная зона: UTC (универсальное скоординированное время) = GMT
- 19 января 2038 года счетчик секунд достигнет 0x7fffffff (макс. полож. зн-е) и станет 0x80000000 (мин. отр. зн-е), что соответствует 1901 году

Массив адресов блоков



Размеры файлов

- Размер блока — 512 байт
- 10 непоср. номеров блоков — 5 KiB
- Номер косв. блока - 128 номеров блоков
  - Итого: 10 + 128 = 138 блоков
- Номер двойного косв. блока - 128^2 блоков
  - Итого: 10 + 128 + 128^2 = ~8 MiB
- Номер тройного косв. блока - 128^3 блоков
  - Итого: 10 + 128 + 128^2 + 128^3 = 1GiB

Структура каталога

- Каталог — файл, содержащий список файлов и каталогов
- Каждая запись в каталоге — 16 байт
  - Имя файла — 14 байтов
  - Номер индексного дескриптора — 2 байта

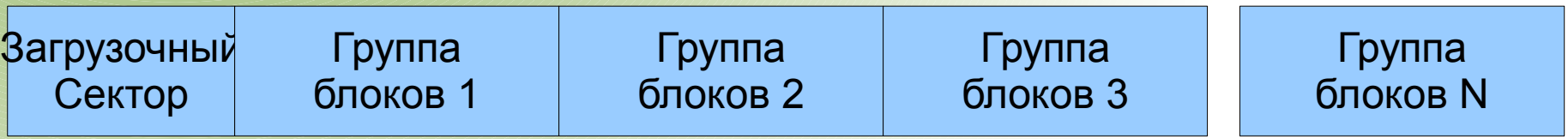
Жесткие ссылки (связи)

- Несколько записей в каталоге могут содержать один и тот же номер индексного дескриптора
- Такие записи называются **жесткими ссылками** на файл
- Все жесткие ссылки неразличимы, невозможно установить, какая из них «главнее»

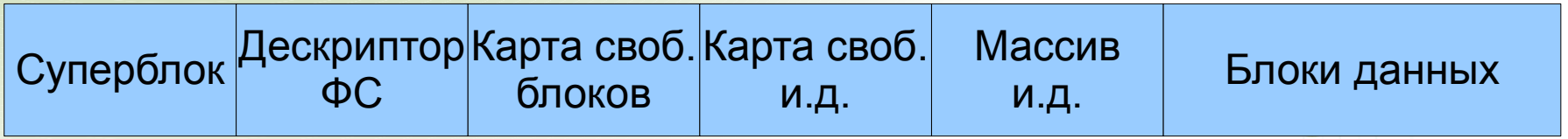
Недостатки

- Суперблок может быть поврежден
- Размер блока недостаточный (низкая скорость передачи)
- Блоки файлов и каталогов разбросаны по диску
- Индексные дескрипторы находятся далеко от блоков данных

Файловая система Ext2 (Linux)



Группа блоков:



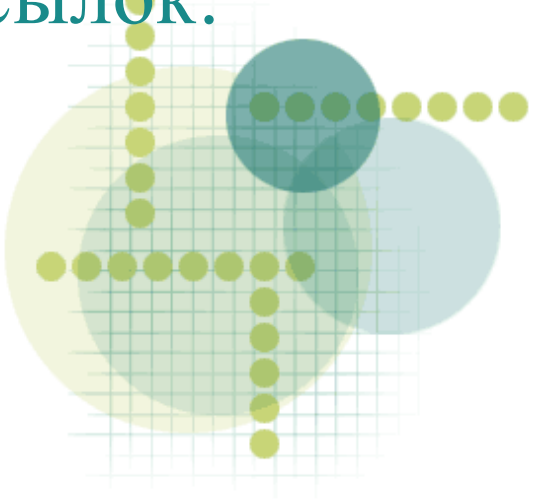
- Размер блока данных: 1024, 2048, 4096 байт
- Номера индексных дескрипторов и блоков — 32 битные беззнаковые
- Размер индексного дескриптора — 128 байт
- Запись в каталоге имеет переменный размер (до 256 с)

Символические ссылки

- Специальный тип файла, содержимое которого содержит путь к другому файлу
- Путь интерпретируется на уровне ядра ОС и прозрачен для программ
- Символическая ссылка на файл вторична по отношению к файлу
  - Удаление символической ссылки не приводит к удалению файла
  - Удаление файла не приводит к удалению символической ссылки — висящая ссылка
- Могут пересекать границы ФС

Символические ссылки

- Символические ссылки могут указывать на файлы на другой файловой системе
- Если длина имени не превышает 60 символов, сама символическая ссылка хранится не в блоках данных, а в индексном дескрипторе
- Команда создания символических ссылок:
  ln -s OLDNAME NEWNAME

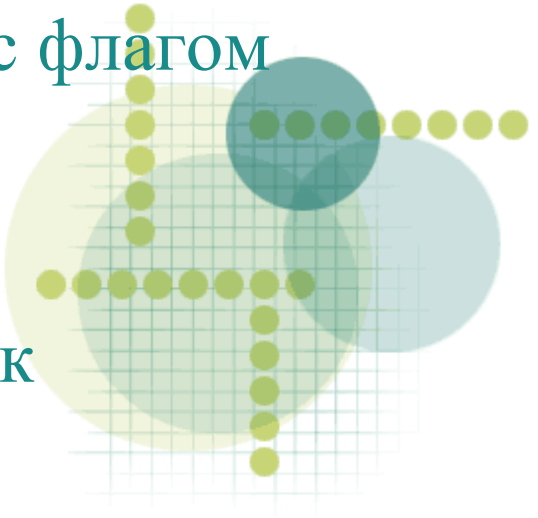


Ext3 (Linux)

- Совместима снизу вверх с ext2
- Обеспечивает журналирование и индексирование больших каталогов
- Максимальный размер файла увеличен до 2 TiB = 2 * 1024 GiB

Журналирование

- Обеспечение целостности файловой системы в случае краха ОС или сбоя питания
- Журнал — специальная область на диске
- Каждая операция, модифицирующая данные, выполняется в три стадии:
  - В журнал записывается операция (с флагом невыполненной)
  - Выполняется операция
  - Операция в журнале помечается как выполненная



Квотирование ФС

- В многопользовательской системе недопустима ситуация, когда один пользователь захватил все дисковое пространство
- Квота: жесткая (нельзя превышать), мягкая (выдается предупреждение, необходимо освободить место до истечения определенного интервала времени)
- Квотируются: блоки данных, индексные дескрипторы

Требования к ФС

- Обеспечение хранения больших объемов информации
- Полное использование возможностей аппаратуры (пропускной способности)
- Обеспечение восстановления информации после сбоев
- Разграничение прав доступа пользователей
- Обеспечение квотирования дискового пространства

Другие ФС

- Iso9660 (+Juliet) — ФС для CD и DVD носителей
- NFS, SMB — сетевые файловые системы
- NTFS — файловая система Windows NT+
- FFS — файловая система BSD

Системные вызовы работы с файлами POSIX

- open
- read, write, read64, write64
- lseek, llseek
- close
- dup, dup2

Файловый дескриптор

- Небольшое неотрицательное целое число, идентифицирует структуры данных в ядре.
- Массив файловых дескрипторов создается для каждого процесса
- Процесс не может иметь открытых файлов больше, чем размер массива файловых дескрипторов
- Максимальный размер массива файловых дескрипторов может быть установлен для каждого процесса или каждого пользователя

Открытие файла

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>

int open(const char *path, int flags, mode_t mode);
```

- Возвращает номер открытого файлового дескриптора при успехе
- Возвращает -1 при ошибке. Переменная errno содержит код ошибки.

Открытие файла: флаги

- Режим открытия: один из:
  - O_RDONLY, O_WRONLY, O_RDWR
- Флаги при открытии на запись:
  - O_CREAT — создание файла
  - O_TRUNC — очистка файла
  - O_APPEND — режим добавления
  - O_EXCL — эксклюзивный режим открытия
  - O_NDELAY — неблокирующий режим
  - O_LARGEFILE — файл большого размера

Права доступа к создаваемому файлу

- Права доступа к файлу:
  perms = mode & ~umask
- umask — параметр процесса
- Чтение/изменение umask
  int umask(int newumask);
- Возвращается старое значение и устанавливается новое (только биты 0777)
- Пример: параметр при создании: 0666, umask: 022, права доступа: 0644

Закрытие файлового дескриптора

int close(int fd);

- Возвращается 0 при успехе и -1 при ошибке
- Игнорирование ошибки закрытия может привести к потере данных

Копирование файловых дескрипторов

```
#include <unistd.h>

int dup(int oldfd);
int dup2(int oldfd, int newfd);
```

- newfd закрывается перед созданием копии

Чтение из файла

ssize_t read(int fd, void *buf, size_t count);

- Возвращается -1 при ошибке
- 0 при достижении конца файла
- Размер реально считанных данных при успешном чтении

Запись в файл

ssize_t write(int fd, const void *buf, size_t count);

- Возвращается -1 при ошибке,
- Возвращается размер записанных данных

Позиционирование

off_t lseek(int fd, off_t offset, int whence);

- Режимы позиционирования:
  - SEEK_SET — от начала файла
  - SEEK_CUR — от текущего положения
  - SEEK_END — от конца файла
- Возвращается -1 при ошибке или предыдущее положение в файле