MicroSoft-Windows

Практическая работа №3:
Архитектура ОС Microsoft Windows.

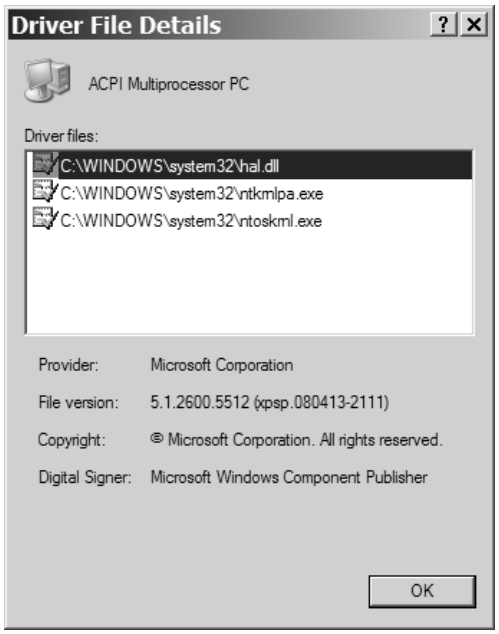
Цель: изучить общие аспекты архитектуры, ключевые компоненты Windows и принципы их взаимодействия.

ВОТ КАК-ТО ТАК ВСЕ ЭТО И РАБОТАЕТ

ЭКСПЕРИМЕНТ: просмотр основных системных файлов Windows, 47

Вы можете просмотреть основные системные файлы Windows следующим образом:

1. Откройте окно свойств системы, дважды щелкнув **System** (**Система**) в окне **Control Panel** (**Панель управления**) или щелкнув правой кнопкой мыши **My Computer** (**Мой компьютер**) на рабочем столе и выбрав из контекстного меню команду **Properties** (**Свойства**).
2. Перейдите на вкладку **Hardware** (**Оборудование**).
3. Щелкните кнопку **Device Manager** (**Диспетчер устройств**).
4. Раскройте объект **Computer** (**Компьютер**).
5. Дважды щелкните дочерний узел объекта **Computer**.
6. Откройте вкладку **Driver** (**Драйвер**).
7. Щелкните кнопку **Driver Details** (**Сведения о драйверах**).



Описание системных файлов Windows

Ntoskrnl.exe	Исполнительная система и ядро
Ntkrnlpa.exe (только 32-разрядные системы)	Исполнительная система и ядро с поддержкой механизма Physical Address Extension (PAE), позволяющего адресовать 64 Гб физической памяти
Hall.dll	Уровень абстрагирования от оборудования
Win32sk.sys	Часть подсистемы Windows, работающая и режиме ядра
Ntdll.dll	Внутренние функции поддержки и интерфейсы (stubs) диспетчера системных сервисов с функциями исполнительной системы
Kernel32.dll Advapi32.dll User32.dll Gdi32.dll	Основные DLL подсистемы Windows

Компоненты режима ядра Windows

Исполнительная система, содержащая базовые сервисы, которые обеспечивают управление памятью, процессами и потоками, защиту, ввод-вывод и взаимодействие между процессами.

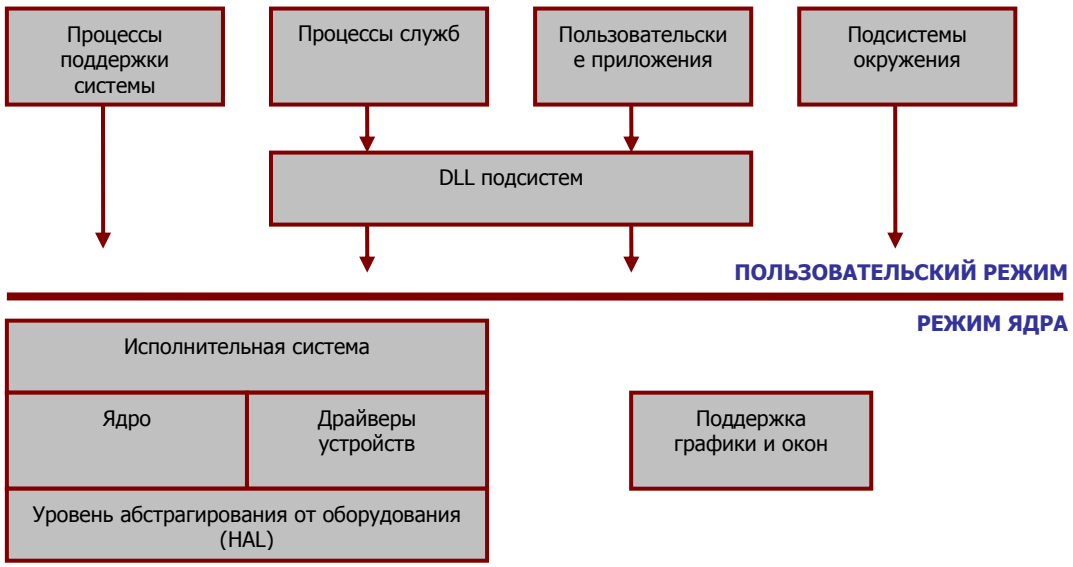
Ядро, содержащее низкоуровневые функции, которые поддерживают, например, планирование потоков, диспетчеризацию прерываний и исключений, а также синхронизацию при использовании нескольких процессоров.

Драйверы устройств, в состав которых входят драйверы аппаратных устройств, транслирующие пользовательские вызовы функций ввода-вывода в запросы, специфичные для конкретного устройства, а также сетевые драйверы и драйверы файловых систем.

Уровень абстрагирования от оборудования, изолирующий ядро, драйверы и исполнительную систему Windows от специфики оборудования на данной аппаратной платформе (например, от различий между материнскими платами)

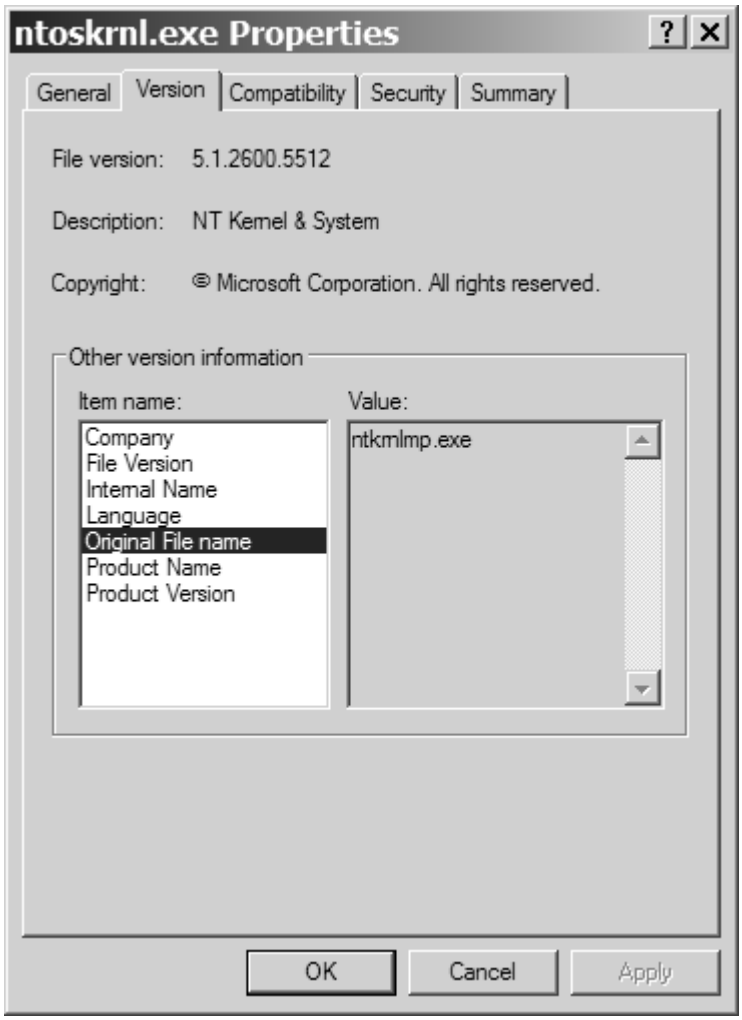
Подсистема поддержки окон и графики, реализующая функции GUI, более известные как Windows-функции модулей USER и GDI. Эти Функции обеспечивают поддержку окон, элементов управления пользовательского интерфейса и отрисовку графики.

Упрощенная схема архитектуры Windows



ЭКСПЕРИМЕНТ: определение текущей версии Ntoskrnl, 48

Чтобы узнать какая версии ядра запускается – одно- или многопроцессорная, отладочная или конечная запустите **Windows Explorer** (**Проводник**), и каталоге **\Windows\System32** щелкните ПКМ файл Ntoskrnl.exe и выберите из контекстного меню команду **Properties** (**Свойства**). Перейдите на вкладку **Version** (**Версия**) и выберите свойство **Original Filename** (**Исходное имя файла**). Если вы работаете с многопроцессорной версией, то увидите диалоговое окно, показанное ниже.

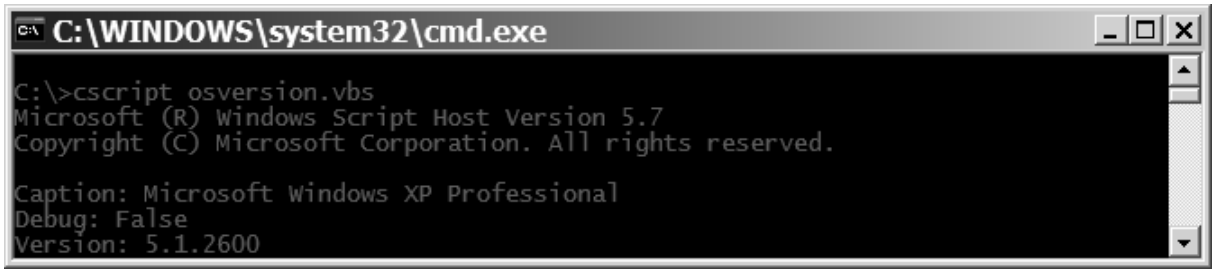


ЭКСПЕРИМЕНТ: определяем, является ли данная система проверочным выпуском, 52

Встроенной утилиты, которая позволяла бы увидеть, с каким выпуском вы имеете дело – проверочным или готовым, нет. Однако эта информация доступна через свойство «**Debug**» **WMI-класса** (**Windows Management Instrumentation**) **Win32_OperatingSystem**. Следующий сценарий на Visual Basic отображает содержимое этого свойства:

```
strComputer = "."
Set objWMIService = GetObject("winmgmts:" _
 & "{impersonationLevel=impersonate}!\\" & _
 strComputer & "\root\cimv2")
Set colOperatingSystems = objWMIService.ExecQuery _
 ("SELECT * FROM Win32_OperatingSystem")
For Each objOperatingSystem in colOperatingSystems
 Wscript.Echo "Caption: " & objOperatingSystem.Caption
 Wscript.Echo "Debug: " & objOperatingSystem.Debug
 Wscript.Echo "Version: " & objOperatingSystem.Version
Next
```

Чтобы опробовать его в действии, наберите код этого сценария и сохраните как файл. Вот пример вывода:



Эта система не является проверочным выпуском, так как флаг **Debug** равен **False**.

Специальная отладочная версия называется **проверочным выпуском**, доступным только подписчикам **MSDN** уровня **Professional**.

Проверочный выпуск предназначен разработчикам драйверов устройств, поскольку эта версия выполняет более строгую проверку ошибок при вызове функций режима ядра драйверами устройств или другим системным кодом. Например, если драйвер (или какой-то иной код режима ядра) неверно вызывает системную функцию, контролирующую передаваемые параметры, то при обнаружении этой проблемы система останавливается, предотвращая повреждение структур данных и возможный крах.

ЭКСПЕРИМЕНТ: просмотр базовых HAL, включенных в Windows, 73

Для просмотра **HAL**, включенных в **Windows**, откройте файл **Driver.cab** в соответствующем подкаталоге, специфичном для конкретной архитектуры. Прокрутите список до файлов, начинающихся с «**Hal**», и вы увидите файлы, перечисленные в таблице.

Список модулей HAL для x86 в \Windows\Driver\Cache\i386\Driver.cab

Hal.dll	Стандартные ПК
Halacpi.dll	ПК с ACPI (Advanced Configuration and Power Interface)
Halapic.dll	ПК с APIC (Advanced Programmable Interrupt Controller)
Halaacpi.dll	ПК с APIC и ACPI
Halmps.dll	Многопроцессорные ПК
Halmacpi.dll	Многопроцессорные ПК с ACPI
Halsp.dll	Compaq SystemPro (только для Windows XP)

Важная задача ядра – абстрагирование исполнительной системы и драйверов устройств от различий между аппаратными архитектурами, поддерживаемыми Windows (т.е. различий в обработке прерываний, диспетчеризации исключений и синхронизации между несколькими процессорами).

Архитектура ядра нацелена на максимальное обобщение кода. Ядро поддерживает набор переносимых между архитектурами интерфейсов.

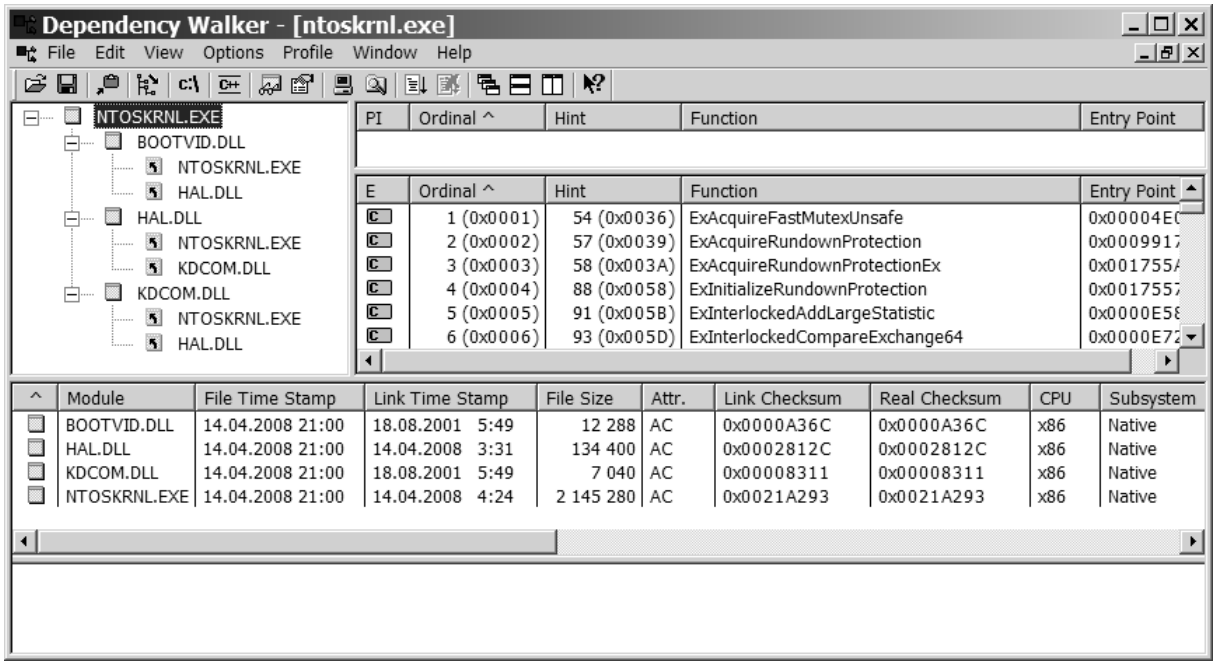
Одной из важнейших особенностей архитектуры **Windows** является переносимость между различными аппаратными платформами. Ключевой компонент, обеспечивающий переносимость – **уровень абстрагирования от оборудования** (hardware abstraction layer). **HAL** – это загружаемый модуль режима ядра (**Hal.dll**), предоставляющий низкоуровневый интерфейс с аппаратной платформой, на которой выполняется Windows. Он скрывает от ОС специфику конкретной аппаратной платформы, в том числе ее интерфейсов ввода-вывода, контроллеров прерываний и механизмов взаимодействия между процессорами, т.е. все функции, зависимые от архитектуры и от конкретной машины.

Когда внутренним компонентам **Windows** и драйверам устройств нужна платформенно-зависимая информация, они обращаются не к самому оборудованию, а к подпрограммам **HAL**, что и обеспечивает переносимость этой операционной системы. По этой причине подпрограммы **HAL** документированы в **Windows DDK**, где вы найдете более подробные сведения о HAL и о его использовании драйверами.

ЭКСПЕРИМЕНТ: просмотр зависимостей NTOSKRNL и HAL, 74

Вы можете просмотреть взаимосвязи образов ядра и **HAL**, изучив их таблицы импорта и экспорта с помощью утилиты **Dependency Walker**. Для исследования файла откройте его командой **Open** из меню **File**.

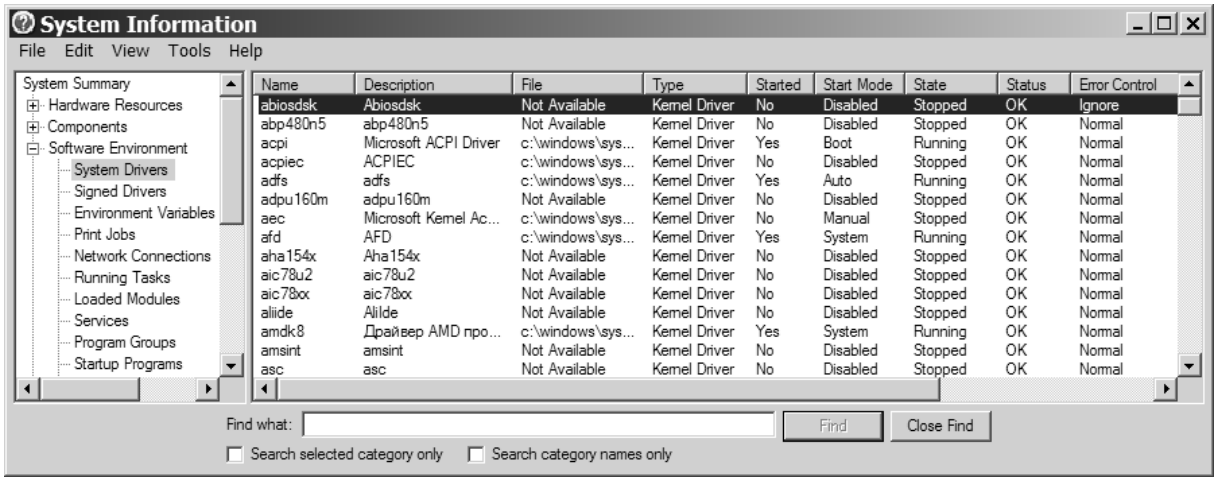
Вот пример вывода этой утилиты при просмотре зависимостей в **Ntoskrnl**.



Обратите внимание, что **Ntoskrnl** связан с **HAL**, который в свою очередь связан с **Ntoskrnl** (оба используют функции друг у друга). **Ntoskrnl** также связан с **Bootvid.dll**, видеодрайвером, используемым для вывода заставки при запуске **Windows**. В **Windows XP** и выше вы увидите в списке дополнительную **DLL**, **Kdcom.dll**. Она содержит код инфраструктуры отладчика ядра, который раньше был частью **Ntoskrnl.exe**.

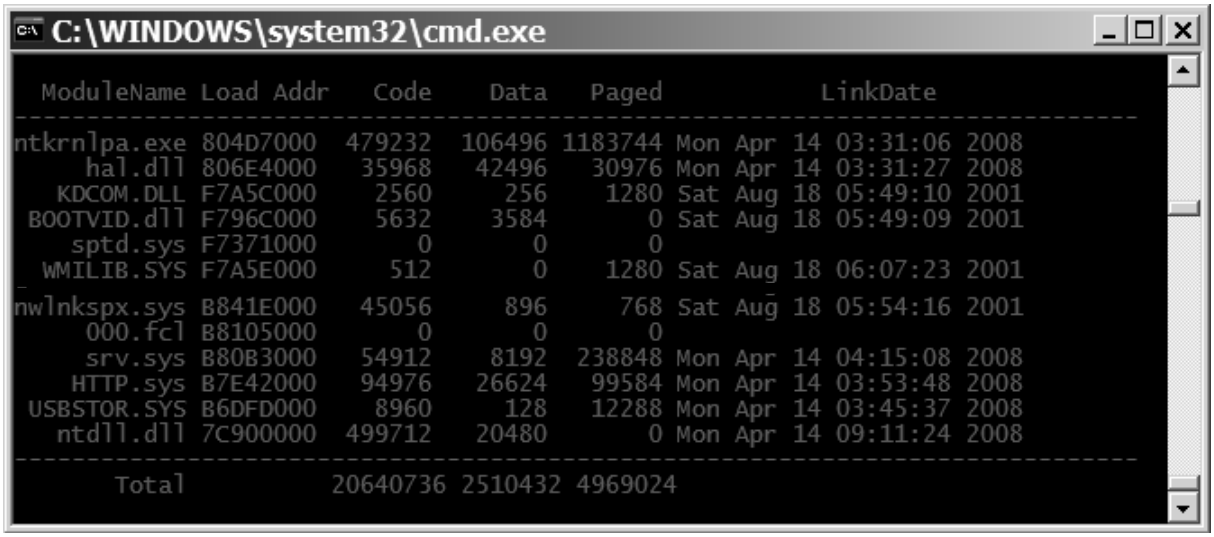
ЭКСПЕРИМЕНТ: просмотр установленных драйверов устройств, 77

Чтобы вывести список установленных драйверов, запустите **System Information** (**Сведения о системе**). Для этого выберите из меню **Start** (**Пуск**) команду **Programs** (**Программы**), затем **Accessories** (**Стандартные**), **System Tools** (**Системные утилиты**). В **System Information** (**Сведения о системе**) выберите **Software Environment** (**Программная среда**) и **Drivers** (**Драйверы**). Ниже приведен пример списка драйверов.



В этом окне выводится список драйверов, определенных в реестре, а также их тип и состояние – **Running** (**Работает**) или **Stopped** (**Остановлена**).

Список загруженных в текущий момент драйверов можно просмотреть и с помощью утилиты **Pstat** (**Pstat.exe** в **Windows XP Support Tools**). Ниже приведен листинг части выходной информации этой утилиты.

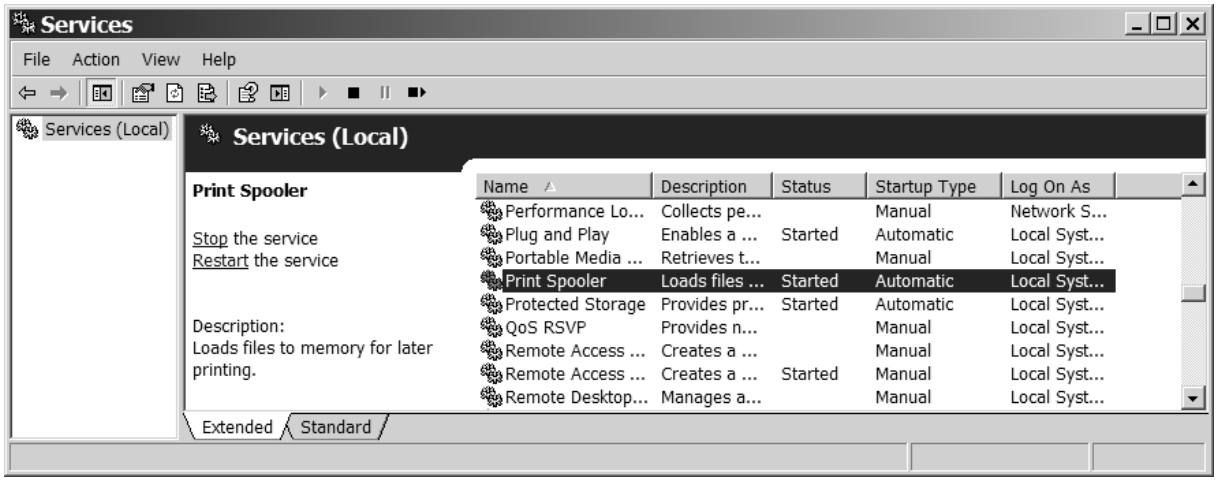


Утилита перечисляет все загруженные компоненты режима ядра (**Ntoskrnl**, **HAL** и драйверы устройств) и сообщает размеры разделов в каждом образе.

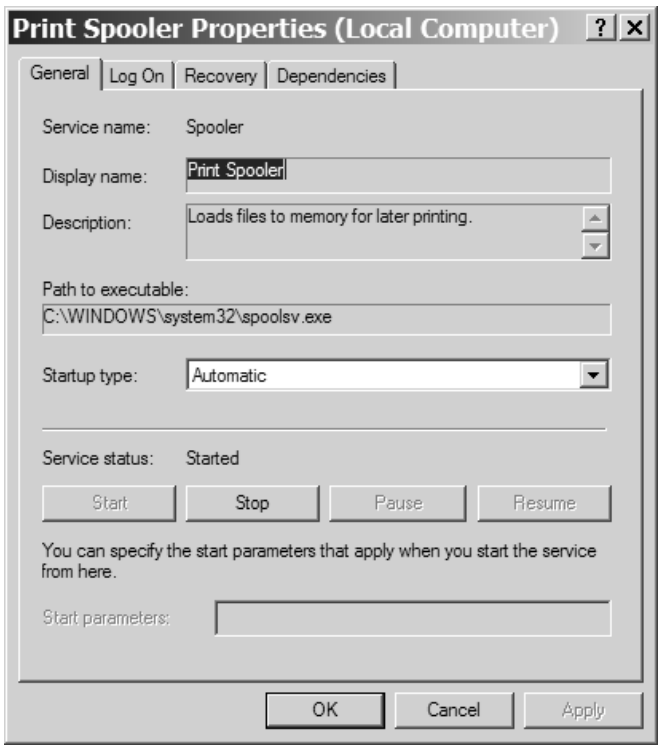
Pstat также выводит список загруженных драйверов после списка процессов и потоков в каждом процессе. Она показывает один вид очень важной информации: адрес загрузки модуля в системном пространстве. Этот адрес нужен для увязки выполняемых системных потоков с драйвером, в котором они существуют.

ЭКСПЕРИМЕНТ: вывод списка установленных сервисов, 89

Чтобы вывести список установленных сервисов (служб), дважды щелкните значок **Administrative Tools** (**Администрирование**) в окне **Control Panel** (**Панель управления**) и выберите **Services** (**Службы**). Вы должны увидеть что-нибудь в таком роде:



Для просмотра детальных сведений о сервисе щелкните правой кнопкой мыши имя сервиса и выберите команду **Properties** (**Свойства**). Ниже показан пример окна свойств для службы **Print Spooler** (**Диспетчер очереди печати**).



Обратите внимание, что поле **Path To Executable** (**Исполняемый файл**) указывает на программу, включающую данный сервис. Помните, что некоторые сервисы разделяют процессы с другими сервисами, поэтому число сервисов и используемых ими процессов не всегда находится в соотношении «один к одному».

КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Перечислите основные системные файлы Windows.
2. Какие компоненты Windows выполняются в режиме ядра, а какие в пользовательском режиме?
3. Каким способом можно узнать версию ядра Windows?
4. Что означает «проверочный выпуск» Windows?
5. Что означает аббревиатура HAL?
6. Что такое драйвер?
7. Каким способом можно просмотреть список загруженных драйверов?
8. Что такое сервис (служба)?
9. Приведите примеры служб, которые на ваш взгляд можно отключить на ПК не подключенному к сети и не имеющего принтера?