
ОПЕРАЦИОННЫЕ СИСТЕМЫ
WINDOWS ДЛЯ ХАКЕРОВ
И НЕ ТОЛЬКО...
НАСТОЛЬНАЯ КНИГА ПЕНТЕСТЕРА

Оглавление

Предисловие

Летом 2023 года (в то самое время, когда ныне покойный товарищ Пригожин организовывал свой поход на Москву, а доллар стоил меньше 80 рублей), я осознал, что, призывая практически в каждом видео (да я немножко видеоблогингом балуюсь, ссылка кликабельна) делать акцент на матчасти и базе, не сообщаю подписчикам в чём же конкретно, по моему мнению, заключается эта самая база.

Особенности архитектуры операционных систем Windows и Linux, сетевые сервисы, логи и системы хранения, нюансы разведки и закрепления, повышение привилегий, работа с дампами памяти, атаки, фильтрация трафика, маршрутизация, конфигурирование Firewall и базовое администрирование с использованием Powershell (будь он проклят).

И это лишь то немногое, что пришло мне в голову в тот момент (и осталось на память в виде корявого текста на обратной стороне чека из местной пятёрочки). Вернувшись из отпуска, данная задумка выкристаллизовалась в намерение сделать полноценный видеокурс для начинающих (и уже весьма опытных) пентестеров. Само-собой с акцентом на «белых» хакеров, как наиболее востребованных сегодня специалистов на рынке ИБ и ИТ в целом.

Примерно полгода ушло на то, чтобы собрать наброски из теоретических материалов, набрать команду высококлассных экспертов из разных направлений кибербезопасности (кодеры, БДшники, пентестеры веба, сетевики ~~и прочие прихлебатели~~), проработать с ними практическую часть курса, оформить это дело в виде классных сценариев и…осознать, насколько же это гигантский объём.

Даже по самым скромным прикидкам на запись и монтаж всех этих роликов ушло бы не меньше 10 лет. А это, на минуточку, примерно столько, сколько я в принципе веду свой канал на ютубе. И, разумеется, ни временных, ни уж тем более финансовых ресурсов (чтобы прожить то время, пока буду записывать курс, не отвлекаясь на основную работу) у меня пока нет и в обозримом будущем не появится.

Пару месяцев я дико прокрастинировал по этому поводу, оплакивая вложенные силы и деньги (консультационные услуги высококлассных экспертов в ИТ нынче стоят дороже эскортниц), попутно уходя с головой в другие, менее вдохновляющие проекты, продавая фрагменты наиболее удачных сценариев на кворке и фриланс.ру (так что не удивляйтесь, если вдруг где-то услышите\уже слышали нечто похожее).

В то время по вечерам я обычно придавался своему любимому хобби - чтению технической литературы. Собственно, именно эта активность в конечном итоге и привела меня к гениальной (во всяком случае очень хочется в это верить) идее, просто изложить наработанный материал в виде серии книг. Да, пусть это не совсем то, что мне изначально хотелось бы видеть. Комьюнити книголюбов значительно меньше аудитории на ютубе.

И соответственно донести свои мысли получится лишь до тех, кто за годы деградационных процессов в системы образования каким-то чудом ещё не забыл, как выглядят буквы. Однако, положа руку на сердце, само качество людей, не утративших понимания пользы от чтения значительно выше. Так что, как минимум, можно наверняка быть уверенным в том, что в руки «случайных прохожих» данная книга точно не попадёт.

И если ты сейчас читаешь данные строки, то, по сути, уже входишь в 2% избранных. Остальные 98%, если когда-нибудь и смогут ознакомиться с этой инфой, то потратят на её самостоятельное освоение и «набивание шишек» минимум лет 10-15 (и то в случае, если наставник попадётся толковый, а суровая жизнь не внесёт свои коррективы).

В данной книге я подробнейшим образом изложил свой собственный опыт и наработки экспертов из индустрии в части эксплуатации операционных систем с точки зрения тестирования на проникновения. И если бы я мог одним щелчком пальцев вернуться в прошлое и прочесть её, будучи студентом шараги или возрастным гуманитарием пожелавшим что-то изменить в жизни и пойти по пути кибербезопасности, я бы незамедлительно это сделал.

Поэтому я искренне, белой завистью вам завидую. Ведь ваша дорожная карта специалиста, после прочтения, будет содержать значительно меньше ухабов и ответвлений с тропинками «в никуда». По сути, это гладкий асфальт, по которому вам просто нужно проехать (возможно не один раз). Так что, если готовы, устраивайтесь по удобней, и будем уже заканчивать эту порядком затянувшуюся прелюдию. Погнали душнить!

Глава 1. Архитектура ОС Windows

Данная глава является наиболее важной с точки зрения базы, поэтому настоятельно рекомендую вам проработать её не «на отшибись» (как проходняковое чтиво), а вдумчиво (делая заметки в мобильнике или блокноте). Ибо всё, что вы узнаете в этой части книги, активно используется при проведении практических пентестов (тестирований на проникновение) в отношении операционной системы Windows.

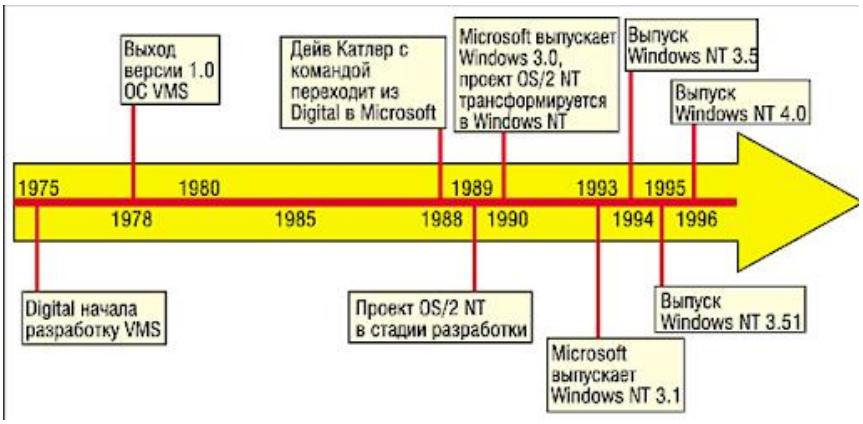
В результате изучения вы будете:

- понимать, как в принципе работает *WIndows*;
- знать разницу между редакциями;
- уметь пользоваться инструментами *Windows Sysinternals*;
- успешно взаимодействовать с реестром *Windows*;
- иметь представление о том, как работают службы в *Windows*.

История и развитие

Так что же такое этот *Windows*? Чтобы ответить на этот вопрос, давайте немного окунёмся в историю появления данной ОС (операционной системы).

Корни ОС *Windows* уходят далеко в 1975 год, когда началась разработка VMS 1.0 под *VAX-11/780 (virtual address extension*, он же «star», имевший на борту аж целых 4 Мб оперативной памяти) Дайвом Катлером в корпорации Digital Equipment.



История развития Windows

После разработки *VMS*, Дейв Катлер приступил к разработке ОС prism, которую, в итоге, ему так и не удалось довести до логического завершения (даже тогда люди ничего не умели доводить до конца). Потом Дейв Катлер со своей командой пришли в *Microsoft* с целью разработки «конкурента» **nix* — подобной серверной ОС для процессоров семейства *intel i860 (N-Ten)*.

История развития Windows NT

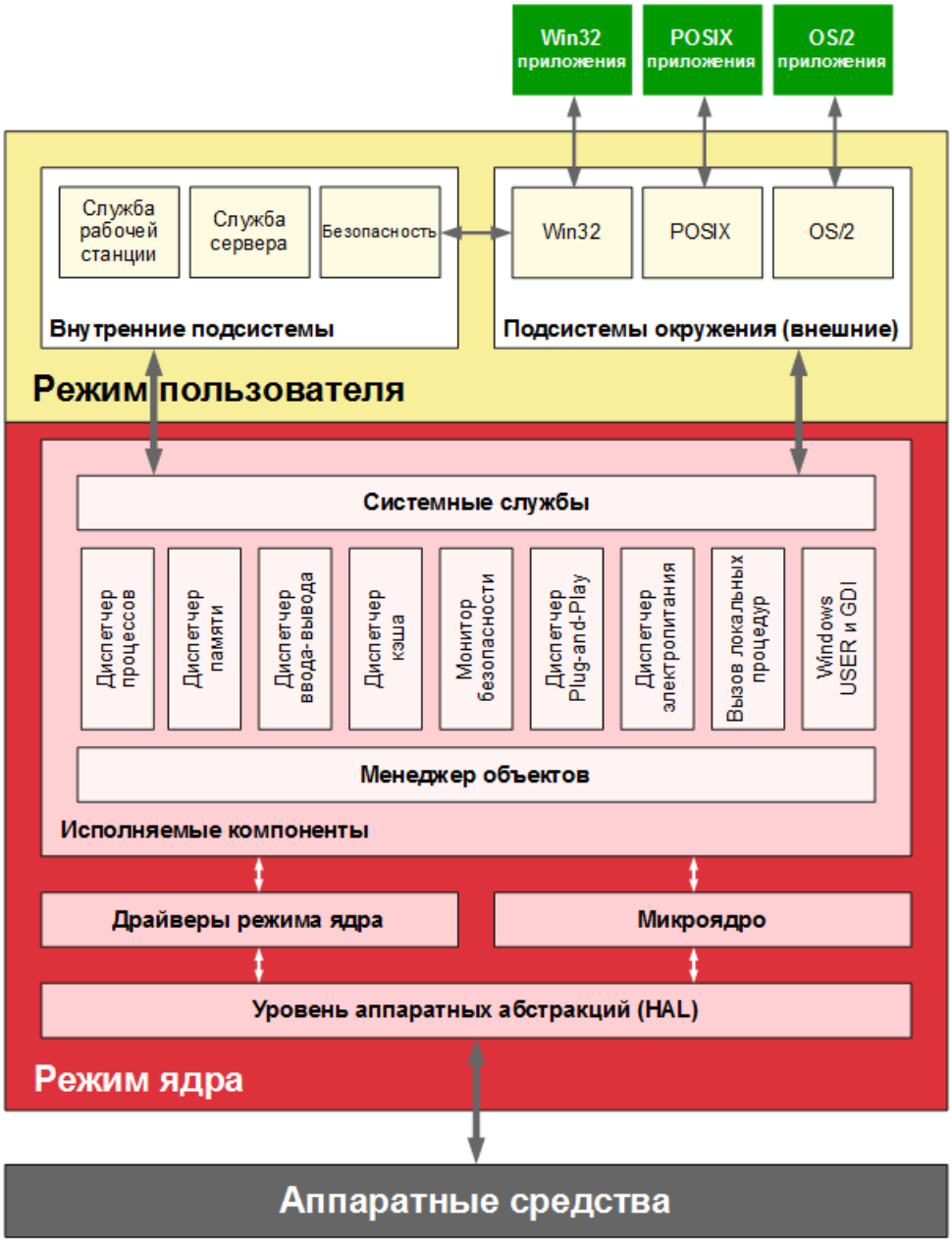
Product Name	Internal Version Number	Release Date
Windows NT 3.1	3.1	July 1993
Windows NT 3.5	3.5	September 1994
Windows NT 3.51	3.51	May 1995
Windows NT 4.0	4.0	July 1996
Windows 2000	5.0	December 1999
Windows XP	5.1	August 2001
Windows Server 2003	5.2	March 2003
Windows Vista	6 0 (Build 6000)	January 2007
Windows Server 2008	6.0 (Build 6001)	March 2008
Windows Server 2008 R2	6.1 (Build 61xx)	October 2009
Windows 7	6.1 (Build 61xx)	October 2009
Windows 8	6.2 (Build 62xx)	October 2012
Windows Server 2012	6.2 (Build 62xx)	October 2012
Windows Server 2012 R2	6.3 (Build 63xx)	June 2013
Windows 10	10.0	July 2015

Windows NT (New Technology) — это многопоточная операционная система на основе микроядра.

Термин «микроядро» означает, что компоненты ядра очень малы и обеспечивают только основные функции, такие как диспетчеризация потоков и обработка аппаратных исключений. Специфичный для оборудования код хранится на отдельном уровне, который называется уровнем аппаратной абстракции *(HAL). HAL* упрощает перенос операционной системы на новые архитектуры процессоров, такие как *IA*-64.

Код основной операционной системы работает в ***«привилегированном режиме процессора».*** Этот режим также известен как ***«защищенный режим»*** (при обращении к центральному процессору) или ***«режим ядра»*** (при обращении к процессу или потоку).

Защищённый режим обеспечивает прямой доступ к системной памяти и другому оборудованию. Приложения работают в непривилегированном режиме процессора, известном как пользовательский режим, и не имеют прямого доступа к оборудованию. Приложения должны использовать системные вызовы — API (интерфейс прикладного программирования) — в базовой операционной системе для выполнения таких задач, как чтение или запись в память или на экран.



Базовая архитектура Windows NT

Архитектура *Windows NT*

Подсистемы и службы *Windows NT*

Службы операционной системы хранятся в отдельных подсистемах, некоторые из которых работают в пользовательском режиме, а другие — в режиме ядра.

В *Windows NT* есть несколько подсистем режима ядра. Они обеспечивают встроенную функциональность *NT* для подсистем пользовательского режима через ntdll.dll (работающую в пользовательском режиме)

Подсистемы режима ядра составляют *Windows NT Executive* и состоят из следующих штук:

- Диспетчер объектов

Архитектура Windows NT не является строго объектно-ориентированной, но внутренние структуры, такие как сегменты разделяемой памяти, процессы и потоки, представлены как объекты, чтобы была возможность обеспечить единый метод обработки таких вещей, как контроль доступа.

*Диспетчер объектов создаёт, управляет и удаляет исполнительные объекты Windows NT***.** Объекты представлены в иерархическом пространстве имён во многом, как файловая система.

- Менеджер процессов:

Отвечает за создание и завершение процессов и потоков с использованием базовых функций ядра.

- Диспетчер виртуальной памяти:

Реализует виртуальную память, используемую для выделения частного адресного пространства каждому процессу.

- Менеджер ввода/вывода:

Предоставляет процессам систему ввода-вывода, не зависящую от устройства. Он отправляет запросы ввода-вывода соответствующему драйверу устройства.

- Средство вызова локальных процедур (*LPC*):

Реализует быструю и облегченную версию удаленного вызова процедур (RPC) для связи между компонентами внутри компьютера.

- Контрольный монитор безопасности (*SRM*):

Обеспечивает соблюдение политик доступа и аудита в системе. Контрольный монитор безопасности обеспечивает проверку доступа, проверку привилегий и создание контрольных сообщений во время выполнения как для процессов в режиме пользователя, так и в режиме ядра.

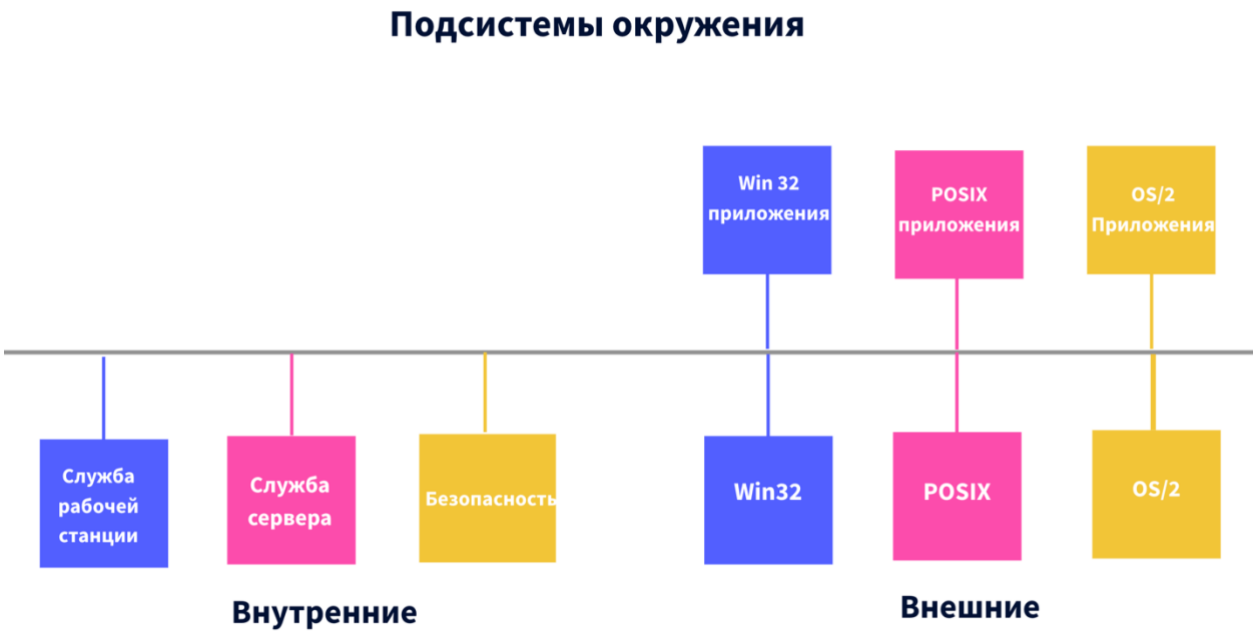
- Диспетчер окон и интерфейс графических устройств (*GDI)*

Эти компоненты составляют часть подсистемы Win32, работающую в режиме ядра. Они обрабатывают ввод данных пользователем и вывод на экран. Вся подсистема Win32 изначально работала в пользовательском режиме, однако из соображений производительности часть его была переведена в режим ядра, начиная с NT 4.0.

Подсистемы, работающие в пользовательском режиме, называются подсистемами окружения.

Существует три подсистемы окружения:

1. **Подсистема in32**: Часть подсистемы Win32 работает в пользовательском режиме. Подсистема Win32 является необходимой частью операционной системы и загружается при запуске. Подсистема состоит из библиотек **DLL Win32 API (kernel32.dll, user32.dll, gdi32.dll)** и процесса подсистемы **Win32 (csrss.exe)**.
2. **Подсистема *POSIX*** (интерфейс портативных операционных систем): обеспечивает поддержку приложений *POSIX.1*. Это дополнительный компонент, который загружается при необходимости. *прежде всего служит для запуска консольных приложений
3. **Подсистема *OS/2 1.x***: обеспечивает поддержку консольных приложений *OS/2 1.x*. Это дополнительный компонент, который загружается при необходимости.



Windows NT также обеспечивает поддержку приложений *MS-DOS* и 16-битной *Windows 3.x* через свою виртуальную машину DOS. Это не нативная подсистема *NT,* а просто приложение *Win*32, которое имитирует среду *DOS*.

Сеть в Windows NT

Первая версия *Windows NT (Windows NT 3.1)* была выпущена в 1993 году. Она позиционировалась, как преемник продуктов LAN Manager от Microsoft и IBM. Для взаимодействия и обратной совместимости с этими продуктами он должен был поддерживать некоторые установленные сетевые стандарты, такие как NetBIOS и SMB. Важно понимать, что это за протоколы и как они используются в *Windows NT*. Они по-прежнему обеспечивают основу для большинства сетевых коммуникаций Windows.

NetBIOS (сетевая базовая система ввода/вывода) — это стандарт для передачи данных между компьютерами по сети. Спецификация NetBIOS была разработана для *IBM* ещё в 1983 году, чтобы обеспечить сетевое взаимодействие между приложениями. NetBIOS предоставляет три основных сервиса:

- **Служба имён** — находит имена *NetBIOS* в сети.
- **Сессионный сервис** — обеспечивает соединение между двумя компьютерами.
- **Служба дейтаграмм** — обеспечивает канал связи между компьютерами без установления соединения.

Первые реализации *NetBIOS* не отделяли программный интерфейс от сетевого протокола. Позднее часть стандарта сетевого уровня получила название *NetBEUI* (расширенный пользовательский интерфейс *NetBIOS*).

Версия *NetBEUI* для *Windows NT* также называется *NBF (NetBIOS Frame*). В настоящее время *NetBIOS* может использовать другой транспорт, кроме немаршрутизируемого протокола NetBEUI, например *TCP/IP (NetBIOS* через *TCP/IP — NetBT*).

Server Message Block (SMB) — это стандарт для отправки команд и данных. Помните, что ***NetBIOS*** — это просто стандарт для поиска ресурсов и передачи битов. Для реального использования поверх *NetBIOS* требуется протокол более высокого уровня. Здесь на помощь приходит *SMB*. В основном он используется для совместного использования файлов и печати, но его также можно использовать для межпроцессорного взаимодействия (*IPC*) для связи с процессами в других системах.

SMB через *NetBIOS* использует порты *udp/137* (служба имен *NetBIOS*) и *udp*/138 (служба дейтаграмм *NetBIOS*) или tcp/139 (служба сеансов *NetBIOS). Windows* 2000 включает поддержку *SMB* без *NetBIOS* через *tcp/445*.

Сетевая архитектура *NT*

Теперь давайте рассмотрим, как устроена сетевая архитектура *Windows NT*.

Сетевая архитектура Windows NT

Диспетчер ввода-вывода в *NT Executive* отвечает за большую часть обработки ввода-вывода, включая дисковый и сетевой ввод-вывод.

Как и все подсистемы в *Executive*, *I/O Manager* предоставляет ряд *API*-интерфейсов процессам пользовательского режима. Эти API включают следующее:

- **Сокеты Windows (Winsock)**

Реализация в *Windows NT* широко используемого *Sockets API*. Приложения, использующие *Winsock*: *Internet Explorer*, *IIS*, *Telnet* и *FTP*.

- **Именованные каналы *SMB***

Односторонние или дуплексные каналы связи между сервером и одним или несколькими клиентами.

- **Почтовые ящики *SMB***

Простой механизм *IPC*, который можно использовать для отправки или получения небольших (менее 425 байт) широковещательных сообщений дейтаграмм.

- **Удалённый вызов процедур (*RPC*)**

Удаленный вызов процедур *Microsoft* (*MS-RPC*) предоставляет механизм для использования обычных вызовов функций для связи с процессами на другом компьютере.

Компоненты объектной модели распределенных компонентов (*DCOM*) используют *MS-RPC*.

Есть **два способа выполнения *RPC*-связи между двумя хостами**:

1. *MS-RPC* через *SMB* – использует каналы с именами *SMB* в качестве транспорта для вызовов *RPC*. Все административные инструменты, такие как *Server Manager*, *User Manager*, *Performance Monitor* и *Event Viewer*, используют *MS-RPC* через *SMB* для подключения к удаленным узлам. Домены *Windows NT* также полагаются на *MS-RPC* через *SMB*.
2. *MS-RPC* с использованием сокетов *Windows* – связь устанавливается с использованием динамически назначаемых портов (> 1023) и служб сопоставления портов *RPC tcp*/135 и *udp*/135. Этот метод *RPC* часто используется приложениями *DCOM*.

- **Драйверы файловой системы *SMB,*** есть два компонента, которые разрешают совместное использование файлов *SMB*:

SMB Redirector — это драйвер файловой системы, который взаимодействует с компонентом драйвера сервера *SMB* в удаленной системе.

SMB Server — драйвер файловой системы сервера и служба сервера работают для соединений, запрашиваемых *SMB Redirector* на стороне клиента, перенаправляя их соответствующему драйверу локальной файловой системы, например *NTFS*.

- ***API* интерфейса *NetBIOS***

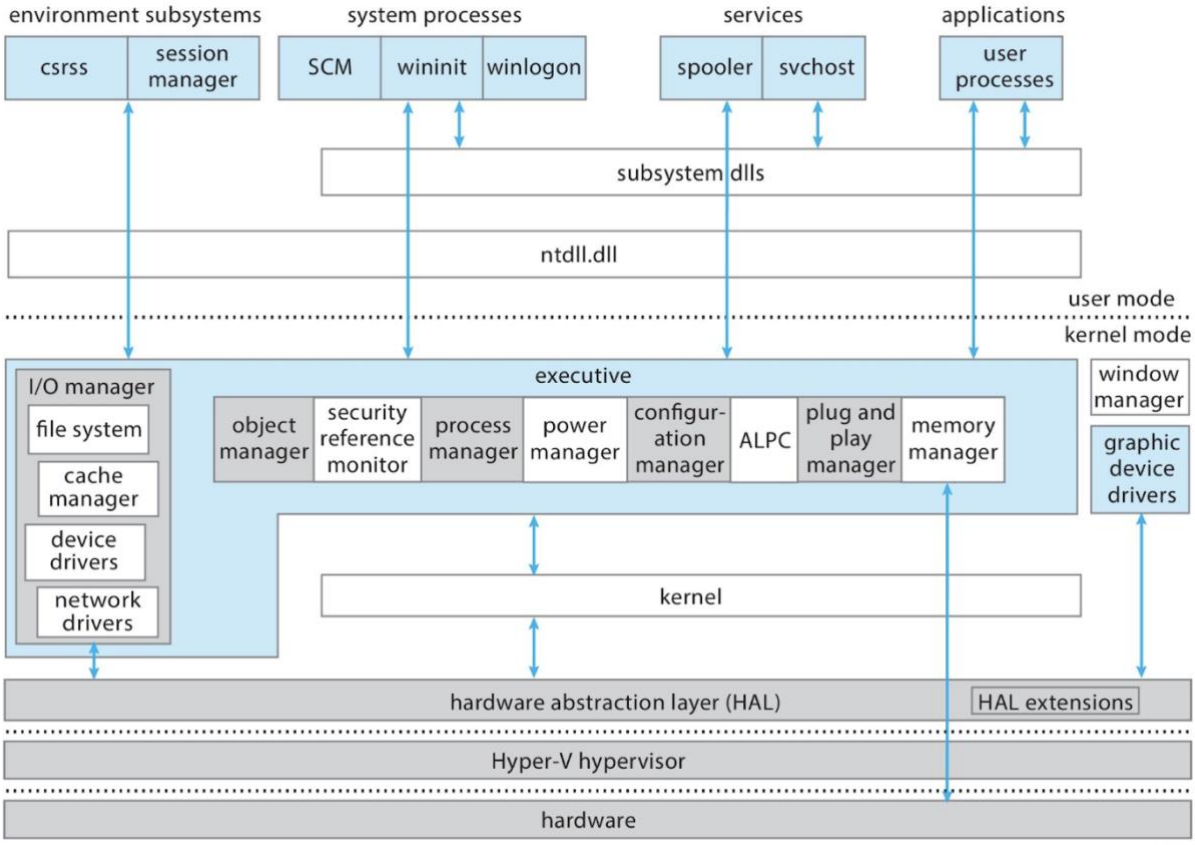
API интерфейса *NetBIOS* предоставляется в первую очередь для существующих приложений написанных на *legacy* (*IBM NetBIOS* 3.0) и нуждающихся в переносе в *API Win*32.

В сетевой архитектуре есть два пограничных уровня:

Интерфейс транспортного драйвера (*TDI*) — предоставляет разработчикам независимый от протокола сетевой *API* для сетевых служб. Разработчикам нужно всего лишь запрограммировать *TDI* для поддержки всех доступных сетевых протоколов.

Спецификация интерфейса сетевого драйвера (*NDIS*) — драйвер *NDIS* взаимодействует с сетевыми протоколами, обеспечивает единый интерфейс между драйверами протокола и драйверами устройств *NDIS* (берёт на себя функцию прокси между железом и низкоуровневыми сетевыми драйверами)

Архитектура Windows 10/11



Версии Windows 10 и *Windows* 11 входят в семейство систем построенных на *Windows NT*. Ядро тут (в плане архитектуры) практически не изменилось. Забавный факт, что, когда *Windows* 10 показали топ-менеджеру *Google*, он рассмеялся и сказал: «Так это же Windows XP, только плоская, как грудь моей бывшей».

А что же в ней изменилось из интересного, кроме графической составляющей? Подвезли поддержку *DirectX* 12, улучшили систему безопасности, изменили систему лицензирования, ну и, конечно, «самое важное» — добавили магазин приложений. На этом, собственно, всё.

Система лицензирования

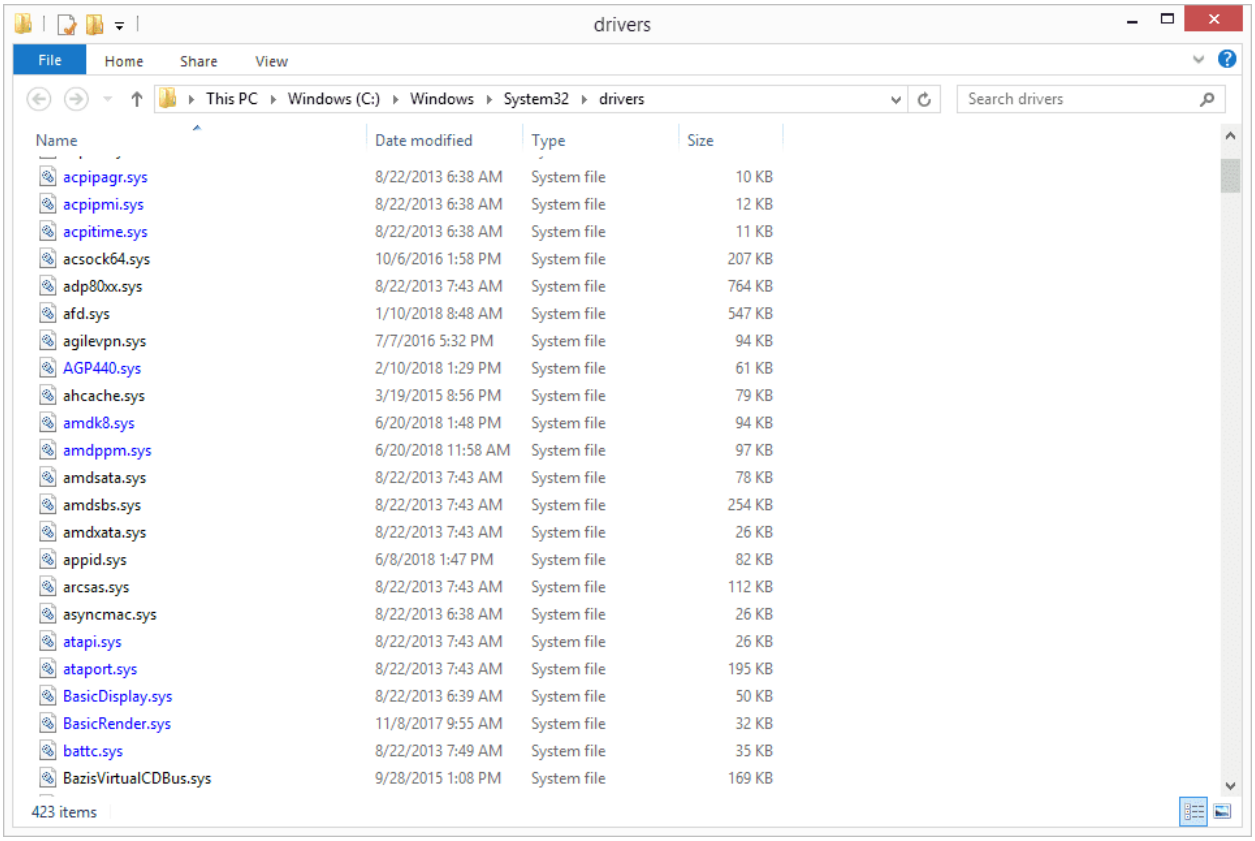
Раньше, если мы говорим о лицензионных версиях, то при установке обязательным условием был ввод ключа, теперь все стало немножечко проще. Можно установить систему без ключа, а активировать ее гораздо позже (никогда, например).

Также теперь при переустановке/обновлении система *Windows* 10/11 напрямую никак не связано с введенным ключом. Вместо этого система сама берет информацию о текущей установленной ОС, а после этого происходит миграция лицензии, во время этого процесса генерируется так называемая «цифровая лицензия», то есть система собирает информацию о железе и «подшивает» в ваше личное дело.

В дальнейшем, если железо меняется незначительно, то *Windows* после установки сама распознает ключ-продукта и не будет просить активацию. Начиная со сборки 1607 «цифровой ключ» жестко привязан к учетной записи *Microsoft*.

Взаимодействие ОС с физическими компонентами системы

Драйвер — это программное обеспечение, которое объясняет операционной системе, как видит и взаимодействует с железом система. Видеокарты, клавиатуры, мышки, принтеры, аудиокарты — всё это работает с помощью драйверов.



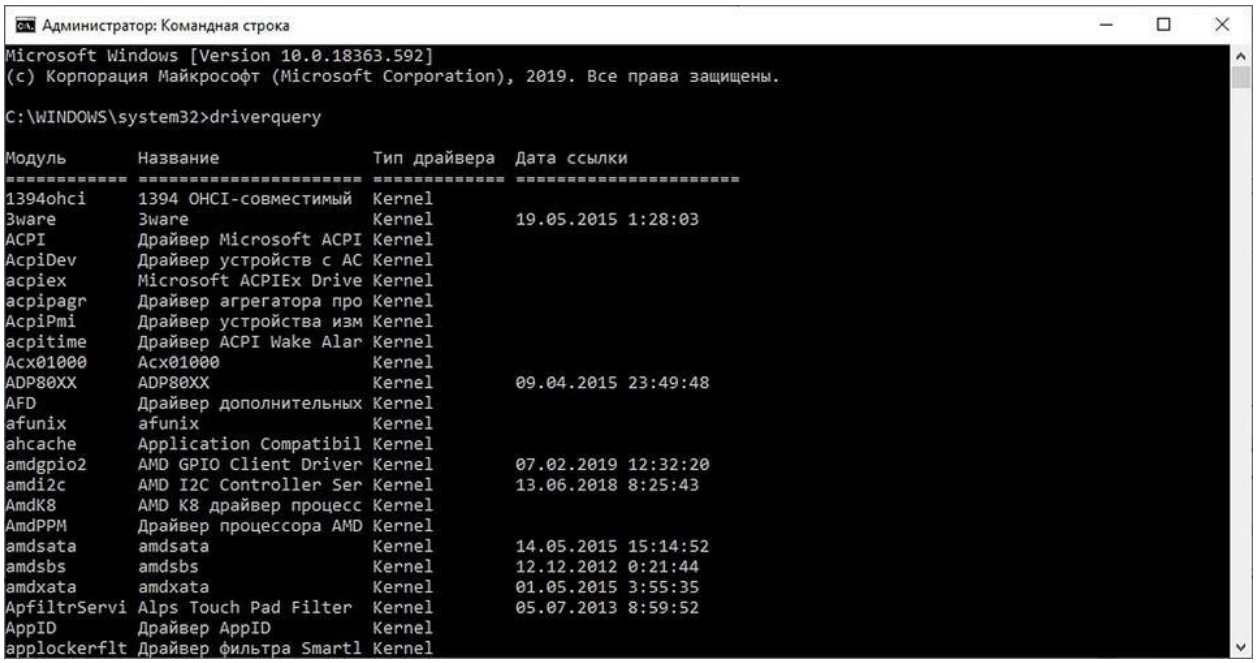
Скриншот каталога C:\Windows\System32\drivers

Все драйверы хранятся по пути диск, на котором стоит винда:\Windows\System32\drivers. Драйвер сообщает системе, как правильно взаимодействовать с оборудованием.

Как посмотреть список драйверов, установленных в системе?

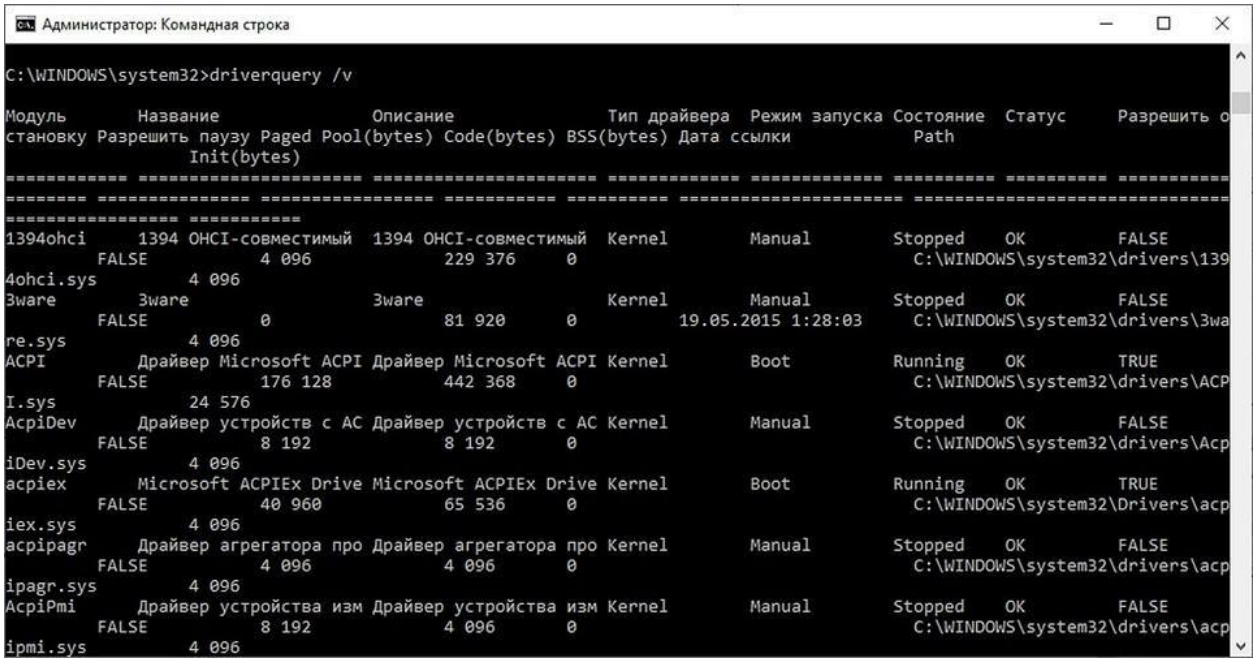
Есть несколько способов. Но мы же с вами «хакеры-мазафакеры, поэтому давайте, как понторезы, воспользуемся терминалом.

Открываем командную строку и пишем driverquery:

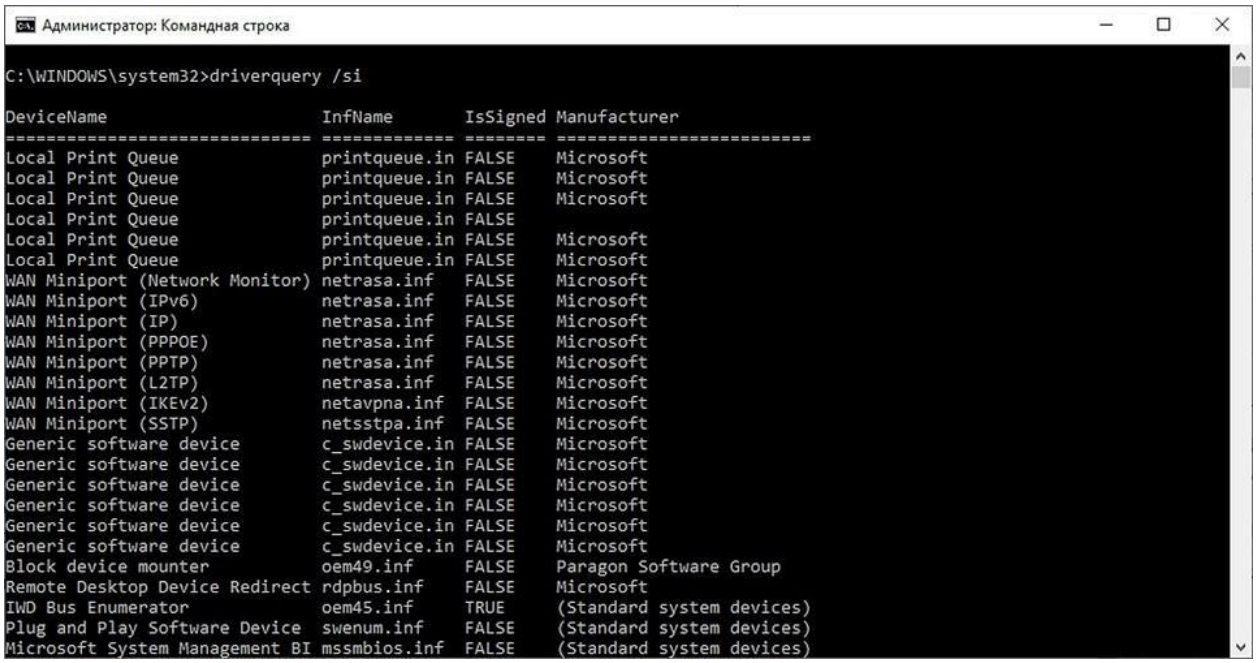


Таким образом можно посмотреть список всех установленных драйверов.

С помощью команды driverquery /v можно вывести подробную информацию про каждый драйвер: путь к драйверу, описание, режим запуска и прочее.

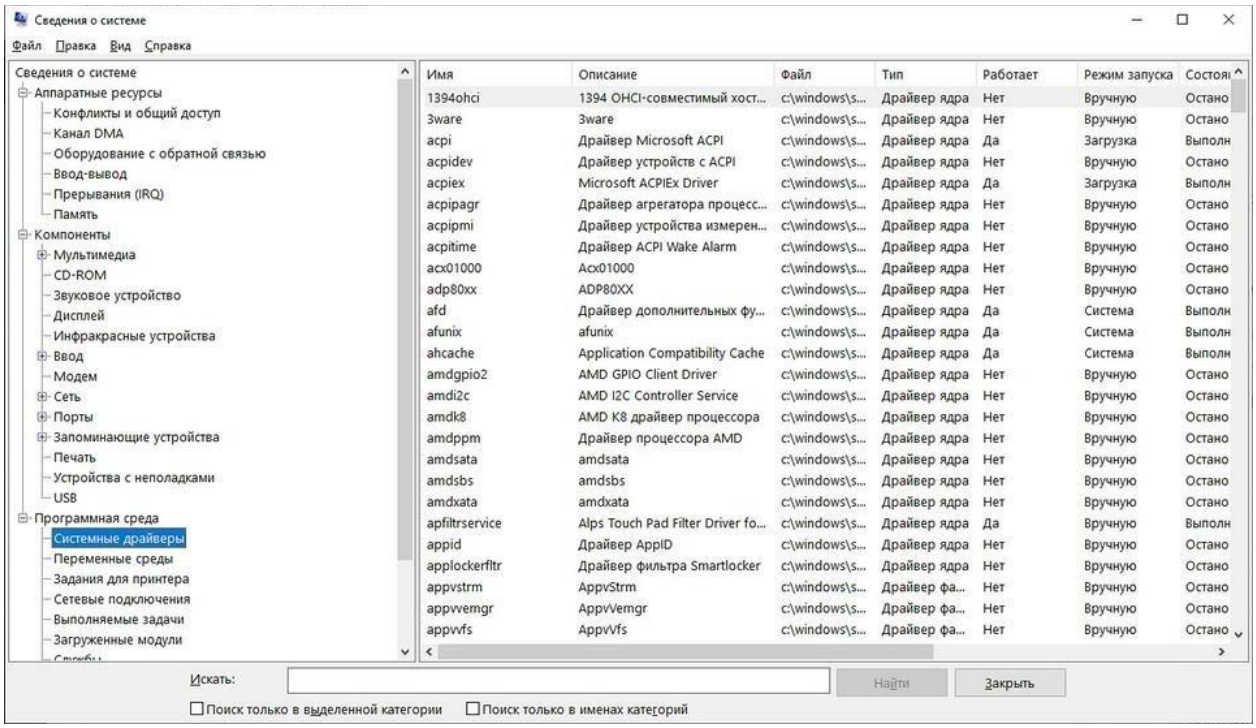


Ещё можно отдельно вывести список подписанных драйверов командой driverquery /si:



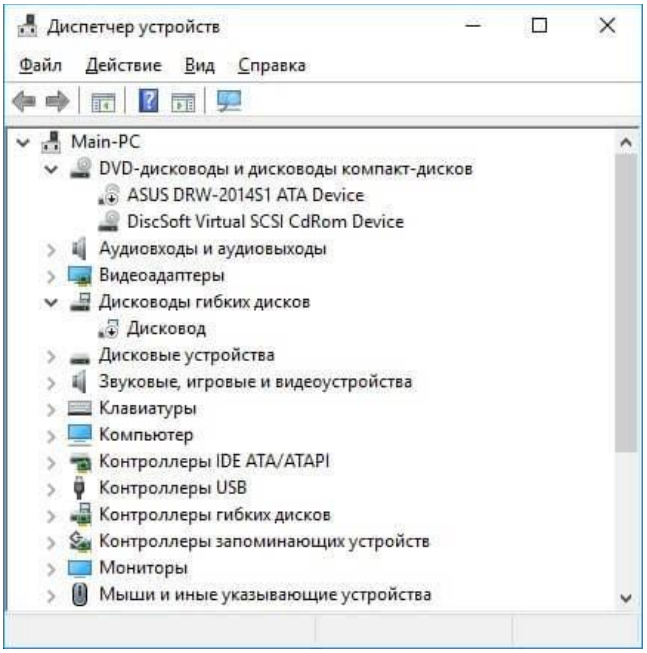
С командной строкой разобрались, давайте рассмотрим ещё один не менее «хакерский» способ просмотра данных, используя хоткеи (горячие, как бабулины пирожки, клавиши).

Нажимаем Win + R и пишем msinfo32, а после переходим в Сведения о системе -> Программная среда -> Системные драйверы:



Данный способ намного нагляднее. Но что, если нам нужно посмотреть конкретный драйвер устройства? Неужели нужно искать так и тратить время на сортировку? Конечно же нет, давайте рассмотрим для этой цели более простой способ.

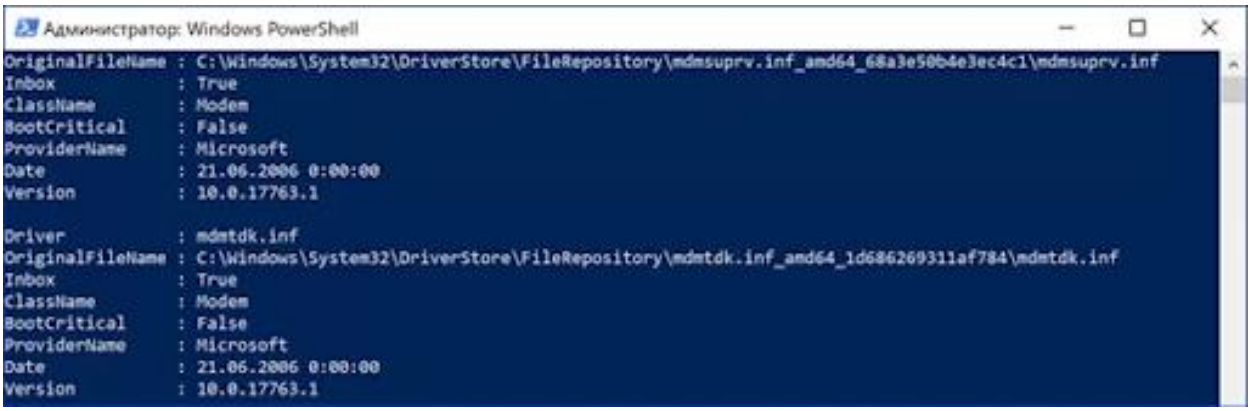
Открываем диспетчер устройств, перед нами открывается список всех устройств, подключённых, опознанных компьютером.



После этого мы можем щёлкнуть по интересующему нас устройству и посмотреть всю необходимую информацию.

Просмотр драйверов через PowerShell

Пойдём по уже отработанной схеме: нажимаем `Win + R` и пишем `PowerShell`, вводим команду `Get-WindowsDriver -online -all`:



Результат работы Get-WindowsDriver -online -all

Виртуальная память

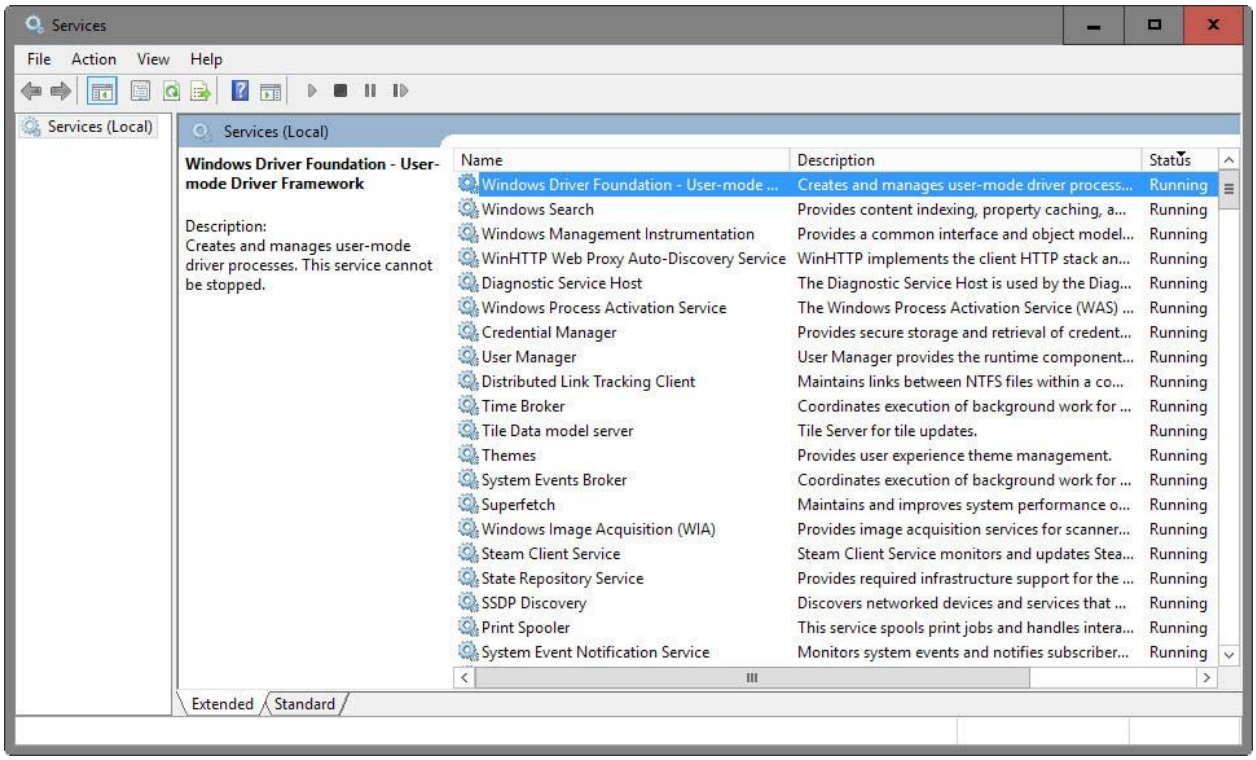
Если оперативная память закончится, то компьютер начнёт использовать виртуальную память *Windows*. Это специальное пространство, выделенное на жёстком диске, которое позволяет сделать временное расширение ОЗУ. Это даёт больше места для всех потребностей в многозадачности и позволяет компьютеру обрабатывать больше программ за раз.

В компьютере есть два типа памяти. *RAM*, *Hard Drive*, *SSD* — это физические аппаратные диски, на которых хранятся ваша операционная система, файлы и фотографии. На жёстком диске есть ещё одно место, которое называется **постоянным запоминающим устройством** (ПЗУ).

В ОЗУ хранятся данные для конкретной программы, которые быстро доступны и изменчивы по своей природе. Какие бы программы и приложения ни запускали, оперативная память создает доступное пространство для хранения временной информации. Можно сравнить его с рабочим местом для решения текущих задач.

Системные службы

Чтобы посмотреть весь список служб, которые работают у вас на ПК, нужно нажать уже знакомую нам комбинацию клавиш `Win + R` и написать `services.msc`:



Существует множество системных служб (или, как некоторые говорят, «сервисов»). Рассматривать их все нет никакого смысла, давайте поговорим про самые важные и распространённые.

- Файл `Svchost.exe` — это основной процесс, который управляет всеми службами в ОС. Через него запускаются и работают все службы для взаимодействия с железом.
- Файл `Spoolsv.exe` — служба, отвечающая за отправку факсов и управление печатью. Полностью процесс называется *Spooler SubSystem App*.
- Процесс `csrss.exe` — отвечает за программы, выполняемые в режиме командной строки (консольные), загрузку `conhost.exe`, а также за процесс выключения.
- Служба `wuauserv` — отвечает за проверку наличия обновлений в системе и установку их.

Файловая система NTFS

Файловая система *NTFS*, также называемая **файловой системой новой технологии**, представляет собой процесс, который операционная система *Windows NT* использует для эффективного хранения, организации и поиска файлов на жестком диске.

NTFS была впервые представлена в 1993 году в составе *Windows NT 3.*1.

Преимущества *NTFS* в том, что по сравнению с другими аналогичными файловыми системами, такими как таблица размещения файлов (*FAT*) и высокопроизводительная файловая система (*HPFS*), *NTFS* ориентирована на:

- **Производительность**: *NTFS* допускает сжатие файлов, поэтому система может получить больше места на диске.
- **Управление безопасным доступом**: *NTFS* позволяет размещать разрешения для файлов и папок, чтобы была возможность ограничить доступ к критически важным данным.
- **Надёжность**: *NTFS* ориентирована на согласованность файловой системы, поэтому в случае какой-то неприятности (например, отключения питания или сбоя системы) присутствует возможность быстро восстановить данные.
- **Использование дискового пространства**: помимо сжатия файлов, *NTFS* также позволяет устанавливать дисковые квоты. Эта функция позволяет предприятиям иметь ещё больший контроль над пространством для хранения.
- **Ведение журнала файловой системы**: это означает, что можно легко вести журнал и проводить аудит файлов, добавленных, изменённых или удалённых на диске. Этот журнал называется **главной файловой таблицей** (*MFT*).

Как работает NTFS

Техническая разбивка *NTFS* выглядит следующим образом:

1. Жёсткий диск отформатирован.
2. Файл разбивается на разделы на жёстком диске.
3. В каждом разделе операционная система отслеживает каждый файл
4. Каждый файл распределяется и хранится в одном или нескольких кластерах или дисковых пространствах заранее определённого равномерного размера (на жёстком диске).
5. Размер каждого кластера будет от 512 байт до 64 килобайт.

При этом присутствует возможность контролировать размер кластера. За счёт этого:

- Эффективно используется дисковое пространство.
- Снижается количество обращений к диску, необходимых для доступа к файлу.

Недостатки NTFS

Основным недостатком *NTFS* является то, что её возможности недоступны для старых технологий. А поскольку *NTFS* предназначена для работы с операционной системой *Windows*, устройства, работающие с *Mac* или *Android*, не всегда совместимы.

Например, компьютеры *Mac OS* могут читать диски, отформатированные в *NTFS*, но их можно записать в *NTFS* только с помощью стороннего программного обеспечения. Такие медиаустройства, как *DVD*-плееры, телевизоры и цифровые камеры слишком стары для использования устройств хранения *NTFS*.

Сегодня *NTFS* чаще всего используется со следующими операционными системами *Microsoft*:

- Windows 11;
- Windows 10;
- Windows 8;
- Windows 7;
- Windows Vista;
- Windows XP;
- Windows 2000;
- Windows NT.

Однако можно использовать *NTFS* и с другими операционными системами, такими как *Linux* и *BSD* (моя слабость).

Управление доступом к файлам

Разрешения *NTFS* обеспечивают контроль доступа к файлам и папкам, контейнерам и объектам в общих системах, обычно в сетевых хранилищах (*NAS*).

Существует пять основных разрешений *NTFS*:

1. **Чтение**: позволяет пользователю или группе читать файл и просматривать его атрибуты, владельца и установленные разрешения.
2. **Запись**: позволяет пользователю или группе перезаписывать файл, изменять его атрибуты, просматривать владельца и просматривать установленные разрешения.
3. **Чтение и выполнение**: позволяет пользователю или группе запускать и выполнять приложение, а также выполнять все действия, предусмотренные разрешением на чтение.
4. **Изменить**: позволяет пользователю или группе изменять и удалять файл, а также выполнять все действия, предусмотренные разрешениями на чтение, запись и чтение, и выполнение.
5. **Полный доступ**: позволяет пользователю или группе изменять набор разрешений для файла, становиться владельцем файла и выполнять действия, предусмотренные всеми другими разрешениями.

Системные процессы

Межпроцессное взаимодействие или IPC

Межпроцессное взаимодействие или *IPC*, как следует из названия, используется для обмена данными между двумя приложениями или процессами. Процессы могут находиться на одном компьютере или где-то ещё в сети. Операционная система *Windows* поддерживает различные методы *IPC*, а именно:

- **Буфер обмена**: метод обмена слабосвязанными данными. Когда пользователь использует команду копирования или вырезания в любом приложении, скопированные данные сохраняются в буфере обмена *Windows* (временное хранилище). Другое приложение может получить доступ к данным из буфера обмена.
- **COM**: *Component Object Model* предлагает платформу для взаимодействия между процессами по шаблону сервера и клиента. *COM*-сервер может быть локальным или внутрипроцессорным сервером. Также может быть несколько клиентов *COM*, которые взаимодействуют с сервером и обмениваются данными.
- **Копирование данных**: *Windows* предоставляет сообщение, например WM_COPYDATA, которое позволяет процессу обмениваться данными с другим процессом. Его можно использовать с *API SendMessage* для *win32*, а COPYDATASTRUCT используется в качестве параметра. Это сообщение используется только в случае копирования на локальном компьютере.
- **DDE**: динамический обмен данными — это протокол, который содержит набор рекомендаций и правил для передачи данных между процессами. Процесс может использовать *API SendMessage* с сообщением *WM_DDE_INITIATE* или *WM_DDE_ACK*, отправленным в ответ на сообщение *WM_DDE_INITIATE*.
- **Отображение файлов**: отображение файлов, быстрый механизм связи между процессами, который даёт эффективный способ использования содержимого файла в виртуальной памяти или путём доступа к совместному использованию памяти. В этом случае данные файла обрабатываются как часть адресного пространства процесса, так что процесс может легко получить доступ к адресу содержимого. Любой другой процесс, имеющий доступ к общей памяти, должен реализовать синхронизацию, чтобы снизить риски повреждения данных. Сопоставление файлов выполняется в той же системе/машине и недоступно для сетевых процессов.
- **Каналы** (*Pipes*): каналы могут использоваться как в качестве однонаправленного, так и двунаправленного механизма обмена данными. *Windows* поддерживает два типа каналов: *именованный* канал и *анонимный* канал. Анонимные каналы могут использоваться только в одной сети или между связанными процессами, в то время как именованные каналы могут использоваться в сети в разных процессах. Именованный канал можно рассматривать как очередь *FIFO*, в которой один конец действует как сервер, а другой как клиент.
- **Сокеты**: представляют собой эффективный способ отправки и получения данных по сети и на локальном компьютере. Он использует несколько протоколов, таких

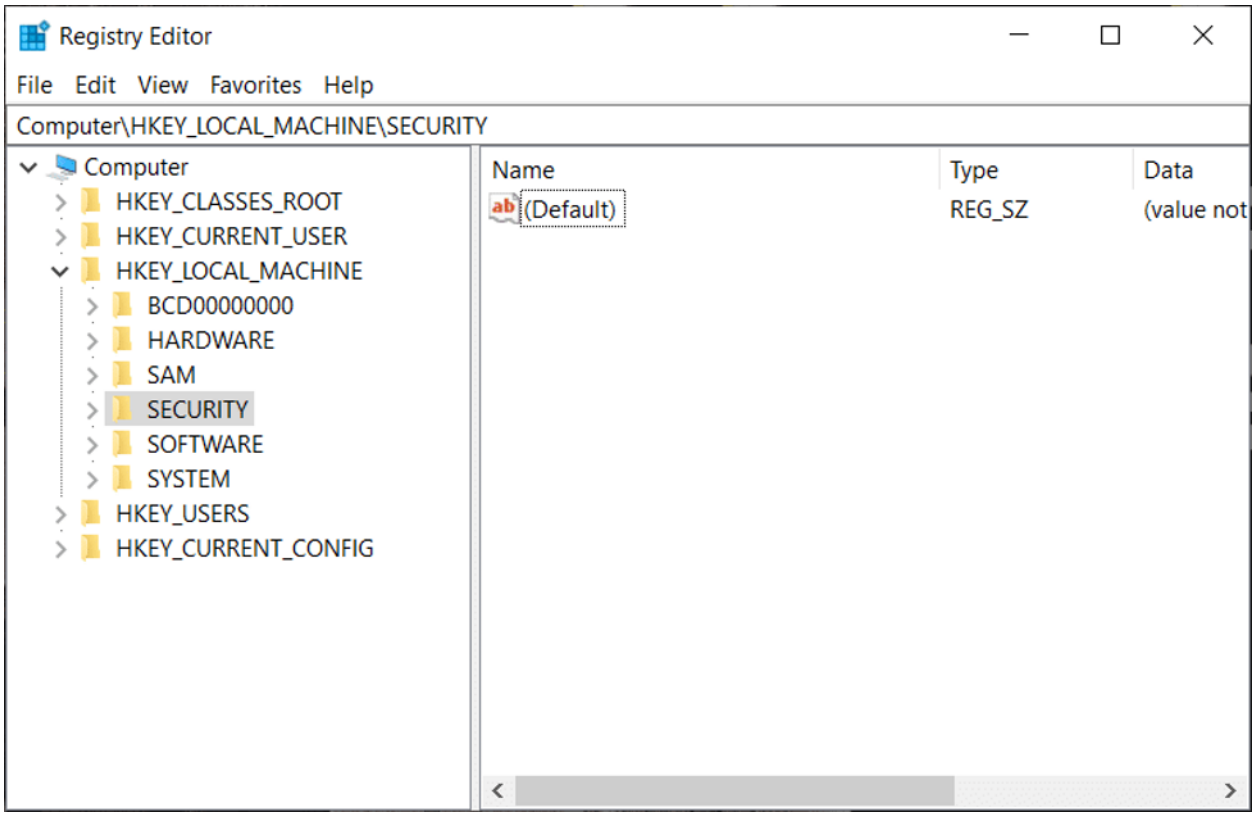
как *TCP/IP* и *UDP*. Он формируется из комбинации *IP*-адреса и адреса порта, по которому могут передаваться данные.

- **RPC**: удалённый вызов процедур обеспечивает способ связи по сети, чтобы процесс мог вызывать функцию в другом процессе. *RPC* поддерживает тесную связь между клиентом и сервером с высокой производительностью.

Реестр Windows

Реестр *Windows* — это набор конфигураций, значений и свойств приложений *Windows*, а также операционной системы, который организован и хранится в иерархическом порядке в едином репозитории.

Всякий раз, когда в системе *Windows* устанавливается новая программа, в реестре делается запись с такими атрибутами, как размер, версия, расположение в хранилище.



Ветка реестра HKEY_LOCAL_MACHINE\SECURITY

Как работает реестр Windows

Реестр *Windows* — это действительно «сердце» *Windows*. Это единственная операционная система, которая использует такой подход центрального реестра. Если бы мы визуализировали, каждая часть операционной системы должна взаимодействовать с реестром *Windows* прямо от загрузки до переименования имени файла.

Проще говоря, это просто база данных, аналогичная базе каталога библиотечных карточек, где записи в реестре подобны стопке карточек, хранящейся в каталоге карточек читателей.

Ключ реестра будет картой, а значение реестра будет важной информацией, записанной на этой карте. Операционная система *Windows* использует реестр для хранения большого количества информации, которая используется для контроля и управления системой и программным обеспечением. Это может быть что угодно: от информации об оборудовании ПК до пользовательских настроек и типов файлов. Практически любая конфигурация системы *Windows* включает редактирование реестра.

История реестра Windows

В начальных версиях *Windows* разработчикам приложений приходилось включать отдельное расширение файла .ini вместе с исполняемым файлом. Этот файл .ini содержал все настройки, свойства и конфигурацию, необходимые для правильной работы программы. Однако это оказалось очень неэффективным из-за избыточности определённой информации, а также представляло угрозу безопасности. В результате возникла очевидная необходимость в новой реализации стандартизированных, централизованных и безопасных технологий.

С появлением Windows 3.1 простая версия этого требования была удовлетворена с помощью центральной базы данных, общей для всех приложений и системы, под названием Windows Registry.

Однако этот инструмент был очень ограничен, поскольку приложения могли хранить только определённую информацию о конфигурации исполняемого файла. С годами *Windows 95* и *Windows NT* продолжали развиваться на этой основе, представив централизацию как основную функцию в новой версии реестра *Windows*.

Тем не менее, хранение информации в реестре *Windows* — опция для разработчиков программного обеспечения. Таким образом, если разработчику достаточно *portable*-версии приложения (для запуска которого не нужна установка), то ему не требуется добавлять информацию в реестр.

Актуальность и преимущество

Windows — единственная операционная система, которая использует такой подход центрального реестра.

Все другие операционные системы, такие как *iOS*, *macOS*, *Android* и *Linux*, продолжают использовать текстовые файлы как способ настройки и изменения.

В большинстве вариантов *Linux* файлы конфигурации сохраняются в формате .txt, это может стать проблемой, когда нам приходится работать с текстовыми файлами, ведь все .txt считаются критическими системными файлами. Поэтому, если мы попытаемся открыть текстовые файлы в этих операционных системах, мы не сможем их просмотреть.

Чтобы обойти эту проблему, как *macOS*, так и *iOS*, развернули совершенно другой подход к расширению текстового файла, реализовав расширение .plist, которое содержит всю информацию о системе, а также информацию о конфигурации приложения.

Поскольку каждая часть операционной системы постоянно взаимодействует с реестром *Windows*, то наша «библиотека с карточками» должна храниться в очень быстром хранилище. Следовательно, эта база данных была разработана для чрезвычайно быстрого чтения и записи, а также для эффективного хранения.

Если открыть и проверить размер БД реестра, то мы бы увидели, что она обычно колеблется между 15-20 мегабайтами, что делает её достаточно малой, чтобы всегда загружать её в оперативную память, которая, кстати, является самым быстрым хранилищем, доступным для операционной системы.

Поскольку реестр должен быть загружен в память постоянно, если бы размер реестра был большой, то он не оставил бы достаточно места для всех других приложений, чтобы они могли работать без сбоев или работать вообще. А теперь ещё приплюсуем сюда хром, который сам отбирает 6-8 гигабайт...

Если несколько пользователей взаимодействуют с одним и тем же устройством и используют одинаковые приложения, то повторная установка одних и тех же приложений дважды будет пустой тратой места, которого, как всем известно – и так не бывает много. Вот в такие моменты реестр вмешивается в «дела пользователей».

Многопользовательские сценарии (когда за одним компьютером могут работать несколько пользователей) очень распространены в корпоративных настройках, здесь необходимо чётко разграничивать права пользователей, это легко реализуется через реестр Windows.

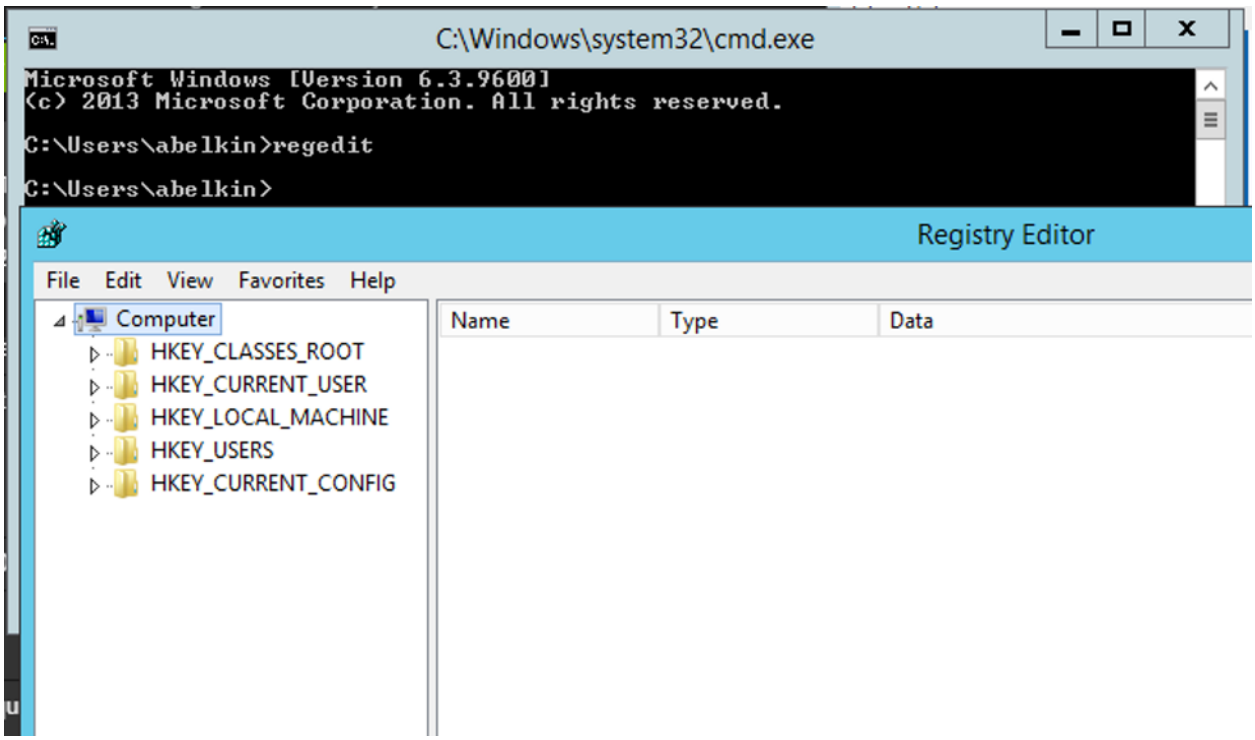
Как получить доступ к реестру Windows?

Реестр Windows состоит из двух основных элементов:

- Ключ реестра (можно сравнить с папкой).
- Значение реестра (можно сравнить с файлами).

Мы можем получить доступ к реестру и настроить его с помощью инструмента «*Редактор реестра*». *Microsoft* включает бесплатную утилиту для редактирования реестра вместе с каждой версией своей операционной системы.

Пойдём по известному нам пути, набрав regedit в командной строке. Этот редактор/утилита является «порталом» для доступа к реестру *Windows* и помогает нам исследовать его и вносить изменения.



Безопасное редактирование

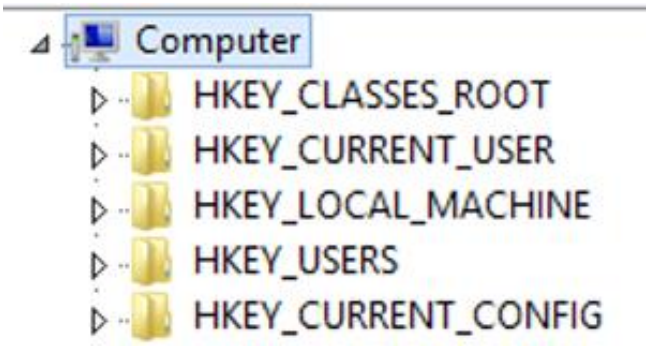
Если не знать, что делать — играть с конфигурацией реестра **опасно**. Каждый раз, когда вносим изменения, нужно убедиться, что следуем правильным инструкциям и изменяем только нужное нам!

Если намеренно или случайно удалить что-то в реестре *Windows*, это может изменить конфигурацию системы, что может привести либо к синему экрану смерти, либо к невозможности загрузки *Windows*.

Поэтому обычно рекомендуется сделать резервную копию реестра *Windows*, прежде чем вносить в него какие-либо изменения. Также можно создать точку восстановления системы (она автоматически создаёт резервную копию реестра), которую можно использовать, если когда-нибудь что-то пойдёт не так.

Структура реестра Windows

Пользователь находится в недоступном хранилище, которое существует только для доступа операционной системы. Эти ключи загружаются в оперативную память на этапе загрузки системы и постоянно передаются в течение определённого интервала времени или при возникновении определённого события или событий системного уровня. Ключи, которые находятся на пике иерархии в реестре, начинающемся с HKEY, считаются ветками.



Ветки реестра

HKEY_CLASSES_ROOT

HKEY_CLASSES_ROOT — это ветка реестра *Windows*, которая состоит из информации об ассоциации расширений файлов, программного идентификатора (*ProgID*), данных идентификатора интерфейса (*IID*) и идентификатора класса (*CLSID*).

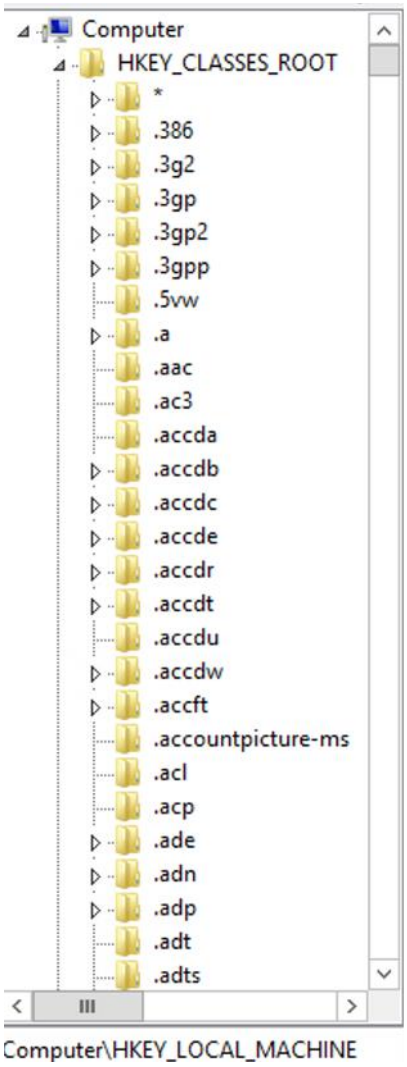
Эта ветка является шлюзом для любых действий или событий, происходящих в операционной системе *Windows*. Предположим, мы хотим получить доступ к некоторым *mp3*-файлам в папке «Загрузки». Операционная система отправляет через него свой запрос, чтобы предпринять необходимые действия.

В тот момент, когда мы обращаемся к *HKEY_CLASSES_ROOT*, можно *офигеть*, глядя на такой огромный список файлов расширений. Тем не менее, это те самые ключи реестра, которые обеспечивают плавную работу *Windows*.

Примеры ключей реестра ветки *HKEY_CLASSES_ROOT*:

- HKEY_CLASSES_ROOT\.otf
- HKEY_CLASSES_ROOT\.htc
- HKEY_CLASSES_ROOT\.img
- HKEY_CLASSES_ROOT\.mhtml
- HKEY_CLASSES_ROOT\.png
- HKEY_CLASSES_ROOT\.dll

Каждый раз, когда мы дважды щёлкаем и открываем файл, скажем, фотографию, система отправляет запрос через *HKEY_CLASSES_ROOT*, где чётко даются инструкции о том, что делать при запросе такого файла. Таким образом, система открывает программу просмотра фотографий.



Расширения

HKEY_LOCAL_MACHINE

Это одна из нескольких веток реестра, в которой хранятся все параметры, относящиеся к локальному компьютеру. Это глобальный ключ, в котором хранимая информация не может быть изменена ни одним пользователем или программой. Из-за таких масштабов этого подраздела вся хранящаяся информация находится в форме виртуального контейнера, непрерывно работающего в ОЗУ. Здесь содержится информация о конфигурации установленного пользователем ПО, про всё обнаруженное «железо».

Этот раздел реестра дополнительно разделен на 6 подразделов:

1. *SAM* (диспетчер учетных записей безопасности) — это файл ключа реестра, в котором хранятся пароли пользователей в защищенном формате (в хеш-значениях *LM* и *NTLM*).

Это заблокированный файл, расположенный в системе по адресу *C:\WINDOWS\system32\config*, который нельзя переместить или скопировать во время работы операционной системы.

Windows использует файл ключа реестра *Security Accounts Manager* для аутентификации пользователей, когда они входят в свои учётные записи *Windows*. Каждый раз, когда пользователь входит в систему, *Windows* использует серию хеш-алгоритмов для вычисления хеш-значения для введённого пароля. Если хеш введённого пароля совпадает с хешем пароля в файле реестра *SAM*, пользователям будет разрешён доступ к своей учётной записи. Это также файл, на который нацелены большинство хакеров при проведении атаки.

2. *Security* — этот раздел реестра является локальным для учётной записи администратора, который вошел в систему. Если система управляется какой-либо организацией, пользователи не могут получить доступ к этому файлу (за исключением администраторов или расширенных настроек прав для конкретного пользователя). Если мы откроем этот файл без прав администратора, он будет пустым. А если с правами администратора, то ключ по умолчанию будет соответствовать профилю безопасности локальной системы, установленному и активно управляемому организацией. Этот ключ связан с *SAM*, поэтому после успешной аутентификации, в зависимости от уровня привилегий пользователя, применяются различные локальные и групповые политики.
3. *System* (критический процесс загрузки и другие функции ядра) — этот подраздел содержит важную информацию, относящуюся ко всей системе, такую как имя компьютера, подключенное оборудование, файловая система и какие автоматические действия могут быть предприняты в определённом событии, например — синий экран смерти из-за перегрева процессора. Этот файл доступен только пользователям с достаточными административными привилегиями.
4. *Software* — здесь хранятся все конфигурации стороннего программного обеспечения, такие как драйверы *plug and play*. Этот подраздел содержит настройки ПО и *Windows*, связанные с уже существующим профилем оборудования, которые могут быть изменены различными приложениями и установщиками системы. Разработчики могут ограничивать или разрешать доступ пользователей к информации при использовании их приложений.
5. *Hardware* — подраздел, который создаётся динамически во время загрузки системы.
6. *BCD.dat* (в папке *\boot* в системном разделе) является важным файлом, который система читает и запускает во время загрузки системы, загружая реестр в ОЗУ.

HKEY_CURRENT_CONFIG

Основная причина существования этого подраздела — хранение видео, а также сетевых настроек. Это может быть вся информация, относящаяся к видеокарте, такая как разрешение, частота обновления, соотношение сторон и т.д., а также сеть.

Это также ветка реестра, в которой хранится информация об используемом в настоящее время профиле оборудования. *HKEY_CURRENT_CONFIG* на самом деле является указателем на ключ:

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\HardwareProfiles\Currentregistry.

Таким образом, *HKEY_CURRENT_CONFIG* помогает нам просматривать и изменять конфигурацию аппаратного профиля текущего пользователя, что мы можем сделать как администратор в любом из вышеперечисленных мест, поскольку все они одинаковы.

HKEY_CURRENT_USER

Часть реестра, которая содержит настройки хранилища, а также информацию о конфигурации для *Windows* и ПО текущего авторизованного пользователя. Например, раскладка клавиатуры, установленные принтеры, обои рабочего стола, параметры отображения, подключенные сетевые диски и т.д.

Многие параметры, которые вы настраиваете в различных апплетах на панели управления, хранятся в *HKEY_CURRENT_USER*. Поскольку эта ветка зависит от пользователя, то для каждого пользователя ключи и значения будут разными. Это не похоже на большинство других веток реестра, которые являются глобальными (в которых для всех пользователей хранится одна и та же информация).

В качестве меры безопасности информация, хранящаяся в *HKEY_CURRENT_USER*, является просто указателем на ключ, расположенный в ветке *HKEY_USERS* в качестве нашего идентификатора безопасности. Изменения, внесённые в любую из областей, вступят в силу немедленно.

HKEY_USERS

Содержит подключи, соответствующие ключам *HKEY_CURRENT_USER* для каждого профиля пользователя.

Здесь регистрируются все пользовательские данные конфигурации. Вся специфическая для пользователя информация, хранящаяся в системе, которая соответствует конкретному пользователю, хранится в ветке *HKEY_USERS*, мы можем однозначно идентифицировать пользователей, используя идентификатор безопасности или *SID*, который регистрирует все изменения конфигурации, сделанные пользователем.

Все пользователи, чья учётная запись существует в этой ветке, в зависимости от привилегии, смогут получить доступ к общим ресурсам, таким как принтеры, локальная сеть, локальные накопители, фон рабочего стола и т.д.

Что касается криминалистической информации, каждый SID хранит огромное количество данных о каждом пользователе, поскольку он ведёт журнал всех событий и действий, предпринимаемых под учётной записью пользователя. Сюда входит имя пользователя, количество раз, когда пользователь входил в систему на компьютере, дату и время

последнего входа в систему, дату и время последнего изменения пароля, количество неудачных попыток входа в систему.

Типы данных в реестре Windows

Все вышеупомянутые ключи и подразделы будут иметь конфигурации, значения и свойства, сохранённые в любом из следующих типов данных:

- строковые значения, такие как *Unicode*;
- двоичные данные;
- целые числа;
- символические ссылки;
- многострочные значения;
- список ресурсов (оборудование *Plug and Play*);
- дескриптор ресурса (оборудование *Plug and Play*);
- 64-битные целые числа.

Мы рассмотрели одну из самых сложных тем для понимания в ОС Windows — реестр. Научились с ним работать, поняли, какие ветки за что отвечают и куда лучше вообще не заглядывать.

Администрирование системы

Разница между Windows и Windows server

Все мы знаем, что такое *Windows*, но как насчёт разных типов? Хотя *Microsoft* предлагает два продукта, которые кажутся похожими, *Microsoft 10/11* и *Microsoft Server*, они выполняют разные функции и предлагают разные возможности. Одна операционная система предназначена для повседневного использования с ПК и ноутбуками, другая подходит для управления несколькими устройствами, службами и файлами через сервер.

Windows Server — это ряд операционных систем, разработанных специально для серверов. Каждая выпущенная версия *Windows Server* имеет соответствующую версию *Windows* — две операционные системы имеют общую кодовую базу.

Он был запущен в апреле 2003 года, ранее *Windows* предлагала серверные версии своих продуктов — *WindowsNT* была разработана как для рабочих станций, так и для серверов.

Из-за характера продукта Windows Server преимущественно используется в бизнес-средах. Продукт позволяет пользователям обмениваться файлами и службами, а также предоставляет администраторам контроль над сетями, хранилищами данных и приложениями.

Windows Server выпускается в трёх редакциях:

- *Datacenter*, как следует из названия, идеально подходит для центров обработки данных и облачных сред.
- *Standart* разработан для небольших физических или минимально виртуализированных сред.
- *Essential* идеально подходит для малых предприятий, насчитывающих до 25 пользователей и 50 устройств.

В чём сходство между *Windows* и *Windows Server*? Поскольку *Windows* и *Windows Server* выпускаются так, чтобы соответствовать друг другу, между ними есть много общего, особенно потому, что они используют одну и ту же кодовую базу.

При использовании *Windows* или *Windows Server* рабочий стол будет выглядеть одинаково. Будет такая же панель задач, значки на рабочем столе и кнопка «Пуск». Всё так же можно выполнять множество одних и тех же функций в любой из операционных систем, поскольку обе позволяют устанавливать одинаковое ПО.

Что предлагает Windows Server?

На первый взгляд две системы кажутся очень похожими. Однако между двумя предложениями гораздо больше различий, чем сходств, от конкретного программного обеспечения *Windows Server* до ценовых категорий. Давайте взглянем.

- Более мощный.

Продукт корпоративного уровня предлагает более мощное оборудование для поддержки более крупной сети. В то время, как та же *Windows 10 Pro* позволяет пользователям устанавливать 2 ТБ ОЗУ (а к этому большинство людей даже не приблизится), *Windows Server* предоставляет до 24 ТБ ОЗУ.

Аппаратное обеспечение *Windows Server* также может обрабатывать больше ядер и процессоров — оно имеет 64 сокета ЦП по сравнению с двумя процессорами в *Windows 10*.

- Не требует графического интерфейса.

Для работы с *Windows 10/11* вам понадобится *графический пользовательский интерфейс* (*GUI*).

Хотя серверная операционная система не требует использования графического интерфейса – его всегда можно установить; *Server* существует в двух формах: *Server Core* или *Desktop Experience*.

Windows Server даёт возможность установить только те роли сервера, которые необходимы под конкретные задачи. Это обеспечивает гибкость для пользователя, позволяя управлять своими операциями тем способом, который больше подходит, а также снижает нагрузку на *Windows Server*.

Если принято решение запустить *Windows Server* без графического интерфейса, вы можете управлять своей системой удалённо из командной строки с помощью *Windows PowerShell*. Кроме того, можно использовать инструмент с графическим интерфейсом, например, *RSAT* или *Windows Admin Center*.

- Предлагает специфичные для сервера инструменты.

Поскольку операционная система предназначена для серверов, *Windows Server* содержит инструменты и программное обеспечение для конкретных серверов, которые нельзя найти в *Windows 10/11*.

Кроме того, *Windows Server* может поддерживать ряд удобного для бизнеса программного обеспечения, разработанного специально для серверов, например *Active Directory* и *DHCP*.

Хотя некоторые из этих инструментов можно использовать в *Windows 10/11*, для них может потребоваться стороннее программное обеспечение. С другой стороны, в *Windows Server* отсутствуют некоторые из наиболее «забавных» функций, представленных в *Windows 10/11*. Поскольку система предназначена для использования в бизнесе, в неё не входят такие потребительские инструменты, как *Edge*, *Microsoft Store* или *Cortana*.

- Более высокий предел подключения.

Windows 10/11 имеет ограничение на подключение до 20 устройств. Это не проблема, если использовать операционную систему для коммерческого использования дома или в рамках малого бизнеса. Однако, если планируется использовать *Windows 10/11* в более крупном масштабе, это может стать помехой.

Windows Server предлагает практически неограниченное количество подключений, идеально подходящих для бизнеса любого размера.

Сравнение версий Windows

На момент написания этих строк на официальном сайте *Microsoft* можно купить всего две версии *Windows*: **Pro** и **Home**. В чём же разница между ними?

С точки зрения безопасности версия *home* **не** будет обладать двумя пунктами:

1. Шифрование устройства *Bitlocker* – если вы потеряете устройство, то *Bitlocker* может его заблокировать, чтобы «воришка» не получил доступ к данным.
2. *Windows Information Protection* – функция, помогающая защититься от утечки конфиденциальных данных.

Вроде бы разница незначительна. Но с точки зрения ведения бизнеса, между этими версиями огромная пропасть, ведь в *pro* версии будут ещё и другие полезные штуки:

- Групповые политики – *pro* версию можно ввести в домен и управлять ей посредством ГП.
- Функция динамической подготовки – история про то, что свежекупленный компьютер легко превратить в корпоративное устройство.
- Поддержка *Active Directory*.

Терминальный сервер

Терминальный сервер, также иногда называют коммуникационным сервером, представляет собой аппаратное устройство или сервер, который предоставляет терминалам, таким как ПК, принтеры и другие устройства, общую точку подключения к локальной или глобальной сети (*WAN*). Терминалы подключаются к терминальному серверу через их последовательный порт *RS-232C* или *RS-423*. Другая сторона терминального сервера подключается через сетевые карты (*NIC*) к локальной сети (*LAN*), обычно к локальной сети *Ethernet* или *Token Ring*, через модемы к *WAN* с коммутируемым/исходящим доступом или к сети *X.25.* или шлюз *3270*. Различные марки терминальных серверов предлагают разные виды межсоединений. Некоторые из них могут быть заказаны в различных конфигурациях в зависимости от потребностей клиента.

Использование терминального сервера означает, что каждому терминалу не нужна собственная сетевая карта или модем.

Ресурсы подключения внутри терминального сервера обычно динамически распределяются между всеми подключенными терминалами.

Некоторые терминальные серверы могут использоваться сотнями терминалов. Терминалами могут быть ПК, терминалы, имитирующие 3270-е, принтеры или другие устройства с интерфейсом *RS-232/423*. Терминалы могут использовать *TCP/IP* для *Telnet*-соединения с хостом, *LAT* для хоста *Digital Equipment Corporation* или *TN3270* для *Telnet*-соединения с хостом *IBM* с приложениями 3270. С некоторыми серверами терминалов конкретный пользователь может иметь несколько подключений хоста к различным типам операционных систем хоста, таким как *UNIX*, *IBM* и *DEC*.

Хотя концепция терминала возникла во времена царства мэйнфреймов, операционная система *Windows Server* долгое время могла выступать в качестве терминального сервера.

Как работает терминальный сервер

Принцип работы терминального сервера может варьироваться от одного поставщика к другому. В случае терминального сервера *Windows* операционная система настроена для поддержки нескольких пользовательских сеансов. Это отличается от других многосессионных сред, таких как файловый сервер *Windows*, потому что операционная система визуализирует пользовательский интерфейс (UI) для каждого из сеансов.

Конечные пользователи подключаются к терминальному серверу с помощью клиента протокола удалённого рабочего стола (RDP), настольного или мобильного приложения, задачей которого является подключение к терминальному серверу и отображение

содержимого сеанса. Клиент RDP связывается с терминальным сервером через порт подключения. Компонент диспетчера сеансов разделяет все пользовательские сеансы, а также обрабатывает такие задачи, как разрешение пользователю повторно подключиться к своему сеансу после случайного закрытия клиента RDP. Сеансы фактически выполняются как часть службы сервера терминалов, но за управление этими сеансами отвечает диспетчер сеансов.

Когда пользователю необходимо взаимодействовать с сеансом с помощью клавиатуры, мыши или сенсорного ввода, эти вводы выполняются в клиенте *RDP*. Затем клиент *RDP* передаёт входные данные на сервер терминалов для обработки. Терминальный сервер также отвечает за выполнение всей графической визуализации, хотя именно клиент *RDP* делает сеанс видимым для пользователя.

Терминальный сервер и удалённый рабочий стол

Терминальный сервер и удалённый рабочий стол служат одной и той же цели. Они позволяют пользователю взаимодействовать с удалённым сеансом через *RDP*-клиент. Основное отличие состоит в том, что терминальные серверы работают на *Windows Server*, и поэтому пользователю предоставляется рабочий стол. И наоборот, в средах удалённых рабочих столов обычно есть настольные операционные системы, такие как *Windows 10/11*, работающие на виртуальных машинах (ВМ). Таким образом, пользователю предоставляется настоящая настольная операционная система, а не сеанс, запущенный на сервере.

Sysinternals

Microsoft приобрела *Sysinternals* в далёком 2006 году, и сегодня Марк Руссинович (один из главных авторов *Sysinternals*) работает техническим специалистом в отделе облачных вычислений и предприятий. Хотя большую часть времени он уделяет платформе *Azure*, он по-прежнему участвует в разработке инструментов, созданных его компанией.

Увы презентации Руссиновича по инструментам *Sysinternals* часто остаются без должного внимания со стороны админов. Блог Руссиновича содержит длинный список статей, в которых рассказывается, как различные системные проблемы, включая вопросы безопасности, анализировались с помощью инструментов *Sysinternals*. Вам стоит потратить своё время на просмотр его блога и записи сеансов *TechEd*. Очень рекомендую.

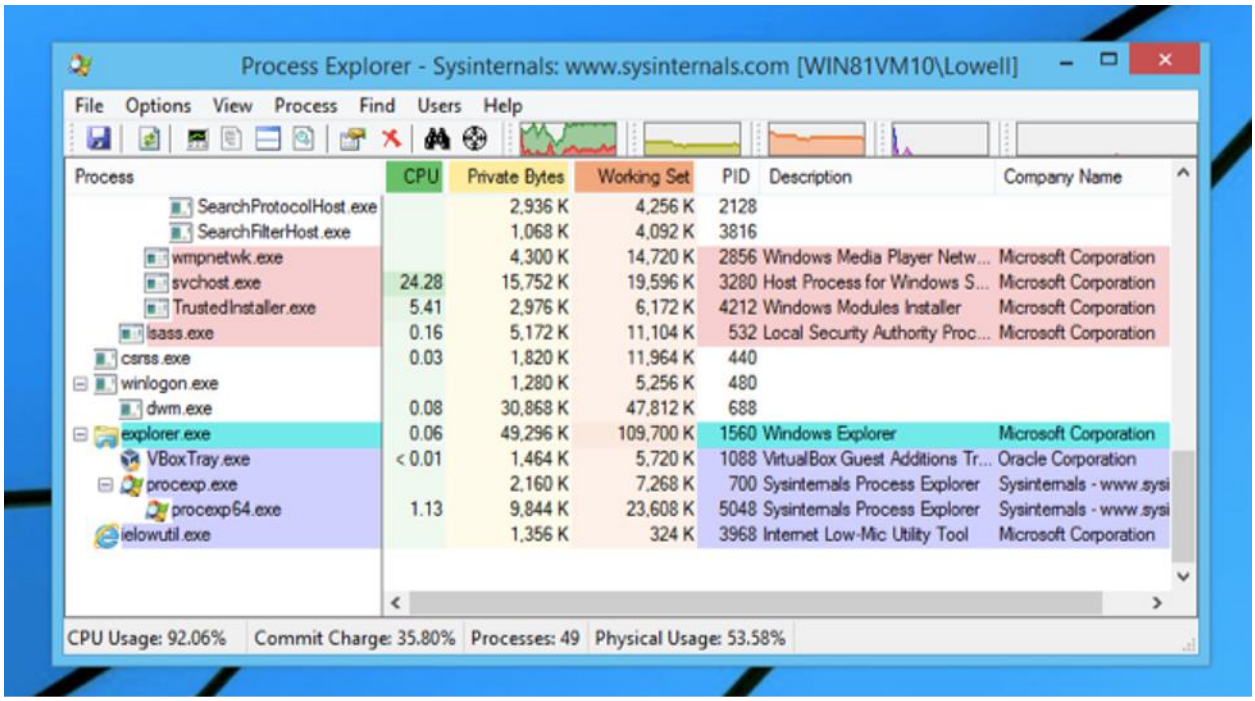
На веб-сайте *Sysinternals* есть ссылки на широкий спектр инструментов, сгруппированных по функциональным областям. Некоторые инструменты перекрывают разные категории и позволяют выполнять как обслуживание системы, так и задачи безопасности. Также можно найти ссылки на веб-трансляции и другие учебные материалы, которые помогут быстрее освоить весь набор инструментов. Если вы действительно хотите копнуть глубже, вам стоит взглянуть на «Справочник администратора Windows Sysinternals». Это мастхэв.

Process Explorer

Все инструменты *Sysinternals* можно загрузить бесплатно, и они предоставляют информацию, которую можно использовать для собственного расследования. Многие опытные системные администраторы держат наготове *USB*-флешку со всем набором инструментов *Sysinternal*. Два самых популярных инструмента, *Process Explorer* и *Process Monitor*, обеспечивают глубокое понимание внутренней работы *Microsoft Windows*.

В настоящее время *Process Explorer* имеет версию 16.03 и существует в той или иной форме со времен *Windows NT*.

Process Explorer — это инструмент, используемый для идентификации файлов, библиотек DLL, ключей реестра и других объектов, прикреплённых к запущенному процессу. Он также покажет владельца каждого запущенного процесса.

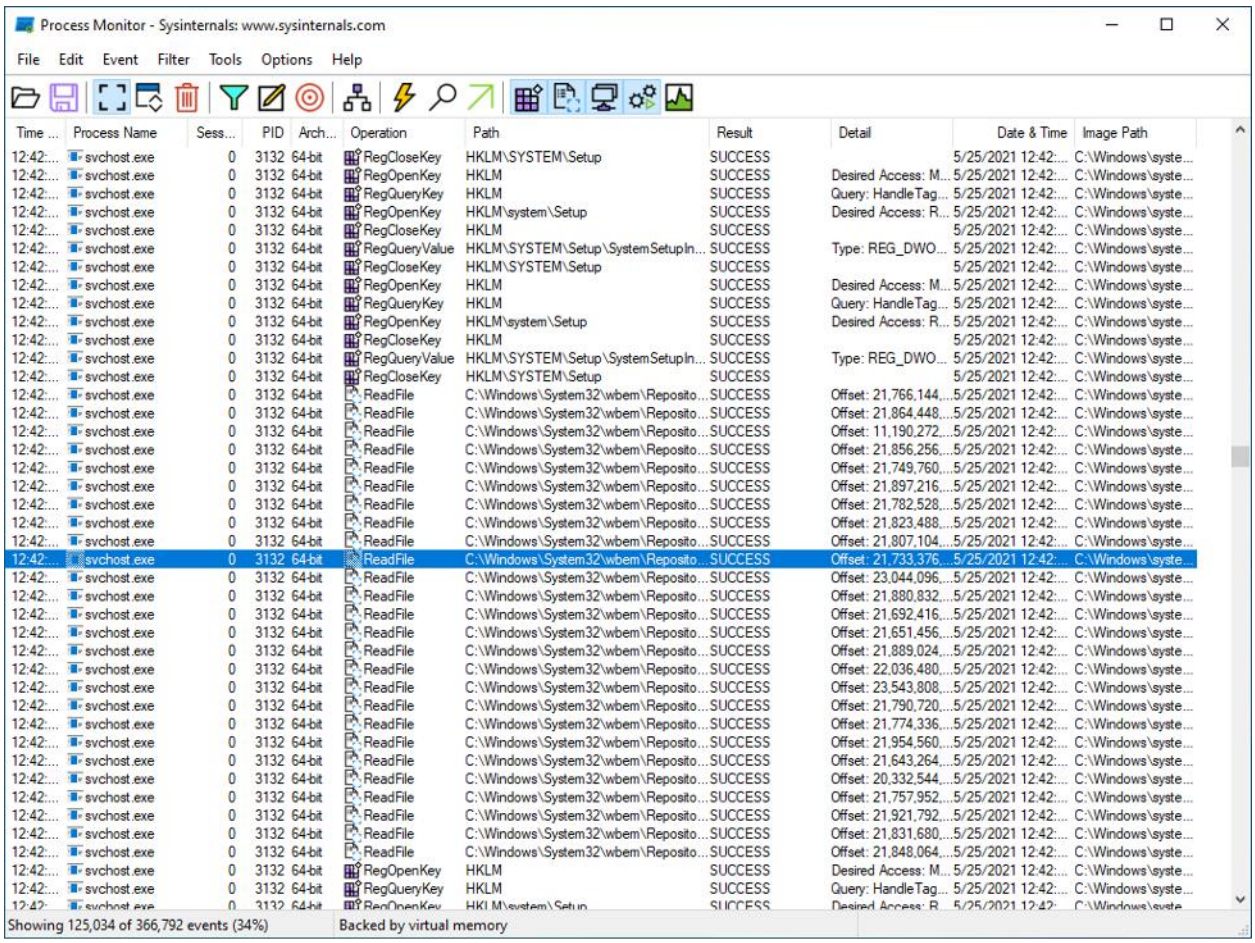


Process Explorer

Process Monitor

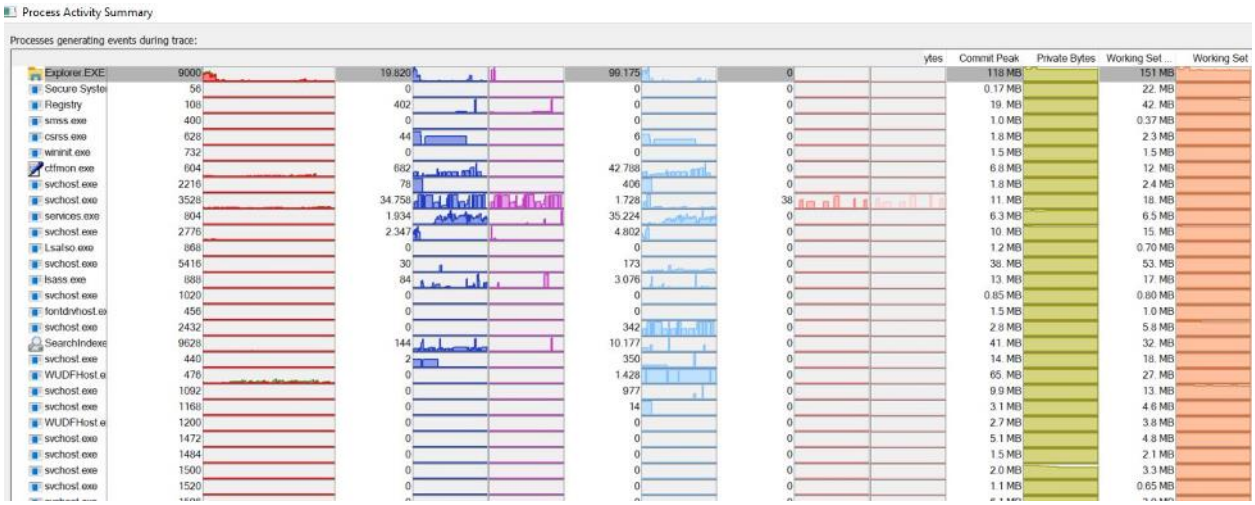
Process Monitor — *один из тех инструментов, о которых вы хотели бы знать давно. Он обеспечивает просмотр в реальном времени всей файловой системы, реестра Windows и активности процессов для каждого запущенного процесса.*

Самая большая проблема, с которой сталкивается большинство людей при использовании Process Monitor, — это огромное количество информации, которую он производит.



Process Monitor

Настоящая мощь *Process Monitor* заключается в инструментах сводки и функции фильтрации. Экран «Сводная информация о процессах», доступный в меню «Инструменты», представляет визуальную картину всех активных процессов с графиками активности файлов, сети и реестра. На этом едином экране легко обнаружить проблемные приложения, а затем использовать инструмент фильтрации, чтобы глубже погрузиться в процесс и точно определить, что происходит.



Process Activity Summary

Sysmon

Sysmon — один из новейших инструментов, добавленных в пакет *Sysinternals*. Его можно загрузить с сайта *Microsoft Technet*. *Sysmon* работает как служба *Windows*, загружаясь очень рано в процессе загрузки, чтобы поймать любой вредоносный код, который может попытаться получить контроль над машиной. Он регистрирует несколько конкретных событий, включая создание процесса, изменение времени создания файла процессом и установление сетевого соединения. Все появления этих событий записываются в журнал событий *Windows*, который при необходимости доступен удалённо.

Фильтрация и поиск в журналах событий могут быть утомительным процессом, но немного попрактиковавшись, можно научиться определять проблемы. Понимание того, как ведут себя определённые типы вредоносных программ, помогает при проверке безопасности. Многие из этих программ делают копию исходного исполняемого файла, пытаясь скрыть его личность. *Sysmon* регистрирует эти типы действий и использует алгоритм хеширования *SHA1* для определения сигнатуры этих мошеннических программ.

Sysmon также отслеживает сетевую активность, чтобы помочь идентифицировать атаки, инициирующие соединения вне хост-системы. Самым большим преимуществом использования такого инструмента, как *Sysmon*, по сравнению с другими инструментами, такими как *Process Monitor*, является использование журнала событий *Windows* для хранения всей необходимой информации. Это избавляет от необходимости запускать дополнительную программу и даёт возможность удалённого доступа к журналам для дальнейшего анализа.

Стандартные сервисы

Microsoft SCCM (*System Center Configuration Manager*) — это платное решение для управления «жизненным циклом» от *Microsoft*, которое отслеживает инвентаризацию сети, помогает в установке приложений и развёртывает обновления и исправления безопасности в сети. Хотя *SCCM* использует систему исправлений *Microsoft WSUS* для проверки и установки обновлений, он даёт пользователям дополнительный контроль управления обновлениями, когда и как они применяются, и включает в себя множество других функций, которые делают его привлекательным вариантом для крупных корпоративных сетей. Однако *Microsoft SCCM* представляет несколько проблем для организаций, которые ищут одно решение для обеспечения обновлений для всех устройств, операционных систем и сторонних приложений, поэтому важно оценить плюсы и минусы.

Плюсы SCCM	Минусы SCCM
Часть системы полного управления жизненным циклом для *Windows*: *SCCM* включает в себя широкий спектр функций, которые обеспечивают гибкость в отношении применения обновлений, создания общесистемных отчетов и позволяют управлять любым компьютером *Windows* в сети с одной центральной консоли. *SCCM* предоставляет набор инструментов для защиты конечных точек и при правильной настройке может стать системой управления полным жизненным циклом для ИТ-отделов с высоким процентом систем *Windows*.	**Высокая стоимость приобретения и запуска**: *SCCM* обычно продается как часть более крупного набора инструментов от *Microsoft* и является чрезмерно дорогостоящим для некорпоративных компаний. Цены на *SCCM* непрозрачны и могут включать отдельные затраты на конечные точки и серверы. *SCCM* также является локальным решением, для работы которого требуется *SQL*-сервер, что приводит к высоким текущим операционным расходам и требованиям к ресурсам для обслуживания.
Полная интеграция с системами *Windows*: будучи продуктом *Microsoft*, *SCCM* очень хорошо интегрируется с системами *Windows* и другими продуктами *Microsoft*. В последние годы *SCCM* пытался адаптироваться к тенденции подключения устройств, предоставляемых сотрудниками, к сетям компании, и теперь поддерживает функцию «Принесите своё собственное устройство», что означает, что устройства, добавленные в сеть отдельными сотрудниками, могут управляться через *SCCM* и помечены, если они не обновляются.	**Создан для систем, в которых фигурирует *Windows*:** *SCCM* создан в первую очередь для систем *Windows*, поэтому его функциональность и обновления сосредоточены на *Windows*. Системами, отличными от *Windows*, включая *Mac* и *Linux*, можно ограниченно управлять через *SCCM* в качестве конечных клиентов, однако этот процесс представляет собой путаницу, поскольку для работы *SCCM* по-прежнему требуется сервер *Windows*, а функциональность для систем, отличных от *Windows*, ограничена. В средах со смешанными ОС элементы ручной установки исправлений остаются даже при установленном *SCCM*, что является серьезным недостатком для компаний, которые уже платят большие суммы за *SCCM*.
Управление через графический интерфейс и поддержка через *Microsoft*: *SCCM* управляется через относительно простой графический интерфейс, что означает, что его легче изучить и внедрить, чем самостоятельно развёртываемые инструменты, такие как *Chef* и *Puppet*. Поскольку *SCCM* — это устоявшаяся и платная служба *Microsoft*, она также имеет хорошую поддержку через форумы сообщества и саму *Microsoft*.	**Ограниченная возможность обновлений сторонних приложений**: хотя *SCCM* добавляет больше поддержки сторонних приложений, чем *WSUS*, возможность *SCCM* обновлять сторонние приложения очень ограничена и является грустью среди ИТ-специалистов. На странице отзывов *Microsoft* о *SCCM* наибольшее внимание уделяется усовершенствованию сторонних исправлений. Что неудивительно, учитывая, что на программное обеспечение сторонних производителей приходится до 76% уязвимостей на среднем ПК, сложность настройки *SCCM* для автоматического исправления сторонних приложений может поставить под угрозу вашу инфраструктуру.

IIS — это веб-сервер, специфичный для платформы *Microsoft .NET*. Таким образом, он также называется *Windows Web Server*. Хотя его можно запустить в *Linux* и *Mac OS* с помощью утилиты под названием *Mono*, делать это крайне нежелательно, потому что он не будет таким стабильным и надёжным, как в «родной» системе.

Итак, что делает *IIS*? Как веб-сервер, это процессор, стоящий за размещением веб-приложений. Вы можете думать об этом как о посреднике, отвечающем за обработку сообщений разных приложений с портов *TCP* по умолчанию.

Outlook Web Access (*OWA*) — это способ доступа к электронной почте *Microsoft Outlook* на основе браузера. Хотя его также называют *Outlook Web App*, сейчас его чаще называют «*Outlook* в Интернете».

OWA когда-то применялся исключительно к онлайн-версии *Outlook*, которая поставлялась с *Microsoft ExchangeServer*. В наши дни доступ к *OWA* чаще осуществляется из учётной записи *Microsoft 365* или учетной записи *Outlook.com*.

Хотя версия *Outlook*, которая работает на ПК в качестве приложения для *Windows* или *Mac*, по-прежнему обеспечивает максимальную гибкость, мощность и возможности, теперь необязательно носить ноутбук с собой, ведь можно открыть всю почту в браузере.

Итоги

По итогам главы мы:

- посмотрели историю семейства ОС *Windows*;
- познакомились с архитектурой системы;
- узнали о системных службах и ФС *NTFS*;
- рассмотрели работу с драйверами устройств;
- попробовали поработать с реестром;
- сравнили пользовательские и серверные типы системы и рассмотрели инструменты администрирования.

Это базовые знания об ОС *Windows*, которые нужны для понимания принципов работы системы. Понятное дело, что это лишь сухая теория, однако непосредственно практика пентеста и изучение более глубинных и ёмких процессов возможна только после знакомства с фундаментальной базой. Её то, вы и освоили в рамках этой главы. Ну а далее, нас уже ждёт погружение в конкретные процессы и практика хакинга.

Конец ознакомительного фрагмента. Полную версию книги можно приобрести тут: https://kurets.ru/windows-for-hackers