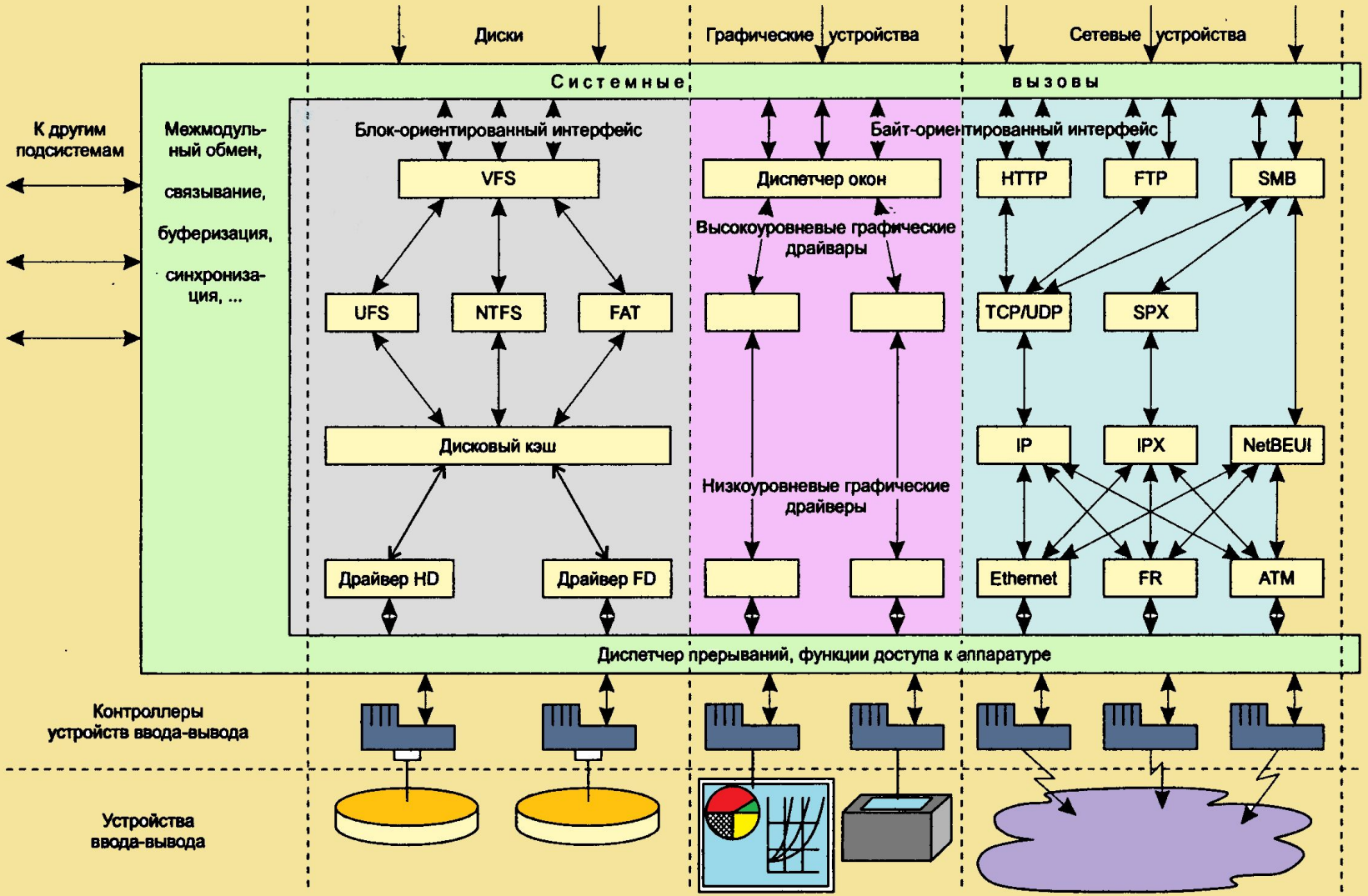
РАЗДЕЛ №4

Подсистема ввода-вывода и файловые системы

Задачи ОС по управлению файлами и устройствами

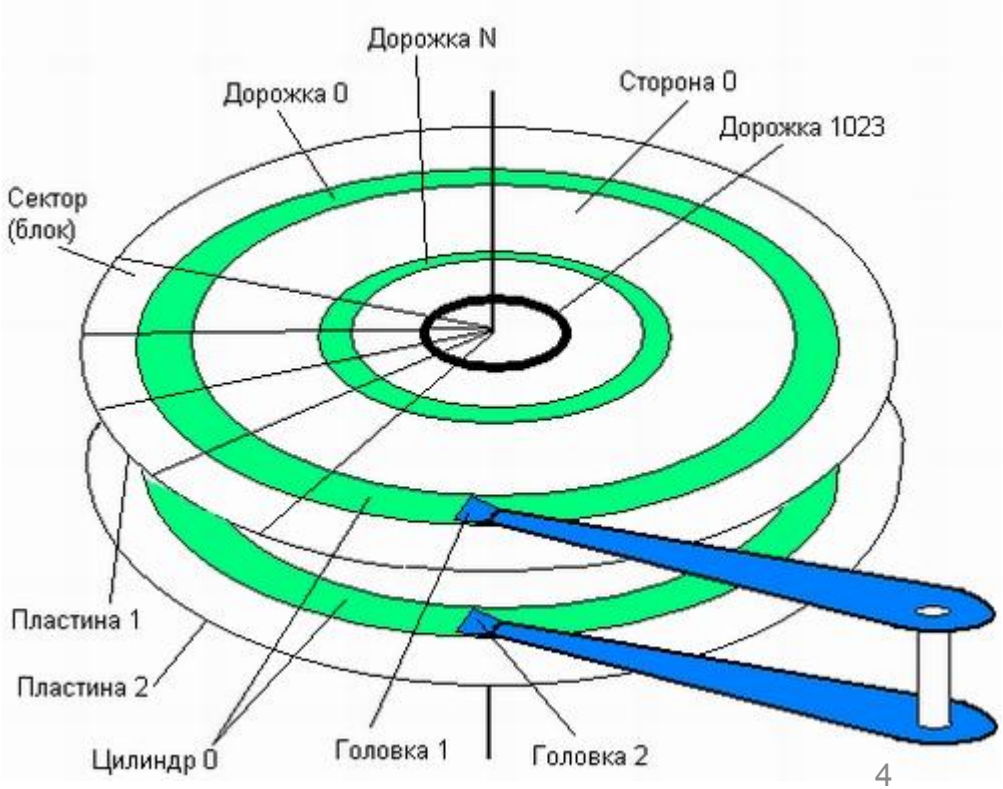
1. Организация параллельной работы устройств ввода-вывода и процессора;
2. согласование скоростей обмена и кэширование данных;
3. разделение устройств и данных между процессами;
4. организация удобного интерфейса между устройствами и остальной частью системы;
5. Поддержка широкого спектра драйверов с возможностью простого включения в систему нового драйвера;
6. динамическая загрузка и выгрузка драйверов;
7. поддержка нескольких файловых систем;
8. поддержка синхронных и асинхронных операций ввода-вывода;

Многослойная модель подсистемы ввода/вывода



Файловые системы

Устройство жесткого диска



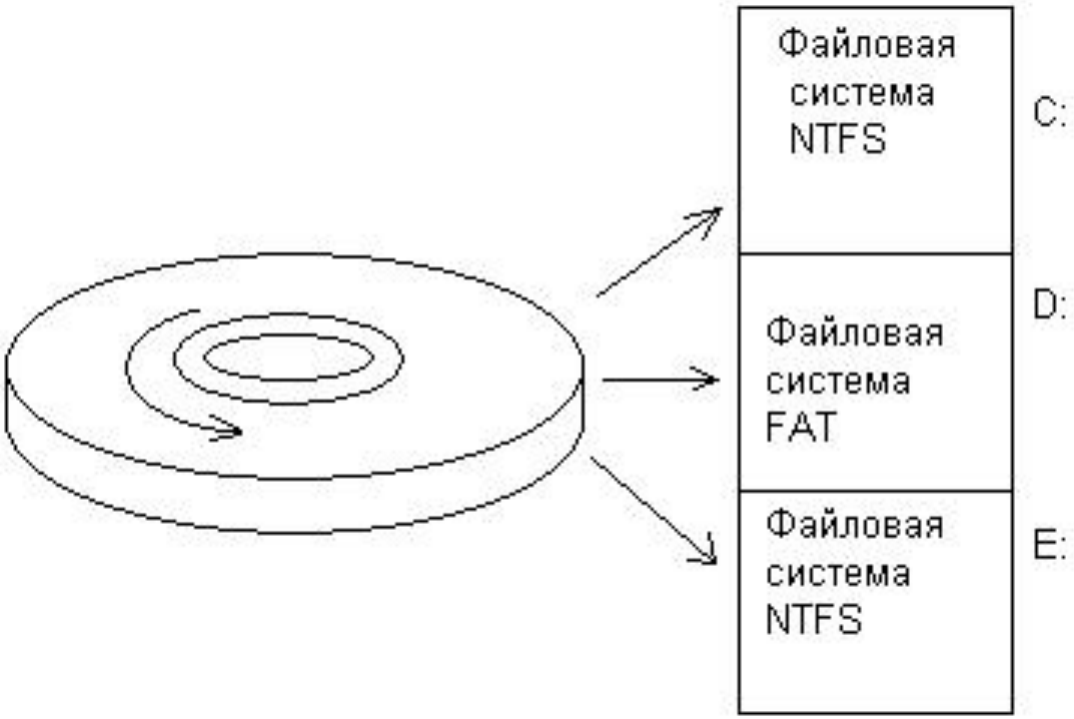
SSD (*solid state drive, накопитель на твёрдотельной памяти, твёрдотельный накопитель*) - накопитель информации, основанный на чипах **энергонезависимой** памяти, которые сохраняют данные после отключения питания.

Достоинства SSD накопителей в сравнении с HDD:

- ✔ Включаются мгновенно, не требуют раскрутки.
- ✔ Значительно более высокая скорость произвольного доступа.
- ✔ Значительно более высокая скорость доступа.
- ✔ Скорость передачи данных значительно выше.
- ✔ Не требуется дефрагментация.
- ✔ Беззвучны, так как не имеют механических частей.
- ✔ Не создают вибраций.
- ✔ Более выносливы в плане температуры, ударов и вибраций.
- ✔ Немного меньшее энергопотребление.

Недостатки SSD накопителей в сравнении с HDD:

Износ ячеек. Хоть в SSD накопителях и отсутствуют механические части, чипы памяти изнашиваются (mlc ~10000 перезаписей, slc ~100000).
Ёмкость значительно меньше.
Цена значительно выше по соотношению ГБ/$
Невозможность восстановить утерянные данные после команды TRIM или просто после форматирования.


Файловая система NTFS
C:
Файловая система FAT
D:
Файловая система NTFS
E:

Файловая система

- Файл – это именованная область внешней памяти, в которую можно записывать и из которой можно считывать данные.

Основные назначения файлов:

✔ долговременное и надежное хранение информации;

✔ совместное использование информации.

- Файловая система - часть операционной системы, включающая:

✔ совокупность всех файлов на диске;

✔ наборы структур данных, используемых для управления файлами, такие, например, как каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске;

✔ комплекс системных программных средств, реализующих управление файлами, в частности: создание, уничтожение, чтение, запись, именование, поиск и другие операции над

Типы файлов:

- **Обычные файлы**
- содержат информацию произвольного характера, которую заносит в них пользователь или которая образуется в результате работы системных и пользовательских программ; подразделяются на текстовые и двоичные.

- **Специальные файлы**
- - это файлы, ассоциированные с устройствами ввода-вывода, которые позволяют пользователю выполнять операции ввода-вывода, используя обычные команды записи в файл или чтения из файла.

- **Каталог** - это особый тип файла, содержащий системную справочную информацию о наборе файлов, сгруппированных пользователем по какому-либо неформальному признаку.

- Другие типы файлов символьные связи;
- именованные конвейеры
- файлы, отображаемые на память

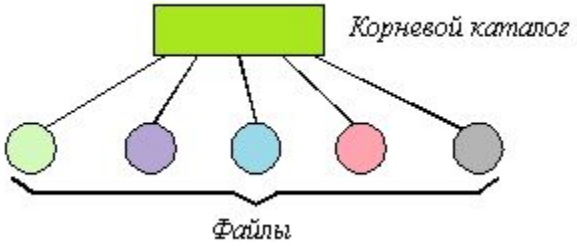
Структура записи каталога FAT

Структура записи каталога UNIX

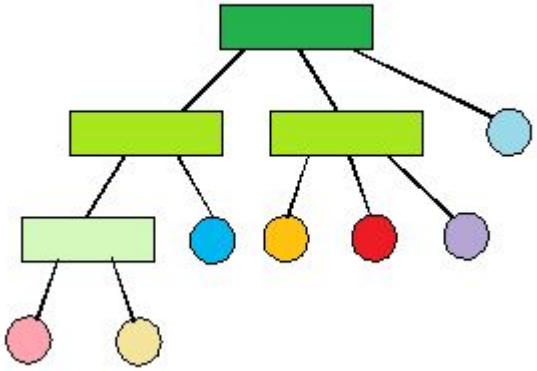
Атрибуты файла

- тип файла (обычный файл, каталог, специальный файл и т. п.);
- владелец файла;
- создатель файла;
- пароль для доступа к файлу;
- информация о разрешенных операциях доступа к файлу;
- времена создания, последнего доступа и последнего изменения;
- текущий размер файла;
- максимальный размер файла;
- признак «только для чтения»;
- признак «скрытый файл»;
- признак «системный файл»;
- признак «архивный файл»;
- признак «двоичный/символьный»;
- признак «временный» (удалить после завершения процесса);
- признак блокировки;
- длина записи в файле;
- указатель на ключевое поле в записи;
- длина ключа.

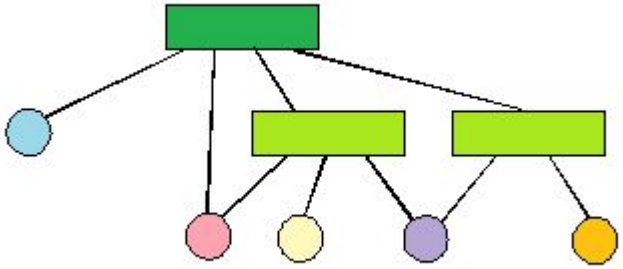
Логическая организация файловой системы



- одноуровневая

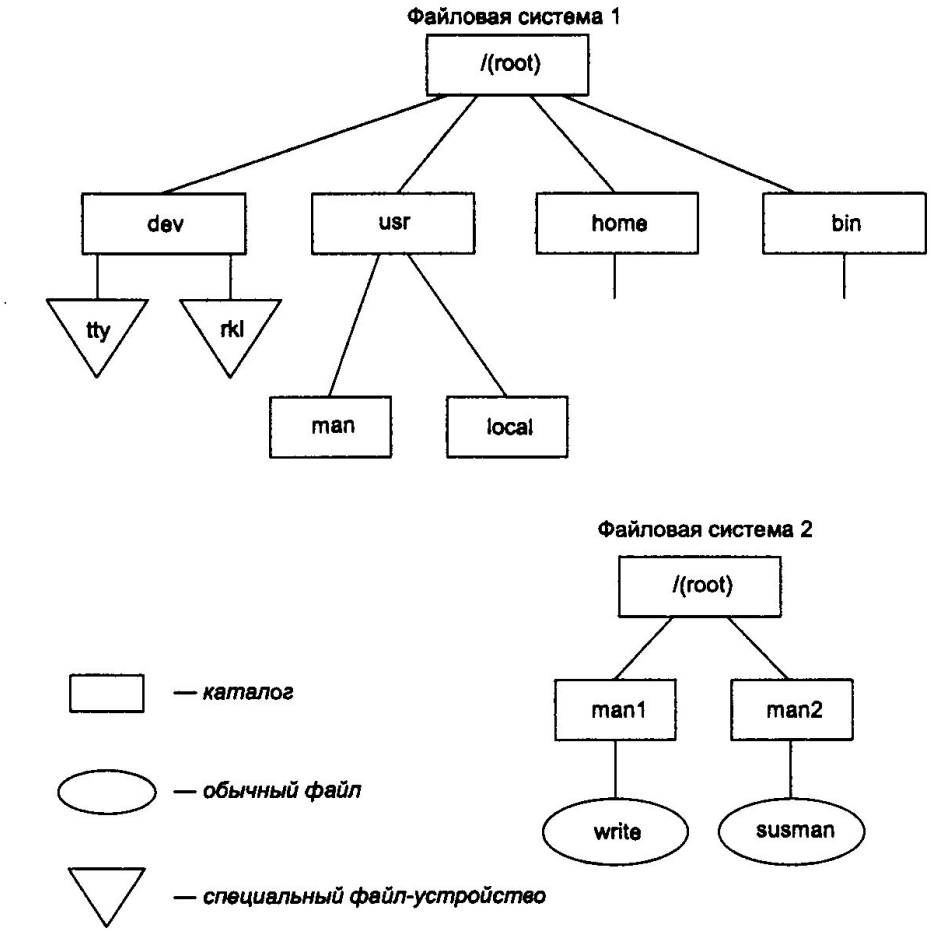


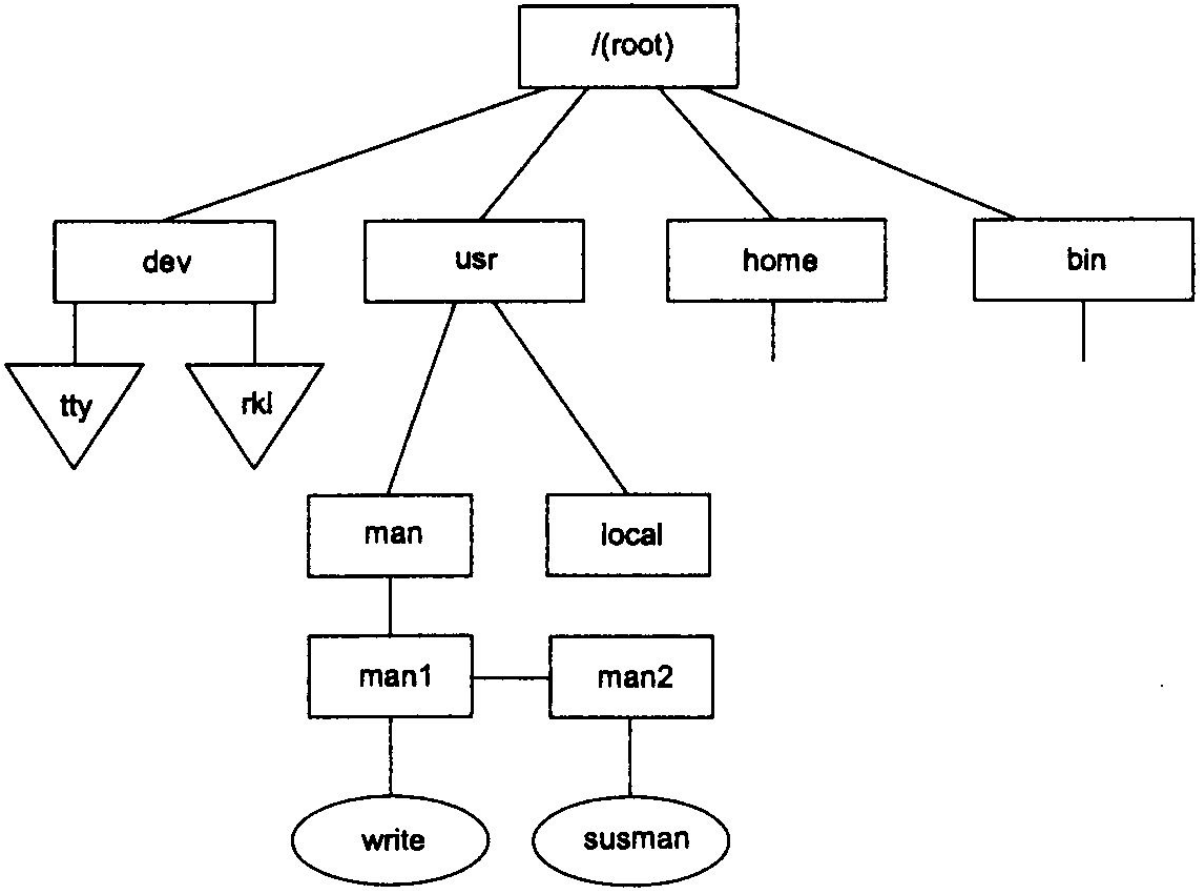
- иерархическая (дерево)



- иерархическая (сеть)

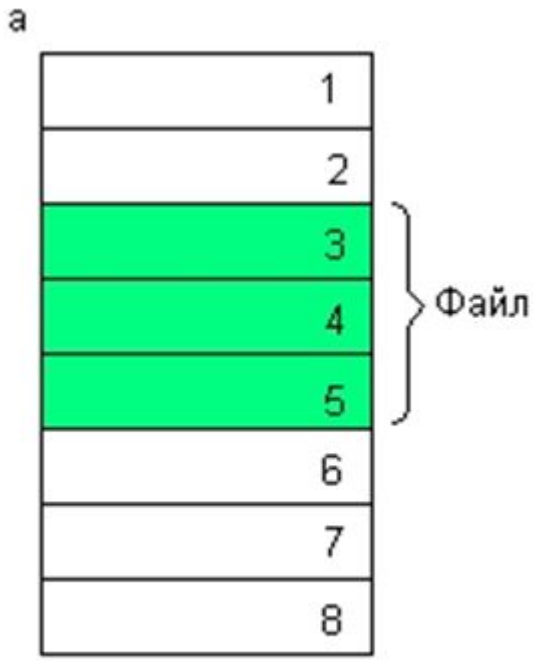
Монтирование



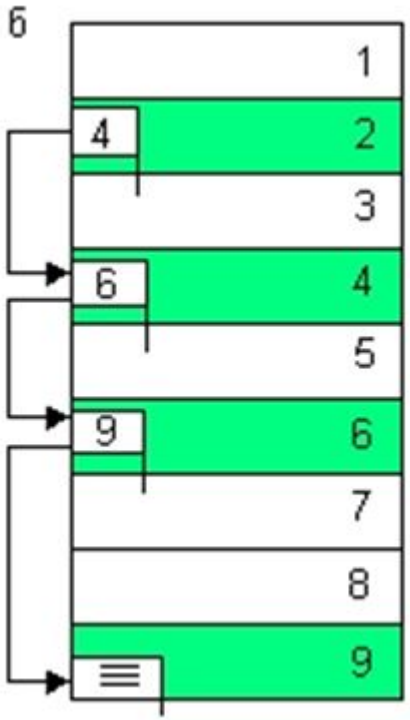

/(root)
dev
usr
home
bin
tty
rkl
man
local
man1
man2
write
susman

Физическая организация и адресация файла

Критерии эффективности:

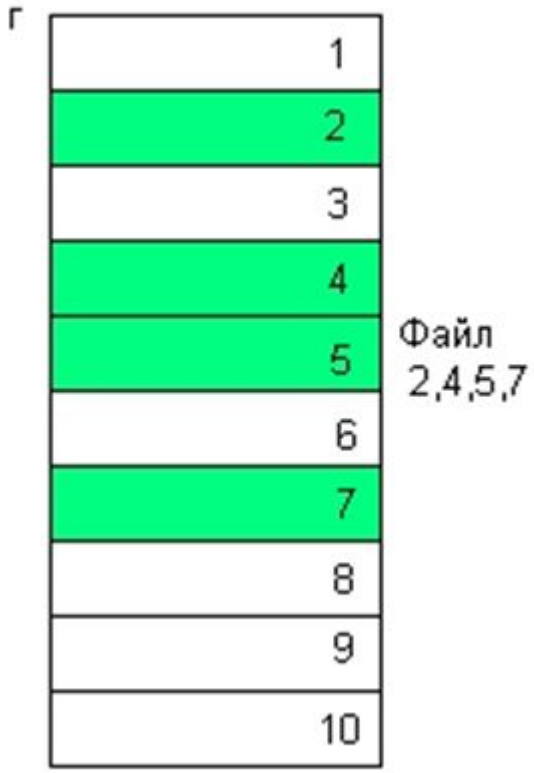


Непрерывное размещение



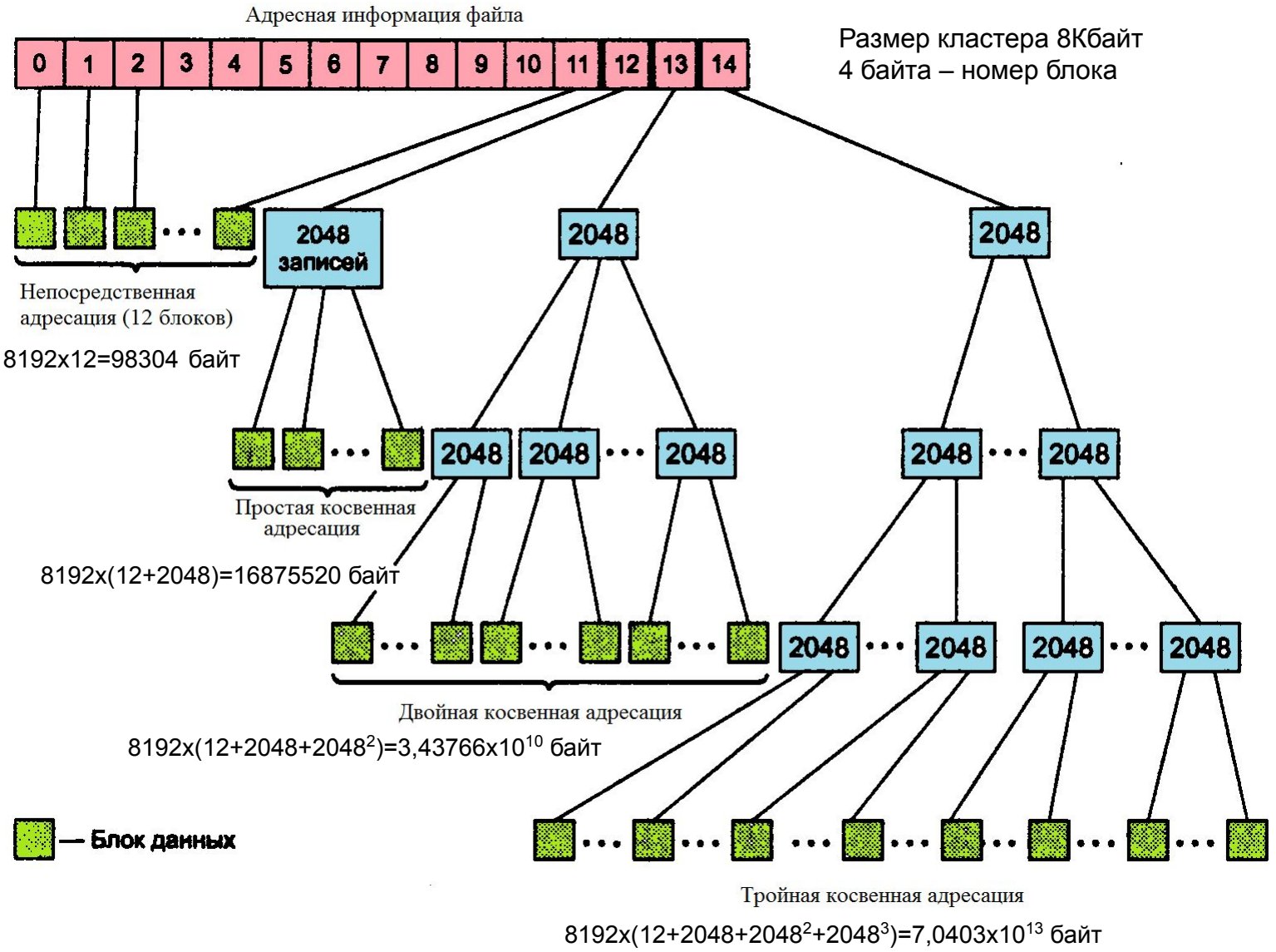
Связанный список кластеров

Связанный список индексов

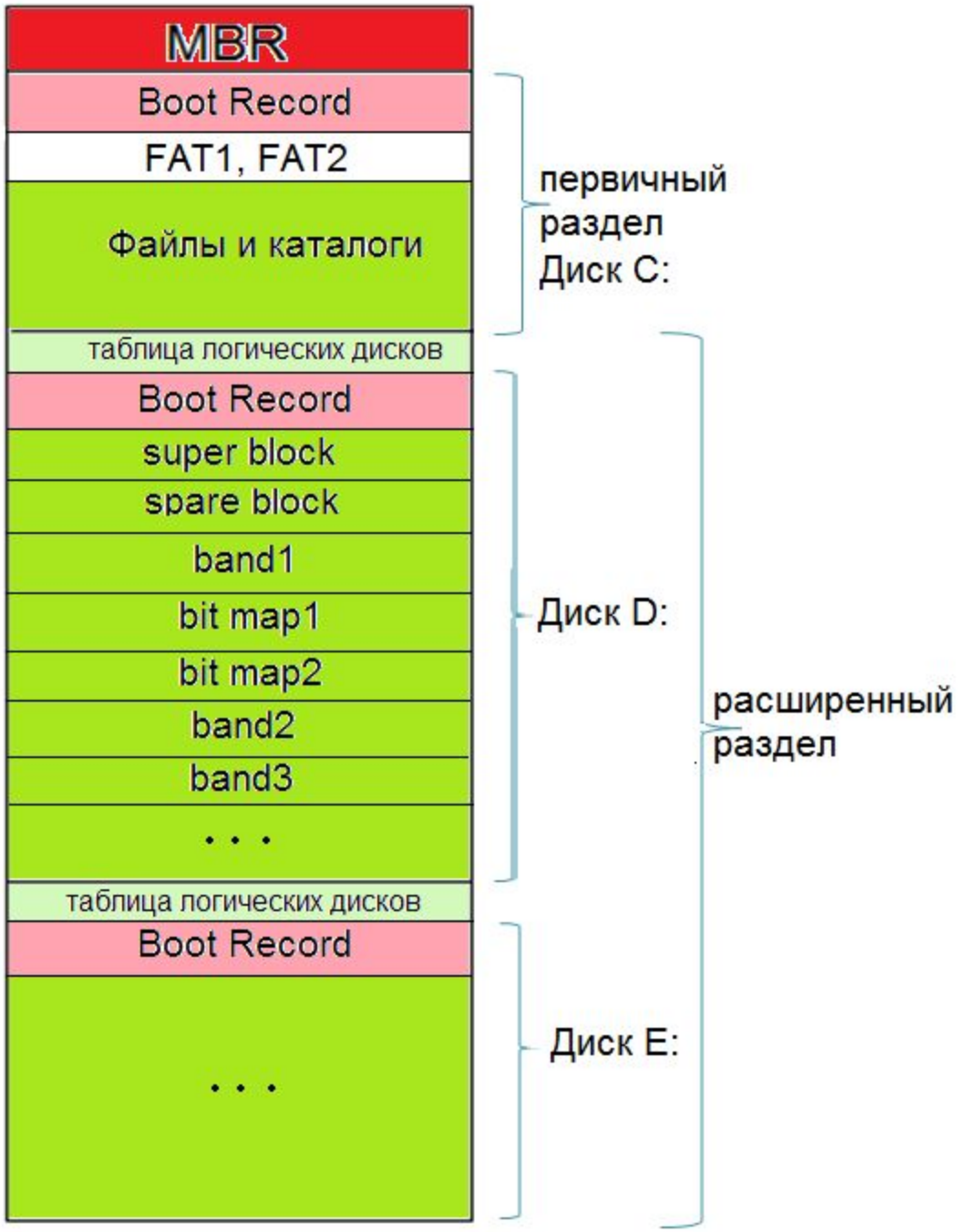


Прямая адресация

Схема адресации файловой системы UFS



Структура главной загрузочной записи



ение	Длина	Описание
	446	Программа начальной загрузки
	64	Таблица разделов
	16	Раздел 1
	16	Раздел 2
	16	Раздел 3
	16	Раздел 4
	2	Сигнатура (55 AAh)

MBR — традиционная структура для управления разделами диска. Поскольку она совместима с большинством систем, то по-прежнему широко Диск MBR допускает только четыре основных раздела. Если нужно больше, то можно назначить один из разделов расширенным разделом, и на нём можно создавать больше подразделов или логических дисков.
MBR использует 32 бита для записи длины раздела, выраженной в секторах, так что каждый раздел ограничен максимальным разделом 2 ТБ.

Преимущества

Совместима с большинством систем.

Недостатки

- Допускает только четыре раздела, с возможностью создания дополнительных подразделов на одном из основных разделов.
- Ограничивает размер раздела двумя терабайтами.
- Информация о разделе хранится только в одном месте — в главной загрузочной записи. Если она повреждена, то весь диск становится нечитаемым.

Таблица разделов GUID (GPT)

GPT — более новый стандарт для определения структуры разделов на диске. Для определения структуры используются глобальные уникальные идентификаторы (GUID).
Это часть стандарта UEFI, то есть систему на основе UEFI можно установить только на диск, использующий GPT.

Преимущества

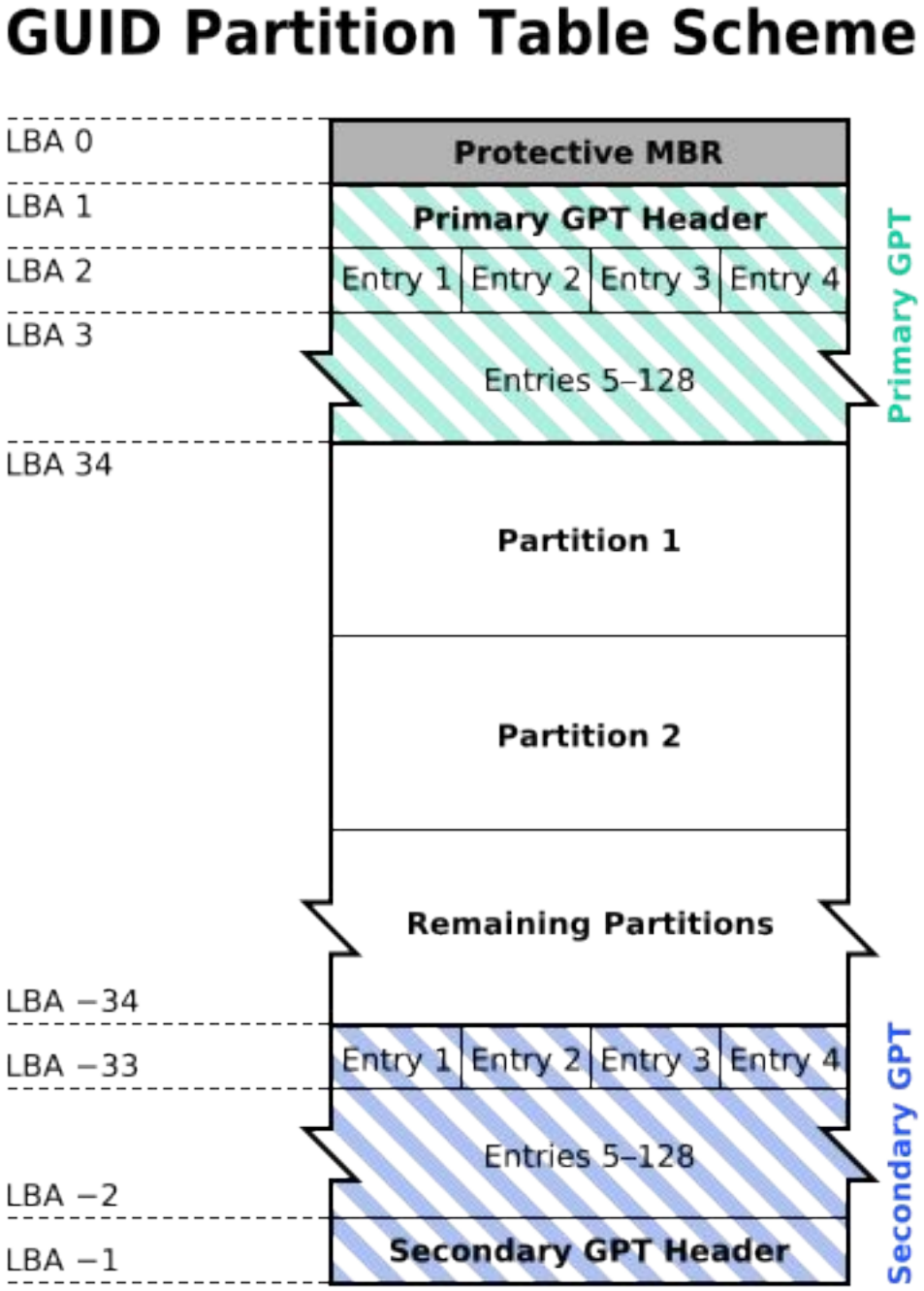
- Допускает неограниченное количество разделов. Лимит устанавливает операционная система, например, Windows допускает не более 128 разделов.
- Ограничение на максимальный размер раздела больше, чем объём любых существующих сегодня дисков. Для дисков с секторами по 512 байт поддерживается максимальный размер 9,4 ЗБ (один зеттабайт равен 1 073 741 824 терабайт)
- GPT хранит копию раздела и загрузочных данных и может восстановить данные в случае повреждения основного заголовка GPT.
- GPT хранит значения контрольной суммы по алгоритму циклического избыточного кода (CRC) для проверки целостности своих данных (используется для проверки целостности данных заголовка GPT). В случае повреждения GPT может заметить проблему и попытаться восстановить повреждённые данные из другого места на диске.

Недостатки

Может быть несовместима со старыми системами.

GUID Partition Table (GPT)

- стандартный формат размещения таблиц разделов на физическом жестком диске. Является частью Extensible Firmware Interface (EFI) — стандарта, предложенного Intel на смену BIOS. EFI использует GPT там, где BIOS использует Главную загрузочную запись (MBR).



Ассоц. платф.	Тип раздела	Глобально уникальный идентификатор (GUID)
(Нет)	Неиспользуемая запись данных	00000000-0000-0000-0000-000000000000
	Схема разделов MBR	024DEE41-33E7-11D3-9D69-0008C781F39F
	Системный раздел EFI	C12A7328-F81F-11D2-BA4B-00A0C93EC93B
Windows	Резервный раздел Microsoft	E3C9E316-0B5C-4DB8-817D-F92DF00215AE
	Раздел основных данных	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
	Менеджер логических томов, раздел мета-данных	5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
	Менеджер логических томов, раздел данных	AF9B60A0-1431-4F62-BC68-3311714A69AD
HP-UX	Раздел данных	75894C1E-3AEB-11D3-B7C1-7B03A0000000
	Раздел Сервиса	E2A1E728-32E3-11D6-A682-7B03A0000000
Linux	Раздел данных	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
	RAID раздел	A19D880F-05FC-4D3B-A006-743F0F84911E
	Своп-раздел	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
	Раздел Менеджера логических томов (LVM)	E6D6D379-F507-44C2-A23C-238F2A3DF928
	Зарезервировано	8DA63339-0007-60C0-C436-083AC8230908
FreeBSD	Загрузочный раздел	83BD6B9D-7F41-11DC-BE0B-001560B84F0F
	Раздел данных	516E7CB4-6ECF-11D6-8FF8-00022D09712B
	Своп-раздел	516E7CB5-6ECF-11D6-8FF8-00022D09712B
	Раздел UFS (Unix File System)	516E7CB6-6ECF-11D6-8FF8-00022D09712B
	Раздел менеджера томов Vinum	516E7CB8-6ECF-11D6-8FF8-00022D09712B
	Раздел ZFS	516E7CBA-6ECF-11D6-8FF8-00022D09712B
Mac OS X	Раздел HFS+ (Hierarchical File System)	48465300-0000-11AA-AA11-00306543ECAC
	Apple UFS	55465300-0000-11AA-AA11-00306543ECAC
	ZFS	6A898CC3-1DD2-11B2-99A6-080020736631
	Apple RAID раздел	52414944-0000-11AA-AA11-00306543ECAC
	Apple RAID раздел, offline	52414944-5F4F-11AA-AA11-00306543ECAC
	Загрузочный раздел Apple	426F6F74-0000-11AA-AA11-00306543ECAC
	Apple Label	4C616265-6C00-11AA-AA11-00306543ECAC
	Apple TV Recovery partition	5265636F-7665-11AA-AA11-00306543ECAC
Solaris	Загрузочный раздел	6A82CB45-1DD2-11B2-99A6-080020736631
	Корневой раздел	6A85CF4D-1DD2-11B2-99A6-080020736631

Совместимость с операционными системами

Первый сектор (сектор 0) на диске GPT содержит защитную запись MBR, в которой записано, что на диске один раздел, который распространяется на весь носитель. В случае использования старых инструментов, которые читают только диски MBR, вы увидите один большой раздел размером с весь диск. Защитная запись сделана для того, чтобы старый инструмент ошибочно не воспринял диск как пустой и не перезаписал данные GPT новой главной загрузочной записью.
MBR защищает данные GPT от перезаписи.

Apple MacBook'и используют GPT по умолчанию, так что невозможно установить Mac OS X на систему MBR. Даже хотя Mac OS X может работать на диске MBR, но установка на него невозможна.
Большинство операционных систем на ядре Linux совместимы с GPT. При установке ОС Linux на диск в качестве загрузчика будет установлен GRUB 2.

Для операционных систем Windows загрузка из GPT возможна только на компьютерах с UEFI, работающих под 64-битными версиями Windows Vista, 7, 8, 10 и соответствующими серверными версиями.
Windows 7 и более ранние системы обычно устанавливают на диски с MBR, но их можно преобразовать разделы в GPT.

Все версии Windows Vista, 7, 8, 10 могут считывать и использовать данные из разделов GPT — но они не могут загружаться с таких дисков без UEFI.

Структура тома FAT

- загрузочный сектор содержит программу начальной загрузки операционной системы
- основная копия FAT содержит информацию о размещении файлов и каталогов на диске
- резервная копия FAT
- корневой каталог занимает фиксированную область размером в 32 сектора (16 Кбайт), что позволяет хранить 512 записей о файлах и каталогах
- область данных предназначена для размещения всех файлов и всех каталогов, кроме корневого каталога

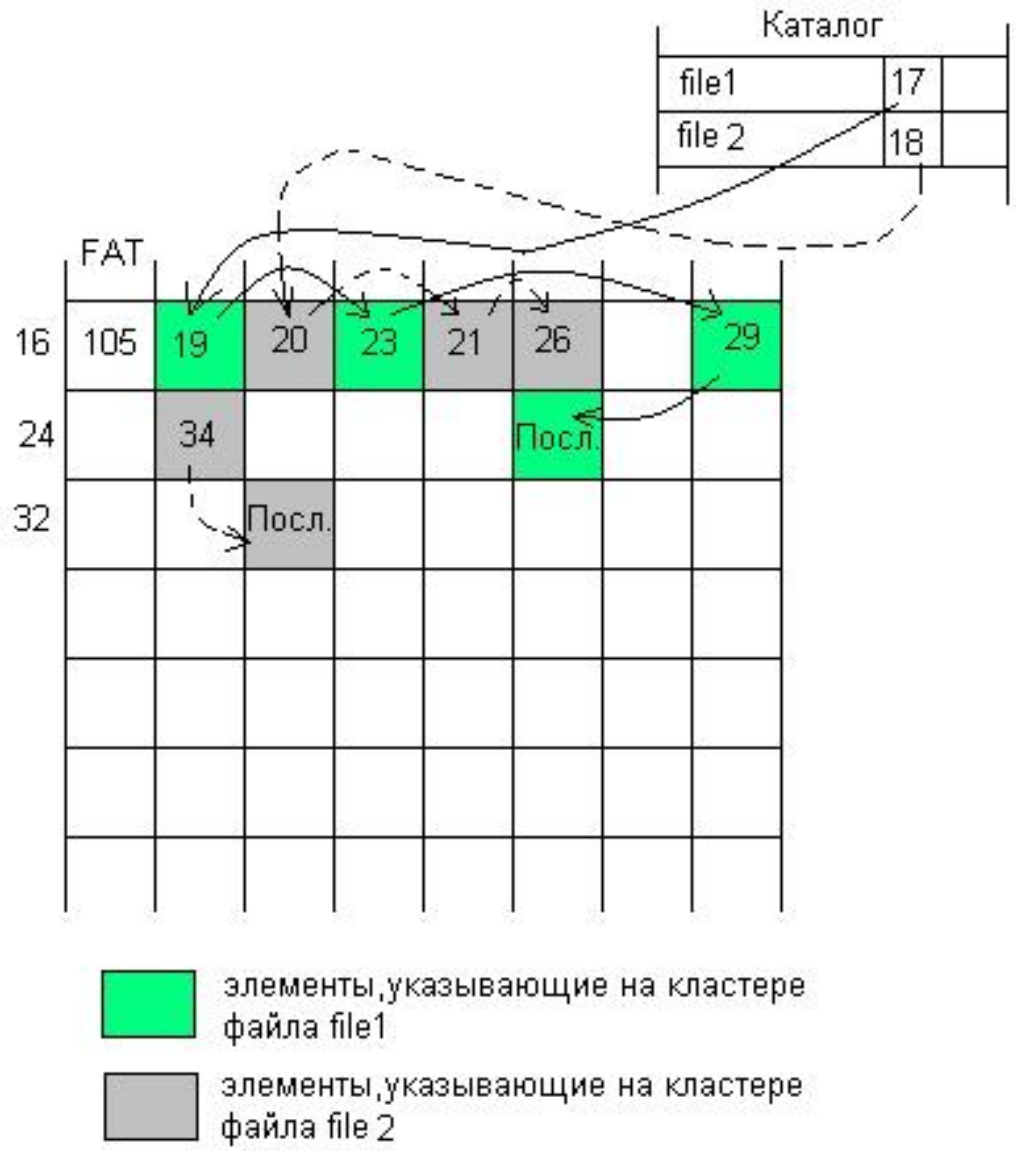
Файловая система FAT

- FAT12 – 4096 кластеров – объем диска не более 16 Мбайт позволяет не использовать кластеры более 4 Кбайт
- FAT16 – 65536 кластеров - объем диска не более 512 Мбайт. Максимальный размер раздела – 4Гбайт (65536 кластеров по 64 Кбайт)
- FAT32 – > 4 миллиардов кластеров – кластеры по 4 Кбайт при работе с дисками объемом до 8 Гбайт, для дисков большего объема – 8, 16 и 32 Кбайт. Максимальный размер раздела –2^{32} кластеров по 32Кбайт)

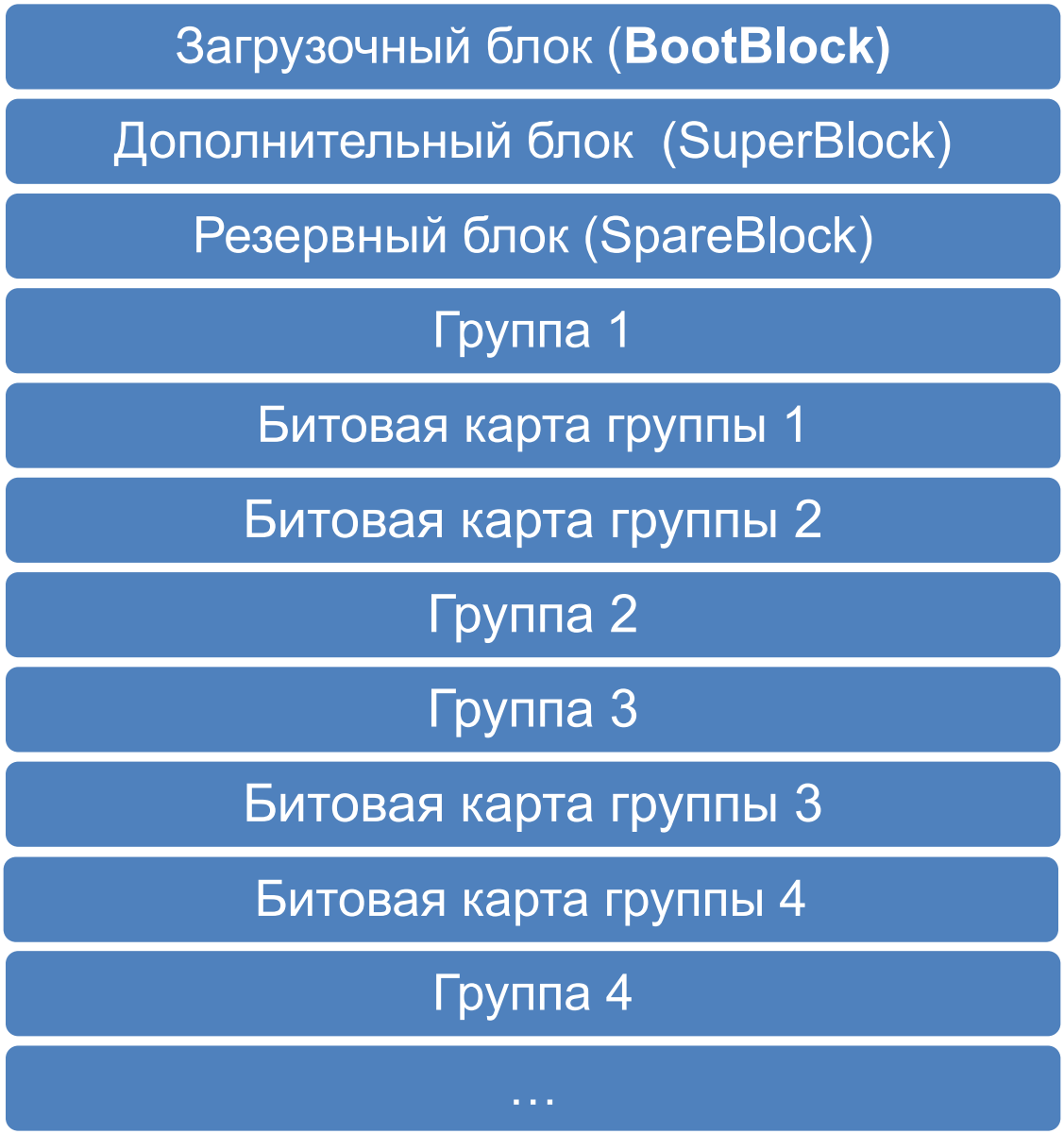
Индексный указатель может принимать следующие значения, характеризующие состояние связанного с ним кластера:

1. • кластер свободен (0000h)
2. • резервный кластер (до 0xFFF6 включительно для FAT16). В этом случае кластер, не соответствуя никаким реальным данным, считается занятым и пропускается при поиске свободного, но никакой другой информации о нем не предоставляется.
3. • последний кластер файла EOC (для FAT12 -число ≥ 0x0FF8, для FAT16 - ≥ 0xFFF8, для FAT32 – ≥ 0x0FFFFFF8)
4. • дефектный кластер (FFF7h)для FAT12 -число ≥ 0x0FF7,для FAT16 - ≥ 0xFFF7, для FAT32 – ≥ 0x0FFFFFF7)
5. • кластер используется файлом и не является последним кластером файла; в этом случае индексный указатель содержит номер следующего кластера файла.

Списки указателей файлов в FAT



Файловая система HPFS



Загрузочный блок Boot Block располагается в секторах с 0 по 15, содержит: имя тома, его серийный номер, блок параметров BIOS, программу начальной загрузки.

Блок Super block содержит:

- указатель на список битовых карт (bitmap block list);
- указатель на список дефектных блоков (bad block list):
- указатель на группу каталогов (directory band);
- указатель на файловый узел (F-node) корневого каталога;
-дату последней проверки раздела программой CHKDSK.

Резервный блок Spare block размещается в 17 секторе диска содержит:

- указатель на карту аварийного замещения (hotfix map или hotfix-areas);
- указатель на список свободных запасных блоков (directory emergency free block list), используемых для операций на почти переполненном диске
- ряд системных флагов и дескрипторов.

Физическая организация NTFS

1. поддержка больших файлов и больших дисков объемом до 2^{64} байт
2. восстанавливаемость после сбоев и отказов программ и аппаратуры управления дисками
3. высокая скорость операций, в том числе и для больших дисков
4. низкий уровень фрагментации, в том числе и для больших дисков
5. гибкая структура
6. устойчивость к отказам дисковых накопителей
7. поддержка длинных символьных имен
8. контроль доступа к каталогам и отдельным файлам

Структура тома NTFS

Номер записи	Системный файл	Имя файла	Назначение файла
0	Главная таблица файлов	$Mft	Содержит полный список файлов тома NTFS
1	Копия главной таблицы файлов	$MftMirr	Зеркальная копия первых трех записей MFT
2	Файл журнала	$LogFile	Список транзакций, который используется для восстановления файловой системы после сбоев
3	Том	$Volume	Имя тома, версия NTFS и другая информация о томе
4	Таблица определения атрибутов	$AttrDef	Таблица имен, номеров и описаний атрибутов
5	Индекс корневого каталога	$.	Корневой каталог
6	Битовая карта кластеров	$Bitmap	Разметка использованных кластеров тома
7	Загрузочный сектор раздела	$Boot	Адрес загрузочного сектора раздела
8	Файл плохих кластеров	$BadClus	Файл, содержащий список всех обнаруженных на томе плохих кластеров
9	Таблица квот	$Quota	Квоты используемого пространства на диске для каждого пользователя
10	Таблица преобразования регистра символов	$Upcase	Используется для преобразования регистра символов для кодировки Unicode
11–15	Зарезервированы для будущего использования		

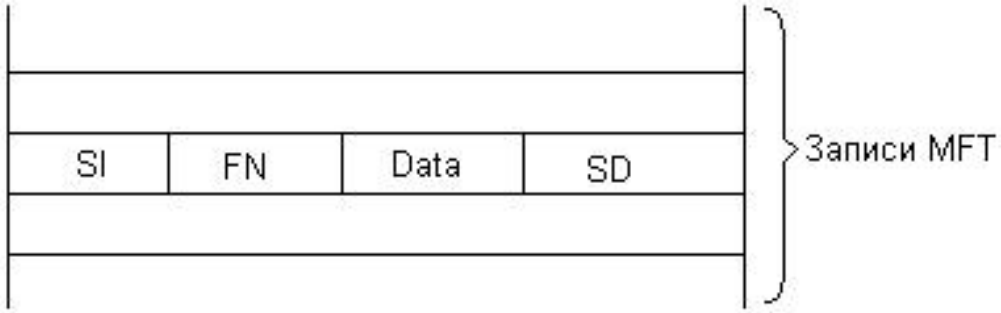
- Основа структуры тома NTFS – главная таблица файлов (MFT), которая содержит хотя бы одну запись для каждого файла тома, включая саму себя.
- Каждая запись MFTимеет фиксированную длину, зависящую от объема диска – 1, 2 или 4 Кбайт.
- Файлы в томе NTFS идентифицируются номером файла, который определяется позицией файла в MFT.
- Весь том NTFS состоит из последовательности кластеров. Порядковый номер кластера в томе NTFS называется логическим номером кластера (LCN). Порядковый номер кластера внутри файла называется виртуальным номером кластера (VCN).
- Единица распределения дискового пространства – отрезок. Адрес отрезка – (LCN, k), логический номер его первого кластера и количество кластеров в отрезке.
- Часть файла, помещенная в отрезок и начинающаяся с виртуального кластера VCN характеризуется адресом (VCN, LCN, k).

Атрибуты файлов

- *Attribute List (список атрибутов)*
- *File Name (имя файла)*
- *MS-DOS Name (имя MS-DOS)*
- *Version (версия)*
- *Security Descriptor (дескриптор безопасности)*
- *Volume Version (версия тома)*
- *Volume Name (имя тома)*
- *Data (данные)*
- *MFT bitmap (битовая карта MFT)*
- *Index Root (корень индекса)*
- *Index Allocation (размещение индекса)*
- *Standard Information (стандартная информация)*

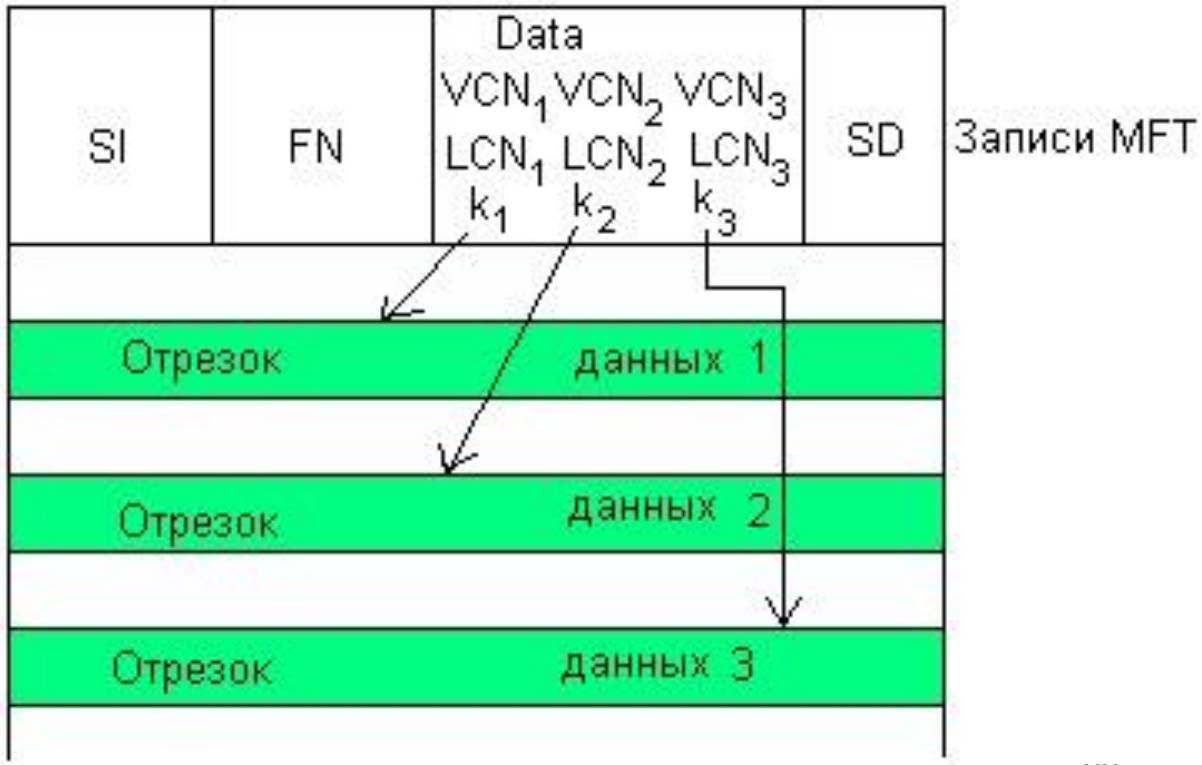
Сведения о владельце файла, биты флагов, временные метки, биты архивирования и т.д.

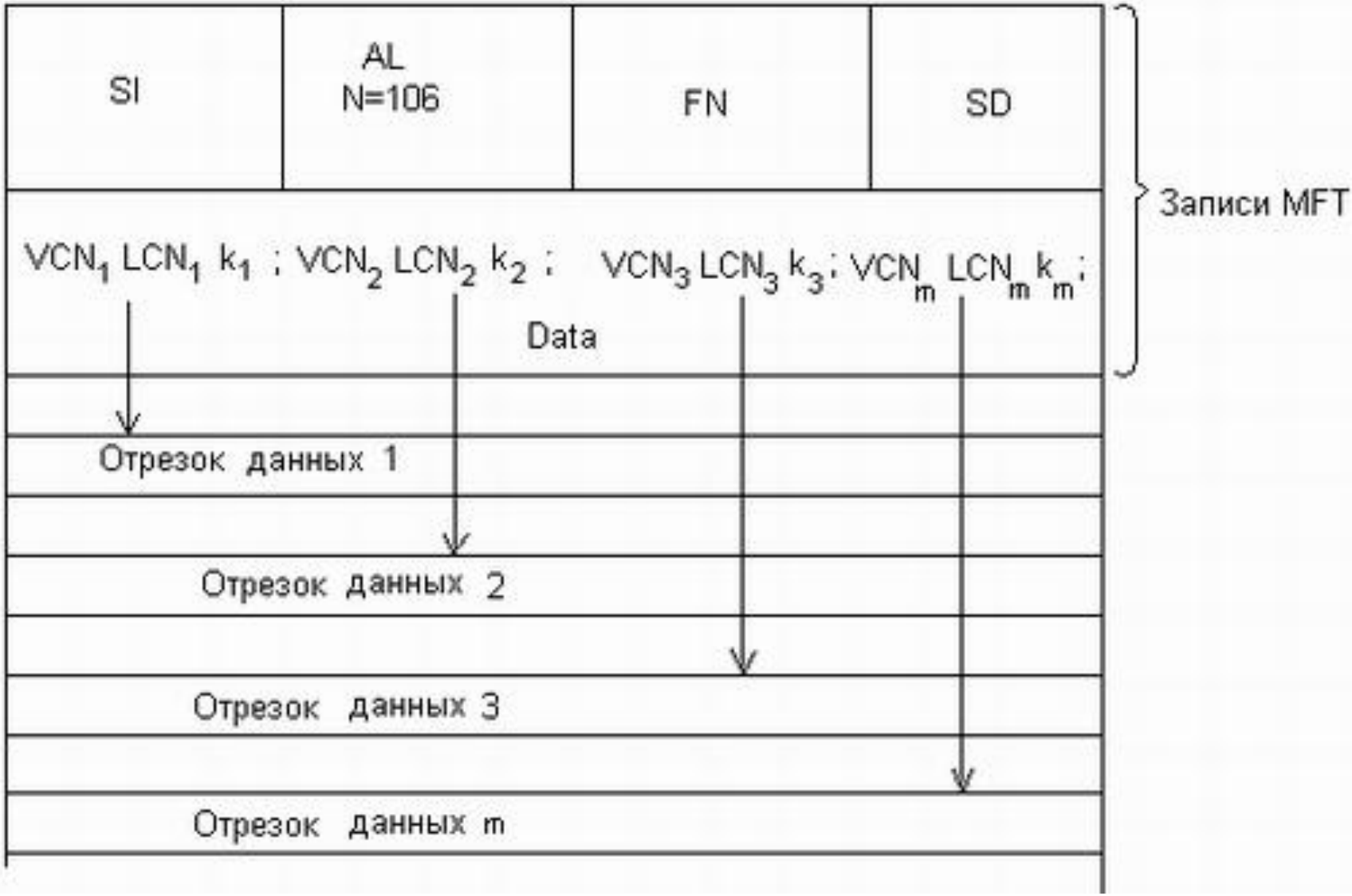
Структура файлов NTFS



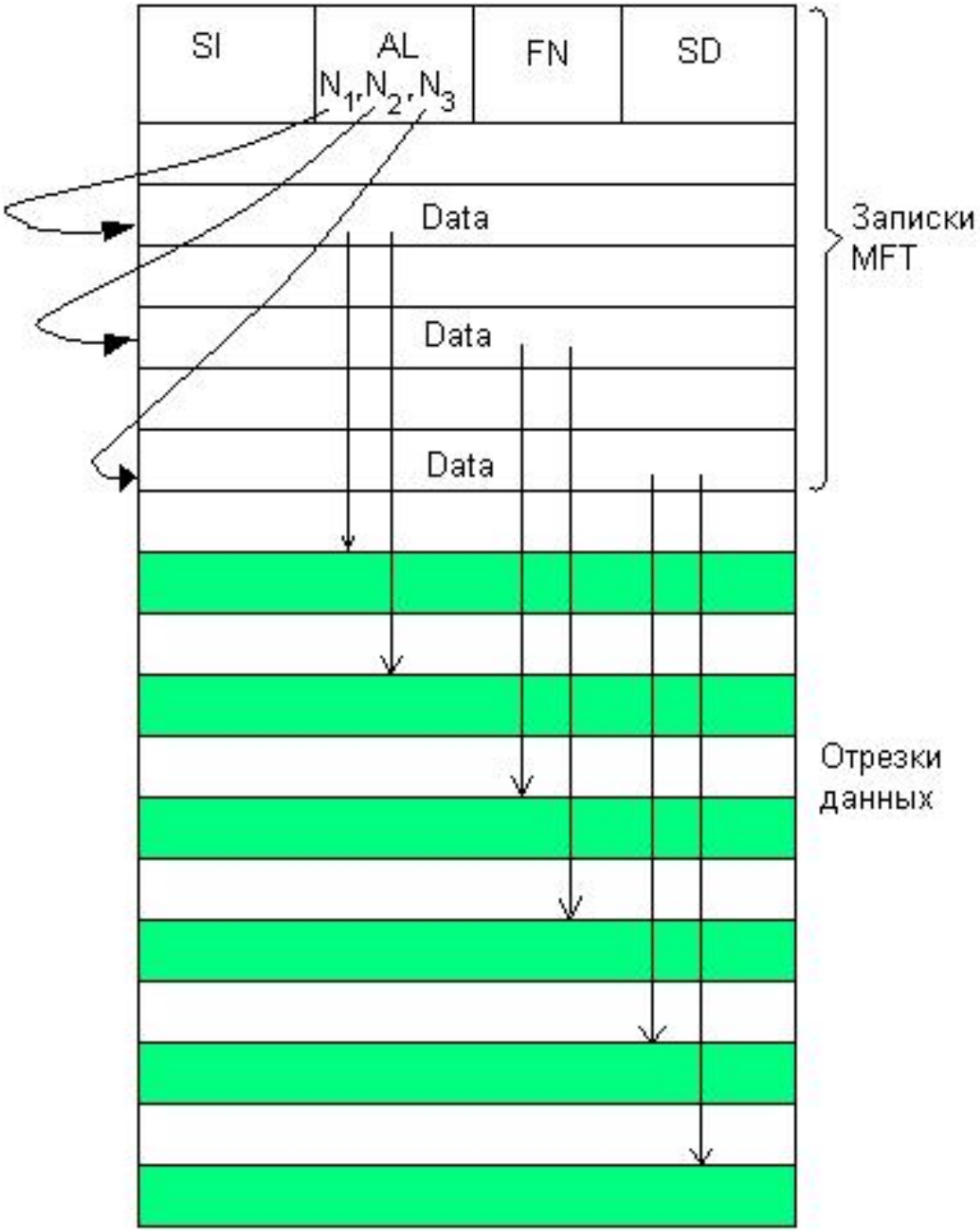
Небольшой файл NTFS

Большой файл





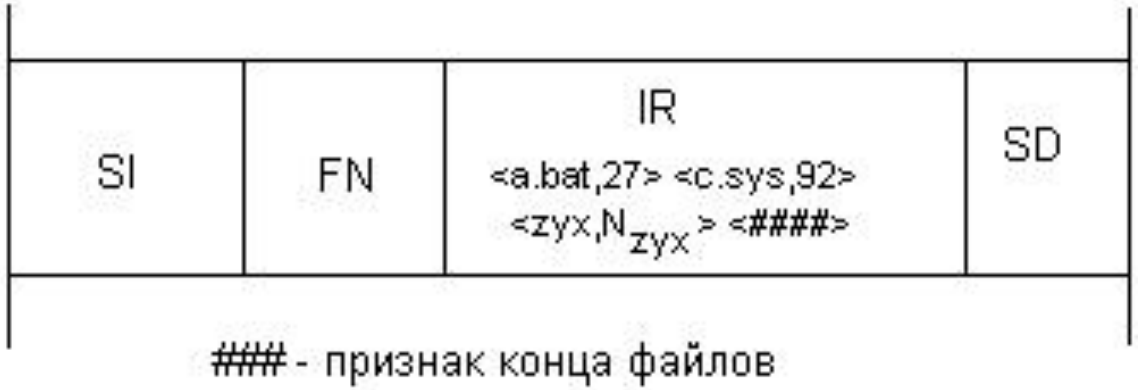
Очень большой файл



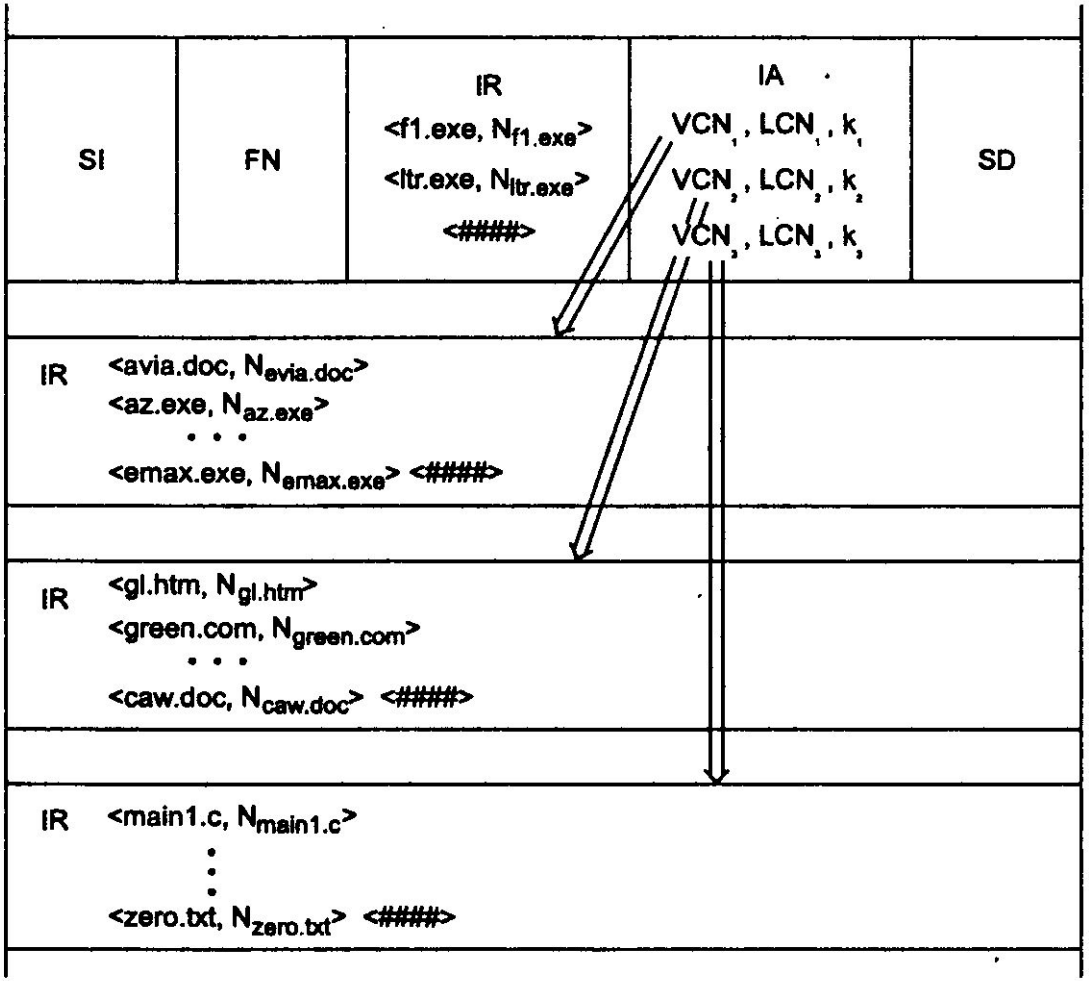
Сверхбольшой файл

Каталоги NTFS

Небольшой каталог



Большой каталог



ReFS (Resilient File System) — отказоустойчивая файловая система

Основные цели создания ReFS:

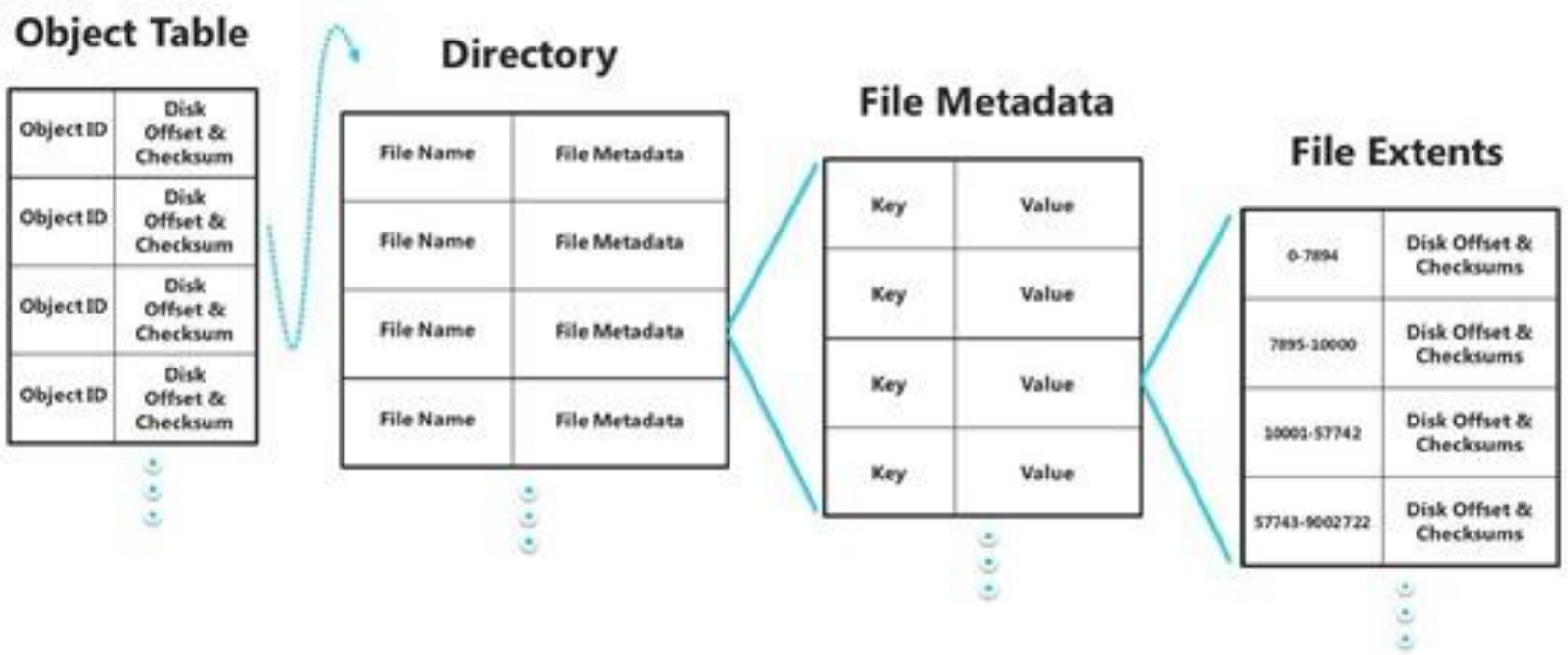
- ✔ Сохранение высокой степени совместимости с подмножеством наиболее востребованных функций NTFS наряду с выводом из употребления прочих, менее полезных.
- ✔ Проверка и автоматическое исправление данных.
- ✔ Оптимизация для экстремальной масштабируемости. Использование масштабируемых структур для всех случаев.
- ✔ Восстановление максимально возможного объема данных без прекращения работы.
- ✔ Обеспечение полной сквозной отказоустойчивой архитектуры.

Практические области применения

Файловый сервер общего назначения. Пользователь разворачивает файловый сервер, подключенный к конфигурации хранилища JBOD с дисками SATA или SAS.

Консолидированное хранилище данных удаленного приложения. Пользователь разворачивает масштабируемый кластер файлового сервера с двумя узлами и дисковыми пространствами, где кластер использует общую конфигурацию хранилища JBOD с дисками SATA или SAS.

Структуры файлов



Сравниваем файловые системы NTFS И ReFS

	NTFS	REFS
МАКСИМАЛЬНЫЙ РАЗМЕР ФАЙЛА	16 Тбайт	18,3 Эбайт
МАКСИМАЛЬНЫЙ РАЗМЕР ТОМА	18,4 Эбайт	402 Эбайт
МАКСИМАЛЬНОЕ ЧИСЛО ФАЙЛОВ В ПАПКЕ (ПРИБЛИЗИТЕЛЬНО)	4,3 млрд	18 трлн
МАКСИМАЛЬНОЕ КОЛИЧЕСТВО СИМВОЛОВ В ИМЕНИ ФАЙЛА	255	32 767
МАКСИМАЛЬНОЕ КОЛИЧЕСТВО СИМВОЛОВ В ИМЕНИ ПУТИ	255	32 767

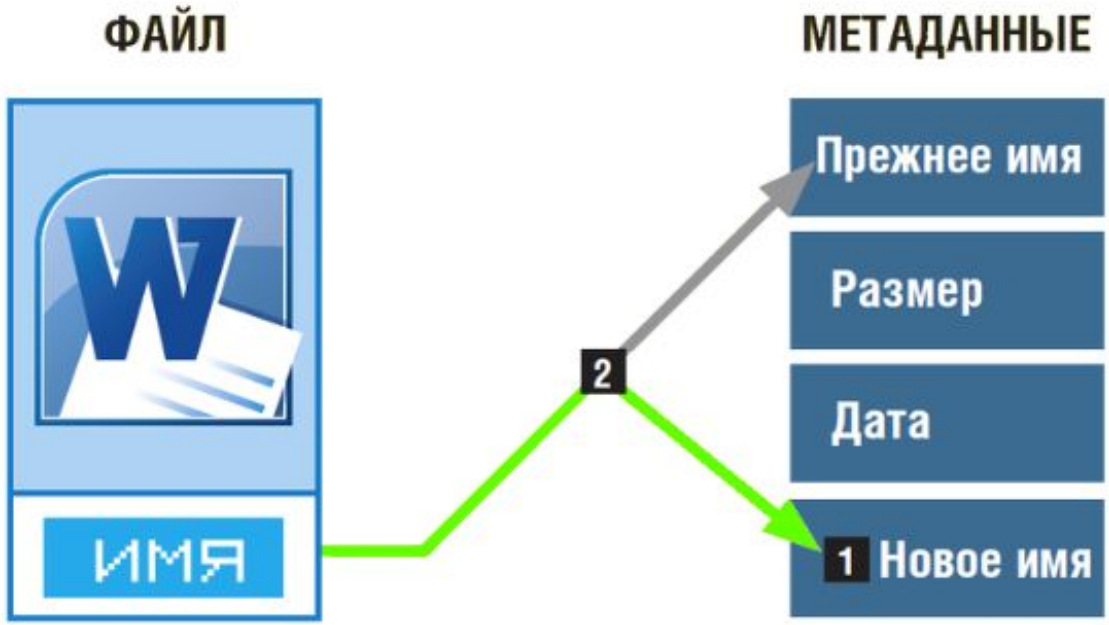
Случай 1. Стандартная задача - переименовать файл

1. NTFS записывает в Журнал, что имя файла должно быть изменено. Там же NTFS регистрирует все действия.

2. Только после этого она на месте меняет имя файла. Таким образом, старое имя переписывается новым.

3. В заключение в Журнале (файле регистрации файловой системы) появляется отметка об успешном завершении заданной операции.

ReFS



1 Новое название записывается в свободное место. При этом очень важно, что прежнее имя поначалу не стирается.

2 Как только новое название записано, ReFS изменяет ссылку на поле имени. Теперь в файловой системе она ведет не на старое имя, а на новое.

Случай 2. Переименование файла при отказе питания

1. NTFS, как обычно, записывает запрос на изменение в Журнал.

2. После этого из-за отказа питания процесс переименования прерывается, и не остается записи ни о прежнем, ни о новом именах.

3. Происходит перезагрузка Windows.

4. Вслед за этим запускается программа для исправления ошибок — Chkdisk.

5. Только теперь с помощью Журнала при применении отката восстанавливается изначальное имя файла.

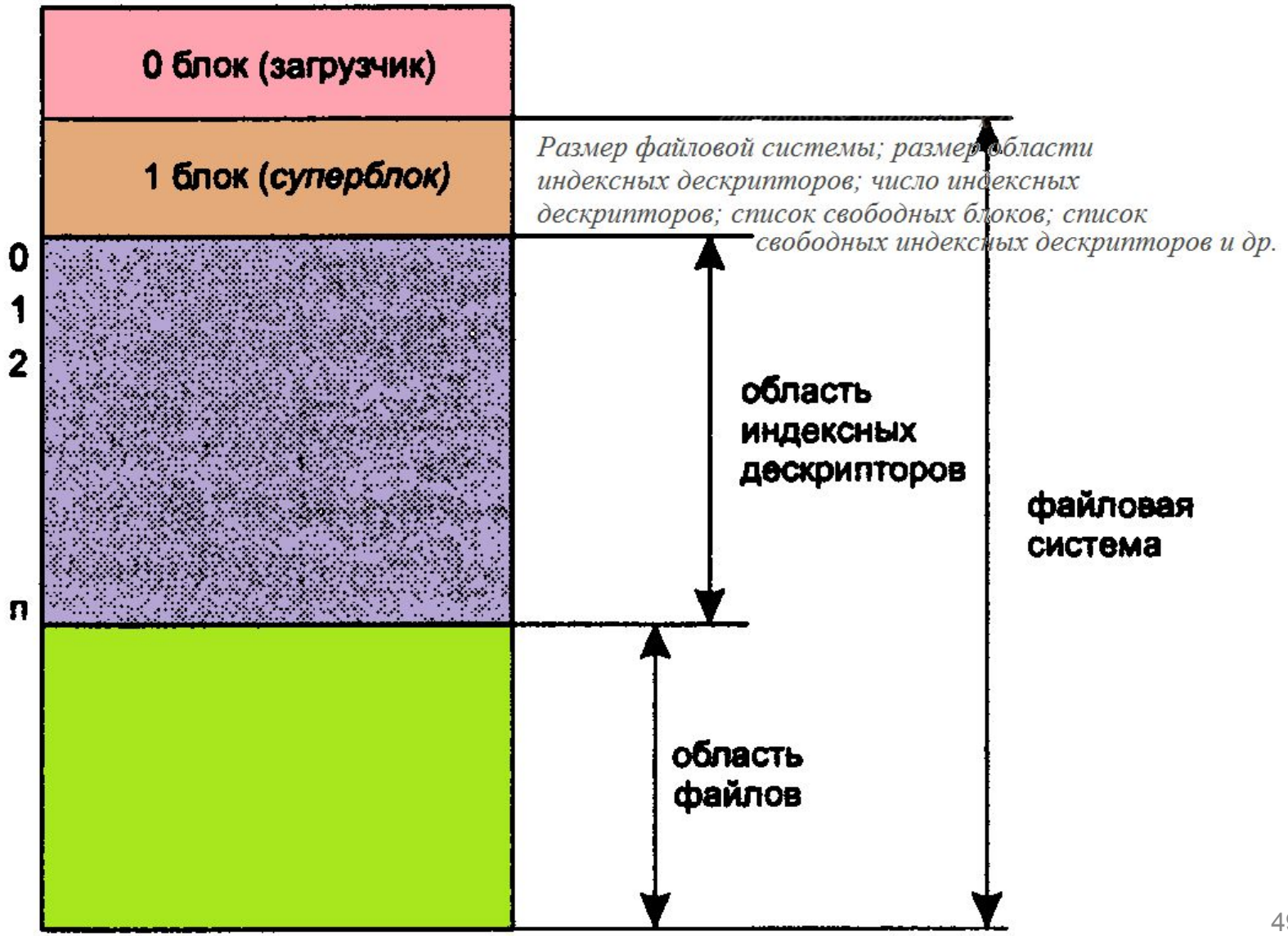
ReFS

1. На первом этапе ReFS записывает новое имя в другом месте файловой системы, однако в этот момент электропитание прекращается.

2. Отказ приводит к автоматической перезагрузке Windows.

3. После нее стартует программа Chkdisk. Она анализирует файловую систему на наличие ошибок и при необходимости исправляет их. Между тем набор данных ReFS находится в стабильном состоянии. Прежнее имя файла снова становится действующим сразу после отказа питания.

Файловые системы S5 и UFS



Индексный дескриптор (64 байта) содержит:

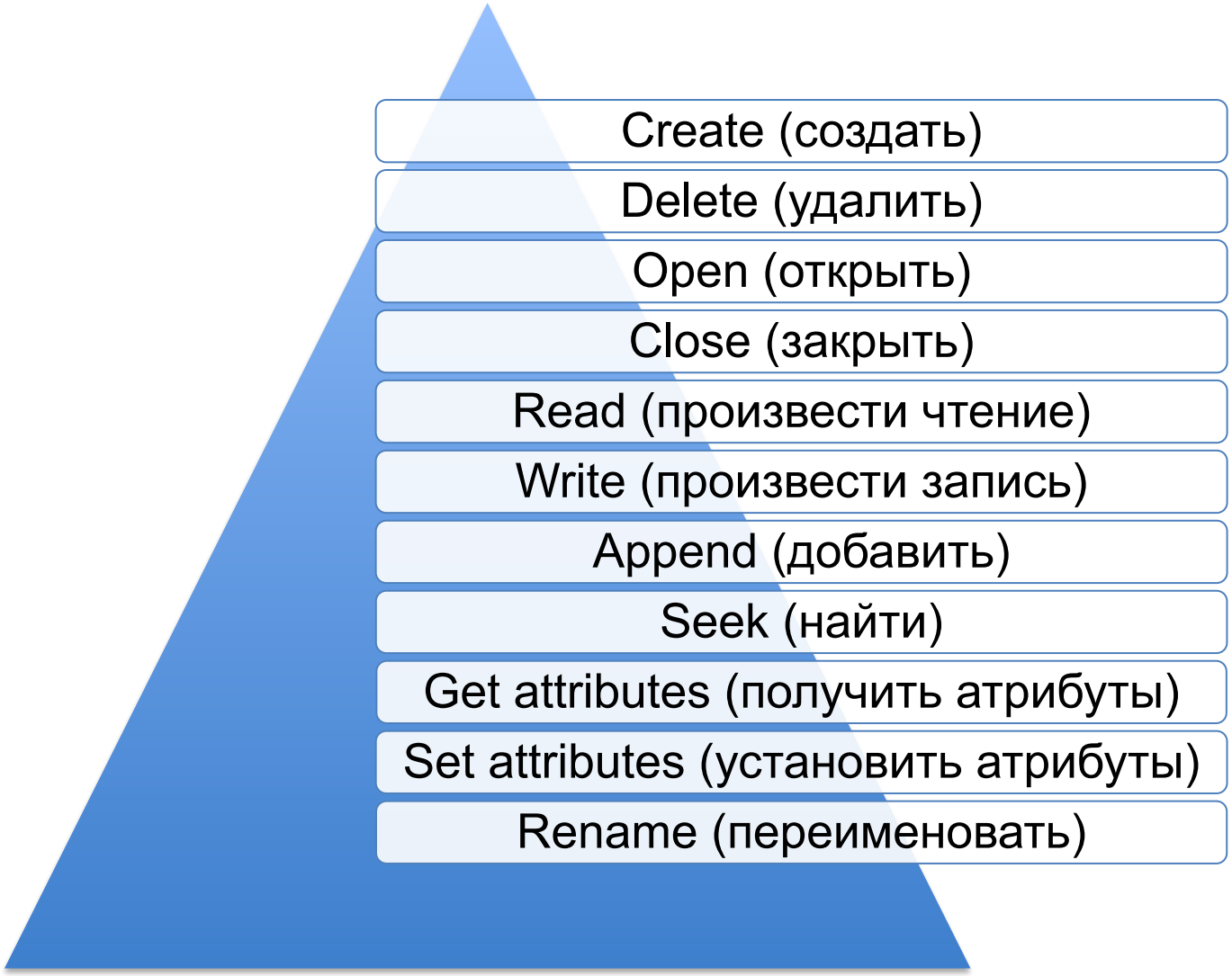
- идентификатор владельца файла;
- тип файла;
- права доступа к файлу;
- временные характеристики: время последней модификации файла, время последнего обращения к файлу, время последней модификации индексного дескриптора;
- число ссылок на данный индексный дескриптор;
- адресная информация;
- размер файла в байтах.

Физическая организация UFS

Unix File System (UFS) — файловая система, созданная для операционных систем семейства BSD и используемая в переработанном и дополненном виде на данный момент как основная в операционных системах-потомках (FreeBSD, OpenBSD, NetBSD).
Поддержка данной файловой системы имеется также в ядре Linux и операционной системе Solaris.

Размещение файловой системы ext2 на диске

Файловые операции

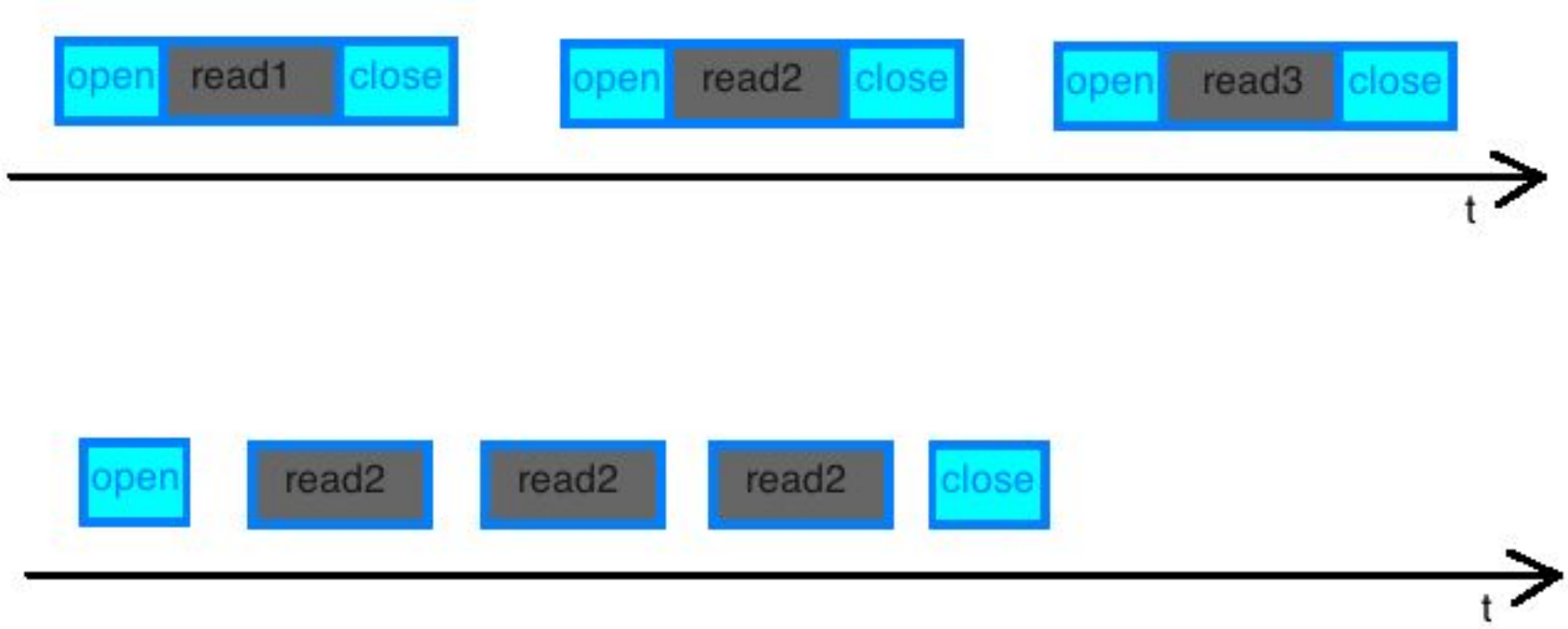



Операция с файлом
Универсальные действия
Поиск характеристик файла на диске по символьному имени
Копирование характеристик файла в ОП
Проверка прав доступа к файлу
Освобождение системных областей при закрытии файла
Уникальные действия
Удаление файла
…
Чтение определенного набора кластеров

конец файла. Как правило, у систем, предоставляющих минимальный набор системных вызовов, вызов append отсутствует, но многие системы предоставляют множество способов получения того же результата, и иногда в этих системах присутствует вызов append.

8. Seek (Найти). При работе с файлами произвольного доступа нужен способ указания места, с которого берутся данные. Одним из общепринятых подходов является применение системного вызова seek, который перемещает указатель файла к считываться или записываться с этой позиции.
9. Get attributes (Получить атрибуты). Процессу для своей работы зачастую необходимо считать атрибуты файла. К примеру, имеющаяся в UNIX программа make обычно используется для управления проектами разработки программного обеспечения, состоящими из множества сходных файлов. При вызове программы make она проверяет время внесения последних изменений всех исходных и объектных файлов и для обновления проекта обходится компиляцией лишьминимально необходимого количества файлов. Для этого ей необходимо просмотреть атрибуты файлов, а именно время внесения последних изменений.
10. Set attributes (Установить атрибуты). Значения некоторых атрибутов могут устанавливаться пользователем и изменяться после того, как файл был создан. Такую возможность дает именно этот системный вызов. Характерным примеромможет

Два способа выполнения файловых операций



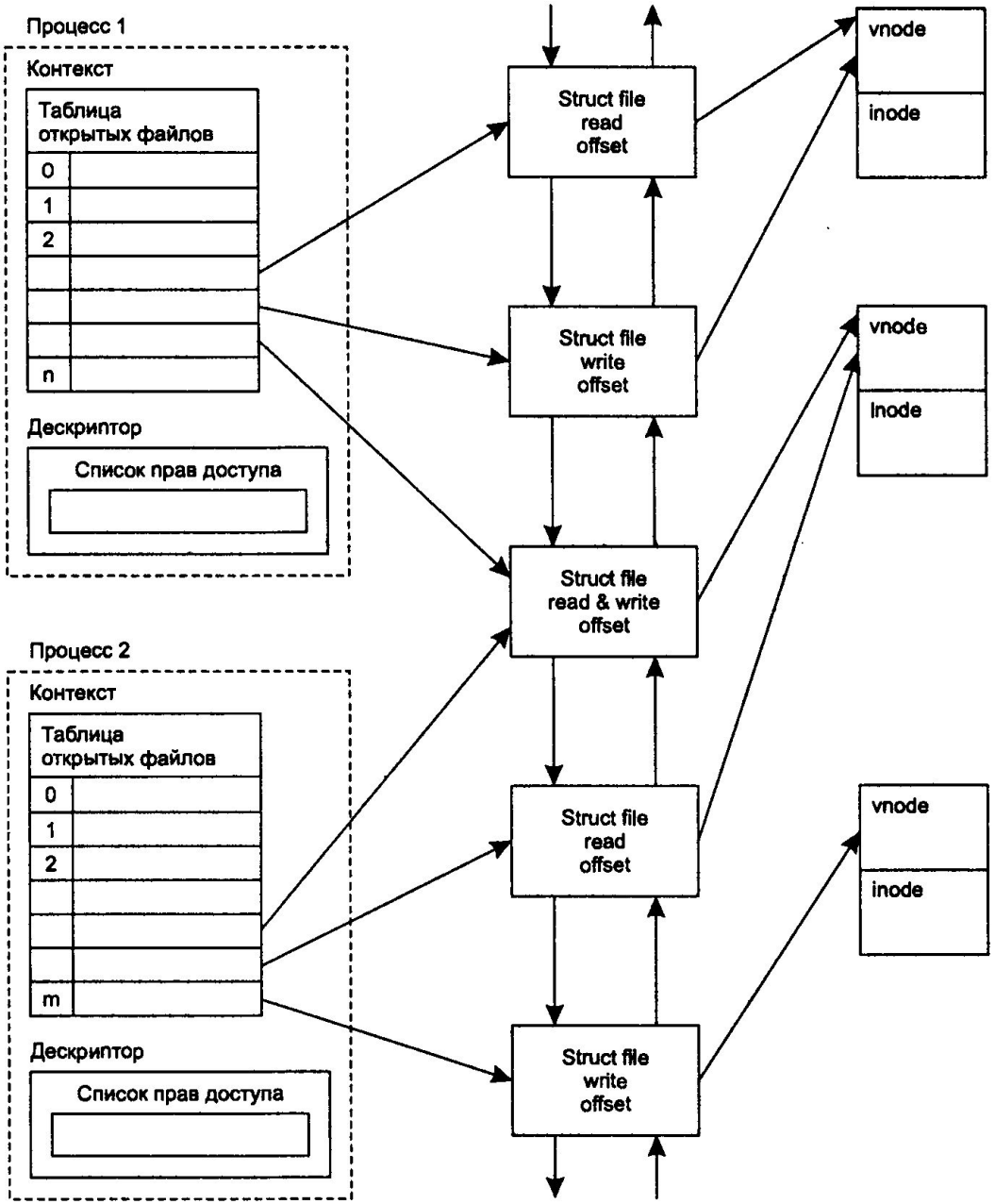
Процедура открытия файла в UNIX

vnode:

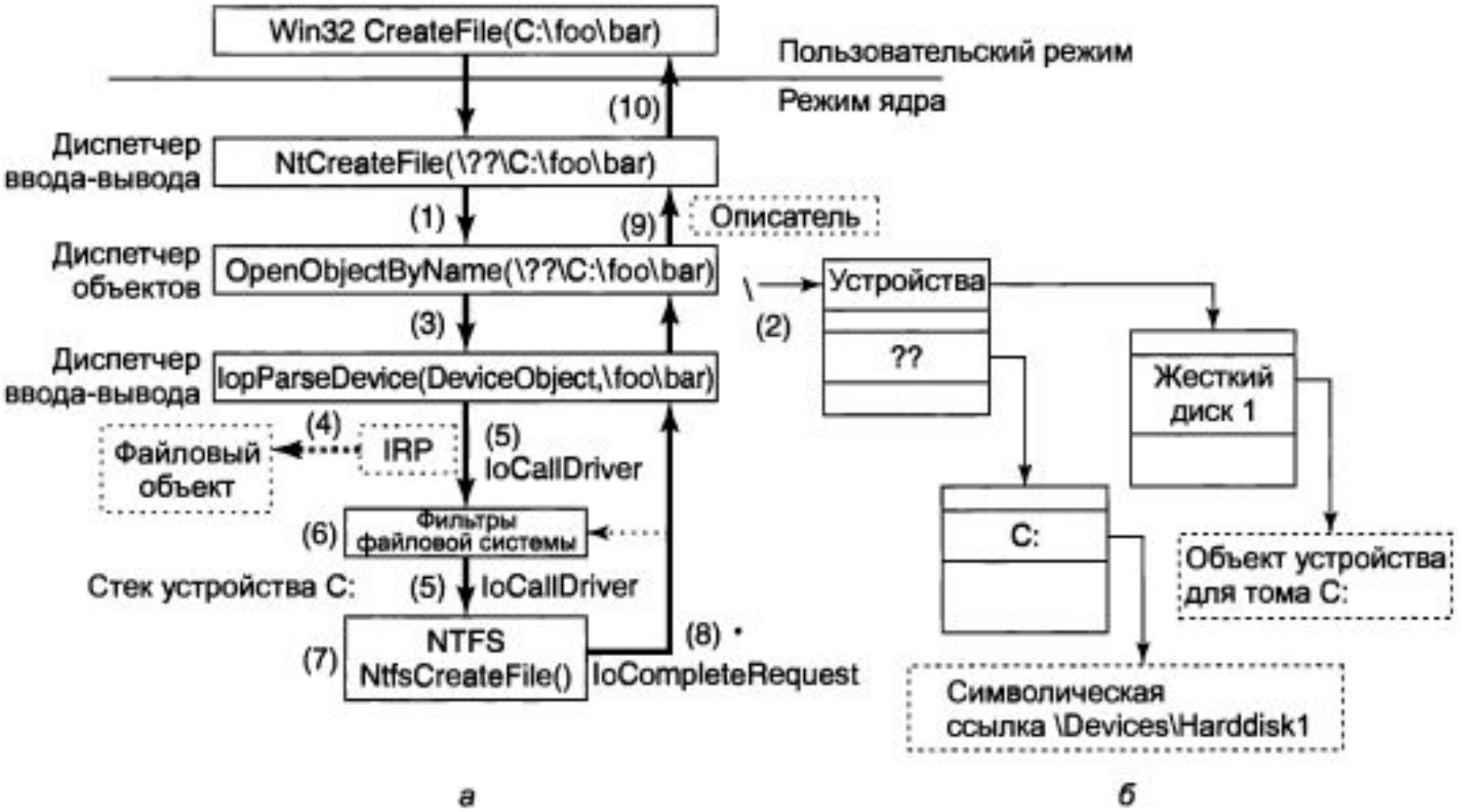
- Состояние индексного дескриптора в памяти.
- Логический номер устройства файловой системы, содержащей файл.
- Номер индексного дескриптора.
- Счетчик ссылок на данную структуру vnode.

Структура file

- признак режима открытия
- указатель на структуру vnode;
- текущее смещение в файле (переменная offset) при операциях чтения/записи;
- счетчик ссылок на данную структуру;
- указатель на структуру, содержащую права процесса, открывшего файл
- указатели на предыдущую и последующую структуры file,

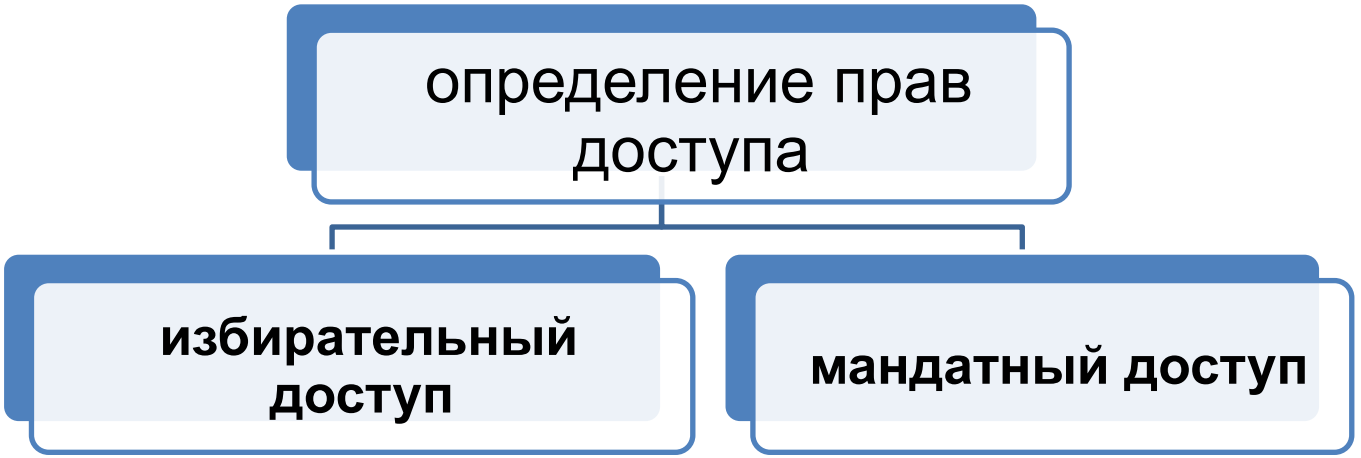

Процесс 1
Контекст
Таблица открытых файлов
0
1
2
n
Дескриптор
Список прав доступа
Процесс 2
Контекст
Таблица открытых файлов
0
1
2
m
Дескриптор
Список прав доступа
Struct file read offset
Struct file write offset
Struct file read & write offset
Struct file read offset
Struct file write offset
vnode
inode
vnode
inode
vnode
inode

Создание или открытие файла в Windows NT



Контроль доступа к файлам

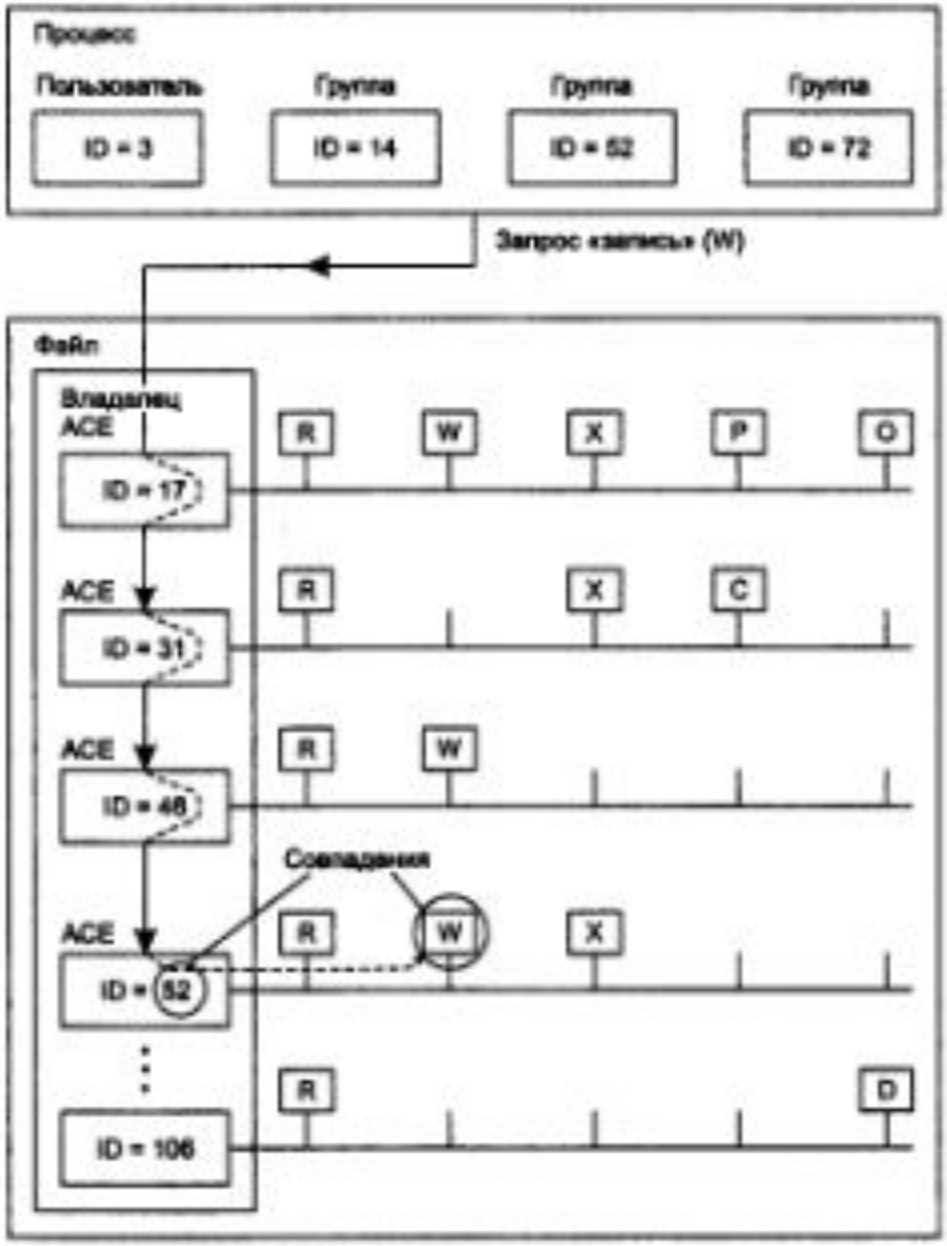
Субъекты доступа	• Пользователи, группы пользователей
Объекты доступа	• Разделяемые ресурсы (файлы, каталоги, устройства, секции памяти, объекты синхронизации и др.)



Матрица прав доступа

Имена пользователей	Имена файлов			
	modern.txt	win.exe	class.dbf	unix.ppt
kira	читать	выполнять	—	выполнять
genua	читать	выполнять	—	выполнять читать
nataly	читать	—	—	выполнять читать
victor	читать писать	—	создать	—

ACL (ACE, ACE, …)


Процесс
Пользователь
ID = 3
Группа
ID = 14
Группа
ID = 52
Группа
ID = 72
Запрос «запись» (W)
Файл
Владелец
ACE
ID = 17
R
W
X
P
O
ACE
ID = 31
R
X
C
ACE
ID = 48
R
W
Совпадения
ACE
ID = 52
R
W
X
R
D
ID = 106

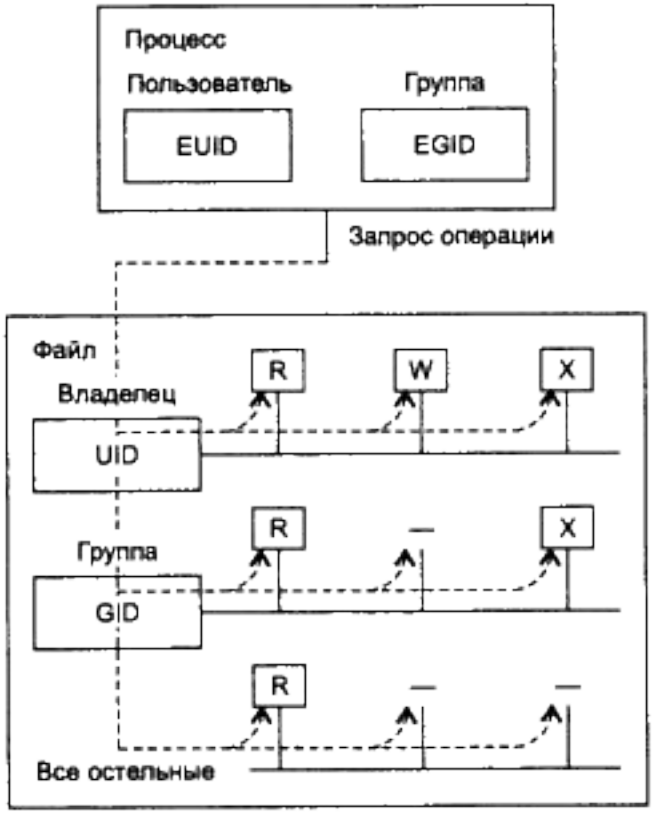
Организация контроля доступа в Unix

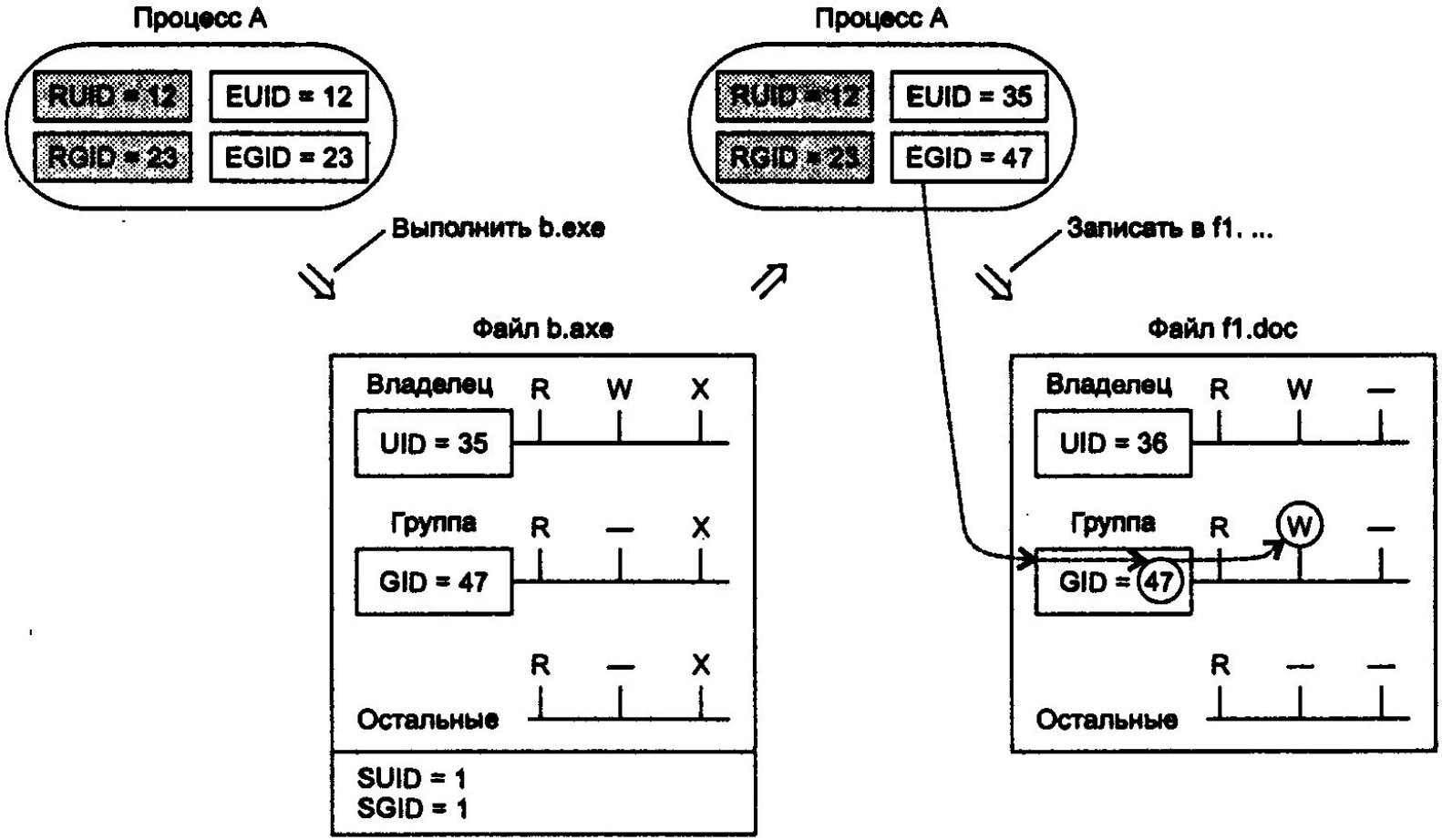
Права доступа к файлу или каталогу определяются для трех субъектов:

- владельца файла (идентификатор User ID, UID);
- членов группы, к которой принадлежит владелец (Group ID, GID);
- всех остальных пользователей системы.

Определены три операции над файлами и каталогами:

чтение, запись, выполнение

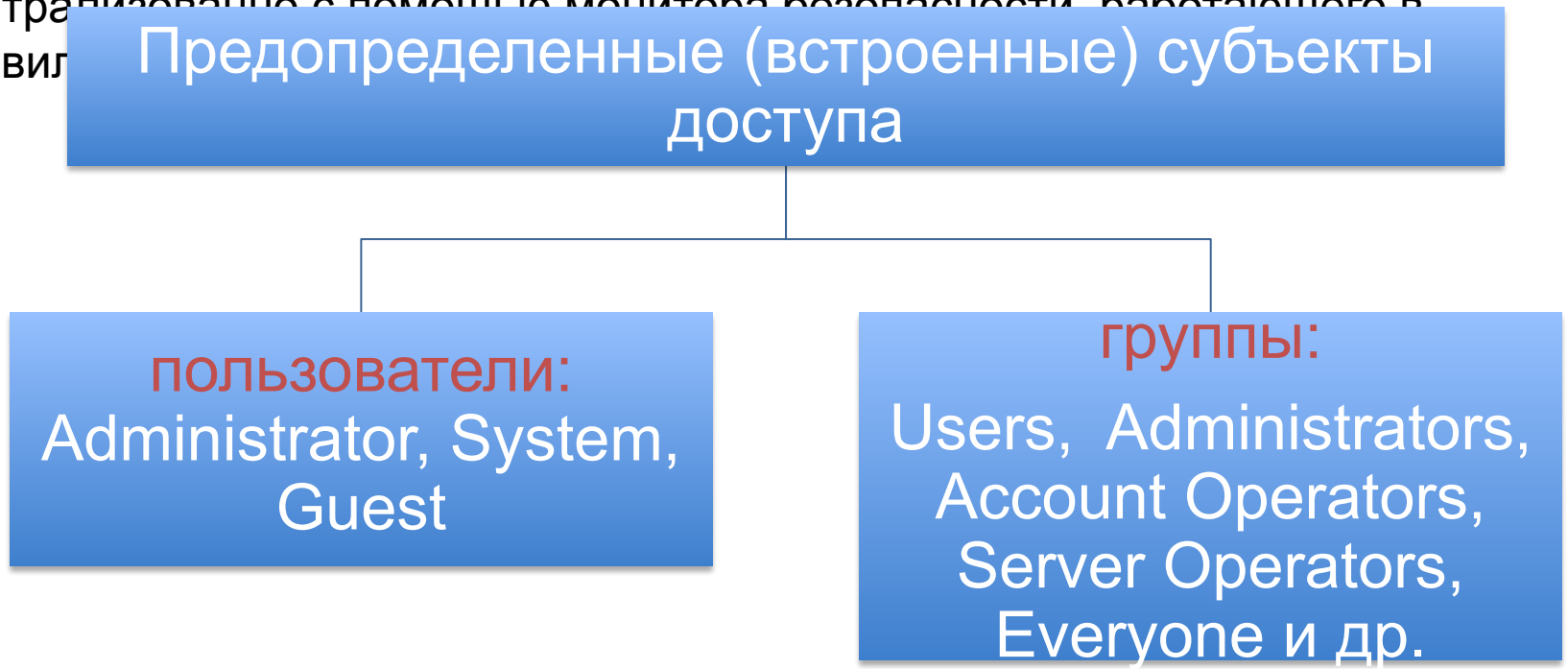



Процесс A
RUID = 12
EUID = 12
RGID = 23
EGID = 23
Выполнить b.exe
Файл b.axe
Владелец
R
W
X
UID = 35
Группа
GID = 47
Остальные
SUID = 1
SGID = 1
Процесс A
RUID = 12
EUID = 35
RGID = 23
EGID = 47
Записать в f1. ...
Файл f1.doc
Владелец
UID = 36
Группа
GID = 47
Остальные

Организация контроля доступа в ОС Windows NT

- Для разделяемых ресурсов в Windows NT применяется общая модель объекта, которая содержит такие характеристики безопасности, как набор допустимых операций, идентефикатор владельца, список управления доступом.

- Проверка прав доступа для объектов любого типа выполняется централизованно с помощью монитора безопасности, работающего в привил



Классы операций доступа в Windows NT

Разрешения – множество операций, которые могут быть определены для субъектов всех типов по отношению к объектам любого типа.

- **Индивидуальные**, относятся к элементарным операциям над каталогами ифайлами
- **Стандартные**, объединение нескольких индивидуальных разрешений

Права – определяются для субъектов типа группа на выполнение некоторых системных операций. В этих операциях участвует особый объект доступа – операционная система в целом. Именно права отличают одну встроенную группу пользователей от другой.

Возможности пользователей – определяются для отдельных пользователей на выполнение действий, связанных с формированием из операционной среды

Индивидуальные разрешения

Разрешение	Для каталога	Для файла
Read (R)	Чтение имен файлов и каталогов, входящих в данный каталог, а также атрибутов и владельца каталога	Чтение данных, атрибутов, имени владельца и разрешений файла
Write (W)	Добавление файлов и каталогов, изменение атрибутов каталога, чтение владельца и разрешений каталога	Чтение владельца и разрешений файла, изменение атрибутов файла, изменение и добавление данных файла
Execute (X)	Чтение атрибутов каталога, выполнение изменений в каталогах, входящих в данный каталог, чтение имени владельца и разрешений каталога	Чтение атрибутов файла, имени владельца и разрешений. Выполнение файла, если он хранит код программы
Delete (D)	Удаление каталога	Удаление файла
Change Permission (P)	Изменение разрешений каталога	Изменение разрешений файла
Take Ownership (O)	Стать владельцем каталога	Стать владельцем файла

Стандартные разрешения

Стандартное разрешение	Индивидуальные разрешения
No Access	Ни одного
Read	RX
Change	RWXD
Full Control	Все

Стандартные разрешения	Индивидуальные разрешения для каталога	Индивидуальные разрешения для файлов каталога при наследоаании
No Access	Ни одного	Ни одного
List	RX	Не определены
Read	RX	RX
Add	WX	Не определены
Add & Read	RWX	RX
Change	RWXD	RWXD
Full Control	Все	Все

Таблица изменяемых прав встроенных групп

Права	Administrators	Server Operators	Account Operators	Print Operators	Backup Operators	Everyone	Users	Guests
Log on locally (локальный логический вход)	Есть	Есть	Есть	Есть	Есть	Нет	Нет	Нет
Access this computer from network (доступ к данному компьютеру через сеть)	Есть	Нет	Нет	Нет	Нет	Есть	Нет	Нет
Take ownership of files (установление прав собственности на файлы)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Manage auditing and security log (управление аудитом и учетом событий, связанных с безопасностью)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Change the system time (изменение системного времени)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Shutdown the system (останов системы)	Есть	Есть	Нет	Нет	Есть	Нет	Нет	Нет
Force shutdown from remote system (инициирование останова с удаленной системы)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Backup files and directories (резервное копирование файлов и каталогов)	Есть	Есть	Есть	Есть	Есть	Нет	Нет	Нет
Restore files and directories (восстановление файлов и каталогов со стримера)	Есть	Есть	Нет	Нет	Есть	Нет	Нет	Нет
Load and unload device drivers (загрузка и выгрузка драйверов устройств)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Add workstation to domain (добавление рабочих станций к домену)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет

Встроенные права встроенных групп

Встроенные права	Administrators	Server Operators	Account Operators	Print Operators	Backup Operators	Everyone	Users	Guests
Create and manage user accounts (создание и управление пользовательской учетной информацией)	Есть	Нет	Есть	Нет	Нет	Нет	Нет	Нет
Create and manage global groups (создание и управление глобальными группами)	Есть	Нет	Есть	Нет	Нет	Нет	Нет	Нет
Create and manage local groups (создание и управление локальными группами)	Есть	Нет	Есть	Нет	Нет	Нет	Нет	Нет
Assign user rights (назначение прав для пользователей)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Manage auditing of system events (управление аудитом системных событий)	Есть	Нет	Нет	Нет	Нет	Нет	Нет	Нет
Lock the server (блокирование сервера)	Есть	Есть	Нет	Нет	Нет	Есть	Нет	Нет
Override the lock of the server (преодоление блокировки сервера)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Format server's hard disk (форматирование жесткого диска сервера)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Create common groups (создание общих групп)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Keep local profile (хранение локального профиля)	Есть	Есть	Есть	Есть	Есть	Нет	Нет	Нет
Share and stop sharing directories (разделение и прекращение разделения каталогов)	Есть	Есть	Нет	Нет	Нет	Нет	Нет	Нет
Share and stop sharing printers (разделение и прекращение разделения принтеров)	Есть	Есть	Нет	Есть	Нет	Нет	Нет	Нет

Отказоустойчивость файловых и дисковых систем

1

- защита от сбоев и отказов аппаратуры

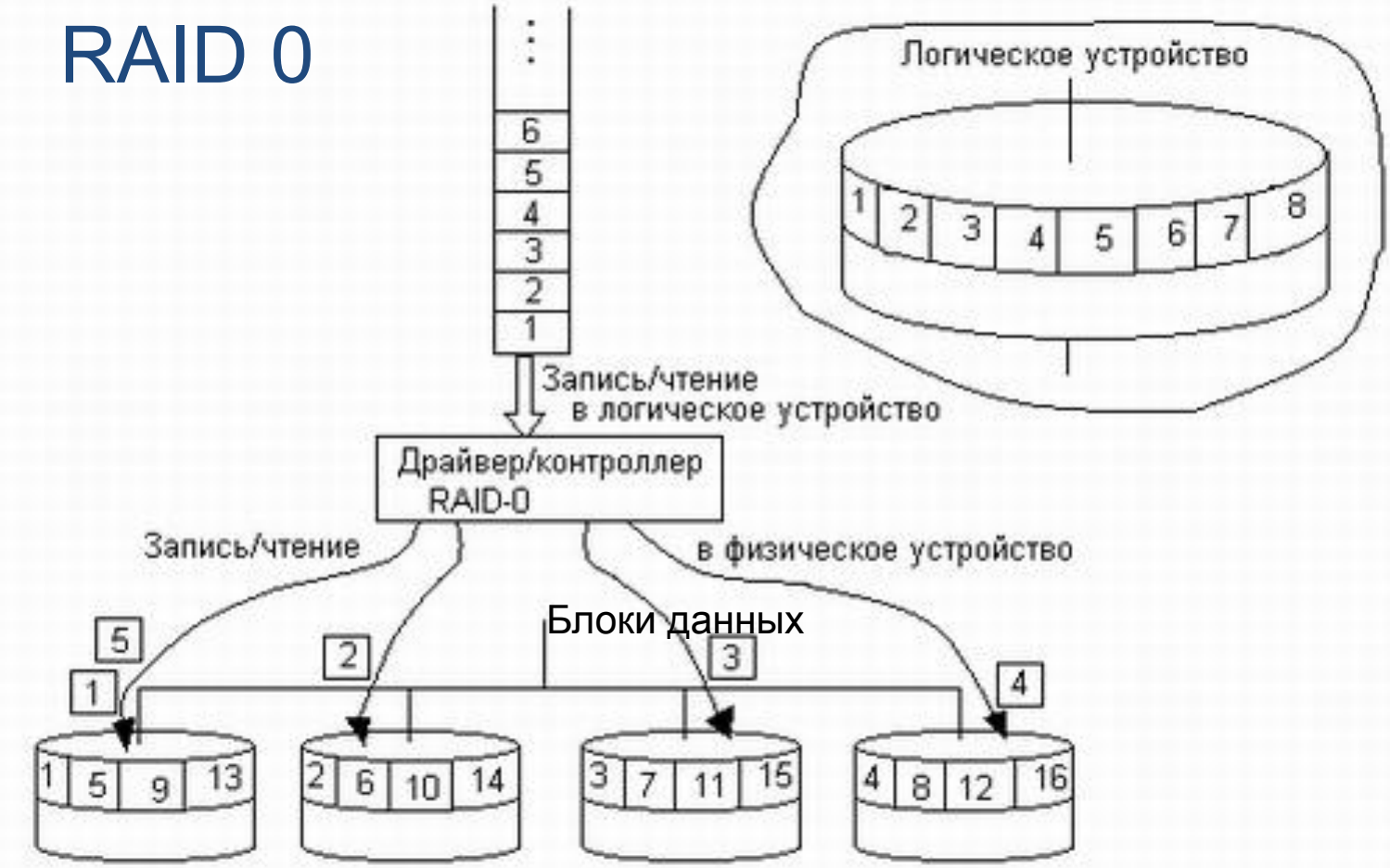
2

- защита от ошибок программного обеспечения

Избыточные дисковые подсистемы RAID (Redundant Array of Inexpensive / Independent Disks, избыточный массив недорогих/независимых дисков)

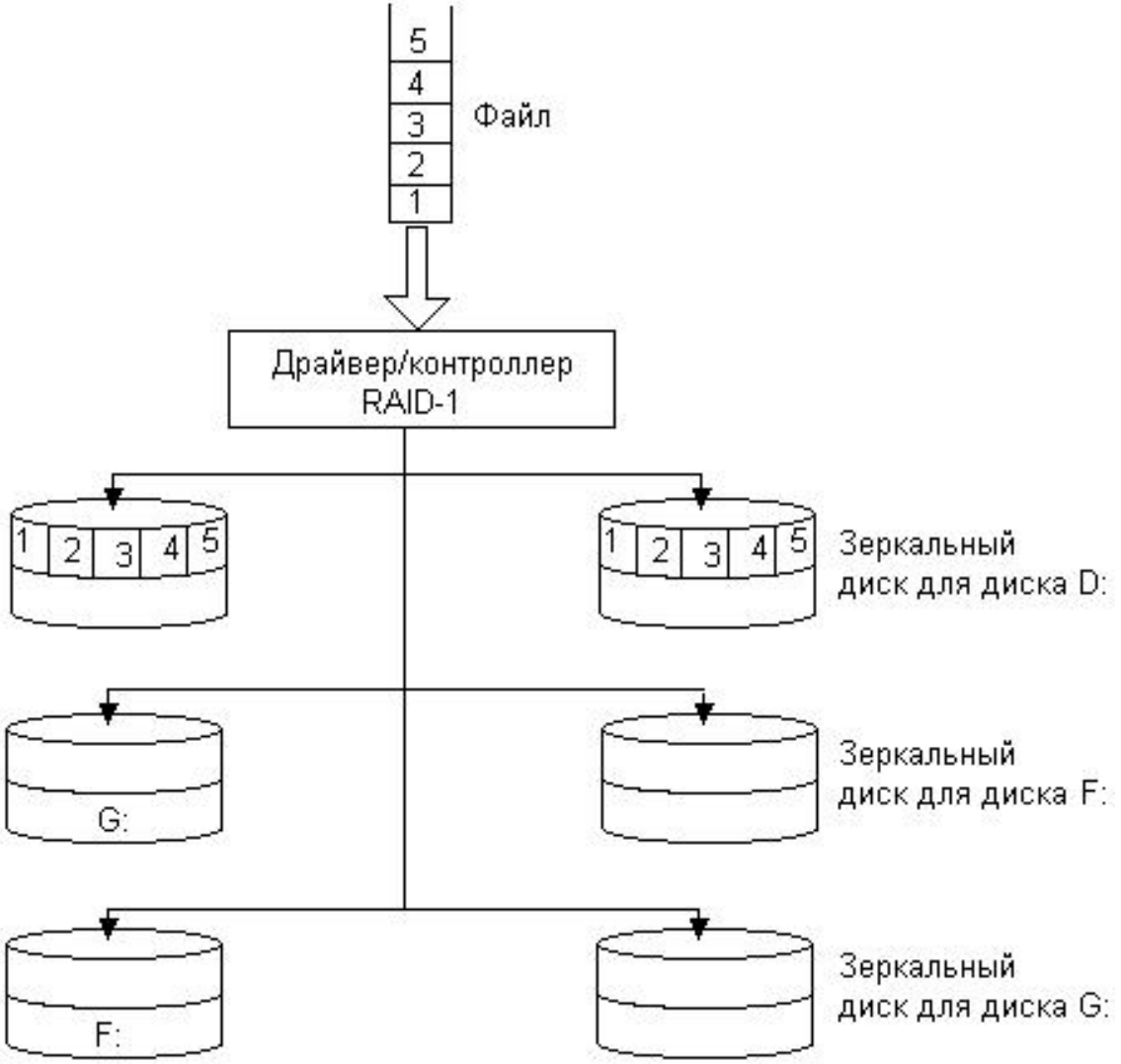
- степень избыточности хранимой информации
- производительность операций чтения и записи
- степень отказоустойчивости

RAID 0



Конфигурация RAID	Избыточность	Отказоустойчивость	Скорость чтения	Скорость записи
RAID 0	Нет	Нет	Повышенная	Повышенная

RAID 1

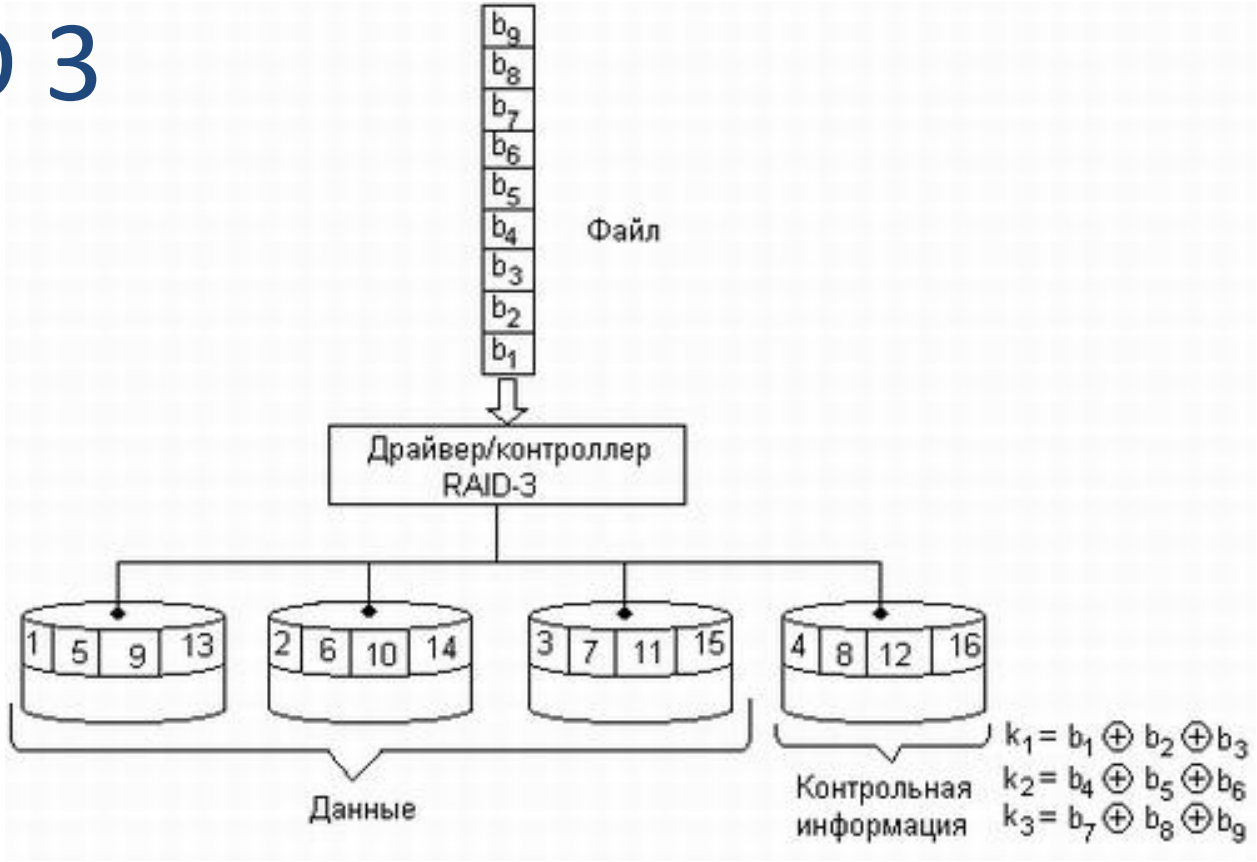


Конфигурация RAID	Избыточность	Отказоустойчивость	Скорость чтения	Скорость записи
RAID 1	50%	Есть	Повышенная	Повышенная

RAID 2

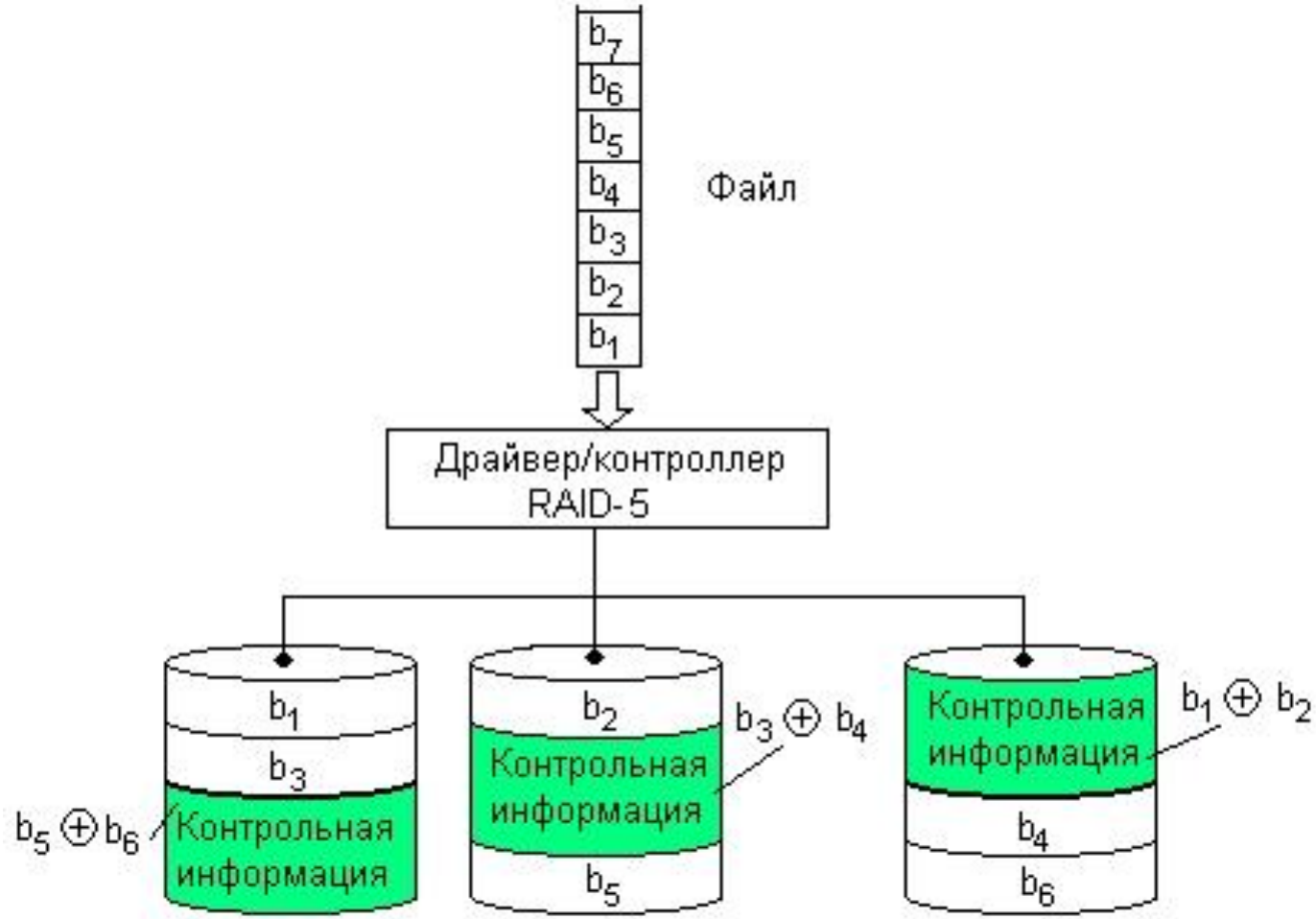
Расщепляет данные побитно – первый бит записывается на первый диск, второй бит на второй диск. Отказоустойчивость реализуется путем использования для кодирования данных корректирующего кода Хемминга. Коды коррекции ошибок записываются на несколько дополнительных дисков. Для массива с числом основных дисков от 16 до 32 необходимо иметь три дополнительных диска для хранения кода коррекции. Применяется в мейнфреймах и суперкомпьютерах. Обеспечивает высокую производительность и надежность, обладает высокой стоимостью реализации.

RAID 3



ДИСК 1	ДИСК 2	ДИСК 3	ДИСК 4
0000 0001	0000 0010	0000 0011	0000 0000
0000 0100	0000 0101	0000 0110	0000 0111
0000 0111	0000 1000	0000 1001	0000 0110
0000 1010	0000 1011	0000 1100	0000 1101

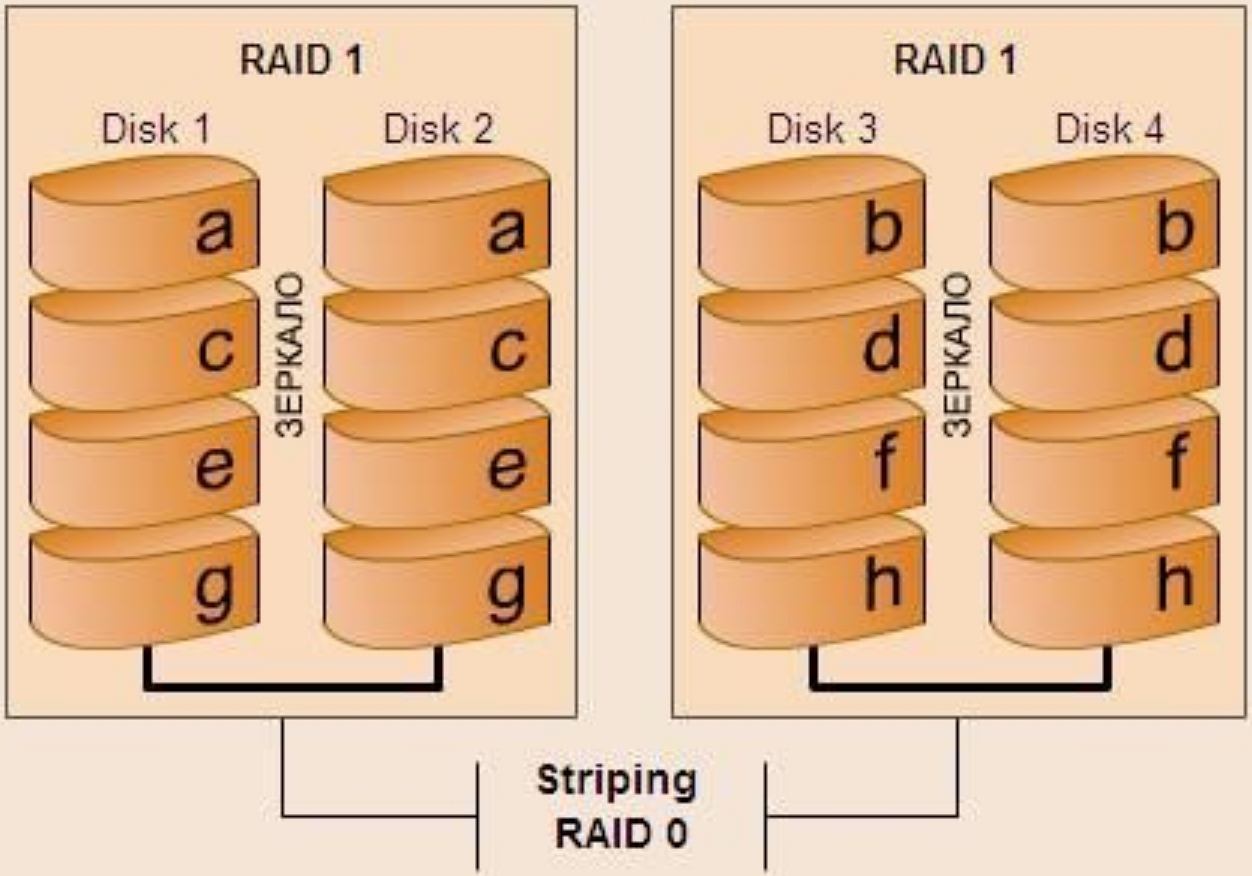
RAID 4 – аналогично, но данные расщепляются не побайтно, а блоками



Конфигурация RAID	Избыточность	Отказоустойчивость	Скорость чтения	Скорость записи
RAID 3, RAID 4, RAID 5	До 33%	Есть	Повышенная	Пониженная (в разной степени)

RAID 10

Отказоустойчивый массив с дублированием и параллельной обработкой



Конфигурация RAID	Избыточность	Отказоустойчивость	Скорость чтения	Скорость записи
RAID 10	50%	Есть	Повышенная	Повышенная

Характеристики уровней RAID

Конфигурация RAID	Избыточность	Отказоустойчивость	Скорость чтения	Скорость записи
RAID 0	Нет	Нет	Повышенная	Повышенная
RAID 1	50%	Есть	Повышенная	Повышенная
RAID 3, RAID 4, RAID 5	До 33%	Есть	Повышенная	Пониженная (в разной степени)
RAID 10	50%	Есть	Повышенная	Повышенная

Отказоустойчивость файловых и дисковых систем

1

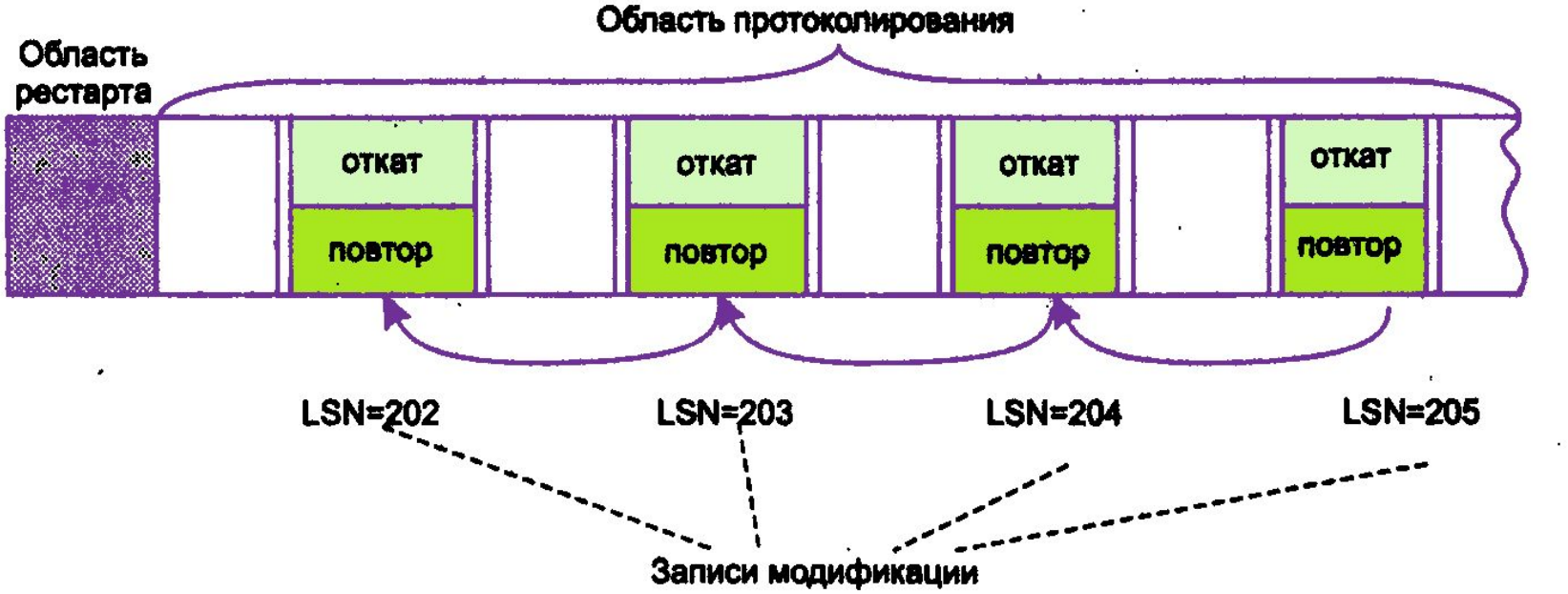
- защита от сбоев и отказов аппаратуры

2

- защита от ошибок программного обеспечения

Самовосстановление файловой системы NTFS

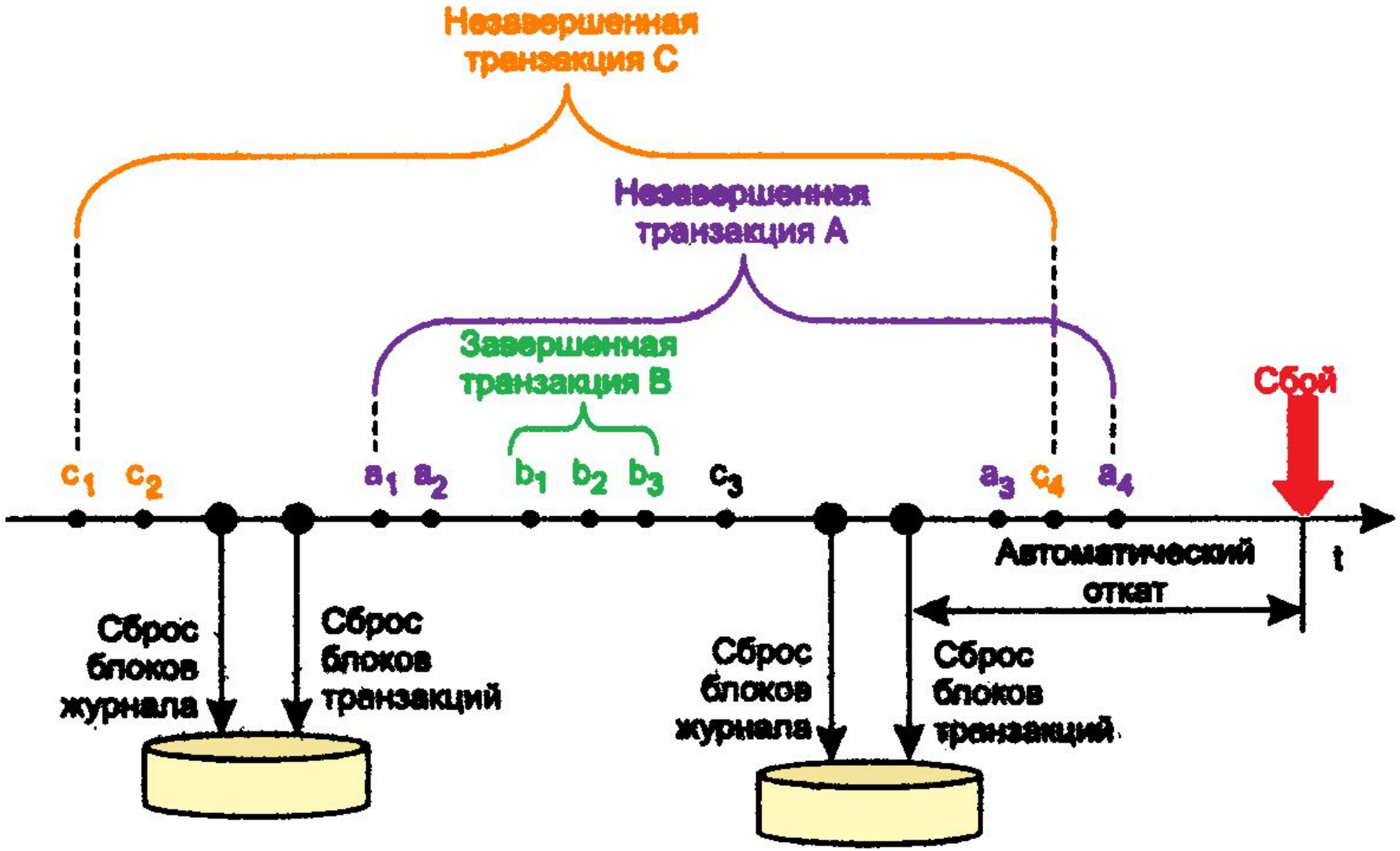
Журнал транзакций состоит из двух частей:
- область рестарта
- область протоколирования



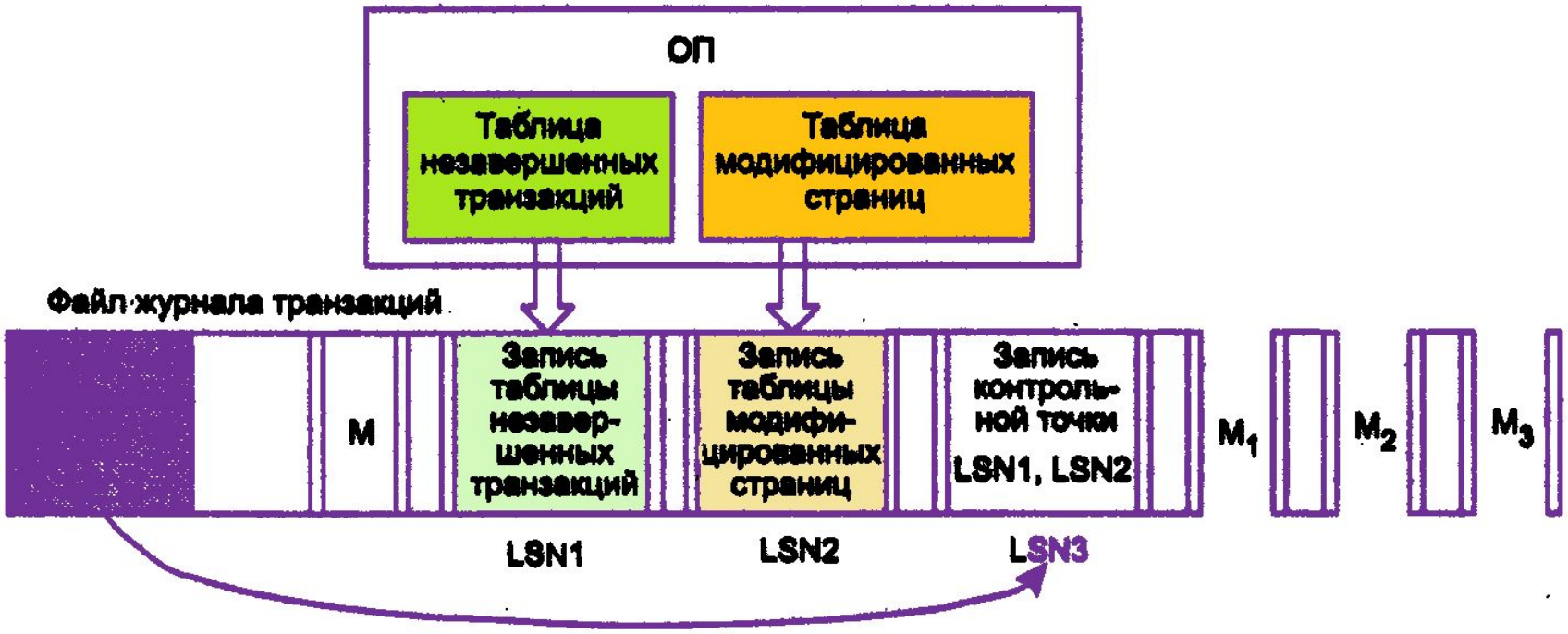
Типы записей: запись модификации; запись контрольной точки; запись фиксации транзакции; запись таблицы модификации; запись таблицы модифицированных страниц

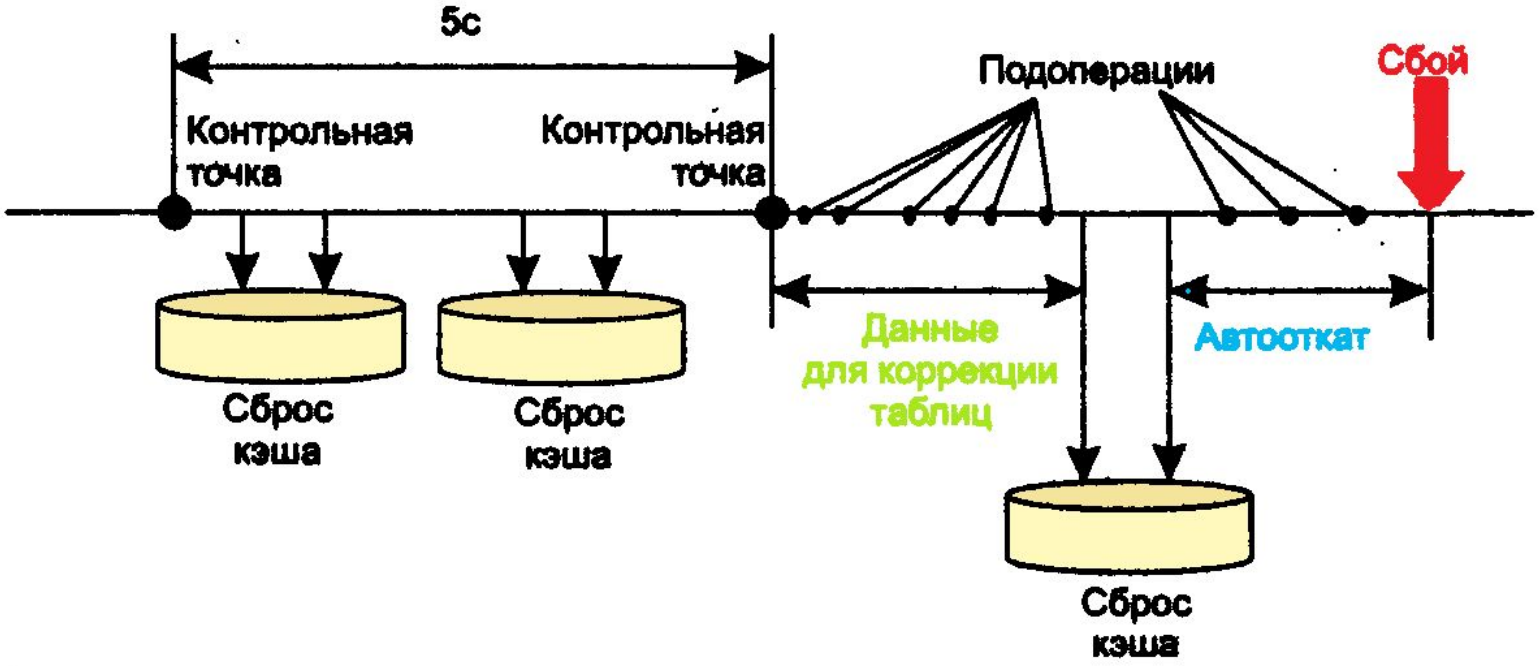
Пример записи модификации

Запись модификации	Информация для повторения транзакции	Информация для отката транзакции
LSN=202	Выделить и инициировать запись для нового файла lotus.doc в таблице MFT	Удалить запись о файле lotus.doc из таблицы MFT
LSN=203	Добавить имя файла в индекс	Исключить имя файла из индекса
LSN=204	Установить биты 3–9 в битовой карте	Обнулить биты 3–9 в битовой карте


Незавершенная транзакция С
Незавершенная транзакция А
Завершенная транзакция В
Сбой
c_1 c_2 a_1 a_2 b_1 b_2 b_3 c_3 a_3 c_4 a_4
t
Автоматический откат
Сброс блоков журнала
Сброс блоков транзакций
Сброс блоков журнала
Сброс блоков транзакций

Записи операции контрольная точка

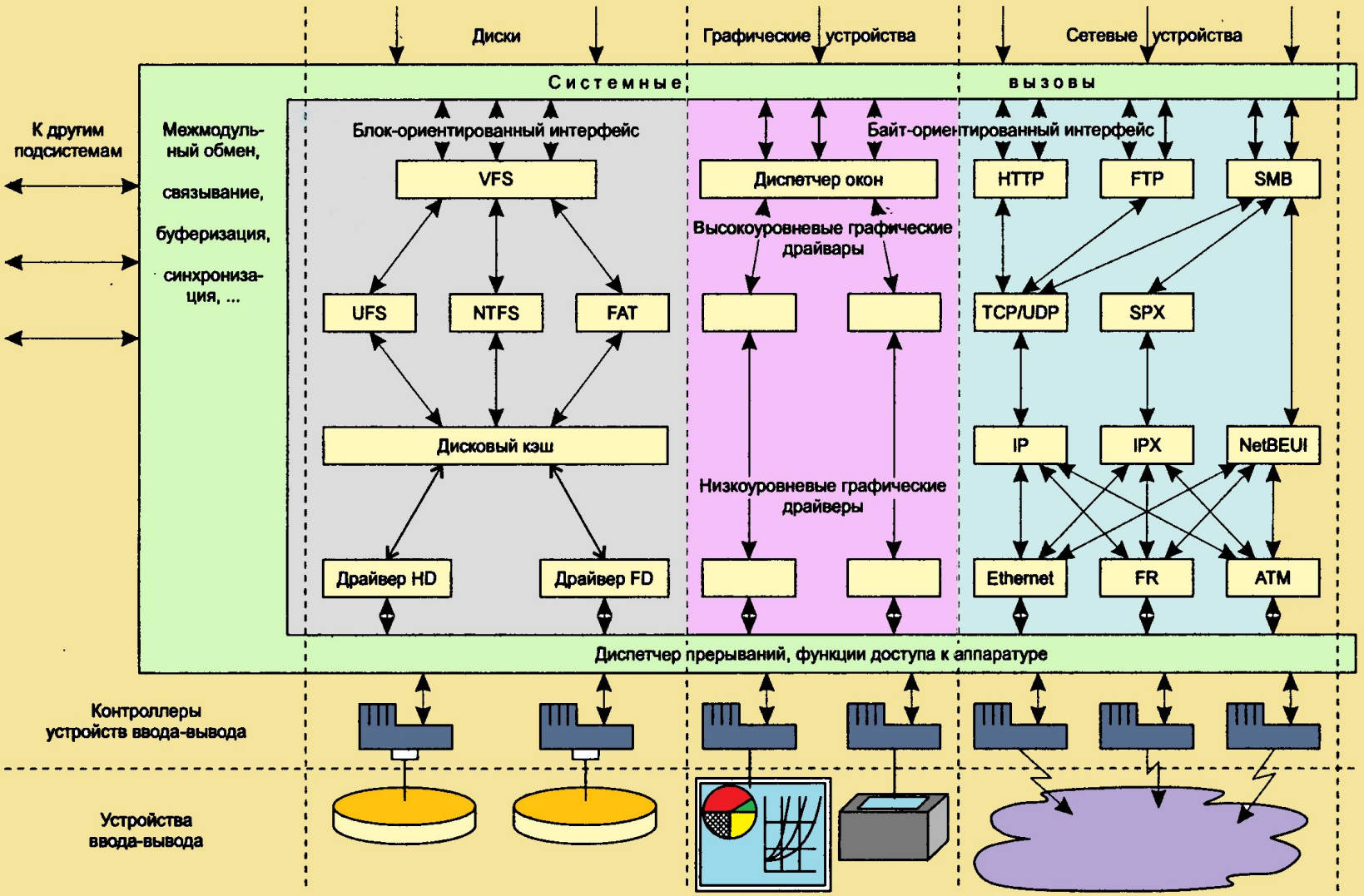




1. Чтение области рестарта из файла журнала транзакций и определение номера самой последней по времени записи о контрольной точке.
2. Чтение записи контрольной точки и определение номеров записей таблицы незавершенных транзакций и таблицы модифицированных страниц.
3. Чтение и корректировка таблиц незавершенных транзакций и модифицированных страниц на основании записей, сделанных в журнале транзакций уже после сохранения таблиц в журнале, но еще до записи журнала на диск
4. Анализ таблицы модифицированных страниц, определение номера самой ранней записи модификации страницы.

5. Чтение журнала транзакций в прямом направлении, начиная с самой ранней записи модификации, найденной при анализе таблицы модифицированных страниц. При этом система выполняет *повторение завершенных транзакций*,
6. Анализ таблицы незавершенных транзакций, определение номера самой поздней подоперации, выполненной в рамках незавершенной транзакции.
7. Чтение журнала транзакций в обратном направлении.
   откат незавершенных транзакций.

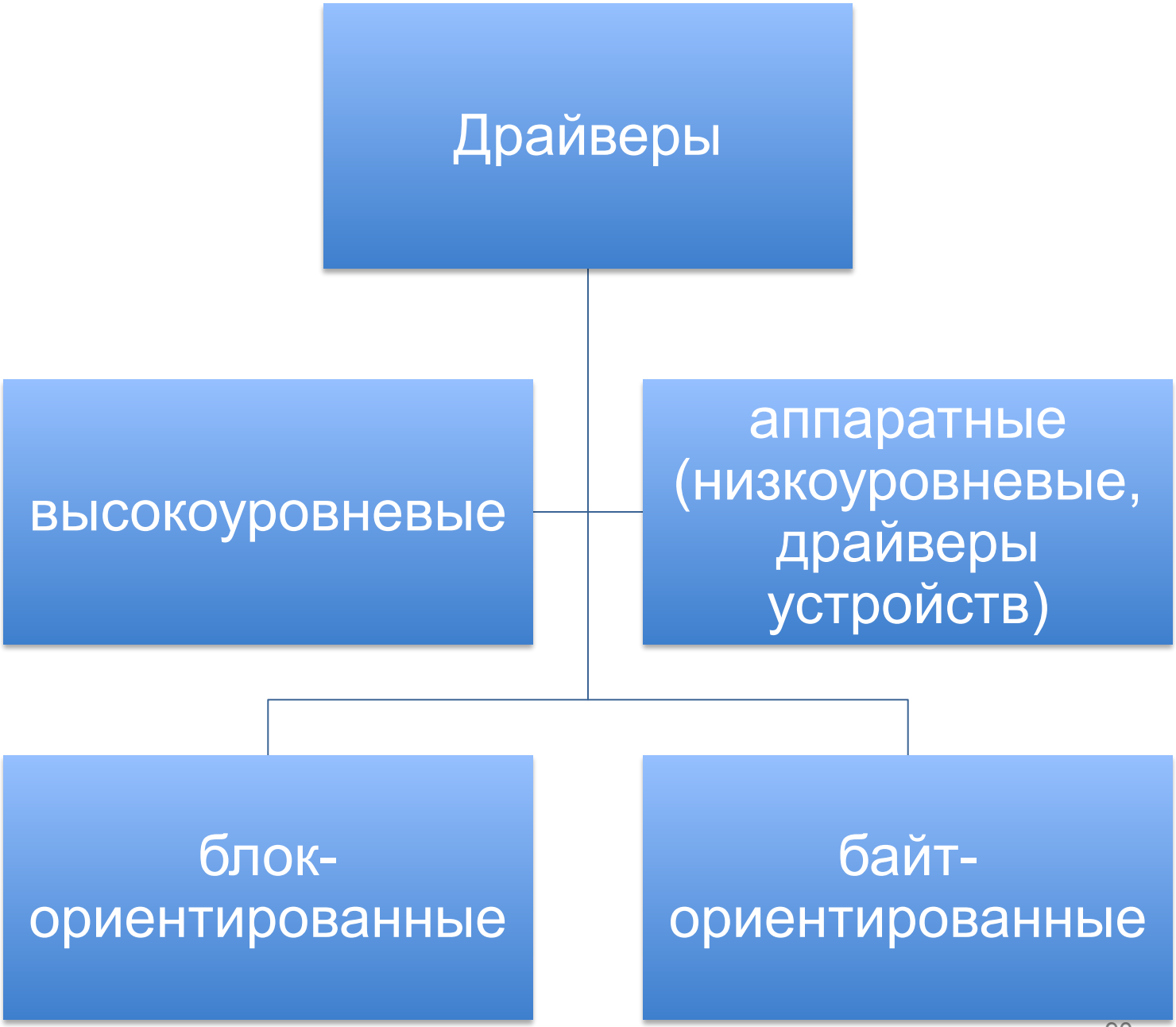
Многослойная модель подсистемы ввода/вывода



Драйверы

Драйвер – это программный модуль, который:

- работает в привилегированном режиме и входит в состав ядра ОС;
- непосредственно управляет внешним устройством, взаимодействуя с его контроллером с помощью команд ввода вывода компьютера;
- обрабатывает прерывания от контроллера устройства;
- предоставляет прикладному программисту удобный логический интерфейс работы с устройством, экранируя от него низкоуровневые детали управления устройством и организации его данных;
- взаимодействует с другими модулями ядра ОС с помощью строго оговоренного интерфейса, описывающего формат передаваемых данных, структуру буферов, способы включения драйвера в состав ОС, способы вызова драйвера, набор общих процедур подсистемы ввода-вывода, которыми драйвер может пользоваться и т.п.


Драйверы
высокоуровневые
аппаратные (низкоуровневые, драйверы устройств)
блок-ориентированные
байт-ориентированные

Структура драйвера Windows NT

Процедура инициализации драйвера. Эта процедура выполняется при загрузке драйвера в подсистему ввода-вывода, при этом создаются системные объекты, которые позволяют менеджеру ввода-вывода найти нужный драйвер для управления определенным устройством или выполнения некоторых высокоуровневых функций

Набор диспетчерских процедур. Эти процедуры составляют основу драйвера, так как именно они выполняют операции ввода-вывода, поддерживаемые данным драйвером, например чтение данных, запись данных, перемотку ленты и т. п.

Стартовая процедура предназначена для приведения устройства в исходное состояние перед началом очередной операции. Выполняет «открытие» (*open*) устройства.

Процедура обработки прерывания (ISR) включает наиболее важные действия, которые нужно выполнить при возникновении очередного аппаратного прерывания от контроллера устройства.

Процедура отложенных вызовов (DPC). процедура DPC обслуживается с более низким значением приоритета IRQL, давая возможность другим приоритетным запросам обслуживаться в первую очередь. Обычно большая часть действий драйвера по обработке прерывания включается в процедуру DPC.

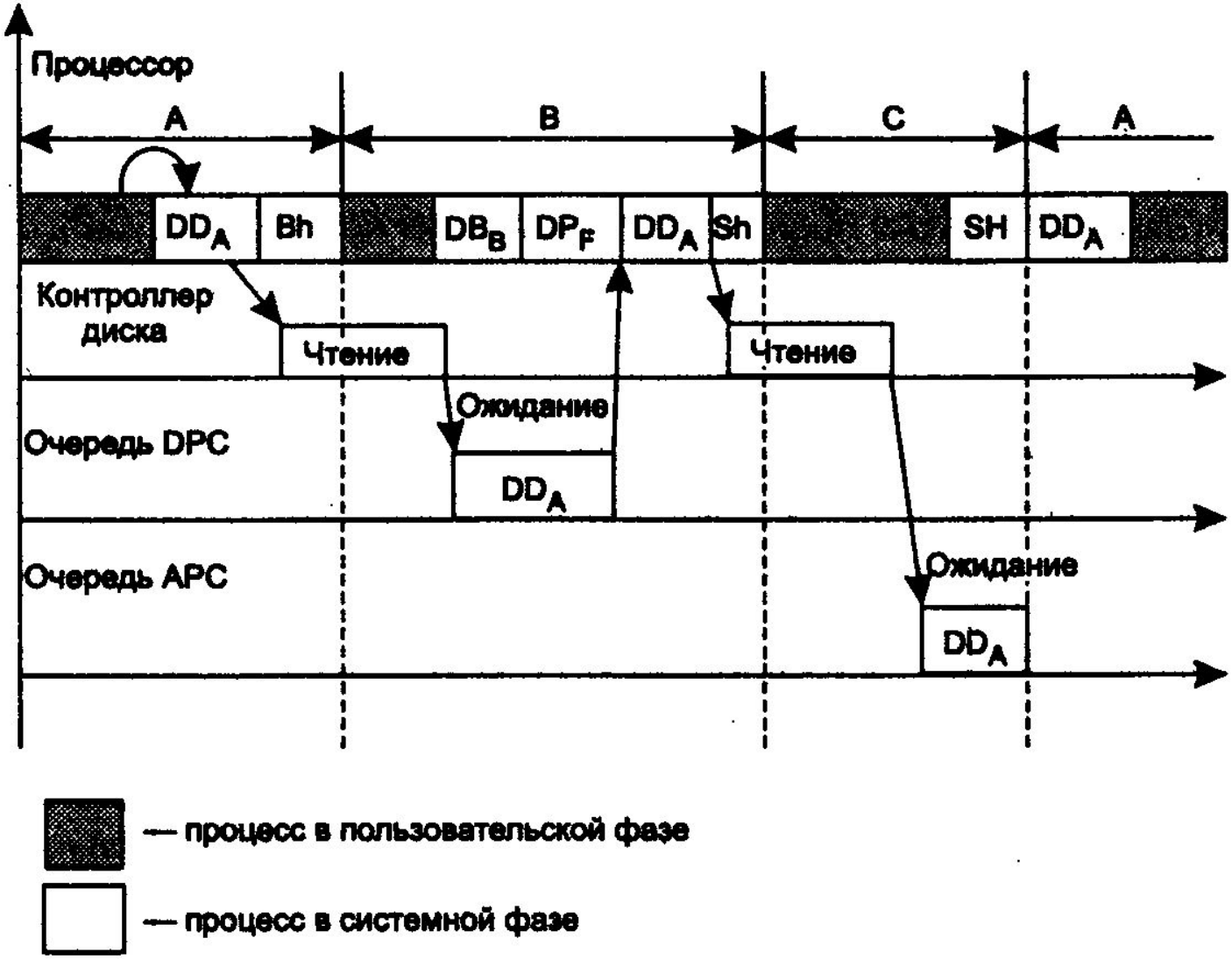
Процедура завершения операции уведомляет менеджер ввода-вывода о том, что операция завершена и данные находятся в системной области памяти.

Процедура отмены ввода-вывода. Для разных стадий выполнения операции могут существовать разные процедуры отмены.

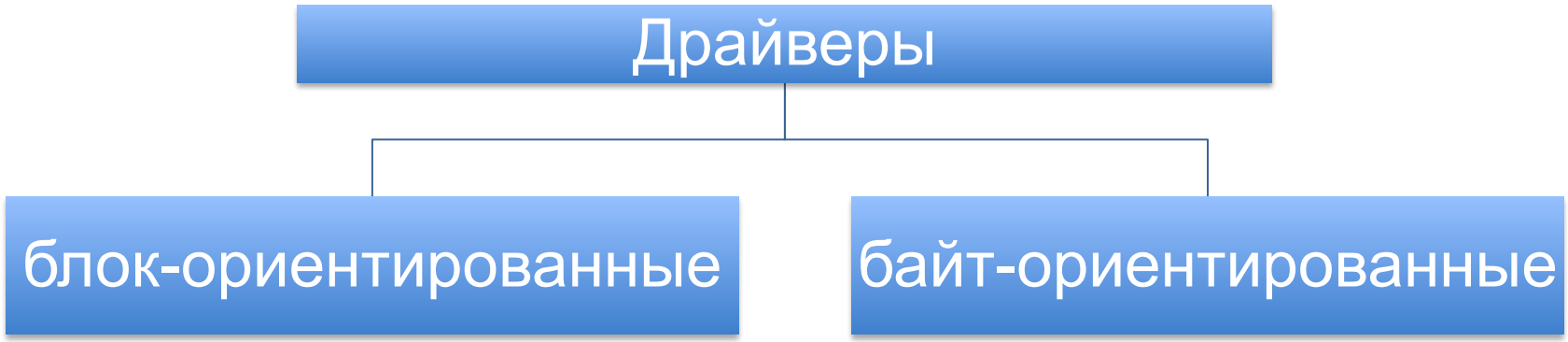
Процедура выгрузки драйвера вызывается при динамической выгрузке драйвера из подсистемы ввода-вывода. Удаляет созданные для драйвера объекты и освобождает системную память.

Процедура регистрации ошибок. При возникновении ошибки в процессе выполнения операции данная процедура уведомляет о ней менеджера ввода-вывода, который в свою очередь делает соответствующую запись в журнале регистрации.

Работа аппаратного драйвера Windows NT



Структура драйвера UNIX



Драйвер блок-ориентированного устройства состоит из следующих функций:

- open — выполняет процедуру логического открытия устройства;
- close — выполняет процедуру логического закрытия устройства;
- strategy — читает или записывает блок;
- print — выводит сообщение об ошибке;
- size — возвращает размер раздела, который представляет данное устройство.

Драйвер байт-ориентированного устройства состоит из следующих стандартных функций:

- open — открывает устройство;
- close — закрывает устройство;
- read — читает данные из устройства;
- write — записывает данные в устройство;
- ioctl — управляет вводом-выводом;
- poll — опрашивает устройство для выяснения не произошло ли некоторое событие;
- mmap, segmap — используются при отображении файла-устройства в виртуальную память.

Дисковый кэш

\+ повышение производительности дисковых операций

\- потенциальное снижение надежности

Традиционный дисковый кэш

- основан на автономном диспетчере кэша, обслуживающем набор буферов системной памяти и при необходимости самостоятельно организующим загрузку блока в буфер, не обращаясь за помощью к другим подсистемам ОС.

Дисковый кэш на основе виртуальной памяти

- использует ее возможности по отображению файлов на память. +функции диспетчера дискового кэша сокращаются, уменьшается объем ядра ОС, повышается его надежность.
- -во многих файловых системах существуют служебные данные, которые не относятся к файлам, и следовательно, не могут кэшироваться. Поэтому в таких случаях наряду с кэшем на основе виртуальной памяти применяется и традиционный дисковый кэш.

Вопросы безопасности вычислительных систем

К безопасности компьютера относят все проблемы защиты данных, хранящихся и обрабатывающихся компьютером. Эти проблемы решаются средствами операционных систем и приложений.

Безопасная система обладает свойствами:

- ✔ *Конфиденциальности* – гарантия того, что секретные данные будут доступны только тем пользователям, которым этот доступ разрешен (авторизованные пользователи);
- ✔ *Доступности* – гарантия того, что авторизованные пользователи всегда получат доступ к данным;
- ✔ *Целостности* – гарантия сохранности данными правильных значений, которая обеспечивается запретом для неавторизованных пользователей каким-либо образом модифицировать, разрушать или создавать данные.

Классификация угроз безопасности

Средства обеспечения безопасности:

- Морально-этические
- Законодательные
- Административные
- Психологические
- Физические
- Технические

Политика безопасности

- Какую информацию защищать?
- Какой ущерб понесет предприятие при потере или раскрытии тех или иных данных?
- Кто или что является возможным источником угрозы, какие атаки возможны в системе?
- Какие средства использовать для защиты каждого вида информации?

Базовые принципы:

- Минимальный уровень привилегий
- Комплексный подход к обеспечению безопасности
- Баланс надежности защиты всех уровней
- Использование средств, переходящих при отказе в состояние максимальной защиты
- Единый контрольно-пропускной пункт
- Баланс возможного ущерба от реализации угрозы и затрат на ее предотвращение

Базовые технологии безопасности

- Аутентификация
- Авторизация
- Аудит
- Технология защищенного канала

Аутентификация – установление подлинности

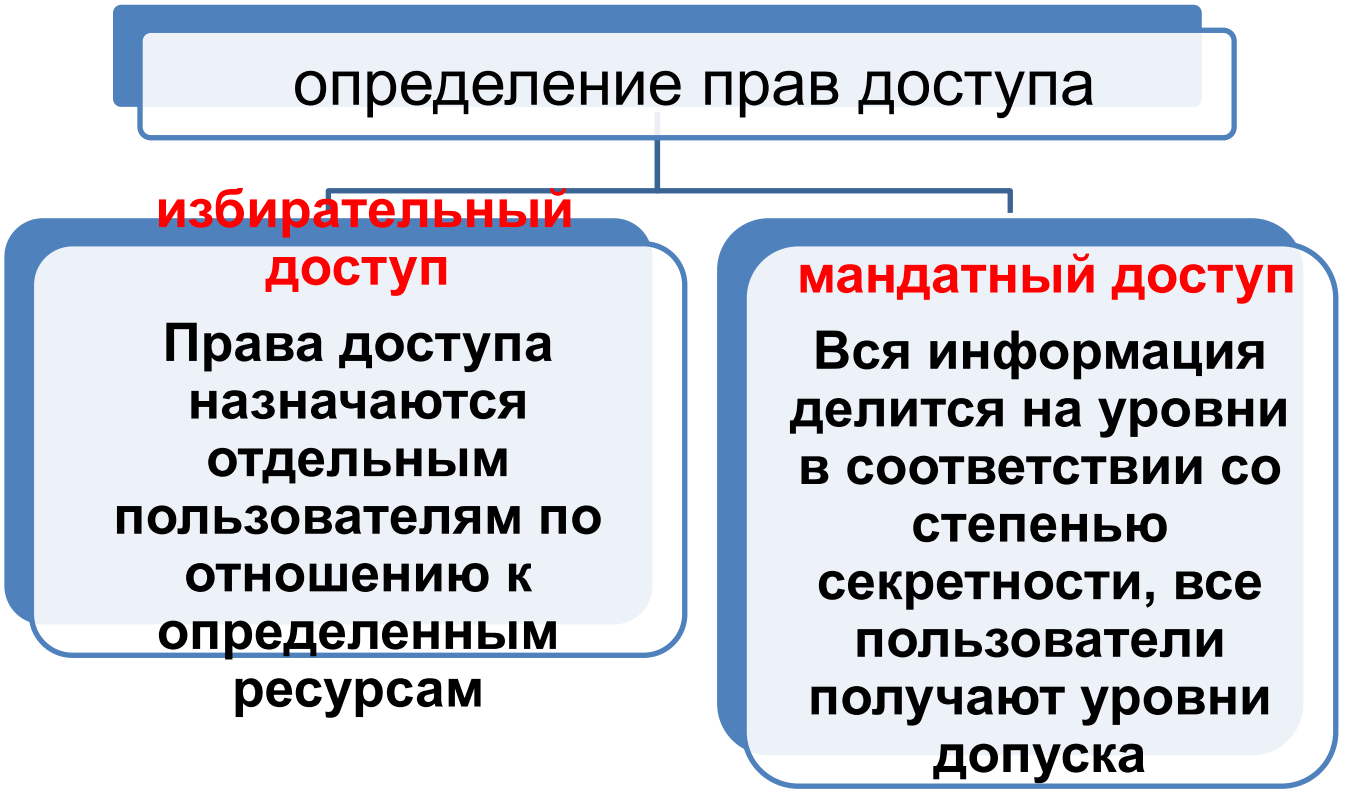
- Предотвращает доступ к сети нежелательных лиц и разрешает вход для легальных пользователей
- Для доказательства аутентичности можно использовать:

- знание некоего общего секрета: слова (пароля) или факта;
- владение неким уникальным предметом (физическим ключом);
- различные биохарактеристики: отпечатки пальцев, рисунок радужной оболочки глаз)


Слабости паролей
Раскрытие и разгадывание паролей
«Подслушивание» путем анализа сетевого трафика
Средства для формирования политики назначения и использования паролей
Шифрование перед передачей в сеть
Задание max и min срока действия
Хранение уже использованн ых паролей
Управление поведением системы после нескольких неудачных попыток входа

Авторизация доступа

- Средства авторизации контролируют доступ легальных пользователей к ресурсам системы, предоставляя каждому из них именно те права, которые ему были определены администратором.



Аудит -

фиксация в системном журнале событий, связанных с безопасностью

Технология защищенного канала

используется для обеспечения безопасности передачи данных в публичных сетях.

Выполняет три основные функции:

- взаимную аутентификацию абонентов при установлении соединения, которая может быть выполнена, например, путем обмена паролями;
- защиту передаваемых по каналу сообщений от несанкционированного доступа, например, путем шифрования;
- подтверждение целостности поступающих по каналу сообщений, например, путем передачи одновременно с сообщением его дайджеста.

Шифрование

- Криптосистема – пара процедур шифрование + дешифрирование.
- Современные алгоритмы шифрования предусматривают наличие параметра – секретного ключа.

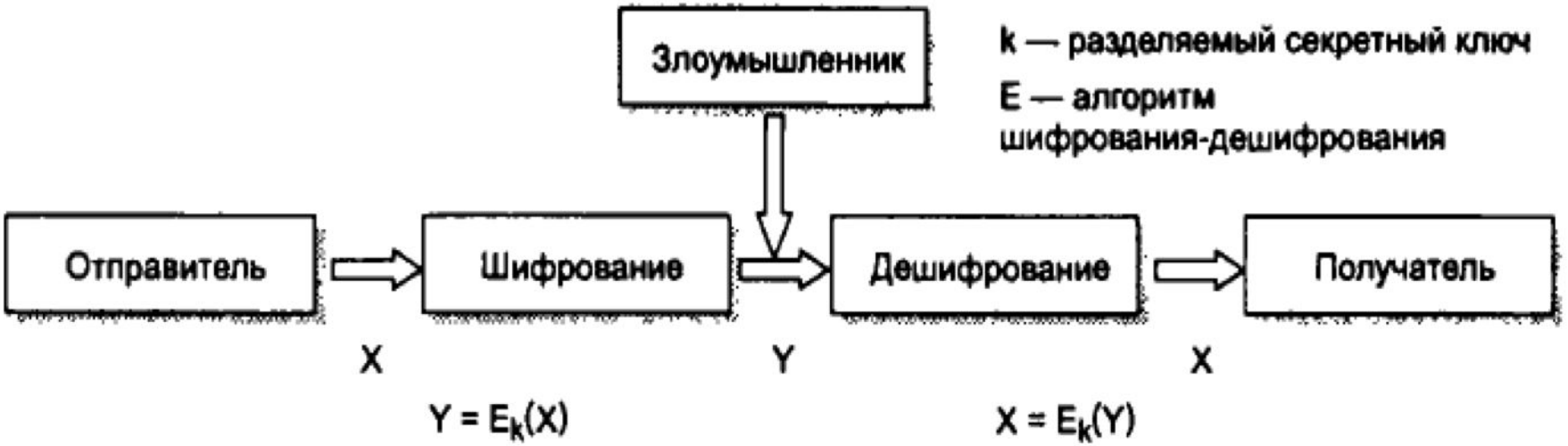
Правило Керкхоффа:

«Стойкость шифра должна определяться только секретностью ключа».

Алгоритм шифрования считается раскрытым, если найдена процедура, позволяющая подобрать ключ за реальное время. Сложность алгоритма раскрытия называется криптостойкостью.


Криптосистемы
Симметричные (классическая криптография)
Секретный ключ зашифровки совпадает с секретным ключом расшифровки
Асимметричные (криптография с открытым ключом)
Открытый ключ зашифровки не совпадает с секретным ключом расшифровки

Симметричные алгоритмы шифрования



Модель симметричного шифрования

Теоретические основы классической модели симметричной криптосистемы были изложены Клодом Шенноном в 1949 году. Модель является универсальной – если зашифрованные данные никуда не передаются, отправитель и получатель совмещаются в одном лице, а в роли злоумышленника выступает некто, имеющий доступ к компьютеру в отсутствии владельца.

Стандартный симметричный алгоритм шифрования DES (Data Encryption Standard)

Разработан IBM и в 1976 году рекомендован Национальным бюро стандартов к использованию в открытых секторах экономики

Схема шифрования по алгоритму DES

Данные шифруются поблочно. На вход шифрующей функции поступает блок данных размером 64 бита, он делится пополам на левую (L) и правую (R) части.

1. На место левой части результирующего блока помещается правая часть исходного блока.
2. Правая часть результирующего блока вычисляется как сумма по модулю два левой и правой части исходного блока.
3. На основе случайной двоичной последовательности по определенной схеме в полученном результате выполняются побитные замены и перестановки.

Используемая двоичная последовательность имеет длину 64 бита, из которых 56 действительно случайны, а 8 предназначены для контроля. Эта последовательность и является *КЛЮЧОМ.*

Для повышения криптостойкости иногда используют тройной алгоритм DES – троекратное шифрование с использованием 2 ключей. Производительность снижается.

AES (advanced Encryption Standard): 128 разрядные ключи (есть возможность использования 192- и 256-разрядных), за один цикл кодируется 128-разрядный блок.

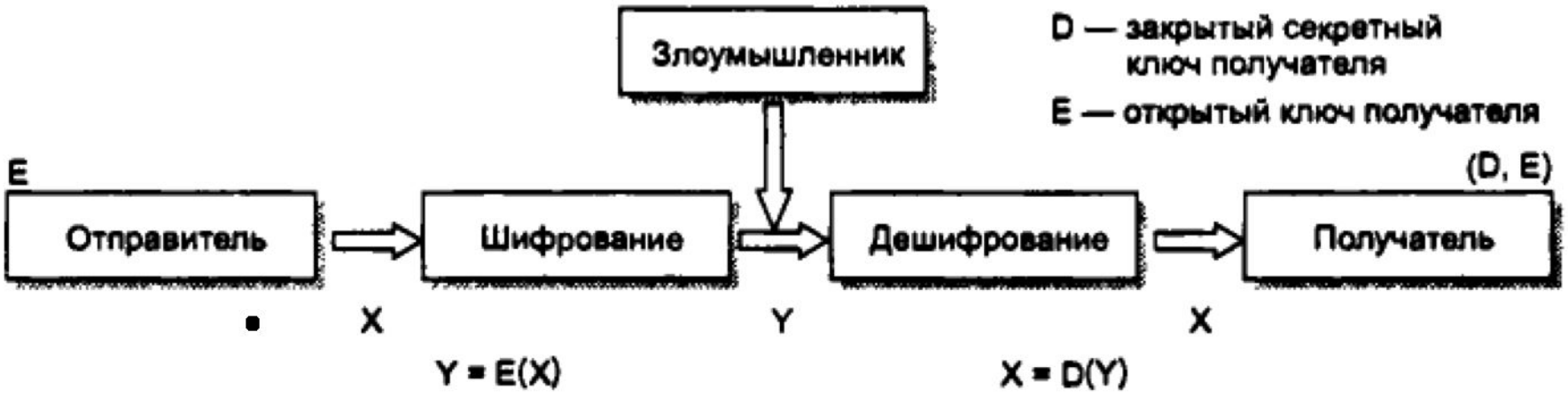
В симметричных алгоритмах планирования главную проблему представляют ключи.

1. Криптостойкость симметричных алгоритмов во многом зависит от качества ключа.

2. Надежность канала передачи ключа второму участнику секретных переговоров. При наличии n абонентов, желающих обмениваться секретными данными по принципу «каждый с каждым», потребуется $nx(n-1)/2$ ключей, которые должны быть сгенерированы и распределены надежным способом. Количество ключей пропорционально квадрату количества абонентов, что при большом количестве абонентов делает задачу чрезвычайно сложной.

Несимметричные алгоритмы шифрования

Винфилд Диффи и Мартин Хеллман в середине 70-х описали принципы шифрования с открытыми ключами.
Одновременно генерируется уникальная пара ключей, такая, что текст, зашифрованный одним ключом, может быть расшифрован только с использованием второго ключа, и наоборот.



Модель криптосхемы с открытым ключом

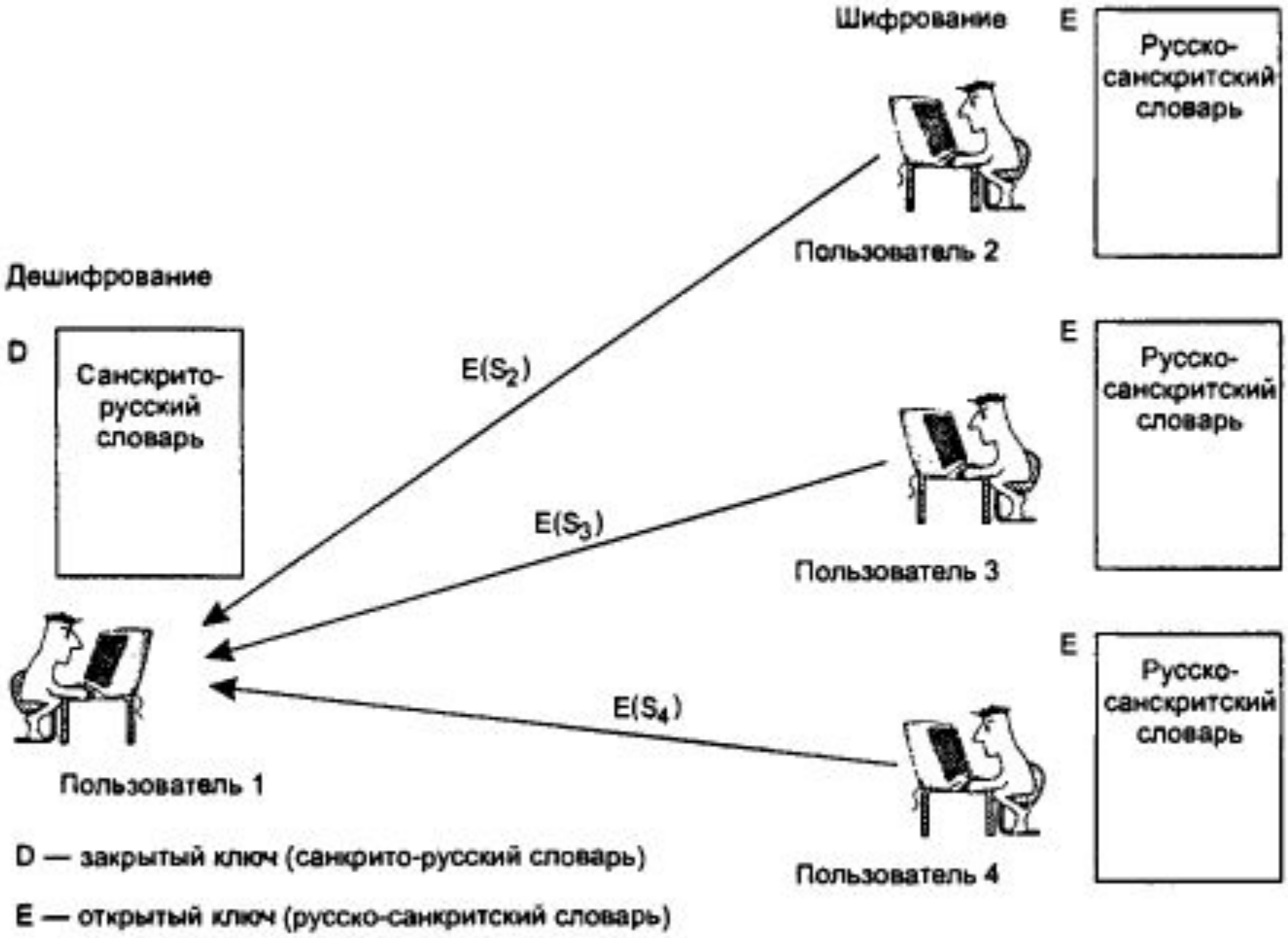
Задача отправителя заключается в том, чтобы по открытому каналу связи передать некоторое сообщение в защищенном виде.

Получатель на своей стороне генерирует два ключа: открытый Е и закрытый Д.

Закрытый ключ абонент должен хранить в защищенном месте, а открытый передает всем, с кем хочет поддерживать защищенные отношения.

Открытый ключ используется для шифрования текста, но расшифрован он может быть только с помощью закрытого ключа.

Открытый и закрытый ключи не могут быть независимы между собой, следовательно, существует теоретическая возможность вычисления закрытого ключа по открытому, но это связано с огромным объемом и временем вычислений.


Шифрование
E
Русско-санскритский словарь
Пользователь 2
Дешифрование
D
Санскрито-русский словарь
E(S2)
E
Русско-санскритский словарь
E(S3)
Пользователь 3
E
Русско-санскритский словарь
E(S4)
Пользователь 1
Пользователь 4
D — закрытый ключ (санкрито-русский словарь)
E — открытый ключ (русско-санкритский словарь)

а

Аутентификация или электронная подпись

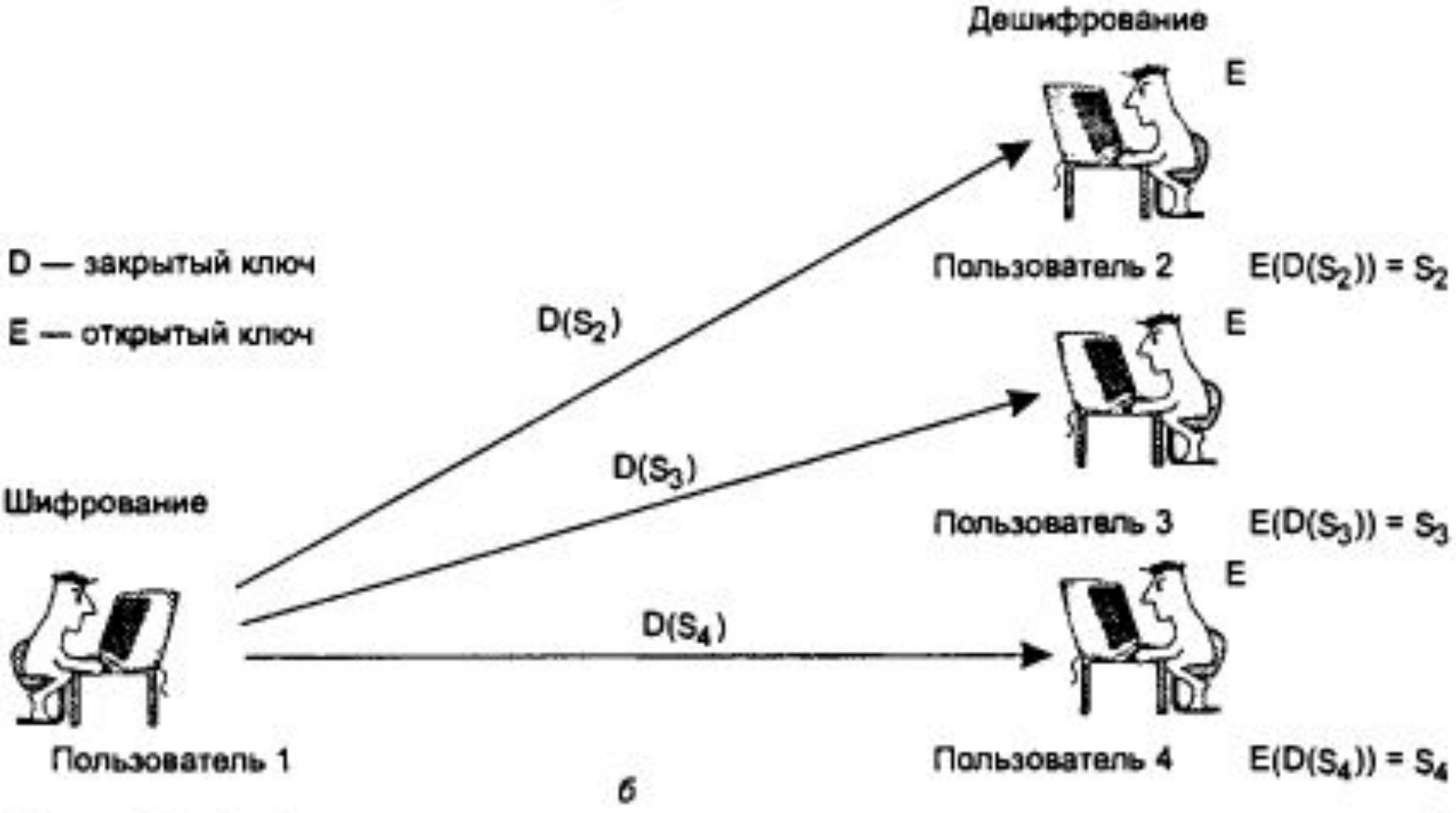


Рис. 11.4. Две схемы использования открытого и закрытого ключей

Если нужна взаимная аутентификация и двунаправленный секретный обмен сообщениями, то каждая из общающихся сторон генерирует свою пару ключей и посылает открытый ключ своему абоненту.
В сети из n абонентов всего будет 2n ключей: n открытых ключей для шифрования и n секретных ключей для дешифрирования.
Таким образом решается проблема масштабируемости – квадратичная зависимость количества ключей от числа абонентов в симметричных алгоритмах заменяется линейной зависимостью в несимметричных алгоритмах. Исчезает задача секретной доставки ключа. Злоумышленнику нет смысла стремиться захватить секретный ключ, поскольку это не дает возможности расшифровать сообщение или вычислить закрытый ключ.

Хотя информация об открытом ключе не является секретной, ее нужно защищать от подлогов, чтобы злоумышленник под видом легального пользователя не навязал свой открытый ключ, после чего он сможет дешифрировать сообщения своим закрытым ключом и рассылать свои сообщения от имени легального пользователя.
Решение этой проблемы – технология цифровых сертификатов.
Сертификат – это электронный документ, который связывает конкретного пользователя с конкретным ключом.

Криптоалгоритм RSA

Наиболее популярный в настоящее время криптоалгоритм с открытым ключом. Разработан в 1978 году.

RSA (буквенная аббревиатура от фамилий Rivest, Shamir и Adleman). RSA стал первым алгоритмом такого типа, пригодным и для шифрования, и для цифровой подписи. Алгоритм используется в большом числе криптографических приложений.

После работы над более чем 40 возможными вариантами, им удалось найти алгоритм, основанный на различии в том, насколько легко находить большие простые числа и насколько сложно раскладывать на множители произведение двух больших простых чисел, получивший впоследствии название RSA.

1. Случайно выбираются два очень больших простых числа p и q.
2. Вычисляются два произведения $n=p\times q$ и $m=(p-1)\times(q-1)$.
3. Выбирается случайное целое число E, не имеющее общих сомножителей с m.
4. Находится D, такое, что $DE=1$ по модулю m.
5. Исходный текст, X, разбивается на блоки таким образом, чтобы $0<X<n$.
6. Для шифрования сообщения необходимо вычислить $C=X^E$ по модулю n.
7. Для дешифрирования вычисляется $X=C^D$ по модулю n.

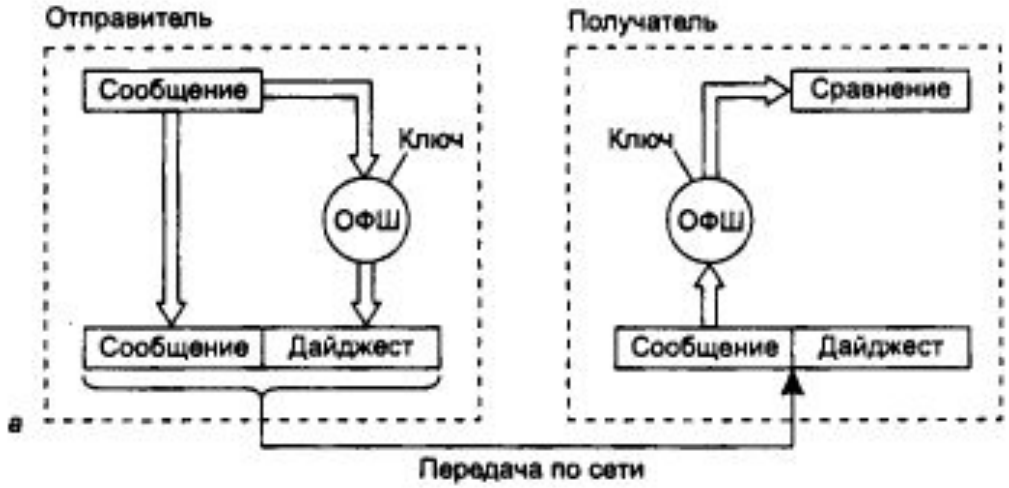
Таким образом, чтобы зашифровать сообщение, необходимо знать пару чисел (E, n), а чтобы дешифрировать – пару чисел (D, n). Первая пара – это открытый ключ, а вторая – закрытый.

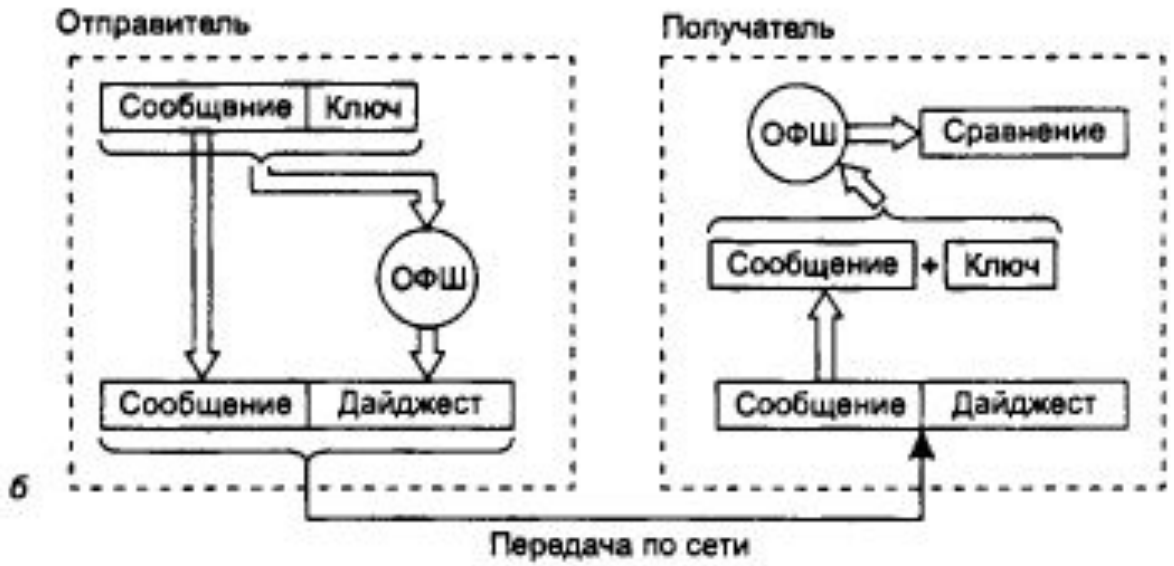
Зная открытый ключ (E, n), можно вычислить значение закрытого ключа D. Необходимым промежуточным действием в этом преобразовании является нахождение чисел p и q, для чего нужно разложить на простые множители очень большое число n, а на это требуется очень много времени. Именно с огромной вычислительной сложностью разложения большого числа на простые множители связана высокая криптостойкость алгоритма RSA. В некоторых публикациях приводятся следующие оценки: для того чтобы найти разложение 200-значного числа, понадобится 4 миллиарда лет работы компьютера с быстродействием миллион операций в секунду. Однако следует учесть, что в настоящее время активно ведутся работы по совершенствованию методов разложения больших чисел, поэтому в алгоритме RSA стараются применять числа длиной более 200 десятичных разрядов.

Программная реализация криптоалгоритмов типа RSA значительно сложнее и менее производительна, чем реализация классических криптоалгоритмов типа DES. Вследствие сложности реализации операций модульной арифметики криптоалгоритм RSA часто используют только для шифрования небольших объемов информации, например для рассылки классических секретных ключей или в

Односторонние функции шифрования

Шифрование с помощью односторонней функции (хэш-функции, дайджест-функции).
Эта функция, примененная к шифруемым данным, дает в результате значение (дайджест), состоящее из фиксированного небольшого числа байтов. Дайджест передается с исходным сообщением. Получатель сообщения, зная какая ОФШ, была применена для получения дайджеста, заново вычисляет его, использую незащищенную часть сообщения. Если полученный и вычисленный дайджесты совпадают, значит, полученное сообщение не подвергалось изменениям.





Построение односторонних функций является трудной задачей. Такого рода функции должны удовлетворять двум условиям:

- по дайджесту, вычисленному с помощью данной функции, невозможно каким-либо образом вычислить исходное сообщение;
- должна отсутствовать возможность вычисления двух разных сообщений, для которых с помощью данной функции могли быть вычислены одинаковые дайджесты.