# кража со взломом
# или "выдирание" программ без дистрибутива

крис касперски, ака мыщъх, a.k.a. nezumi, a.k.a. souriz, a.k.a elraton, no-email

**хочется перенести программу на другой компьютер, но дистрибутив утерян или не обозначен, а если даже и обозначен — жаль терять свои настройки. знакомая ситуация, не правда ли? существует множество утилит "клонирования" программ, но все они требуют обязательного наличия инсталлятора. а у нас его нет. зато есть лапы, усы, голова и хвост. головой мы будем думать, хвостом — шевелить, а лапами стучать по клавиатуре.**

## *введение*

Во времена викторианской Англии и безраздельного господства MS-DOS перенос программ решался тривиальным копированием базового каталога (например, \GAMES\DOOM2\) с одного компьютера на другой. Эта техника не утратила своей значимости и до сих пор, но используется все реже и реже. Современные программы своим подавляющем большинством нагло лезут (читай — серут) в реестр, в директории типа \WINNT\System32 и еще десяток подобных "злачных" мест, разбрасывая свой код, данные и помет по всей системе, а потому копирование базового каталога уже ничего не решает и на новом компьютере программа категорически отказывается работать.

Этим активно пользуются многие фирмы, специализирующиеся на обслуживании офисной техники. Устанавливая программы без дистрибутивов они сажают пользователей на "иглу", вынуждающую постоянно обращаться к "дилерам" за новой дозой "кокаина". С хакерской точки зрения переход с "кокаина" на "траву" — это даже не взлом, а так… мелкое хулиганство в стиле детской шалости. Но… не всем же ломать программы! Кому-то приходится и хулиганить! (Особенно в целях производственной необходимости :-).

## *метод каменных стрел и топоров*

Нашим предкам было тяжело. Они одевались в звериные шкуры и добывали огонь трением. А все потому, что каменные орудия были тупые. Как мозги. И еще неизвестно – кто был тупее. Пользователи, переносящие программы варварскими методами, недалеко ушли от них. Что же это за методы такие?

Начнем с классики. Копируем базовый каталог на новую машину. Запускаем. Программа ругается, сообщая чего ей там не хватает (а не хватает ей обычно динамических библиотек). Находим их, копируем. Запускам снова. И продолжаем сей процесс до тех пор, пока программа не запустится или пользователь не обломается. А обломаться он может по очень многим причинам. Стоит только программе вместе внятного сообщения об ошибке выдать что-то типа "неправильная установка" и все. Кранты.

## *путешествие по времени*

"*...доктор Брук ткнул пальцем в одну из строчек на доске. Похоже на антигравитацию, а если развернуть принцип на сто-восемьдесят градусов, мы получим формулу путешествия во времени*" (с) Роберт Хайнлайн "Имею скафандр — готов путешествовать".

Хорошо, инсталлятора у нас нет. Но деинсталлятор в подавляющем большинстве случаев все-таки остается. Чаще всего он кладется в базовый каталог с программой, реже — в папку \WINNT\Installer\. А деинсталлятор это… тот же самый инсталлятор, только наоборот! Развернув принцип на 180 градусов мы получим инструмент, который, собственного говоря, и искали. На самом деле, просто так взять и превратить деинсталлятор в инсталлятор не получится — как его не крути и какую траву не кури. Но вот "выдрать" из него список устанавливаемых файлов/драйверов и ветвей реестра вполне возможно.

Нам потребуется декомилятор, коих развелось в огромном числе. Правда, количество самих инсталляторов еще больше. Они плодятся как кролики в условиях невесомости и к тому же далеко не всякий декомилиятор поддерживает файлы, созданные деинсталлятором (они зачастую имеют слегка другой формат данных, а программисты у нас ленивые – пока не

покурят ничего вообще ничего не кодят, а как покурят, накодят такое, что обзавидуешься где же они такую траву раздобыли, ведь не сезон).

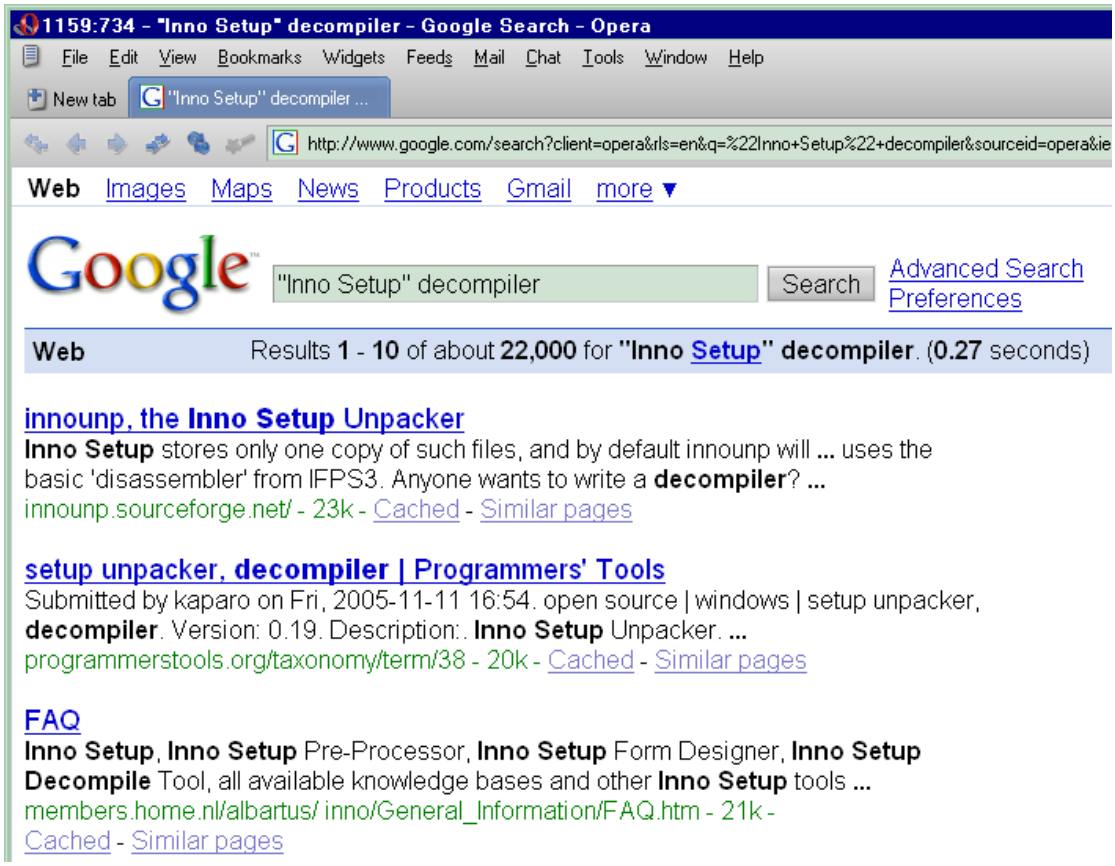


**Рисунок 1 парад декомпиляторов на Google'е**

Впрочем, в половине случаев ничего трансмутировать не приходится и в основном каталоге программы лежит лог (файл с расширением .log или типа того), созданный инсталлятором с перечнем всех, совершенных им действий. Загрузив его в любой текстовой редактор, например, в FAR по <F4> или <F3>, мы можем видеть какие файлы, динамические библиотеки и драйвера были скопированы и какие ключи реестра созданы. Остается только повторить эти действия вновь и… программа встанет на соседний компьютер как родная!

ОК, довольно теории. Займемся практикой. Откроем для примера базовый каталог программы "Macro Express 3", найдем в нем файл "INSTALL.LOG" и загрузим его в любой текстовой редактор:

```
Made Dir: C:\Program Files\Macro Express3
File Copy: C:\Program Files\Macro Express3\UNWISE.EXE
RegDB Key: Software\Microsoft\Windows\CurrentVersion\Uninstall\Macro Express 3
RegDB Val: Macro Express 3
RegDB Name: DisplayName
RegDB Root: 2
RegDB Key: Software\Microsoft\Windows\CurrentVersion\Uninstall\Macro Express 3
RegDB Name: UninstallString
RegDB Root: 2
Made Dir: C:\Documents and Settings\All Users\Главное меню\Программы\Macro Express
Shell Link: C:\Documents and Settings\All Users\Главное меню\Программы\Macro
Express\Macro Express 3.lnk
File Copy: C:\Program Files\Macro Express3\MacExp.exe
File Copy: C:\Program Files\Macro Express3\WhatsNew.txt
…
Made Dir: C:\PROGRA~1\COMMON~1\Insight Software Solutions
File Copy: C:\Program Files\Common Files\Insight Software Solutions\QkStart.exe
File Copy: C:\Program Files\Common Files\Insight Software Solutions\ISSBugRp.exe
```

**Листинг 1 протокол инсталляции программы Macro Express 3**

Мы видим не только копируемые файлы/ярлыки/динамические библиотеки/etc, но и ветви реестра вместе с их значениями!!! Кстати, обратим внимание, что помимо базового каталога, "Macro Express 3" добрался до директории "Common Files" и начал там слегка безобразничать.

Впрочем, нам просто повезло. В некоторых (достаточно редких) случаях в лог-файл попадают только сами ветви реестра, создаваемые инсталлятором, без их значений, что, собственно говоря и неудивительно, поскольку лог-файл чаще всего создается для деинсталлятора, которому достаточно знать лишь имя ключа реестра. На конкретное значение — ему плевать. Вот оно и не попадает в лог. Нашли чем пугать! Щас! Берем "Редактор Реестра" и быстро-быстро извлекаем все значения из обозначенных ключей на автопилоте.

Гораздо хуже, когда никакого лога в нашем распоряжении нет. Возьмем, например, достаточно известную утилиту "PDF Creator". В базовом каталоге (из всех интересующих нас вещей) лежат лишь unins000.exe и unins000.dat. Ну, первый из них мы отбросим сразу (это исполнительный "движок" — общий для всех программ, созданных инсталлятором данного типа), а вот unins000.dat откроем в FAR'e по <F3> или в HIEW'e (см. рис. 3).

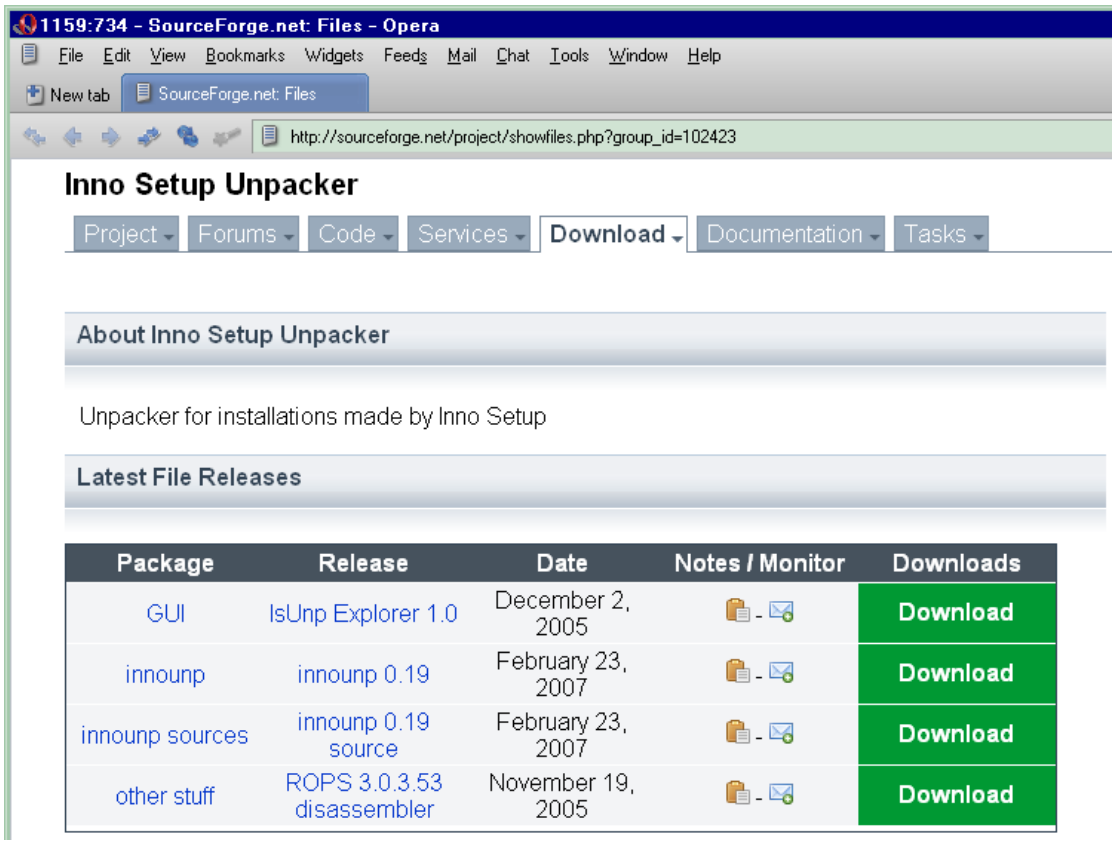


**Рисунок 2 Inno Setup Unpacker – неплохой свободный декомпилятор**

Первой же строкой мы видим: "**Inno Setup Uninstall Log (b)**". Ага, значит, это лог созданный инсталлятором "Inno Setup". И хотя лог неупакован (смотри по <F3> сколько хочешь), он представлен в неудобном для нас человеков нетекстовом формате — пока прочитаешь очкариком стать можно! Лучше заплатить десяток баксов за декомпилятор (или найти бесплатный) чем всю жизнь работать на аптеку!

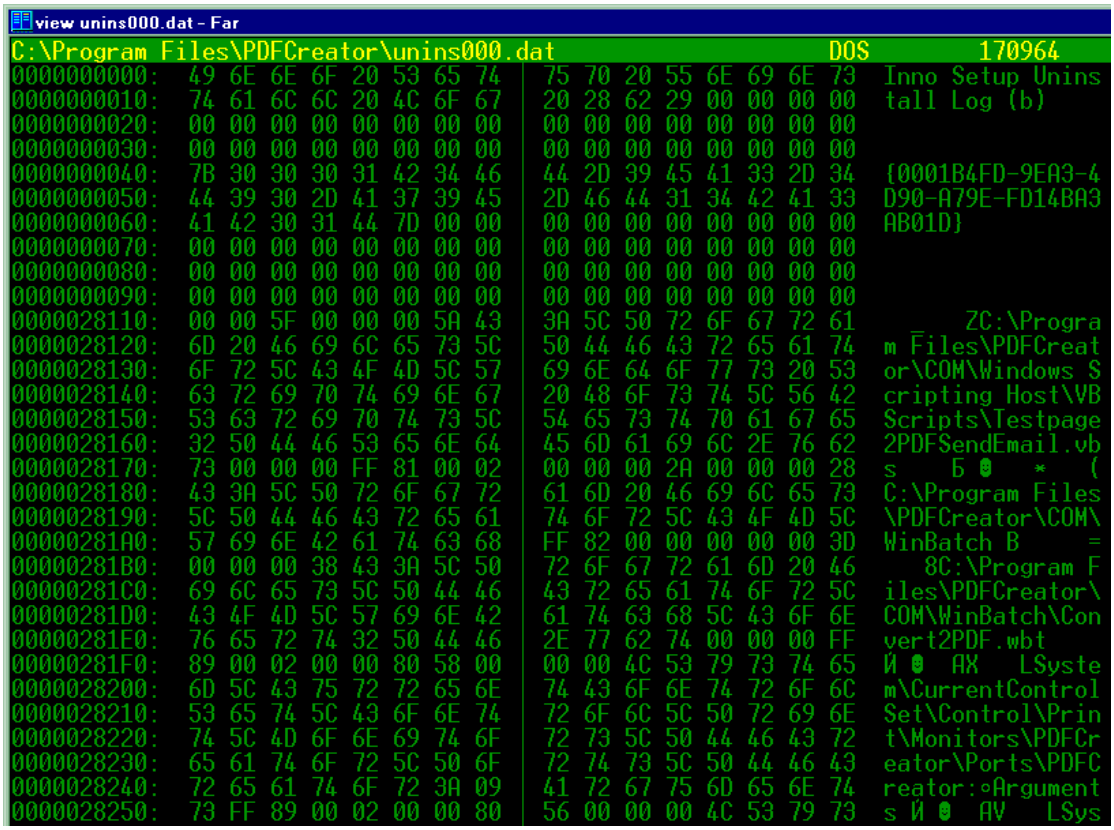


**Рисунок 3 лог-файл, созданный инсталлятором "Inno Setup" в FAR'e**

Набираем в Google "Inno decompiler" и получаем внушительный список из которого мыщъх'у больше всего понравился бесплатный InstallExplore от Сергея Ванина, выполненный в виде plug-in'a для FAR'a — http://plugring.farmanager.com/downld/files/instexpl_v0.3.rar. Просто наводим курсор на файл, который мы хотим декомпилировать, нажимаем <Shift-F3> и получаем список файлов/ключей реестра в удобно-читаемой форме или… красное ругательное окошко (красное — это оно от стыда, наверное), язвительно поздравляющее нас с исключением. Приплыли! Сушите весла! А пока они сохнут, самое время отправится на поиски другого декомпилятора, благо, старик Google всегда под рукой…

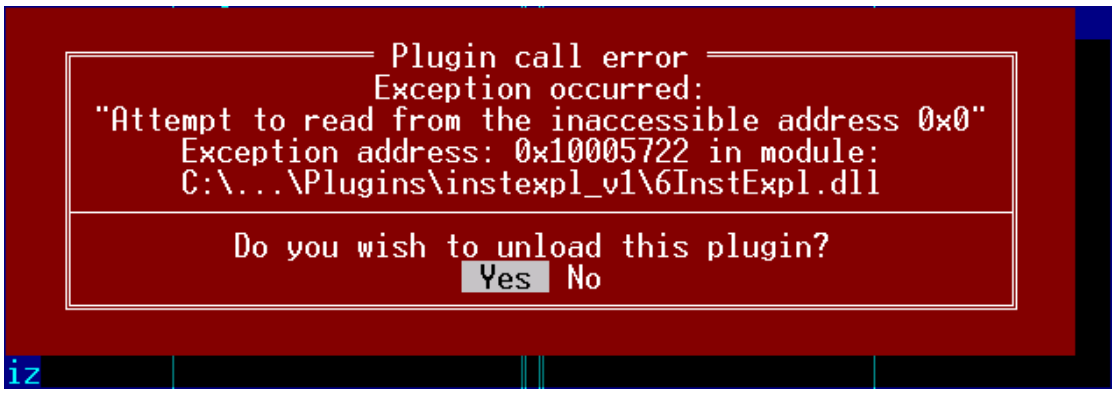


**Рисунок 4 краснота спасет мир!**

К сожалению, существует большое количество инсталляторов, для которых до сих пор нет достойных декомпилияторов. Взять, например, "Nullsoft Install System" (да-да, тот самый Nullsoft, подаривший нам WinAmp), который используется для установки "Abyss Web Server". Мы видим всего лишь один "uninstall.exe" файл, включающий в себя как исполнительный движок, так и лог.

Просматривая лог в HIEW'e, мы видим следующие текстовые строки (см. листинг 2), из которых заключаем, что программа что-то заносит в файл wininit.ini (находим этот файл в каталоге Windows, открываем его и видим текстовую строку "[Rename] NUL=C:\DOCUME~1\

KRISKA~1\LOCALS~1\Temp\A~NSISu_.exe", так что с этим пунктом все ясно. Далее видно, что "Abyss Web Server" копирует себя в C:\Program Files в ProgramFilesDir — каталог с именем программы (в данном случае это "Abyss Web Server"), после чего лезет в ключ реестра Software\Microsoft\Windows\CurrentVersion и чего-то там создает. Ну, что он там создает — догадаться нетрудно — "Abyss Web Server" и создает. После чего добавляет себя на панель быстрого запуска Quick Launch и на этой радостной ноте считает свою миссию выполненной.

```
.00409320:  00 00 00 00-0A 5B 00 00-5B 52 65 6E-61 6D 65 5D      ◙[  [Rename]
.00409330:  0D 0A 00 00-5C 77 69 6E-69 6E 69 74-2E 69 6E 69  ♪◙  \wininit.ini
.00409340:  00 00 00 00-25 73 3D 25-73 0D 0A 00-4D 6F 76 65      %s=%s♪◙ Move
.00409350:  46 69 6C 65-45 78 41 00-43 3A 5C 50-72 6F 67 72  FileExA C:\Progr
.00409360:  61 6D 20 46-69 6C 65 73-00 00 00 00-50 72 6F 67  am Files    Prog
.00409370:  72 61 6D 46-69 6C 65 73-44 69 72 00-53 6F 66 74  ramFilesDir Soft
.00409380:  77 61 72 65-5C 4D 69 63-72 6F 73 6F-66 74 5C 57  ware\Microsoft\W
.00409390:  69 6E 64 6F-77 73 5C 43-75 72 72 65-6E 74 56 65  indows\CurrentVe
.004093A0:  72 73 69 6F-6E 00 00 00-43 6F 6D 6D-6F 6E 46 69  rsion   CommonFi
.004093B0:  6C 65 73 44-69 72 00 00-5C 4D 69 63-72 6F 73 6F  lesDir  \Microso
.004093C0:  66 74 5C 49-6E 74 65 72-6E 65 74 20-45 78 70 6C  ft\Internet Expl
.004093D0:  6F 72 65 72-5C 51 75 69-63 6B 20 4C-61 75 6E 63  orer\Quick Launc
.004093E0:  68 00 00 00-2A 3F 7C 3C-3E 2F 22 3A-00 00 00 00  h   *?|<>/":
```

**Листинг 2 фрагмент деинсталлятора, созданного инсталлятором от Nullsoft**

Вот мы и декомпилировали двоичный файл деинсталлятора в HIEW'e (или в FAR'e по <F3>) без всяких дополнительных утилит, то есть вручную. Конечно, нам повезло, что uninstall.exe не был упакован никаких протектором (в жизни и такое случается), а лог лежал в незашифрованном виде. Иначе, без помощи отладчика и дизассемблера нам бы уже не обойтись. Однако, это клинические случаи, которые практически не случаются в реальной жизни, а если даже и случаются, то существуют гораздо более короткие пути, чем отладка и дизассемблирование.

## *>>> врезка охота за динамическими библиотеками*

В некоторых руководствах по "выдиранию" приложений встречается утверждение, что определить набор используемых динамических библиотек можно с помощью Olly или Process Explorer'a. Все они показывают список DLL, загруженных в адресное пространство исследуемого процесса, делая тайное явное. И не нужно ковырять логи деинсталляторов. ОК, запускаем Process Explorer и смотрим какие DLL использует ну, например, Опера (см. рис. 5).

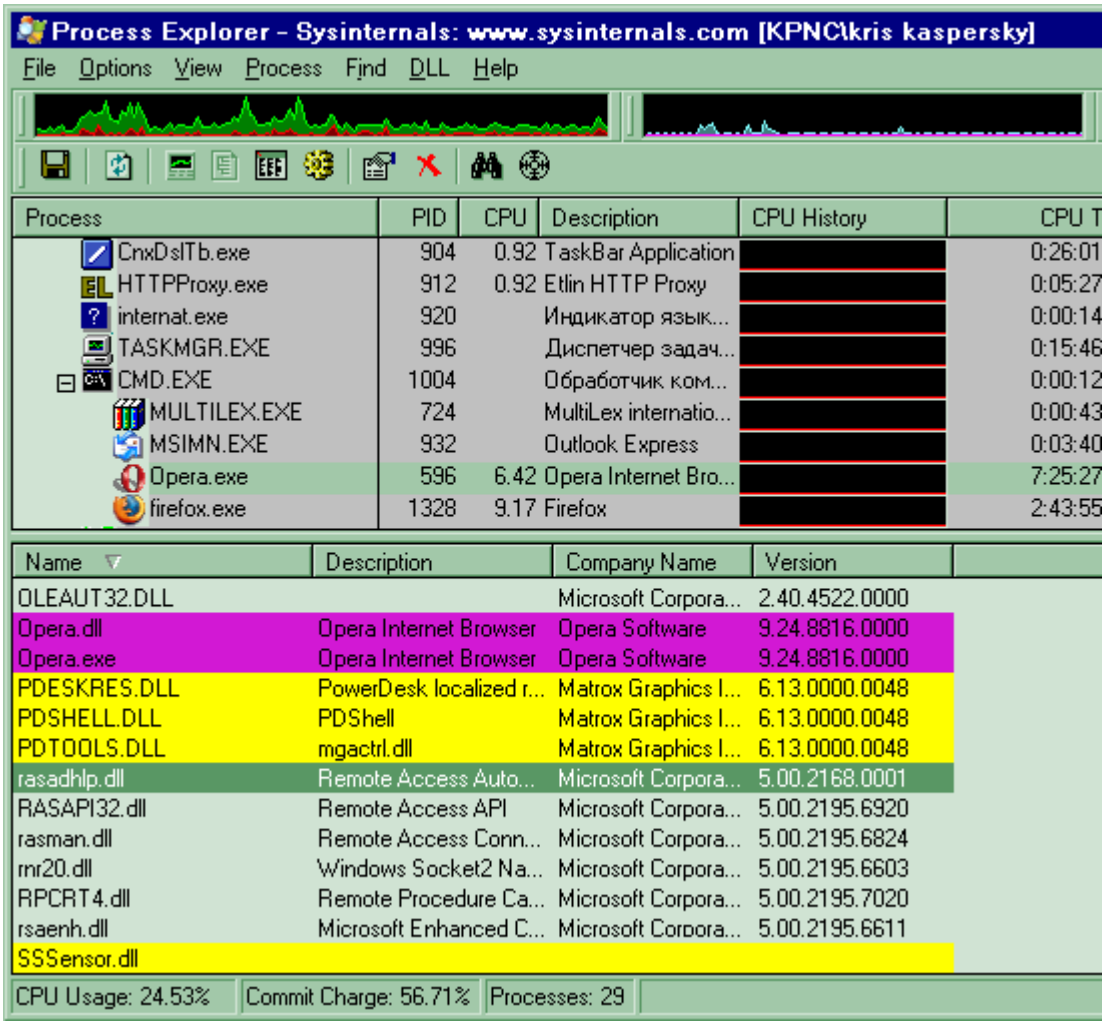


**Рисунок 5 определение перечня загруженных динамических библиотек с помощью Process Explorer'a**

И вот тут нас ждет "приятный" сюрприз типа "граблей". Оказывается, что посторонние программы весьма активно внедряют свои динамически библиотеки во все запускаемые приложения для организации межпроцессорного взаимодействия. В данном случае мы видим PDESKRES.DLL, PDSHELL.DLL, PDTOOLS.DLL принадлежащие "оснастке" карты Matrox G450, а так же SSSensor.dll от SyGate Personal Firewall. И хотя от того, что мы перетащим их на соседнюю машину, никакого вреда не будет (без соответствующих EXE эти библиотеки будут лежать балластом мертвого груза), но ведь и пользы от них никакой! А винчестеры все-таки не резиновые. Но даже это не самое страшное!

Некоторые динамические библиотеки подгружаются лишь при строго определенных ситуациях, например, при нажатии на кнопку "Печать" или вызове определенного пункта меню. Естественно, до тех пор, пока данные действия не будут совершены, список DLL, "выдернутый" из адресного пространства, будет одновременно и избыточный, и неполный.

Короче, ситуация…

## *камеры наружного наблюдения или мониторинг файлов и реестра*

Утилиты для наблюдения за обращением к файлу и ветвям реестра (filemon и regmon соответственно, которые можно бесплатно скачать с сайта www.sysinternals.com, автоматически перенаправляющем нас в замок имении зла) достаточно популярны в среде хакеров. Казалось бы, что может быть проще — запускаем filemon/regmon и смотрим — куда лезет наша подопытная программа. Конечно, с полученной простыней протокола еще предстоит повозиться, выкидывая из нее повторные обращения, но уж это всяко проще, чем ковыряться в двоичном файле по <F3>.

На самом деле средства мониторинга это last resort, к которому прибегают, когда по другому перенести программу с одного компьютера на другой никак не получается. Дело в том, что всякая программа активно обращается и к тем ветвям реестра, которые сама не создает. Продемонстрируем это на примере популярного почтового клиента The Bat, протокол "общения" с реестром которого представлен ниже (см. рис. 6):

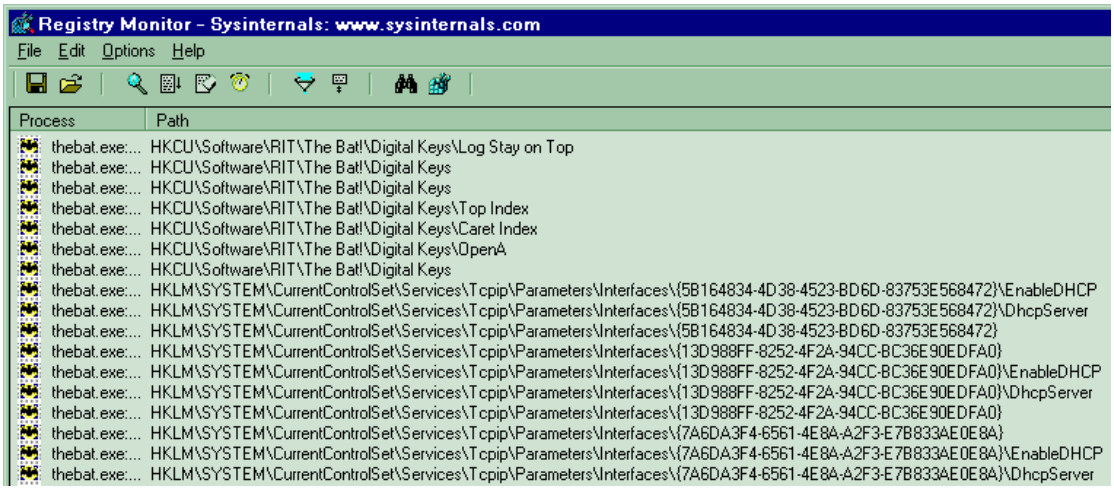


**Рисунок 6 протокол обращений к реестру программы The Bat**

Достаточно очевидно, что ветвь "HKCU\Software\RIT\The Bat!\Editor\Font Size" принадлежит саму The Bat'y и должна быть перенесена на соседний компьютер вместе с ним, а вот ветвь типа "HKLM\System\CurrentControlSet\Services\Tcpip\Parameters\Interfaces\{5B16484-4D38-4523-D6D-83753E568472}" уже является частью системы и относится к TCP/IP-стеку со всеми его интерфейсами, идентификаторы которых на разных компьютерах навряд ли будут совпадать.

Возникает резонный вопрос: как отделить зерна от плевел, то есть выделить лишь те ветви реестра (файла), которые были созданы самой программой при ее установке? Ответ прост. Копируем базовый каталог подопытной программы на соседнюю машину, на ней же запускаем монитор реестра и смотрим — с открытием каких именно ветвей она обламывается (в этом случае в статусе операции будет написано "ERROR" или "NOT FOUND"), однако, следует учесть, что некоторые ветви реестра отсутствуют не просто так, а по творческому замыслу разработчика программы. В этом случае на целевом компьютере (с правильно установленной программой) монитор реестра выдаст тот же самый результат — "NOT FOUND".

Перенос же ветвей реестра осуществляется элементарно, через его редактор. Просто выделяем требуемую ветвь, щелкнув по ней. В меню "Реестр" выбираем пункт "Экспорт файла реестра" и в появившемся диалоговом окне говорим, что хотим экспортировать только выбранную ветвь, а не весь реестр целиком. Мы получаем reg-файл, запустив который на соседней машине, добавляем эту ветвь в реестр. Порядок добавления ветвей произволен.

С файлам же все обстоит еще проще и они могут быть "выдернуты" даже без всяких мониторов. Достаточно воспользоваться поиском по дате.

## *следы времени на песке файловой системы*

Заходим в базовый каталог программы и смотрим когда был создан главный исполняемый файл (правая клавиша мыши → свойства), например, VMWare.exe, олицетворяющую собой одноименную виртуальную машину. На компьютера автора время ее установки равно "10 июня 2004 г., 18:12:30" (версия уже устарела как мамонт, но для мыщъх'а сойдет).

Давим Пуск → Найти → Файлы и Папки, и в параметрах поиска вводим Дата → Файлы, созданные → с 10.06.2004 по 10.06.2004. Говорим Фас! то есть "Найти" и… находим всю дичь, спрятанную не только в базовом каталоге VMWare, но и в каталогах C:\WINNT\System32 и C:\WINNT\System32\Drivers (см. рис. 7). Красота!!!!

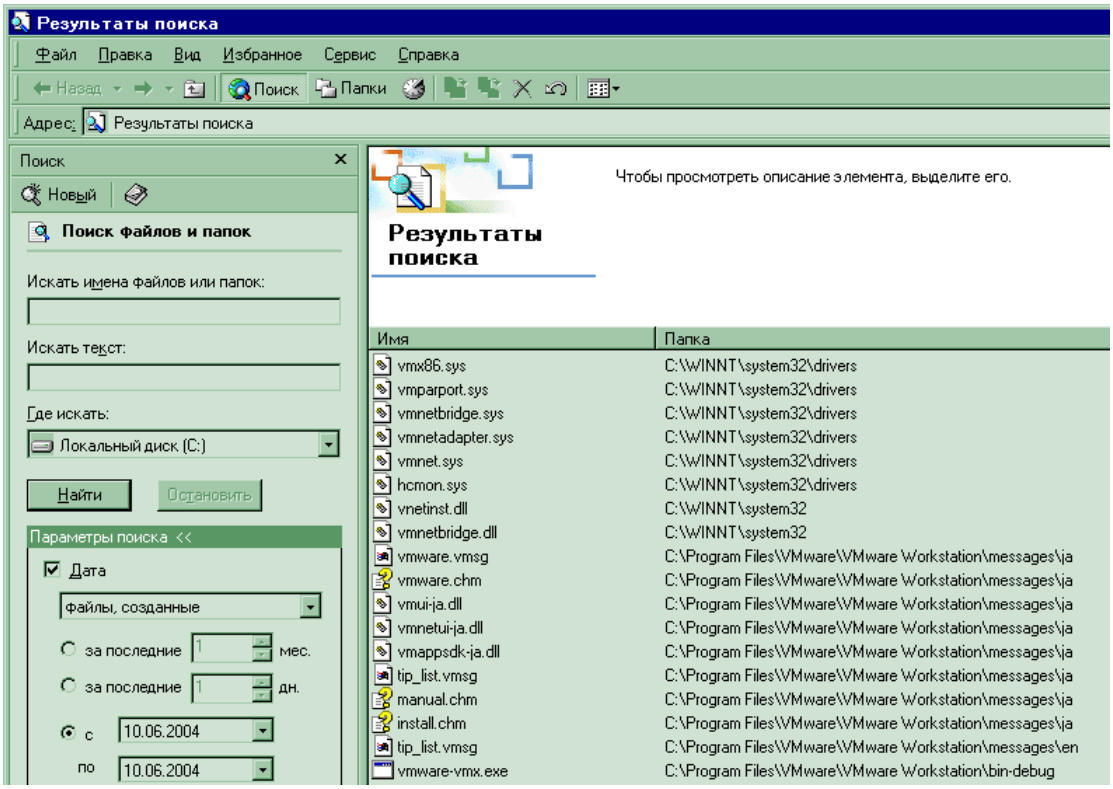


**Рисунок 7 ищем все файлы, принадлежащие VM Ware**

К сожалению, данный способ не лишен недостатков. Если две или более программы устанавливались в один и тот же день, то поиск по времени покажет их всех, поскольку, глупая оболочка Windows не позволяет задавать в критериях поиска минуты и секунды (да мы и не должны их задавать, ведь процесс установки иной раз занимает полчаса или около того).

Впрочем, просматривая "свойства" всех найденных файлов через контекстное меню (там время создания указывается с точностью до секунды) мы легко отсеем посторонних кандидатов, конечно, при том условии, что разные программы ставились с приличным разносом во времени. Если же установка проводилась на "конвейере" (то есть одна программа за другой), отделить какие файлы принадлежат какой программе практически невозможно.

Другой "тонкий" момент. Если на момент установки подопытной программы требуемые ей библиотеки уже имелись на компьютере (например, ".NET Framework"), то, очевидно, они не были установлены, а даже если и были, то… при перезаписи новых файлов поверх старых, время их создания не меняется (так уж устроен NTFS-драйвер). Как результат — после переноса на соседний компьютер программа может не найти каких-то библиотек и не запуститься. Каких именно — поможет выяснить файловый монитор, хотя в большинстве случаев удается обойтись и без него.

## *заключение*

Выдирание программ лишенных инсталлятора — процесс не то, чтобы сложный, но сильно утомительный, особенно если мы нарвемся на клинический случай, хотя в целом, никаких непреодолимых проблем на этом пути не возникает и специальных хакерских навыков не требуется. Решение обозначенной задачи по силам любому продвинутому пользователю!

## *>>> врезка регистрация OLE- и ActiveX-компонентов*

OLE- и AcriveX-компоненты (заключенные в файлах с расширениями .dll или .ocx) поимо переноса на соседний компьютер требуют обязательной регистрации (это не та регистрация, которая требует денег, это просто процесс их "прописки" в системном реестре). Без прописки будет сплошной мрак и полный облом.

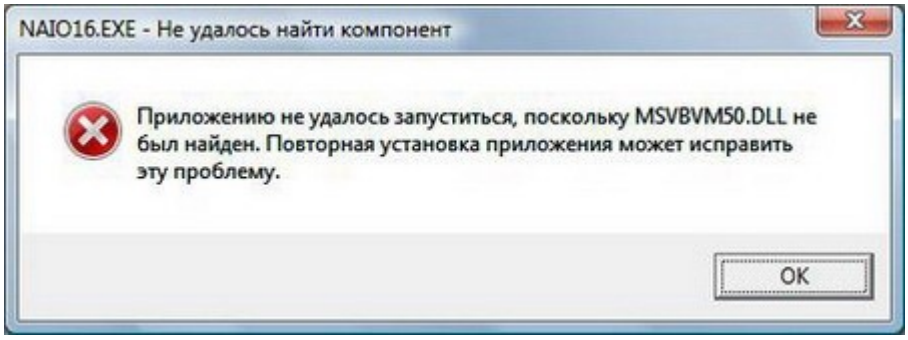


**Рисунок 8 библиотека установлена, но не зарегистрирована**

Регистрация осуществляется штатной утилитой regsvr32.exe запускаемой с именем регистрируемой библиотеки в командой строке. Где взять командную строку? Ну, ты что, братец?! Совсем ламер, да?! FAR, cmd.exe – к твоим услугам. А еще есть Total Commander, но в среде хакеров старого поколения он непопулярен. Как говорится, скажи мне кто твой коммандер и я скажу кто ты. Впрочем, мы крупно отвлеклись. Вернемся к нашим баранам.

Берем, значит, .dll или .ocx и передаем ее имя утилите regsvr32.exe в качестве параметра. Если регистрация прошла успешно, то все ОК, если же нет – мы увидим следующее ругательство (см. рис. 9). Беда в том, что если практически все .ocx являются OLE-компонентами, то в случае с DLL об этом не скажешь. Как узнать – кто из них кто? Без дизассемблирования (и даже без подглядывания в таблицу импорта) — только методом тыка!

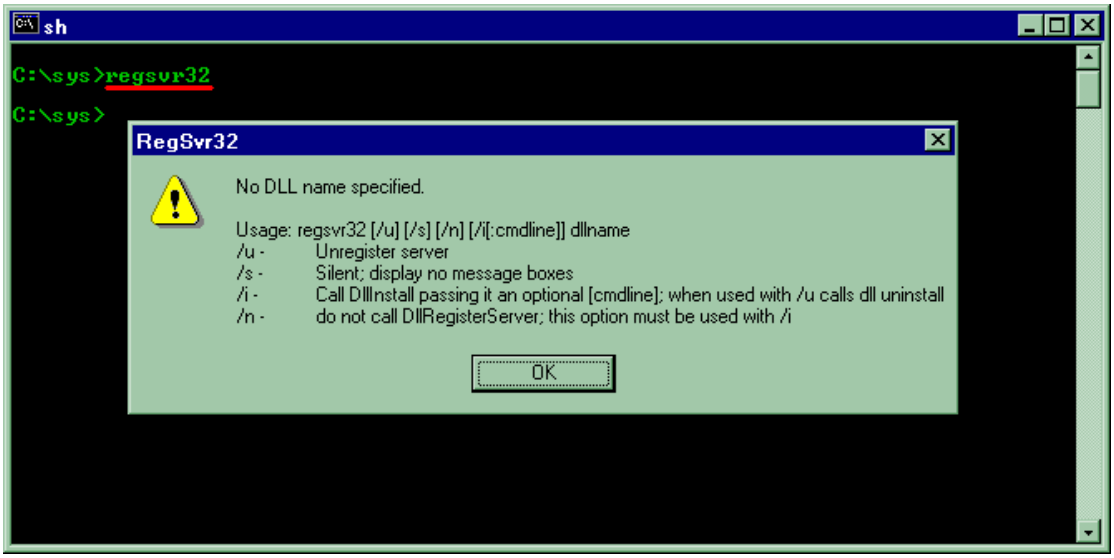


**Рисунок 9 регистрация OLE/ActiveX-компонентов**

Впрочем, процесс регистрации — обратимый и ключ "/u" удаляет зарегистрированный компонент из системы. К сожалению, описанный способ не универсальный и далеко не всегда срабатывающий. Некоторые компоненты требуют регистрации с ключом "/I", ожидающим увидеть строку параметров, которых мы не знаем и о которых даже не догадываемся. А ведь без правильной регистрации всех компонентов программа работать не будет!!!

Особенно много компонентов содержат программы, написанные на Visual Basic'e и DELPHI. Впрочем, не будем отчаиваться. Раз регистрация по сути своей сводится к созданию новых ключей в системном реестре, то все они могут быть найдены по методике, описанной выше. То есть, через монитор.