Операционные системы и сети ЭВМ

Operating Systems and Networking

Лекция 26

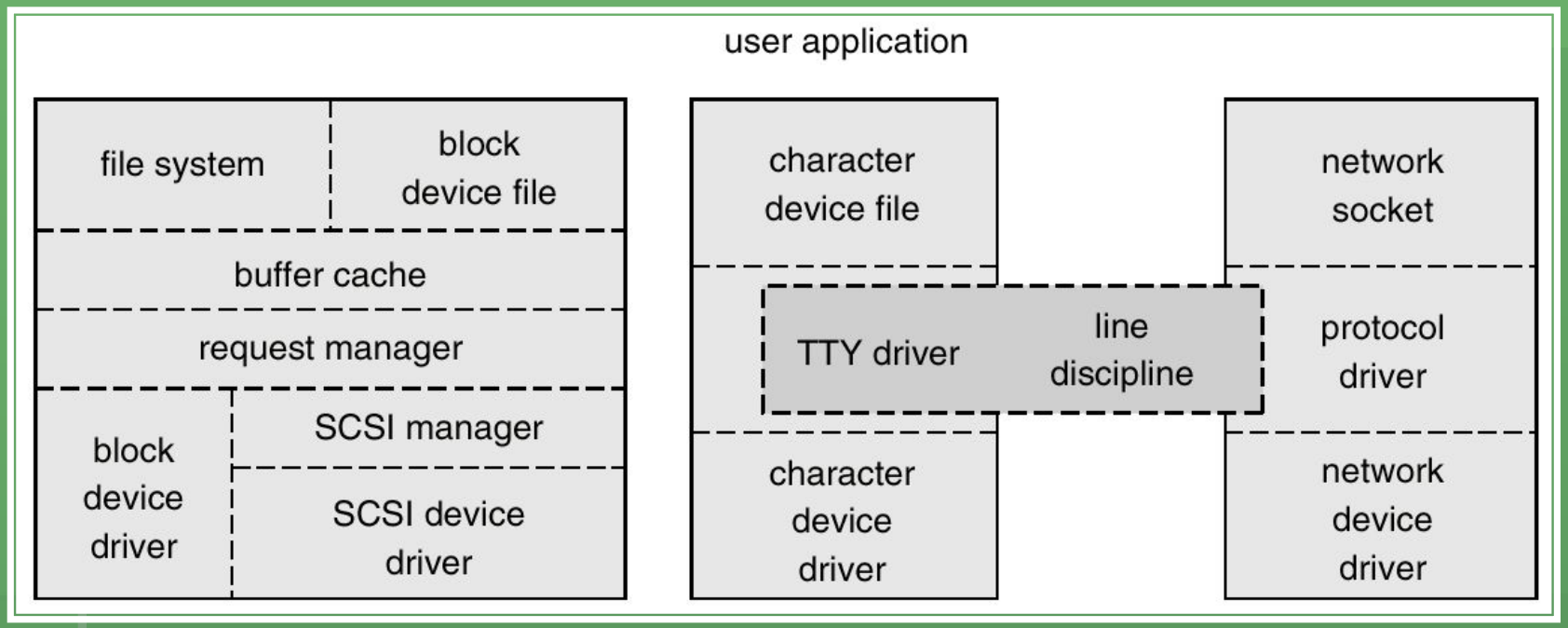
Сафонов Владимир Олегович,
профессор кафедры информатики,
руководитель лаборатории
Java-технологии
НИИММ СПбГУ

Email: **v_o_safonov@mail.ru**

Linux: Ввод и вывод

- **Система файлов Linux , ориентированная на устройства, осуществляет доступ к дисковой памяти с помошью двух кэшей:**
  - **Данные хранятся в кэше страниц, который объединен с системой виртуальной памяти**
  - **Метаданные хранятся в буферном кэше, причем каждый кэш индексируется блоком диска.**
- **Linux разбивает устройства на три класса:**
  - ***Блочные устройства* допускают произвольный доступ к полностью независимым блокам данных фиксированного размера**
  - ***Символьные устройства* включают большую часть всех других устройств; они не нуждаются в поддержке функциональности обычных файлов.**
  - ***Сетевые устройства* взаимодействуют с сетевой системой ядра**

Блочная структура драйверов устройств



Блочные устройства

- **Обеспечивают основной интерфейс ко всем дисковым устройствам в системе.**
- **Блочный буферный кэш служит для двух основных целей:**
  - **Как буферный пул для активного ввода-вывода**
  - **Как кэш для завершенного ввода-вывода**
- **Менеджер запросов управляет чтением и записью содержимого буферов с помощью драйвера блочного устройства.**

Символьные устройства

- Драйвер устройства, которое не поддерживает произвольный доступ к фиксированным блокам данных.
- Драйвер символьного устройства должен зарегистрировать набор функций, реализующих разнообразные требуемые операции ввода-вывода.
- Ядро не выполняет почти никакой предварительной обработки запроса на чтение или запись в файл символьного устройства, но просто передает данный запрос драйверу устройства.
- Основное исключение из этого правила – это особый набор драйверов символьных устройств, которые реализуют доступ к терминальным устройствам – для них ядро поддерживает стандартный интерфейс.

Взаимодействие процессов

- **Как и UNIX, Linux информирует процессы о наступлении событий с помощью сигналов.**
- **Существует ограниченный набор сигналов, и они не могут нести какую-либо информацию: только факт, что сигнал имеет место, доступен процессу.**
- **Ядро Linux не использует сигналы для коммуникации процессов, исполняемых в режиме ядра. Коммуникация внутри ядра осуществляется с помощью структур планировщика – states (состояния) и wait.queue (очередь ожидания).**

Передача данных между процессами

- **Механизм конвейера (pipe) позволяет дочернему процессу наследовать коммуникационный канал от процесса-родителя. Данные, записываемые с одного конца конвейера, могут быть прочитаны на другом конце.**
- **Общая память обеспечивает очень быстрый способ коммуникации; любые данные, записанные одним процессом в регион общей памяти, могут быть немедленно прочитаны любым другим процессом, который отобразил этот регион в свое адресное пространство.**
- **Однако с целью синхронизации общая память должна использоваться в сочетании с каким-либо другим комуникационным механизмом.**

Объект в общей памяти

- **Объект в общей памяти используется как файл откачки для регионов общей памяти, так же как файл может быть использован для откачки информации из региона, отображаемого в память.**
- **Отображения в общую память перенаправляют отказы страниц в регион памяти, занятый разделяемым объектом.**
- **Разделяемые объекты помнят свое содержимое, даже если в данный момент никакие процессы не отображают их в свои виртуальные пространства памяти.**

Структура сети

- **Работа в сети – ключевая область функциональности в Linux.**
  - **Сетевая система Linux поддерживает основные Интернет-протоколы для коммуникаций UNIX - UNIX.**
  - **Она также реализует протоколы, характерные для ОС, отличных от UNIX, в частности, протоколы, используемые в сетях PC, таких как Appletalk и IPX.**
- **Внутри сетевая система Linux реализована в виде трех уровней абстракции:**
  - **Сокетный интерфейс**
  - **Драйверы протоколов**
  - **Драйверы сетевых устройств**

Структура сети (прод.)

- **Наиболее важный набор сетевых протоколов в Linux – это набор протоколов Интернета.**
  - **Она обеспечивает маршрутизацию между различными машинами на любом участке сети.**
  - **На верхнем уровне протокола маршрутизации поддерживаются UDP-, TCP- и ICMP-протоколы.**

Безопасность

- Подключаемые аутентифакационные модули (*pluggable authentication modules - PAM)* доступны в системе Linux.
- PAM основана на общей библиотеке, которая может быть использована любыми компонентами, которым требуется аутентифицировать пользователя.
- Управление доступом в системах типа UNIX, включая и Linux, осуществляется с помощью уникальных числовых идентификаторов пользователя и группы (uid и gid).
- Управление доступом выполняется путем присваивания объектам *маски защиты*, которая указывает, какие операции (чтение, запись, исполнение) доступны для владельца, группы и всех остальных пользователей.

Безопасность (прод.)

- Linux дополняет стандартный механизм UNIX – setuid - двумя способами:
  - Реализует этот механизм по спецификации POSIX, что позволяет каждому процессу process многократно освобождать и вновь получать свой действующий uid.
  - Добавлена характеристика процесса, которая предоставляет лишь подмножество полномочий по действующему uid.
- Linux обеспечивает другой механизм, который позволяет клиенту выборочно передавать доступ к отдельному файлу некоторому серверному процессу без предоставления ему каких-либо других привилегий.

Windows 2000

- История
- Принципы проектирования
- Компоненты системы
- Подсистемы окружения
- Файловая система
- Работа в сети
- Интерфейс программиста

Windows 2000

- 32-битовая многозадачная операционная система для микропроцессоров типа Intel (продолжает линию NT).
- Основные цели системы:
  - переносимость
  - безопасность
  - соответствие POSIX
  - поддержка многопроцессорности
  - расширяемость
  - поддержка интернационализации
  - совместимость приложений с MS-DOS и MS-Windows.
- Использует архитектуру микроядра.
- Доступна в нескольких версиях - Professional, Server, Advanced Server, National Server.
- В 1996 году было продано больше лицензий на NT server, чем лицензий на UNIX

История

- **В 1988 Microsoft приняла решение о разработке переносимой ОС “new technology” (NT), которая поддерживала бы и OS/2, и POSIX APIs.**
- **Первоначально NT должны была использовать OS/2 API как свое естественное окружение, однако в процессе разработки NT была изменена и стала использовать Win32 API, что отражает популярность Windows 3.0.**

Принципы проектирования

- **Расширяемость – многоуровневая архитектура.**
  - **Ядро (Executive), исполняемое в защищенном режиме, обеспечивает базовые системные сервисы.**
  - **Поверх ядра реализованы несколько серверных подсистем, работающих в пользовательском режиме.**
  - **Модульная структура позволяет добавлять новые подсистемы окружения без модификации ядра.**
- **Переносимость — Windows 2000 может быть перенесена с одной аппаратной архитектуры на другую со сравнительно небольшими изменениями.**

  Написана на C и C++.

  Код, зависящий от процессора, изолирован в динамически линкуемую библиотеку (DLL), называемую “уровень абстрагирования от аппаратуры”- “hardware abstraction layer” (HAL).

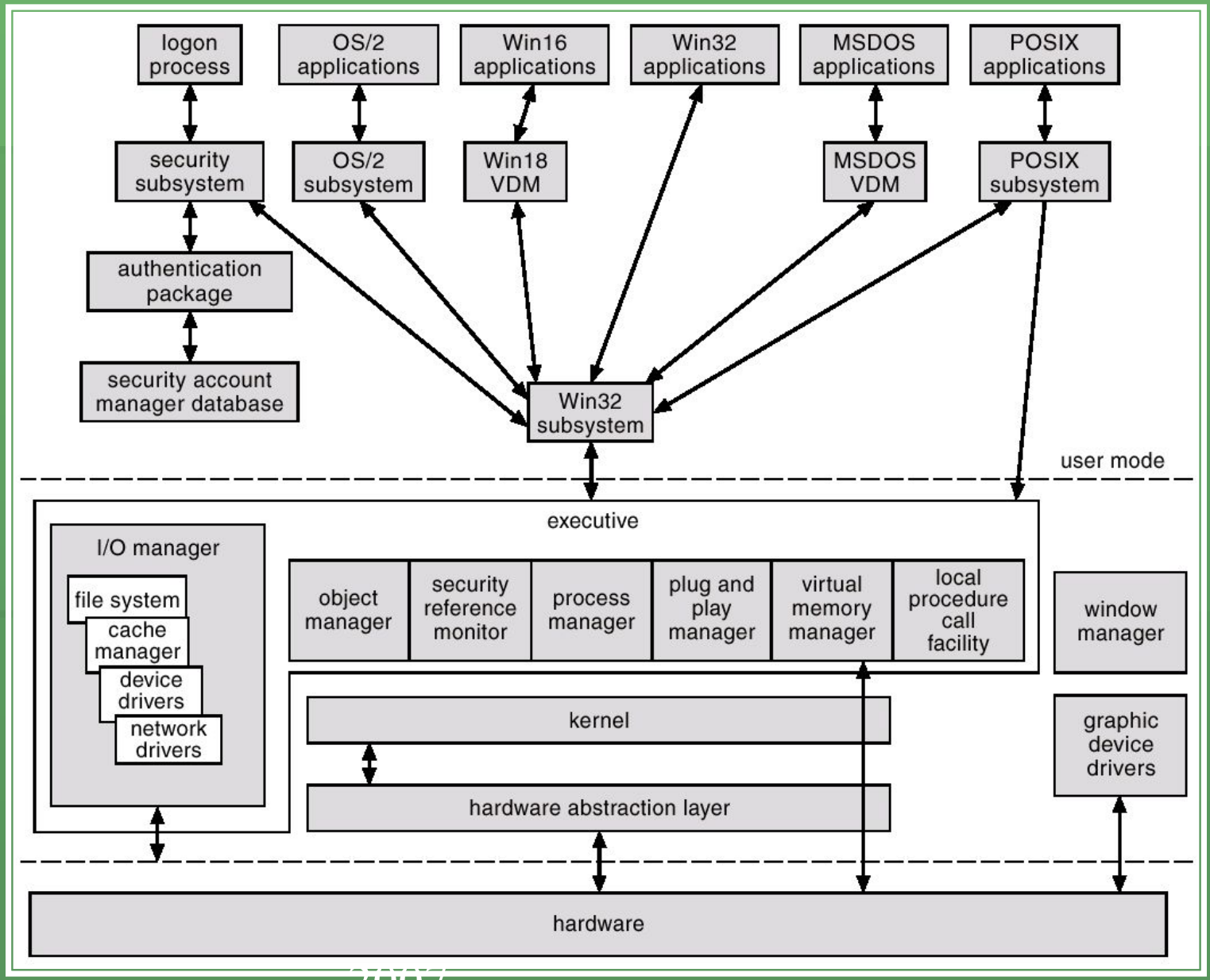
Принципы проектирования (прод.)

- **Надежность — Windows 2000 использует аппаратную защиту для виртуальной памяти and и программные защитные механизмы – для ресурсов ОС.**
- **Совместимость— приложения, которые следуют IEEE 1003.1 (POSIX) – стандарту, могут компилироваться для Windows 2000 без изменений в исходном тексте.**
- **Производительность — подсистемы Windows 2000 могут взаимодействовать друг с другом с помощью высокопроизводительной передачи сообщений.**
  - **Прерывание низкоприоритетных потоков позволяет системе быстро реагировать на внешние события.**
  - **Спроектирована для симметричного мультипроцессирования.**
- **Поддержка интернационализации (i18n) и локализации (l10n) — поддерживает различные языки и “культуры” с помощью NLS API.**

Архитектура **Windows 2000**

- **Многоуровневая система модулей.**
- **Защищенный режим — HAL, ядро, executive.**
- **Пользовательский режим – набор подсистем**
  - **Подсистемы окружения эмулируют различные ОС.**
  - **Подсистемы защиты реализуют различные функции безопасности.**

Схема архитектуры **Windows 2000**



Системные компоненты - ядро

- **Основа для executive и подсистем.**
- **Отказы страниц исключены; исполнение никогда не прерывается.**
- **Четыре основных функции:**
  - **Планирование потоков**
  - **Обработка прерываний и исключений**
  - **Низкоуровневая синхронизация процессов**
  - **Восстановление после отказов электропитания**
- **Ядро объектно-ориентированное, использует два набора объектов.**
  - ***Объекты-диспетчеры*** **управляют диспетчеризацией и синхронизацией (события, mutex'ы, семафоры, потоки, таймеры).**
  - ***Управляющие объекты*** **(асинхронные вызовы процедур, прерывание, нотификация об электропитании, состояние электропитания, профилирование.)**

Ядро – процессы и потоки

- **Процесс имеет адресное пространство в виртуальной памяти, информацию (например, базовый приоритет) и тесную связь с одним или несколькими процессами.**
- **Потоки – единицы исполнения, планируемые диспетчером ядра.**
- **Наждый поток имеет свое собственное состояние, включая приоритет, связь с процессором и статистическую информацию.**
- **Поток может быть в следующих состояниях: *ready, standby, running, waiting, transition*, and *terminated*.**

Ядро - планирование

- **Диспетчер использует 32-уровневую схему приоритетов для определения порядка выполнения потоков. Приоритеты разбиты на два класса:**
  - **Класс real-time содержит потоки с приоритетами от 16 до 31.**
  - **Класс variable содержит потоки с приоритетами от 0 до 15.**
- **Характеристики стратегии приоритетов Windows 2000.**

  Хорошее время ответа для потоков, использующих мышь и окна.

  Дает возможность потокам, связанным с вводом-выводом, обеспечивать занятость устройств ввода-вывода.

Ядро – планирование (прод.)

- **Планирование выполняется, кодга поток переходит в состояние ready или wait, когда поток завершается, оибо когда приложение изменяет приоритет потока или связь с процессором.**
- **Real-time потокам отдается предпочтение при выделении процессора; но ОС не гарантирует, что поток начнет выполняться в течение какого-либо определенного интервала времени. (такой подход известен как *soft real-time*.)**

Windows 2000: уровни запросов на прерывания

interrupt levels	types of interrupts
31	machine check or bus error
30	power fail
29	interprocessor notification (request another processor to act; e.g., dispatch a process or update the TLB)
28	clock (used to keep track of time)
27	profile
3-26	traditional PC IRQ hardware interrupts
2	dispatch and deferred procedure call (DPC) (kernel)
1	asynchronous procedure call (APC)
0	passive

Ядро – обработка прерываний

- **Ядро обеспечивает обработку прерываний, если исключения и прерывания генерируеются аппаратурой и программным обеспечением (NB: в ОС введены средства обработки исключений!).**
- **Исключения, которые не могут быть обработаны программно, обрабатываются *диспетчером исключений* ядра ОС.**
- **Диспетчер прерываний в ядре обрабатывает прерывание либо путем вызова подпрограммы, обслуживающей прерывание (например, драйвера устройства), либо путем вызова внутренней подпрограммы ядра.**
- **Ядро использует блокировщики (spin locks), находящиеся в основной памяти, для взиамного исключения процессов.**

Executive — менеджер объектов

- **Windows 2000 использует объекты для всех своих служб и представления сущностей; менеджер объектов управляет использованием всех объектов.**
  - **Генерирует object *handle* (ссылку на объект)**
  - **Выполняет проверки безопасности.**
  - **Следит за тем, какие процессы используют каждый объект.**
- **Объекты управляются стандартным набором методов: `create, open, close, delete, query name, parse, security`.**

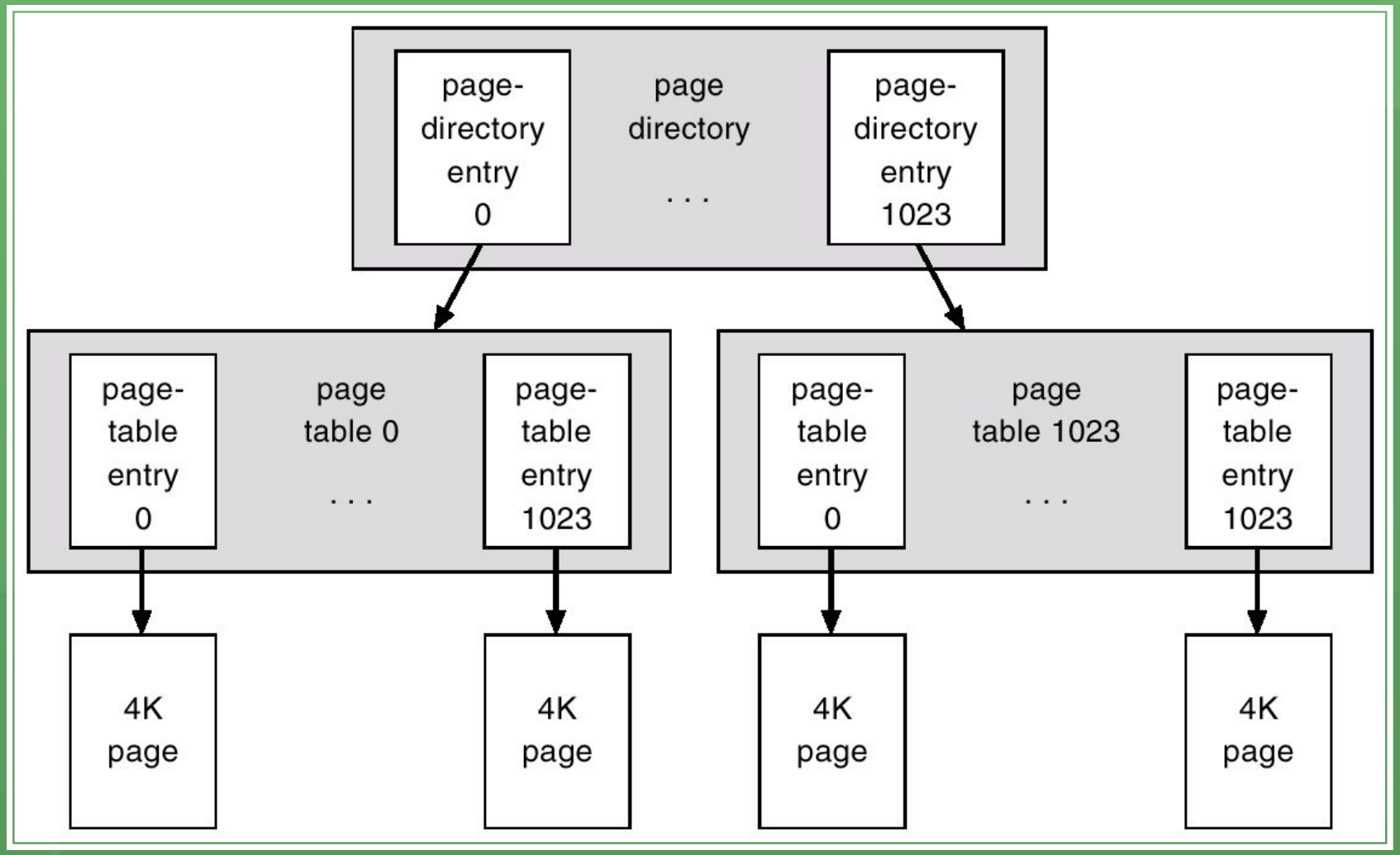
Executive — именование объектов

- Модуль executive поддерживает именование объектов. Имя может быть постоянным или временным.
- Имена объектов структурируются, как имена путей доступа к файлам в MS-DOS или UNIX.
- Реализованы *объекты-символические ссылки*, которые подобны символическим ссылкам в UNIX и дают возможность иметь несколько синонимов для одного файла.
- Процесс получает ссылку на объект при его создании, при открытии уже существующего объекта, при получении скопированной ссылки от другого процесса, либо путем наследования ссылки от процесса-родителя.
- Каждый объект защищен списком управления доступом.

Executive — менеджер виртуальной памяти

- **При проектировании менеджера виртуальной памяти предполагалось, что процессор поддерживает для отображения виртуальных адресов в физические механизм страничной организации, прозрачный кэш для многопроцессорных систем, а также алиасы для виртуальных адресов.**
- **VM – менеджер в Windows 2000 использует страничную организацию с размером страницы 4 KB.**
- **Используется двухуровневая схема выделения памяти.**
  - **На первом шаге резервируется часть адресного пространства процесса.**
  - **На втором шаге данное выделение поддерживается выделением пространства в файле откачки (paging file).**

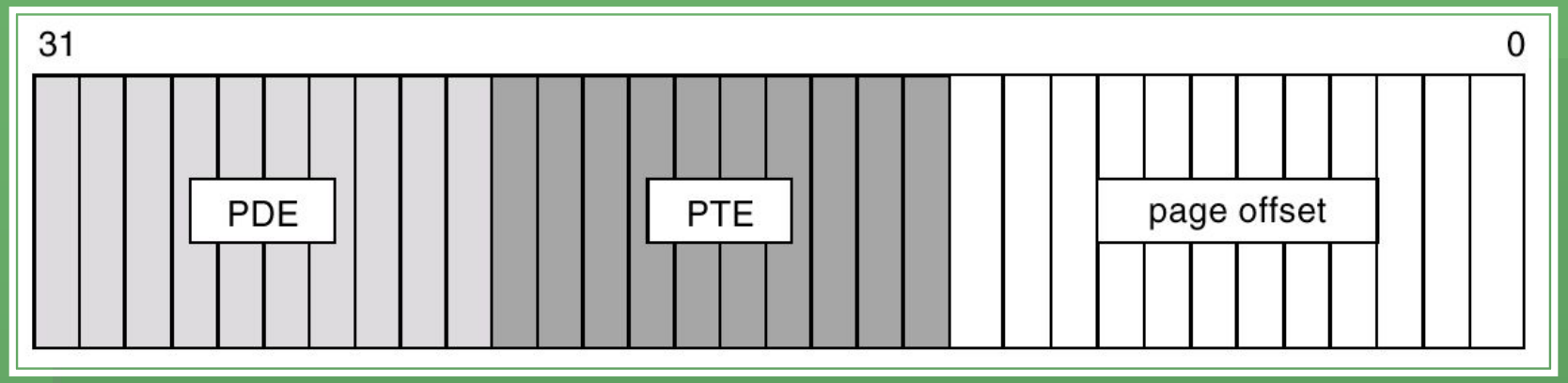
Распределение виртуальной памяти



Менеджер виртуальной памяти (прод.)

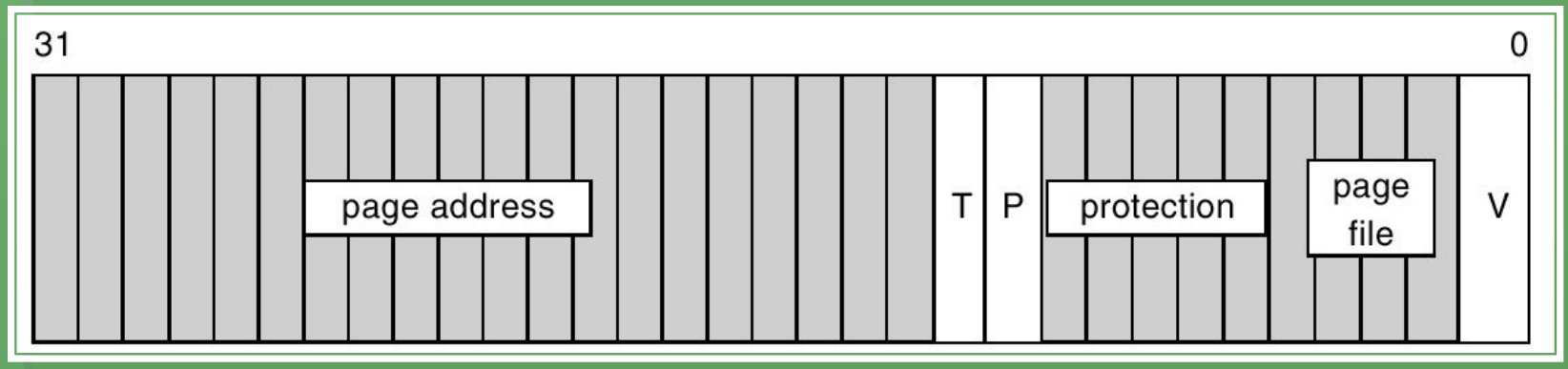
- **Трансляция виртуальных адресов в Windows 2000 использует несколько структур данных.**
  - **Каждый процесс имеет справочник страниц (*page directory*), содержащий 1024 *элемента справочника страниц* размером по 4 байта.**
  - **Каждый элемент справочника страниц ссылается на *таблицу страниц*, которая содержит 1024 *элемента таблицы страниц* (page table entries - PTEs) размером по 4 байта.**
  - **Каждый PTE ссылается на фрейм страницы (4 KB) в физической памяти.**
- **Ссылка на элемент всегда занимает 10 битов (0..1023).**
- **Это свойство используется при трансляции виртуальных адресов в физические.**
- **Станица может находиться в следующих состояниях: valid, zeroed, free standby, modified, bad.**

Трансляция виртуальных адресов в физические



- **10 битов для page directory entry, 10 битов для page table entry, 12 битов для смещения в байтах на странице.**

Элемент таблицы страниц файла откачки



- **5 битов для защиты страницы, 20 битов для адреса фрейма страницы, 4 бита для выбора файла откачки, 3 бита для описания состояния страницы. V = 0**

Executive — менеджер процессов

- **Обеспечивает сервисы для создания, удаления и использования потоков и процессов.**
- **Связи родительских процессов с дочерними и иерархии процессов обрабатываются конкретной подсистемой окружения, которая владеет данным процессом.**

Executive — локальный вызов процедуры **(LPC)**

- **LPC передает запросы и результаты между клиентским и серверным процессами на локальной машине.**
- **В частности, он используется для запросов к сервисам различных подсистем ОС.**
- **При создании канала для LPC должно быть указано сообщение одного из трех типов.**
  - **Первый тип – маленькие сообщения, до 256 байтов; в качестве промежуточной памяти используется очередь сообщений порта, и сообщения копируются от ного процесса к другому.**

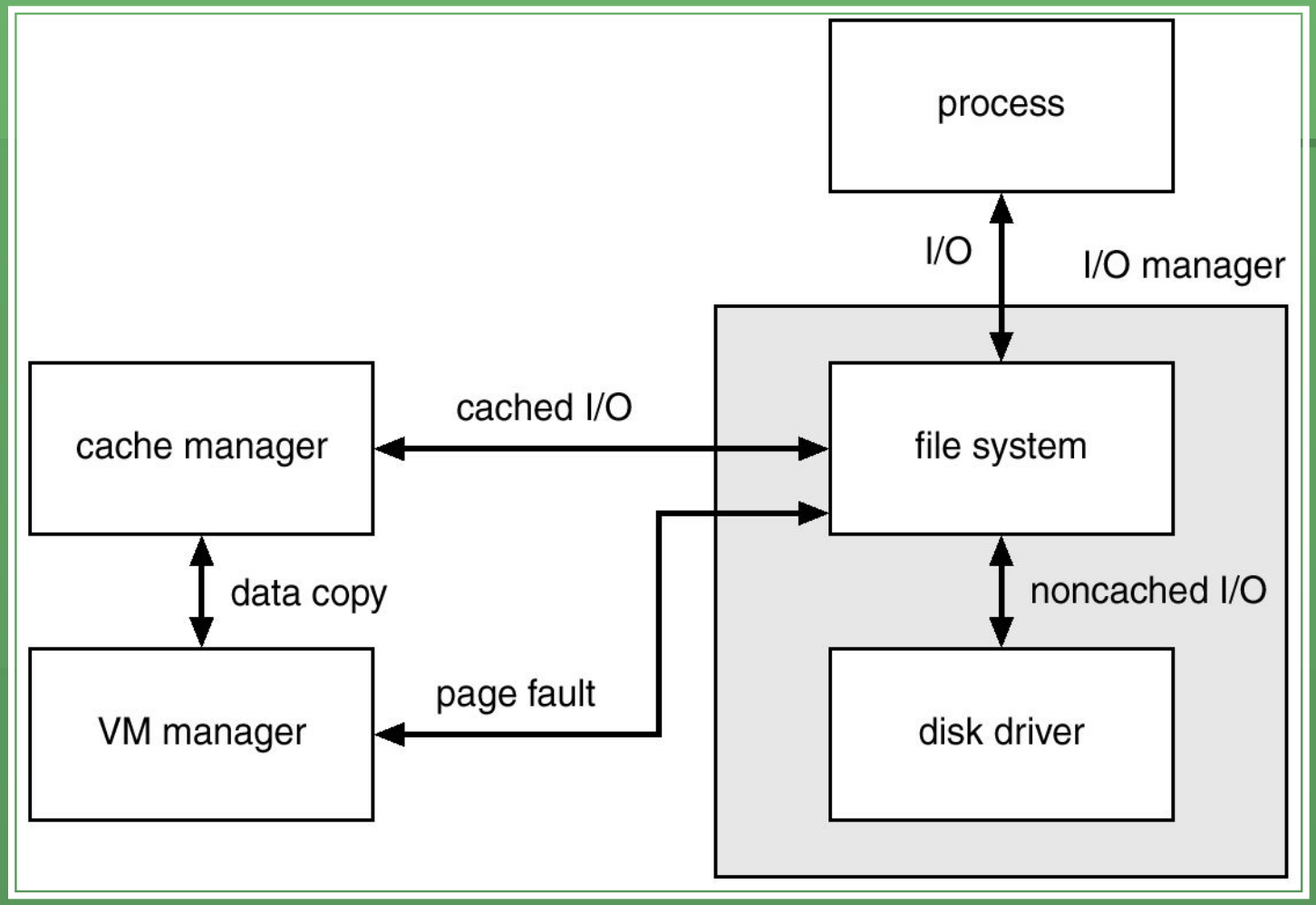
    Второй тип – во избежании копирования больших сообщений, передаются ссылки на разделяемые объекты, содержащие сообщения.

    Третий тип - быстрый LPC – используется графическими подсистемами Win32.

Executive — менеджер ввода-вывода

- Менеджер ввода-вывода отвечает за:
  - Файловые системы
  - Управление кэш-памятью
  - Драйверы устройств
  - Сетевые драйверы
- Следит за тем, какие файловые системы загружены, и управляет буферами для запросов на ввод-вывод.
- Взаимодействует с менеджером виртуальной памяти для обеспечения ввода-вывода в файлы, отображаемые в память.
- Управляет кэш-менеджером, который обеспечивает кэширование для всей системы ввода-вывода.
- Поддерживает как синхронные, так и асинхронные операции, обеспечивает тайм-ауты для драйверов, имеет механизмы для вызова одного драйвера другим драйвером.

Файловый ввод-вывод



Executive — Монитор безопасности

- **Объектно-ориентированная архитектура Windows 2000 обеспечивает использование единого механизма для контроля доступа во время выполнения и аудита всех объектов системы.**
- **Каждый раз, когда процесс получает ссылку на объект, монитор безопасности проверяет маркер безопасности процесса и список управления доступом к объекту для проверки того, имеет ли процесс необходимые права.**

Executive – Менеджер **Plug-and-Play**

- **Менеджер Plug-and-Play (PnP) используется для распознавания изменений в конфигурации оборудования и адаптации к ним (установки соответствующих драйверов).**
- **Когда добавляются новые устройства (например, PCI или USB), менеджер PnP загружает соответствующий драйвер.**
- **Менеджер PnP также следит за ресурсами, используемыми каждым устройством.**

Подсистемы окружения

- **Над executive надстраиваются процессы пользовательского режима, обеспечивающие исполнение программ, разработанных для других ОС.**
- **Windows 2000 использует подсистему Win32 как основное операционное окружение; Win32 используется для запуска всех процессов. Она также обеспечивает средства работы с мышью, клавиатурой и средства графики.**
- **Окружение MS-DOS обеспечивается приложением Win32, называемым *virtual dos machine* (VDM), процессом пользовательского уровня, для которого поддерживается страничная организация и диспетчеризация, как и для всех других потоков.**

Подсистемы окружения (прод.)

- **Окружение для 16-битовых Windows:**
  - **Обеспечивается VDM, которая содержит подсистему *Windows on Windows*.**
  - **Предоставляет процедуры ядра Windows 3.1 для менеджера окон и функций GDI.**
- **Подсистема POSIX спроектирована для исполнения POSIX-приложений, следующих POSIX.1 – стандарту, который базируется на модели UNIX.**

Подсистемы окружения (прод.)

- **Подсистема OS/2 выполняет OS/2 - приложения.**
- **Подсистема входа и безопасности аутентифицирует пользователей, входящих в систему Windows 2000. Требуется, чтобы пользователи имели имя учетной записи и пароль.**

  - Пакет аутентификации аутентифицирует всех пользователей, которые пытаются осуществить доступ к какому-либо объекту системы. Windows 2000 использует Kerberos как пакет аутентификации по умолчанию.

Система файлов

- **Фундаментальная структура системы файлов Windows 2000 (NTFS) – *том (volume*).**
  - **Создается утилитой администрирования диска.**
  - **Основан на логическом диске (partition).**
  - **Может занимать часть диска, целый диск или распределяться по нескольким дискам.**
- **Все метаданные, такие как информация о томе, хранятся в обычном файле.**
- **NTFS использует *кластеры* как базовую единицу выделения дисковой памяти.**

  Кластер – число секторов диска, размер которгого – степень двойки.

  Поскольку размер кластера меньше, чем в FAT16, внутренняя фрагментация уменьшается.

Система файлов – внутреннее представление

- **NTFS использует логические номера кластеров 0 *logical cluster numbers* (LCNs) в качестве дисковых адресов.**
- **Файл в NTFS – не просто байтовый поток, как в MS-DOS или в UNIX, но это структурированный объект, состоящий из атрибутов.**
- **Каждый файл в NTFS описывается одной или несколькими записями в массиве, хранящемся в специальном файле, называемом Master File Table (MFT).**
- **Каждый файл в томе NTFS имеет уникальный идентификатор (ID), называемый ссылкой на файл - *file reference.***
  - **64-битовое число, состоящее из 48-битового номера файла и 16-битового номера последовательности.**
  - **Может использоваться для выполнения внутренних проверок целостности.**
- **Пространство имен NTFS организовано в иерархию директорий; индексный корень (*index root*) содержит верхний уровень B+ - дерева.**

Файловая система - Восстановление

- **Все изменения структуры данных в файловой системе выполняются как транзакции, для которых используется журнал.**
  - **Перед тем, как структура данных изменяется, транзакция заноситв журнал специальную запись, которая содержит информацию для повторного выполнения (redo) и отмены (undo) данного изменения.**
  - **После изменения структуры данных в журнал заносится информация об успешном выполнении операции.**
  - **В случае порчи информации файловая система может быть восстановлена до целостного состояния с использованием журнальных записей.**

Файловая система – восстановление (прод.)

- **Эта схема не гарантирует, что все данные пользовательского файла могут быть восстановлены в случае порчи информации, а гарантирует лишь, что все структуры данных о файлах в системе (метаданные) не повреждены и отражают какое-либо целостное состояние данных до порчи информации.**
- **Журнал хранится в третьем файле метаданных каждого тома.**

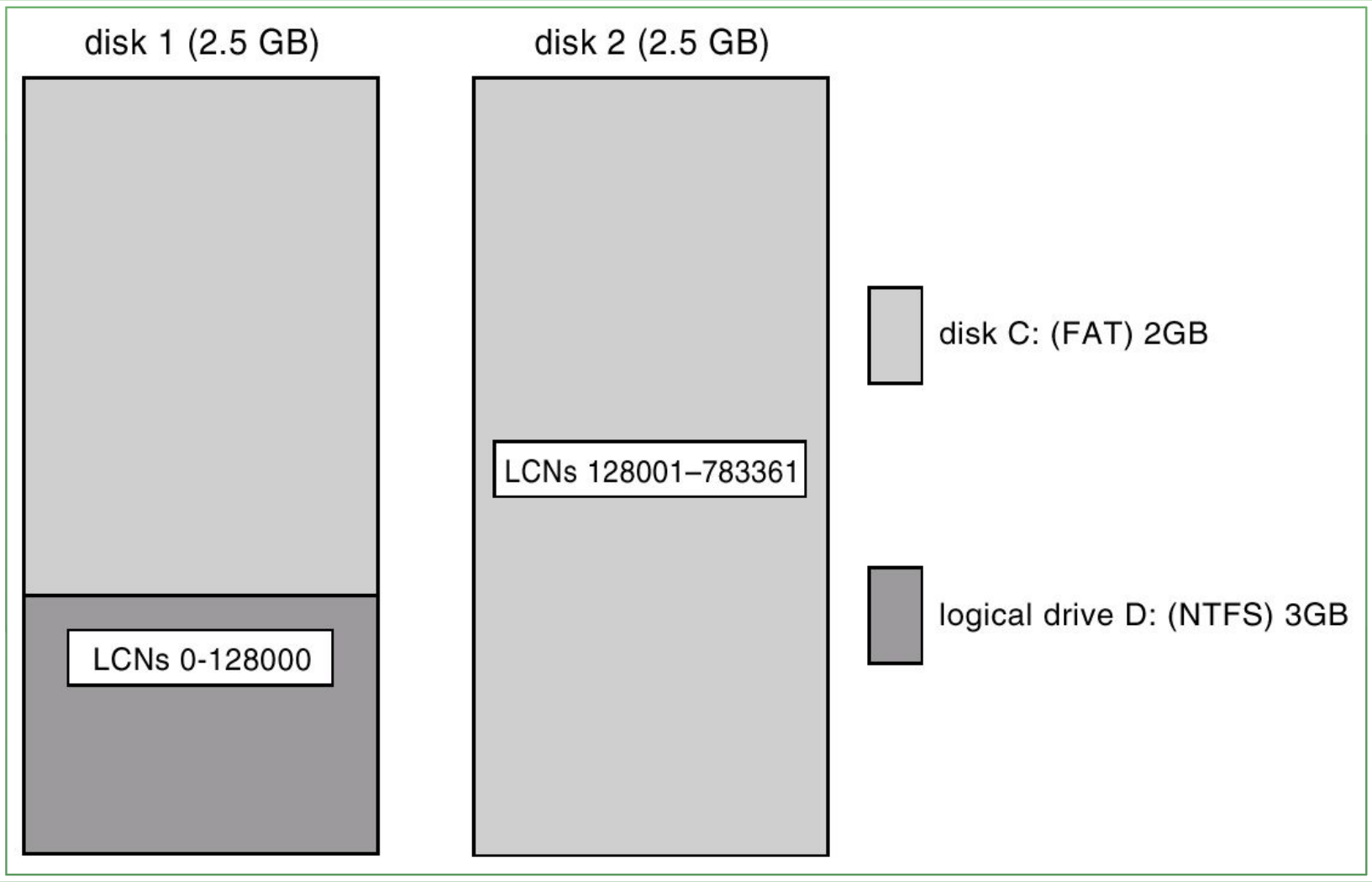
Файловая система - Безопасность

- **Безопасность тома NTFS реализована на основе объектной модели Windows 2000.**
- **Каждый файловый объект имеет дескриптор безопасности, хранящийся в записи MFT.**
- **Данный атрибут содержит маркер доступа владельца файла, а также список управления доступом, устанавливающий права каждого пользователя для доступа к данному файлу.**

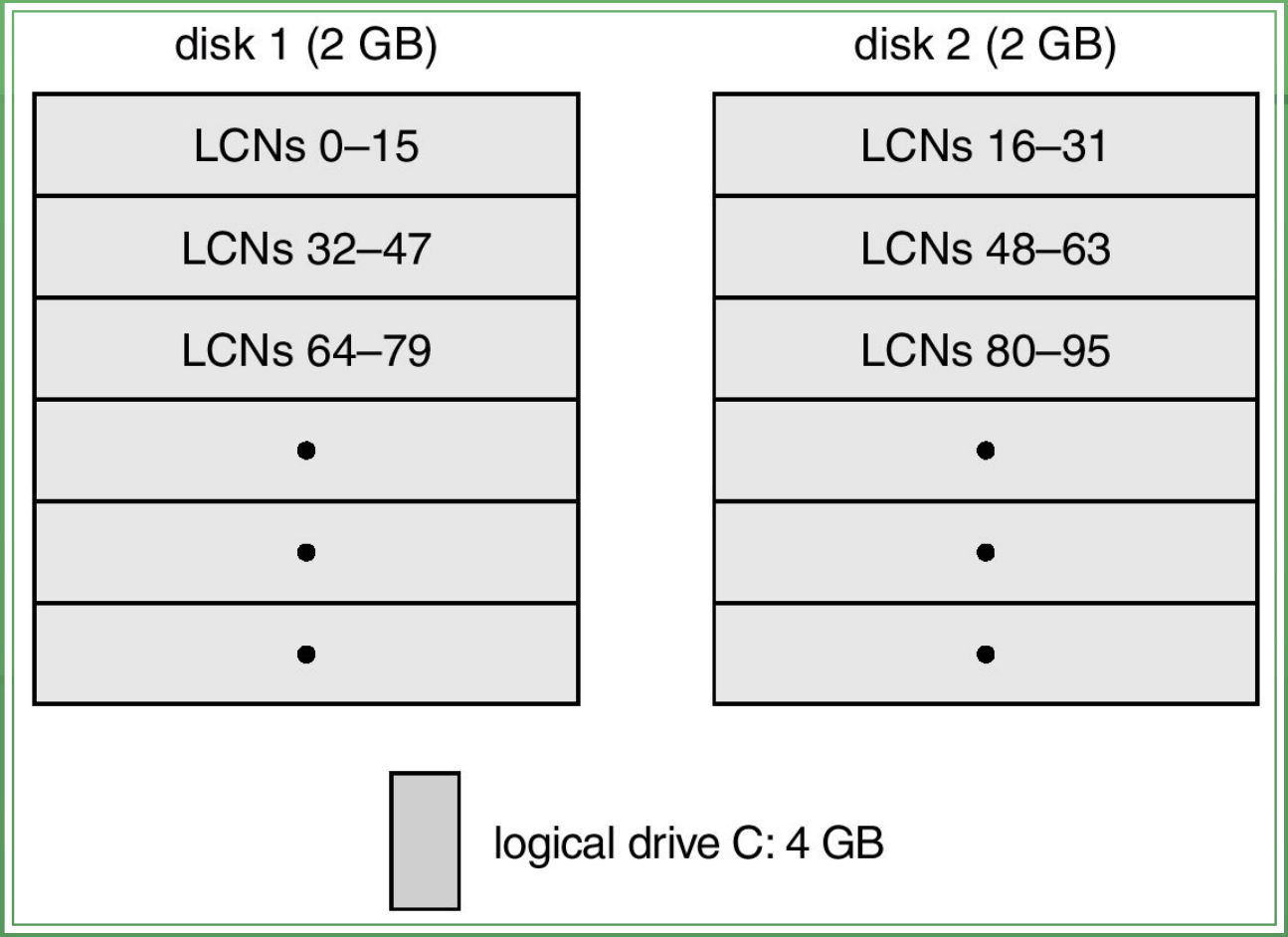
Управление томами и устойчивость к сбоям

- `FtDisk`, дисковый драйвер Windows 2000, устойчивый к сбоям, обеспечивает несколько способов объединения нескольких SCSI-дисков в один логический том.
  - Логически конкатенирует диски, образуя один логический том (набор дисков тома – volume set).
  - Обработка нескольких частей тома по принципу round-robin для формирования “полосатого множества” (stripe set), также называемого RAID уровня 0, или “disk striping”).
    - Вариант: *stripe set with parity,* или RAID уровня 5.
  - Зеркальное отображение дисков (Disk mirroring), или RAID уровня 1, - это надежная схема, использующая множество “зеркал” (*mirror set*) — две секции одного размера на разных частях диска с идентичным содержимым.
  - Для обработки запорченных дисковых секторов, `FtDisk` использует аппаратный метод, называемый предохранением секторов (*sector sparing*), а NTFS использует программный метод, называемый повторным отображением кластеров (*cluster remapping*).

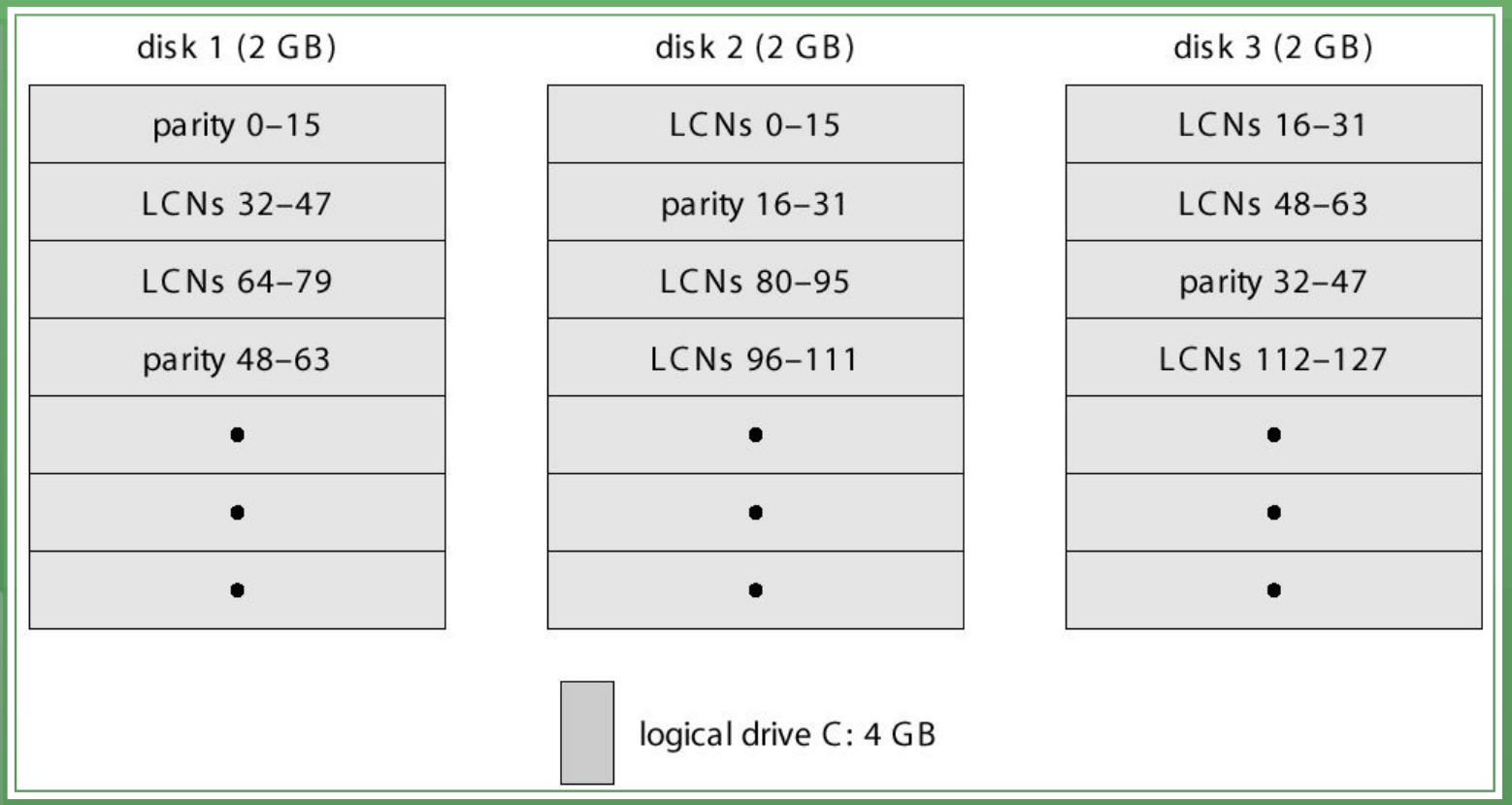
Том, размещаемый на двух дисках



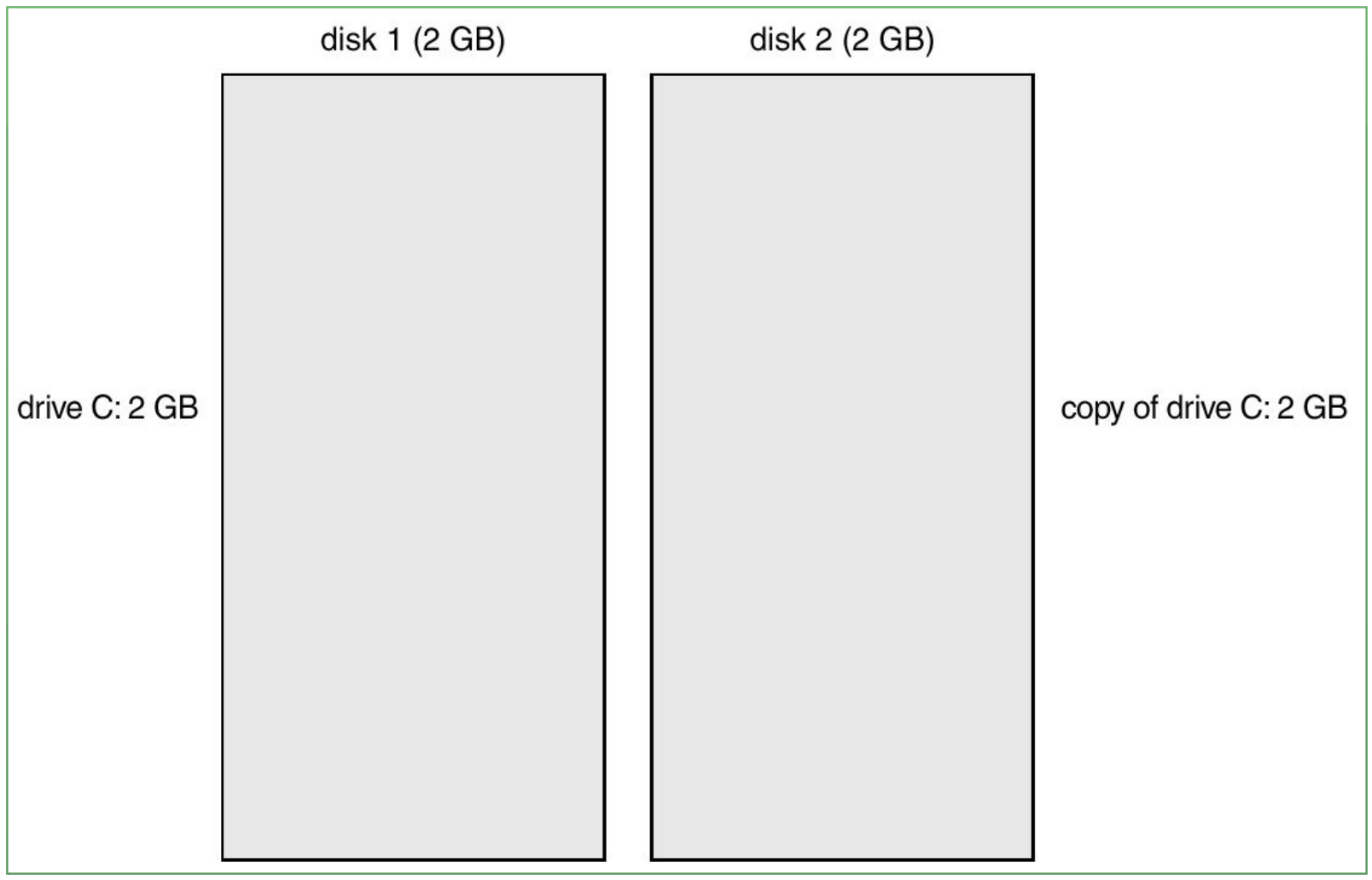
Stripe Set на двух дисках



Stripe Set With Parity на трех дисках



Mirror Set на двух дисках



Файловая система - сжатие

- **Для сжатия файла NTFS разделяет данный файла на модули сжатия (*compression units*) - блоки по 16 смежных кластерах.**
- **Для не смежно расположенных файлов NTFS использует другой метод экономии памяти.**
  - **Кластеры, содержащие только нули, фактически не хранятся на диске.**
  - **Вместо этого, в последовательности виртуальных номеров кластеров оставлены пропуски, информация о которых хранится в элементе MFT для данного файла.**

  При чтении из файла, если найден пропуск в нумерации виртуальных кластеров, NTFS просто заполняет нулями соответствующую часть буфера.

Файловые системы – точки повторного анализа **(reparse points)**

- **Точки повторного анализа при обращении вызывают генерацию кода ошибки. Они содержат информацию для менеджера ввода-вывода, какие действия выполнять дальше.**
- **Точки повторного анализа могут быть использованы для обеспечения функциональности монтирования, как в UNIX.**
- **Они могут быть также использованы для доступа к файлам, которые перенесены в отдельно размещаемую память.**

Сетевые средства

- Windows 2000 поддерживает как одноранговую связь, так и клиент-серверную связь в сетях; она также содержит средства для управления сетями.
- Для описания сетевых средств в Windows 2000, будем ссылаться на два внутренних интерфейса:
  - NDIS (Network Device Interface Specification) — отделяет сетевые адаптеры от транспортных протоколов, так чтобы каждый из них можно было изменить, не оказывая влияния на другой.
  - TDI (Transport Driver Interface) — обеспечивает, чтобы каждая из компонент уровня сеанса могла использовать любой транспортный механизм.
- Windows 2000 реализует транспортные протоколы как драйверы, который могут быть динамически добавлены к системе или удалены из нее.

Сетевые средства – Протоколы

- **Протокол “server message block” (SMB) используется для передачи через сеть запросов на ввод-вывод. Он имеет четыре типа сообщений:**
  - `Session control`
  - `File`
  - `Printer`
  - `Message`
- **Система Network basic Input/Output system (NetBIOS) - сетевой интерфейс с абстрагированием от аппаратуры. Используется для:**
  - **Установки логических имен в сети.**
  - **Установления логической последовательности сеансов между двумя логическими именами в сети.**
  - **Поддержки надежной передачи данных для сеанса с помощью запросов NetBIOS или *SMB*.**

Сетевые средства – протоколы (прод.)

- **NetBEUI (NetBIOS Extended User Interface): протокол по умолчанию для одноранговых сетей Windows 95 и Windows for Workgroups; используется для совместного использования ресурсов в подобных сетях.**
- **Windows 2000 использует протокол Интернета TCP/IP для соединения с различными ОС и аппаратными платформами.**
- **PPTP (Point-to-Point Tunneling Protocol) используется для коммуникации между модулями Remote Access Server, работающими на машинах под Windows 2000, соединенных через Интернет.**
- **Протокол NWLink соединяет сети NetBIOS и Novell NetWare.**

Сетевые средства – протоколы (прод.)

- **Протокол Data Link Control (DLC) используется для доступа к mainframe-компьютерам IBM и принтерам HP, непосредственно подсоединенным к сети.**
- **Системы на базе Windows 2000 могут взаимодействовать с компьютерами Macintosh с помощью протокола Apple Talk, если сервер в сети, работающий под Windows 2000, использует пакет Windows 2000 Services for Macintosh.**

Сетевые средства – механизмы распределенной обработки

- **Windows 2000 поддерживает распределенные приложения с помощью именованных NetBIOS, именованных конвейеров (pipes), mailslots, Windows Sockets, Remote Procedure Calls (RPC) и Network Dynamic Data Exchange (NetDDE).**
- **NetBIOS могут взаимодействовать через сеть, используя NetBEUI, NWLink или TCP/IP.**
- **Именованные конвейеры – это механизм передачи сообщений через сетевую коннекцию. Они именуются с использованием *uniform naming convention* (UNC).**
- **Mailslots – это механизм передачи сообщений без непосредственного использования коннекции, основанный на приложениях типа поиска компонент в сети.**
- **Winsock, API для реализации сокетов под Windows, - это интерфейс уровня сеанса, который обеспечивает стандартизованный интерфейс для многих транспортных протоколов, которые могут иметь различные схемы адресации.**

Механизмы распределенной обработки (прод.)

- **Механизм RPC в Windows 2000 следует широко используемому стандарту Distributed Computing Environment для RPC - сообщений, так что прогнаммы, использующие RPC для Windows 2000, имеют высокую степень переносимости.**
  - **RPC – сообщения посылаются с использованием NetBIOS, или Winsock в сетях TCP/IP, или именованные конвейеры в сетях LAN Manager.**
  - **Windows 2000 предоставляет Microsoft *Interface Definition Language* для описания имен, аргументов и результатов удаленных процедур.**

Сетевые средства – перенаправления и серверы

- **В Windows 2000 приложение может использовать API для ввода-вывода Windows 2000 для доступа к файлам удаленного компьютера, как к локальным файлам, при условии, что на удаленном компьютере исполняется MS-NET server.**
- **Перенаправитель (*redirector*) - это объект клиентской стороны, который пересылает запросы на ввод-вывод удаленных файлов. Эти запросы затем удовлетворяются сервером.**
- **Для повышения производительности и обеспечения безопасности, перенапрпавители и серверы выполняются в режиме ядра.**

Доступ к удаленному файлу

- Приложение вызывает менеджер ввода-вывода для запроса на открытие файла (предполагается, что имя файла – в стандартном формате UNC).
- Менеджер ввода-вывода конструирует пакет запроса на ввод-вывод.
- Менеджер ввода-вывода распознает, что это запрос к удаленному файлу, и вызывает специальный драйвер, называемый Multiple Universal Naming Convention Provider (MUP).
- MUP посылает пакет запроса на ввод-вывод асинхронно всем зарегистрированным перенаправителям.
- Перенаправитель, который может удовлетворить данный запрос, отвечает MUP.
  - Для того, чтобы не задавать тот же вопрос перенаправителям в будущем, MUP использует кэш-память для запоминания того, какой перенаправитель может работать с этим файлом.

Доступ к удаленному файлу (прод.)

- Перенаправитель посылает сетевой запрос удаленной системе.
- Сетевые драйверы удаленной системы получают запрос и передают его драйверу сервера.
- Драйвер сервера перепоручает этот запрос драйверу соответствующей файловой системы.
- Соответствующий драйвер вызывается для доступа к данным.
- Результаты возвращаются драйверу сервера, который пересылает данные перенаправителю, передавшему запрос.

Сетевые средства - Домены

- **Windows NT использует концепцию домена (domain) для управления глобальными правами доступа между группами.**
- **Домен – это группа машин, использующих Windows NT Server, которые используют одну и ту же политику безопасности и одну и ту же пользовательскую базу данных.**
- **Windows 2000 обеспечивает три модели установки доверительных связей.**
  - ***Однонаправленную - А доверяет В***
  - ***Двунаправленную, или транзитивную - А доверяет В, В доверяет С, следовательно, А, В, С доверяют друг другу***

    Перекрестную (Crosslink) – допускает, чтобы аутентификация миновала иерархию, с целью сокращения аутентификационного трафика.

Разрешение имен в сетях **TCP/IP**

- **В сети IP разрешение имен – это процесс преобразования имени компьютера в IP-адрес.**

  Например, `www.bell-labs.com` преобразуется в `135.104.1.14`

- **Windows 2000 обеспечивает несколько методов разрешения имен:**
  - **Windows Internet Name Service (WINS)**
  - **broadcast name resolution**
  - **domain name system (DNS)**
  - **a host file**
  - **an LMHOSTS file**

Разрешение имен (прод.)

- **WINS состоит из одного или более WINS – серверов, поддерживающих динамическую базу данных о связях между именами и IP-адресами, а также клаентское программное обеспечение для запросов к серверам.**
- **WINS использует Dynamic Host Configuration Protocol (DHCP), который автоматически обновляет адресные конфигурации в базе данных WINS без вмешательства пользователя или системного администратора.**

Программный интерфейс – Доступ к объектам ядра.

- Процесс получает доступ к объекту ядра, называемому `XXX`, путем вызова функции `CreateXXX` для получения (открытия) ссылки (*handle*) на `XXX`; ссылка уникальна для данного процесса.
- Ссылка может быть закрыта вызовом функции `CloseHandle`; система может удалить данный объект, если счетчик ссылок на него стал равным нулю.
- Windows 2000 три способа совместного использования объекта несколькими процессами.
  - Дочерний процесс наследует ссылку на объект.
  - Один процесс дает объекту имя при его создании, а другой процесс открывает данное имя.
  - `Функция DuplicateHandle`:
    - Если известна ссылка на процесс и значение ссылки, то другой процесс может получить ссылку на тот же объект.

Программный интерфейс – Управление процессами

- **Процесс запускается функцией `CreateProcess`, которая загружает все DLL, используемые процессом, и создает первичный поток (*primary thread*).**
- **Дополнительные потоки могут создаваться функцией `CreateThread`.**
- **Каждай DLL или exe-файл, загружаемые в адресное пространство процесса, идентифицируются ссылкой на экземпляр (*instance handle*).**

Управление процессами (прод.)

- Планирование в Win32 использует четыре класса приоритетов:
  - `IDLE_PRIORITY_CLASS` (уровень приоритетов 4)
  - `NORMAL_PRIORITY_CLASS` (уровень 8 — типичный для большинства процессов
  - `HIGH_PRIORITY_CLASS` (уровень 13)
  - `REALTIME_PRIORITY_CLASS` (уровень 24)
- Для обеспечения уровней произвыодительности, необходимых для интерактивных программ, Windows 2000 использует специальное правило планирования для процессов с `NORMAL_PRIORITY_CLASS`.

  Windows 2000 различает основной процесс (*foreground process),* который в данный момент выбран на экране, и фоновые процессы (*background processes*), которые не выбраны в данный момент.

  Когда процесс становится основным, Windows 2000 увеличивает его квант планирования в несколько раз, как правило – в три.

Управление процессами (прод.)

- Ядро динамически изменяет приоритет потока, в зависимости от того, связан ли он с вводом-выводом или с процессором.
- Для синхронизации доступа к общим объектам несколькими потоками ядро предоставляет синхронизирующие объекты, такие как семафоры и мьютексы (mutexes).
  - Кроме того, потоки могут синхронизироваться с использованием функций `WaitForSingleObject` или `WaitForMultipleObjects`.

    Другой метод синхронизации в Win32 API – критическая секция.

Управление процессами (прод.)

- **Волокно (fiber) – это код пользовательского режима, исполнение которого планируется по алгоритму, определенному пользователем.**
  - **В каждый момент времени разрешено исполняться только одному волокну, даже на многопроцессорной аппаратуре.**
  - **Windows 2000 поддерживает концепцию волокон с целью переноса унаследованных (legacy) UNIX-приложений, написанных на основе модели исполнения волокон.**

Программный интерфейс – Взаимодействие процессов

- **Win32 – приложения могут выполнять взаимодействие между процессами путем совместного использования разделяемых объектов ядра.**
- **Альтернативный способ взаимодействия процессов – передача сообщений; он наиболее популярен для Windows GUI - приложений.**
  - **Один поток посылает сообщение другому потоку или окну.**
  - **Вместе с сообщением поток может также посылать данные.**
- **Каждый поток Win32 имеет свою входную очередь, из которой данный поток получает сообщения.**
- **Это более надежно, чем общая входная очередь, применяемая в 16-битовой версии Windows, так как при использовании отдельных очередей одно подвисшее приложение не может блокировать другие.**

Программный интерфейс – Управление памятью

- **Виртуальная память:**
  - **`VirtualAlloc` резервирует или согласует для резервирования виртуальную память.**
  - **`VirtualFree` освобождает виртуальную память.**
  - **Эти функции дают возможность приложению запомнить виртуальный адрес, по которому была выделена виртуальная память.**
- **Приложение может использовать память, отобразив файл в свое адресное пространство.**
  - **Многоэтапный процесс.**
  - **Два процесса совместно используют память, отображая один и тот же файл в свою виртуальную память.**

Управление памятью (прод.)

- Куча (heap) в окружении Win32 – это область (region) зарезенвированного адресного пространства.
  - Процесс Win 32 создается с кучей, размер которой по умолчанию равен 1 MB.
  - Доступ к ней синхронизирован, с целью защиты структур данных, связанных с распределением памяти в куче, от разрушения при совместном доступе из нескольких потоков.
- Поскольку функции, которые основаны на глобальных или статических данных, одычно неправильно работают в многопоточном окружении, предоставлен механизм выделения глобальной, но связанной с конкретным потоком памяти (thread-local storage).

  Данный механизм предоставляет как статические, так и динамические методы выделения памяти, связанной с потоком.

Q & A

- Вопросы и ответы