поиск малвари на Server 2003/XP своими руками

крис касперски, no-email

глобальных эпидемий не вспыхивало уже несколько лет, но локальные очаги заражения продолжают разрастаться, осваивая новые территории. хакеры проявляют большую активность и разработчики защитных комплексов (как правило, распространяемых на коммерческой основе) за ними, увы, не поспевают, увеличивая разрыв с каждым днем. pro-активные технологии элементарно обходится даже простейшими вирусами. root-kit'ы все глубже зарываются в ядра операционных систем и очень трудно найти администратора, ни разу не сталкивающегося с вредоносным программным обеспечением. баста! надоело! покажем как обнаруживать свыше 90% малвари бесплатным программным обеспечением (руки, голова и хвост по умолчанию прилагаются). главным образом мы будем говорить о Server 2003/XP, однако, сказанное во многом справедливо и для Server 2008/Vista[1]

введение

Антивирусы, изначально рассчитанные на борьбу с вирусами (то есть с саморазмножающимися программами, внедряющими свою, возможно слегка измененную копию, в другие файлы) к нашествию троянских коней оказались совершенно не готовы. Троянская программа никуда не внедряется, пишется (на основе готовых компонентов) чуть ли не за несколько минут, после чего обрабатывается упаковщиком/протектором и "сливаться" в Сеть, например, путем широковещательной/избирательной почтовой рассылки. К тому времени, пока пользователи заподозрят что-то неладное и пошлют файл в антивирусный центр на исследование (если еще найдут его на диске!), вредоносное программное обеспечение (далее по тексту именуемое малварью) уже сделало свое черное дело, например, разослало спам, нашло и передало конфиденциальную информацию, создало новую учетную запись, допускающую удаленное подключение хакера и т. д. Грамотно спроектированная малварь обычно уничтожает себя буквально через несколько минут, а то и секунд после запуска вредоносного файла.

Основную зацепку представляют готовые компоненты, используемые хакерами, чтобы не писать каждый экземпляр малвари с нуля. Теоретически их можно занести в базу сигнатур, детектируя заразу на ранних стадиях внедрения. Практически же, универсальный распаковщик, встроенный в антивирусы, справляется лишь с простейшими упаковщиками. Все, что посложнее – распаковывается набором статических распаковщиков, запрограммированных вручную и "тупо" повторяющих алгоритм оригинального упаковщика. Проблема в том, что этот алгоритм может быть легко изменен прямо в hex-редакторе. Хакеру достаточно перенести оригинальную точку входа (OEP – Original Entry Point) в другое место, чтобы обхитрить антивирус или воткнуть в начало файла конструкцию, вызывающую у антивируса несварение желудка.

Обычно это что-то из набора команд SSE/SSE2/SSE3 — в настоящий момент их не эмулирует ни один антивирус, а переменная длинна x86 команд не позволяет продолжить декодирование инструкций, после встречи с перовой же неизвестной командой. Тоже самое относится и к самомодифицирующемуся коду (особенно, выполняющемуся на стеке), структурным исключениям… Да что там говорить, хакеру достаточно "закрутить" цикл, исполняющийся секунду или около того, чтобы эмулятор антивируса "отвалился" по тайм-ауту. Никакой антивирус не эмулирует кодов ошибок, возвращаемых API-функциями, которые могут быть использованы для расшифровки остального тела файла (скажем, малварь открывает заведомо несуществующий файл, но антивирус-то этого не знает! а "подобрать" правильный ключ расшифровки — выше его сил!).

Таким образом, хакеру достаточно взять широко распространенную зловредную программу, обработать ее новейшим упаковщиком/протектором, слегка изменив код последнего так, чтобы статический антивирусный распаковщик не смог его распаковать и все! Антивирус будет молчать как партизан! Что же касается C# программ, то их вообще элементарно декомпилировать штатным дизассемблером ildasm.exe, слегка модернизировать CLR-код и

[1] извините, я не помню какие у вас требования к оформлению статей и какой максимальный объем аннотации к статье, если она слишком велика, скажите до какого кол-ва символов ее необходимо сократить

ассемблировать его заново штатным же ассемблером ilasm.exe — антивирусы вновь отдыхают, а хакеры размножаются со страшной силой — легкость программирования, большое количество готовых руководств, компонентов и библиотек вызывает активный приток "пионеров" всех мастей, в то время как еще буквально 5-10 лет назад написание вируса представляло собой нетривиальную задачу, требующую глубокого знания внутренностей операционной системы, ассемблера и… уймы свободного времени, которое нечем занять.

И хотя некоторые антивирусы (такие, например, как Norman) "прогоняют" подозреваемые файлы через своеобразную "песочницу" (sandbox) — в грубом приближении представляющую собой эмулятор операционной системы, с тщательной проверкой изменений состояния реестра и файловой системы после завершения работы подопытной программы, они не панацея, а очередное "плацебо" для успокоение пользователей, что "в Багдаде все спокойно". Достаточно сказать, что экзотические API-функции типа NtSetLdtEntries() ни сам Norman, ни его "коллеги" не эмулируют, тоже самое относится и к NtSystemDebugControl(), что позволяет зловредному коду в лучшем случае остаться незамеченным, а в худшем — вырваться из-под контроля эмулятора, попав на живую машину со всеми вытекающими отсюда последствиями.

А чего вы хотели? Антивирус — тупая машина. Мыслит шаблонно и всякий нормальный хакер обязательно проверяет созданный им вредоносный код на большой коллекции свежих антивирусов, совершенствуя алгоритмы маскировки до тех пор, пока антивирусы не перестанут его обнаруживать. Мораль: поиском заразы должен заниматься человек, чей живой и неординарный ум оперирует отнюдь не шаблонами, а логикой, размышлениями, интуицией. Близкое знакомство с малварью показывает, что подавляющее большинство вредоносных программ элементарно обнаруживаются буквально с первого взгляда. И не нужно платить никаких лицензионных отчислений, постоянно качать самые свежие базы, мириться с тормозами pro-активных технологий и прочих прелестями прогресса. Мы будем рассматривать только бесплатные общедоступные утилиты и простые технологии обнаружения малвари, не требующие глубокого знания внутренностей системы и умения читать дизассемблерный код. А раз так, чего мы сидим! Вперед!

Рисунок 1 фото автора

>>> врезка укрепляем линию обороны

Чтобы затруднить атаку на систему рекомендуется заблаговременно предпринять ряд несложных действий. Во-первых, переименовать ядро (файл ntoskrnl.exe) во что-то другое (например, nezumi.exe), а вместо ntoskrnl.exe положить ядро от другой версии системы (не забывая обновить его и в кэше SFC, иначе весь труд пойдет насмарку и она тут же его восстановит). Фокус в том, что для определения адресов перехватываемых функций многие rootkit'ы вызывают функцию LoadLibrary("ntoskrnl.exe"), даже не догадываясь о том, что ядро переименовано и полученные адреса весьма далеки от действительности! Корректно написанный rootkit в таком случае просто откажется от перехвата, некорректный (коих большинство) грохнет систему, что неприятно, но все-таки лучше заражения.

А как же сообщить самой системе где искать новое ядро? Очень просто — добавить в файл boot.ini "волшебный" ключик "/kernel=", после чего строка с описанием операционной системы будет выглядеть приблизительно так:

```
multi(0)disk(0)rdisk(0)partition(1)\WINNT="Server 2003" /fastdetect /kernel=nezumi.exe
```

Листинг 1 фрагмент "защищенного" boot.ini файла

Только не забудьте перед установкой пакетов обновления вернуть прежнее ядро на место, иначе инсталлятор аварийно завершит свою работу, поскольку, он (как и rootkit'ы) не догадывается о возможности переименования ядра.

Кстати, большинство создателей вредоносных программ свято верят в то, что Windows _всегда_ стоит на диске "C:" и нагло используют абсолютные пути, даже не заглядывая в соответствующие переменные окружения. Следовательно, поставив систему на любой другой диск (например, "D:" или "F:") мы отсечем часть зловредных программ еще на подлете. Windows Update, кстати говоря, обрабатывает такую ситуацию вполне адекватно.

закулисные игры автозагрузки

Чтобы "окопаться" на вражеском компьютере, малварь должна пустить корни, прописав себя в автозагрузку. Разумеется, термин "автозагрузка" в данном контексте очень условен — это может быть и сетевой фильтр, и сервисная служба, и расширение к Internet Exploder'у — в системе существует _сотни_ мест, в которые можно незаметно внедриться, получая управления сразу же после загрузки Windows или при наступлении определенных событий (запуска IE или другого приложения).

Просматривать все ключи реестра, прямо или косвенно связанные с автозапуском, не только утомительно, но и нецелесообразно, поскольку, "невооруженным глазом" очень трудно отличить легальные компоненты от нелегальных. Что делать?! И тут на помощь приходит бесплатная утилита Autoruns от Марка Руссиновича, показывающая огромное количество "злачных" мест, куда любит внедряться вредоносное программное обеспечение (см. рис. 2).

Для облегчения восприятия вся информация разбита на 16 вкладок (точнее — 15, 16'я вкладка включает содержимое остальных). Но даже такое изобилие отнюдь не гарантия полноты. Увы! Существуют десятки мест вполне подходящие для внедрения, которые Autoruns не показывает. Например, малварь может вклиниться в цепочку ассоциаций расширений. Скажем, при открытии .doc файлов управление получает не MS Word, а зловредный файл, делающий свое черное дело и запускающий оригинальный MS Word, так что с точки зрения пользователя не происходит ничего подозрительного. Тоже самое относится и к пунктам контекстного меню. Допустим, у нас есть пункт "Add to zip/rar", но вместо архиватора сначала запускается малварь (подменившая эту ветвь в реестре) и только потом передающая управление оригинальной программе.

Так вот! Всего этого Autoruns не показывает!!! Это вовсе не призыв к отказу от его использования (более достойных утилит мне пока не попадалось), просто, к полученным результатам следует относится со здоровой долей скептицизма, не забывая о необходимости ручной проверки системы.

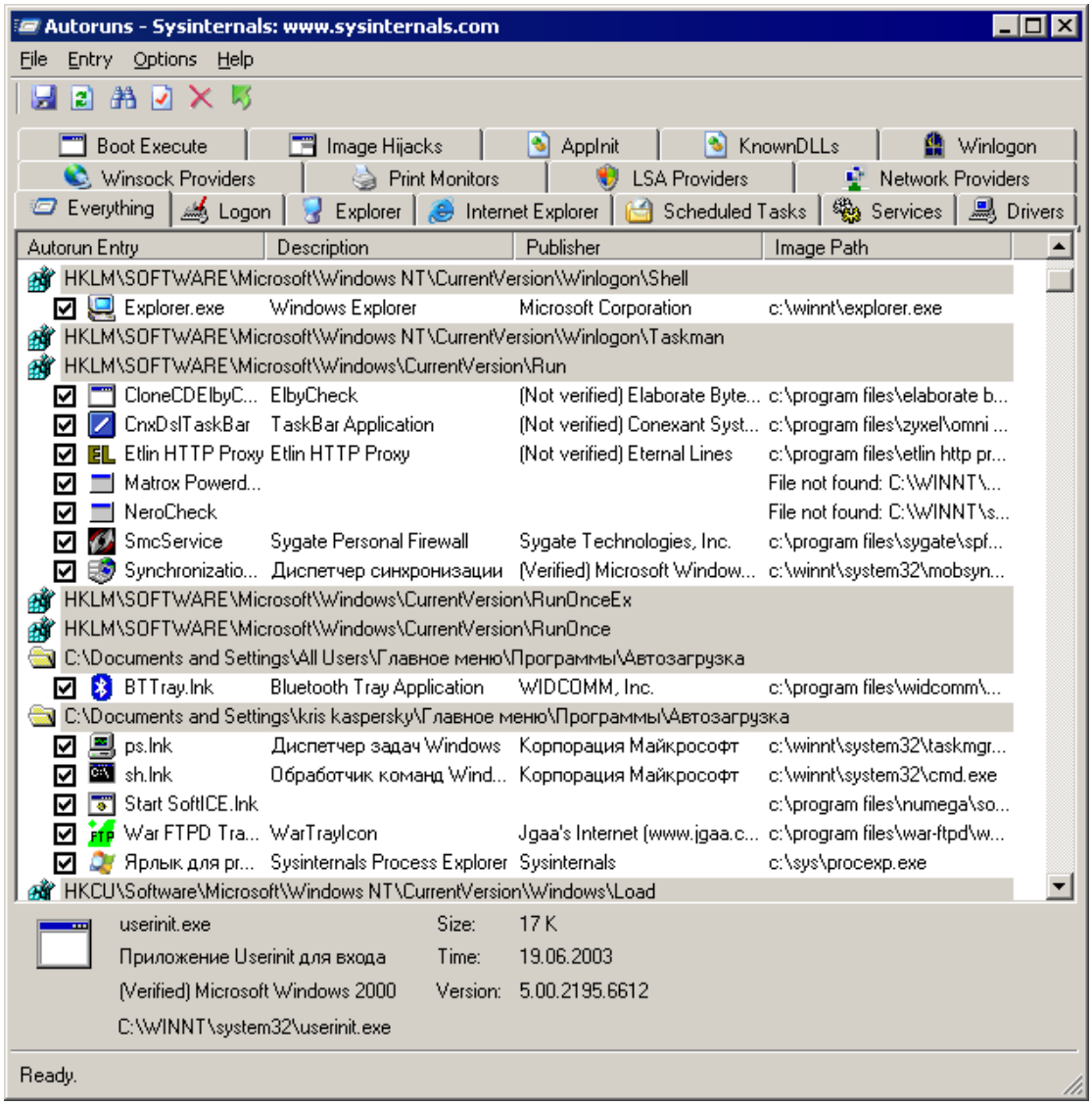

Рисунок 2 внешний вид утилиты Autoruns от Марка Руссиновича

Самым ценным свойством программы является возможность сохранять список элементов автозагрузки в текстовой файл, а затем в любой момент времени сравнивать его с текущим (File → Save As, File → Compare). "Новоявленные" элементы (см. рис. 3) помечаются зеленным цветом (хм, почему зеленым, а не красным?! ведь самое время бить тревогу!). Сравнение списков очень мощное оружие, помогающие обезвреживать не только малварь, но и некорректно написанные программы, лезущие своими грязными лапами туда, куда их не просят.

К минусам утилиты Autoruns можно отнести невозможность просмотра всех пользовательских профилей, даже будучи запущенной из-под администратора она показывает только элементы автозагрузки _текущего_ профиля, но никогда — всех профилей сразу. Правда, при запуске из под администратора у нас появляется меню "User" со всеми пользователями, зарегистрированными в локальной системе, и для получения достоверного результата нам необходимо сохранить в log-файлы их все. Главное, потом не запутаться какой log какому пользователю принадлежит, иначе можно получить очень неожиданный результат.

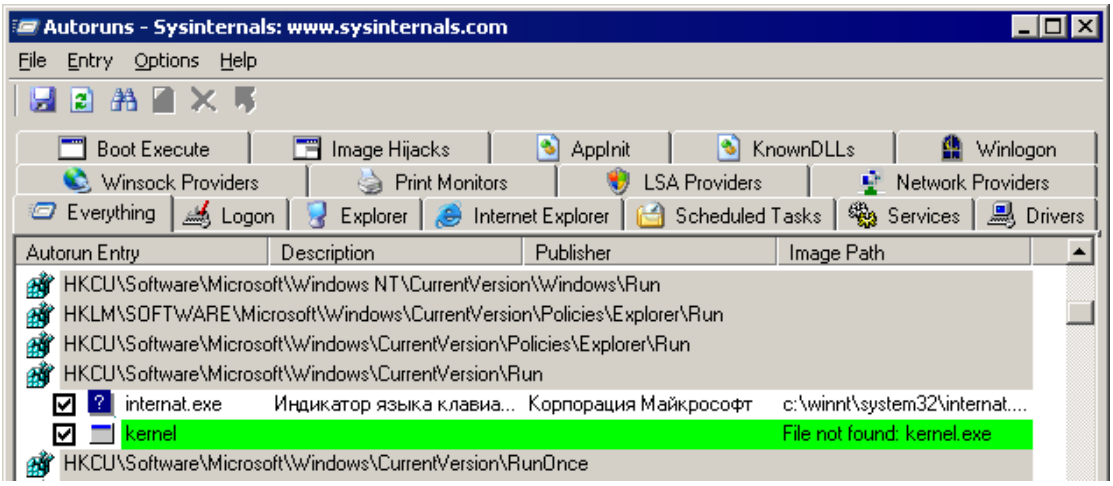

Рисунок 3 сравнение логов элементов автозагрузки позволяет выявлять внедрение новой малвари

Рядом с каждым элементом автозагрузки находится галочка, сброс которой приводит к отключению данного элемента, причем, программа хранит состояние всех элементов и при желании отключенные элементы можно включить вновь, если конечно, их отключение не приведет к краху системы так, что нам вообще не удастся загрузится и тогда придется прибегнуть к танцам с бубном. Отключенные элементы автозапуска хранятся непосредственно в самом реестре, в частности, элементы из ветки \Software\Microsoft\Windows\CurrentVersion\Run\ попадают в подветку AutorunsDisabled, что необходимо помнить при запуске "Консоли восстановления" для подъема рухнувшей системы, а лучше — перед всякими экспериментами просто зарезервировать реестр любой из многочисленных утилит, коих за последнее время развелось…

Другая полезная функция — верификации цифровой подписи от Microsoft, для этого достаточно щелкнуть правой клавишей мыши по интересующему нас элементу автозагрузки и сказать "Verify". Если цифровая подпись совпадает, то рядом с именем программы появляется статус (Verified), означающий, что данному компоненту автозагрузки можно на 99% доверять. К сожалению, с компонентами от сторонних разработчиков верификация не работает, но с другой стороны в _большинстве_ случаев ничего плохого не случиться если их временно отключить и посмотреть, что у нас упало. Ага, перестал работать Proxy-сервер или ADSL-модем. Значит, все правильно, полет нормальный и можно возвращать элемент автозагрузки на место. А вот если ничего не упало, то следует призадуматься, выбрав в том же контекстом меню пункт "Search Online" и поискать, что пишут о данном файле на WEB. До тех пор, пока System Internals не продалась Microsoft'у поиск работал через Google, теперь же он работает через MSN, то есть никак не работает (во всяком случае у меня) и приходится либо вбивать имя exe-файла в строку поиска Google вручную, либо изменять параметр "onlinesearchcommand" в ветке реестра программы Autoruns с MSN на Goodle и радоваться прогрессу, весне и автоматизации.

Обычно, малварь прописывает себя в фиксированные каталоги под фиксированными именами и поиск на WEB'е оказывается довольно плодотворным, хотя и дает большое количество ложных срабатываний от ложно-позитивных, до ложно-негативных. Однако, при возникновении подозрений, файл можно немедленно отправить на www.virustotal.com — специальную on-line службу, собравшую под одной крышей пару десятков популярных антивирусов со свежими базами. Естественно, как мы уже говорили выше, верить антивирусу может только наивный, но дополнительная проверка никому не помешает.

Так же стоит обратить внимание, что далеко не все элементы, причастные к автозагрузке, на самом деле являются таковыми. В частности, содержимое ветки реестра HKLM\System\CurrentControlSet\Services перечисляет драйверы которые _могут_ быть запущены вызовом ZwLoadDriver или каким-либо другим путем, однако, их присутствие в данной ветке само по себе еще не приводит к автоматической загрузке драйвера, поэтому, ее следует рассматривать лишь как потенциальную угрозу. Обратная ситуация наблюдается с веткой HKLM\System\CurrentControlSet\Services, в которой Autoruns показывает лишь запущенные службы, "забывая" упомянуть службы, загружаемые вручную или вызываемые из других служб при наступлении определенных обстоятельств.

Наведя курсор на любой файл и нажав <ALT-ENTER> мы получим стандартное диалоговое окно со свойствами, среди которых иногда можно обнаружить цифровую подпись (если она есть), а так же имя производителя и прочие данные, содержащиеся в секции ресурсов исполняемого файла/динамической библиотеки и, естественно, подверженных угрозе фальсификации. Если цифровой подписи нет (а чаще всего ее нет), то кто угодно может написать здесь что угодно, однако, как показывает практика, хакеры крайне редко утруждают себя подделками, а если и пытаются выдать малварь за продукцию известной компании, то подделывают ее имея столь неумело, что тут же разоблачают себя.

А вот что действительно важно — из диалога "свойств" можно извлечь дату создания исполняемого файла/динамической библиотеки на диске. Если, допустим, вы никаких программ уже полгода как не устанавливали, и тут вдруг появляется файл, созданный совсем недавно, это наводит на размышления (естественно, малварь может изменить дату своего создания, но пока с такой "умными" вирусами мне сталкиваться еще не приходилось) и чтобы исследовать его свойства, достаточно вызвать "Process Explorer" из контекстного меню. Это еще одна утилита от Марка Руссиновича. Очень мощная и очень полезная в плане поиска малвари. И на этой ноте мы заканчиваем описание Autoruns и переходим к Process Explorer'у.

Process Explorer

Грубо говоря, "Process Explorer" это аналог "Диспетчера Задач", только намного более мощный (см. рис. 4). Искать малварь с его помощью — одно удовольствие. Первое, что мы видим при запуске — это список процессов (включая процессы, не отображаемые в "Диспетчере Задач"). Многие rootkit'ы предпринимают неслабые усилия для удаления себя из списка процессов и в этом плане доверять Process Explorer'у нельзя. Эх, взять бы отладчик Soft-Ice, напрямую работающий с низкоуровневыми структурами ядра и не оставляющий rootkit'ам никакого шанса, но… во-первых, Soft-Ice это платный продукт, да к тому же еще и мертвый. Старые версии не работают под Server 2008/Vista, а новых не предвидится…

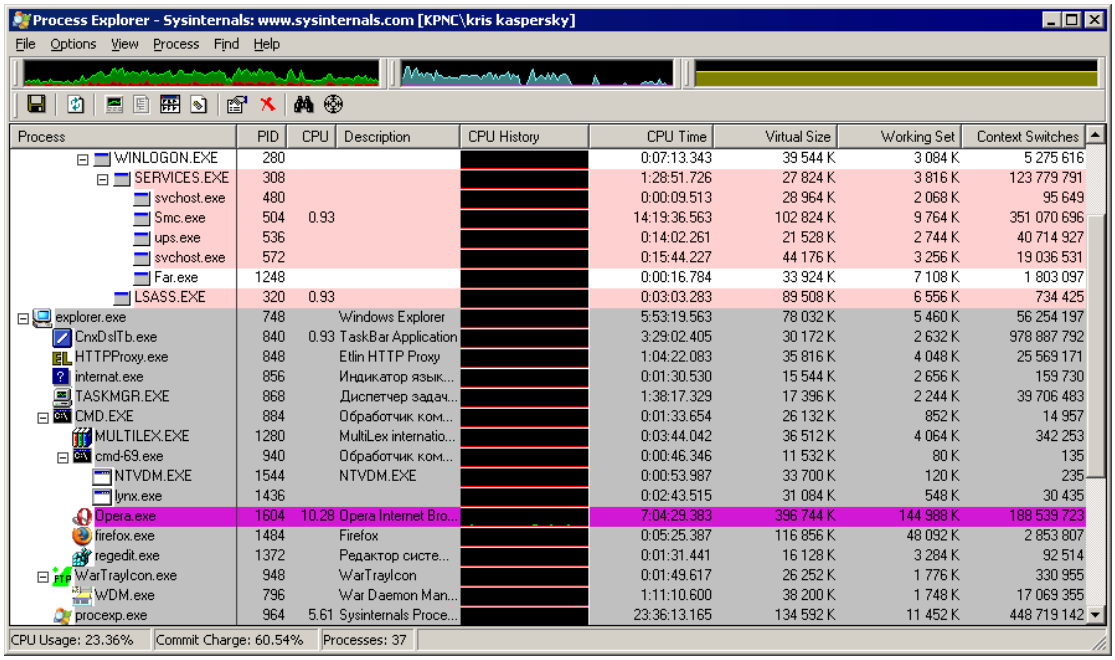

Рисунок 4 внешний вид утилиты Process Explorer

Тем не менее, здесь есть один очень хитрый трюк, способный разоблачить большое количество rootkit'ов. Дело в том, что операционные системы семейства NT поддерживают два различных механизма перечисления процессов — средства TOOLHELP32, выдающие информацию по _каждому_ процессу, и счетчики производительности (performance counters), подсчитывающие общее количество процессов в системе, обмануть которые намного сложнее, да и существующие rootkit'ы даже не пытаются это делать (наверное, просто не знают, что есть такая лазейка).

Жмем <CTRL-I> для вызова диалогового окна "System Information" (см. рис. 5), где и видим показания счетчиков производительности: "Processes: 38", в то время как статусная

строка Process Explorer'a насчитывает всего 37 процессов, то есть на один процесс меньше! Кто не верит — может сосчитать процессы вручную!

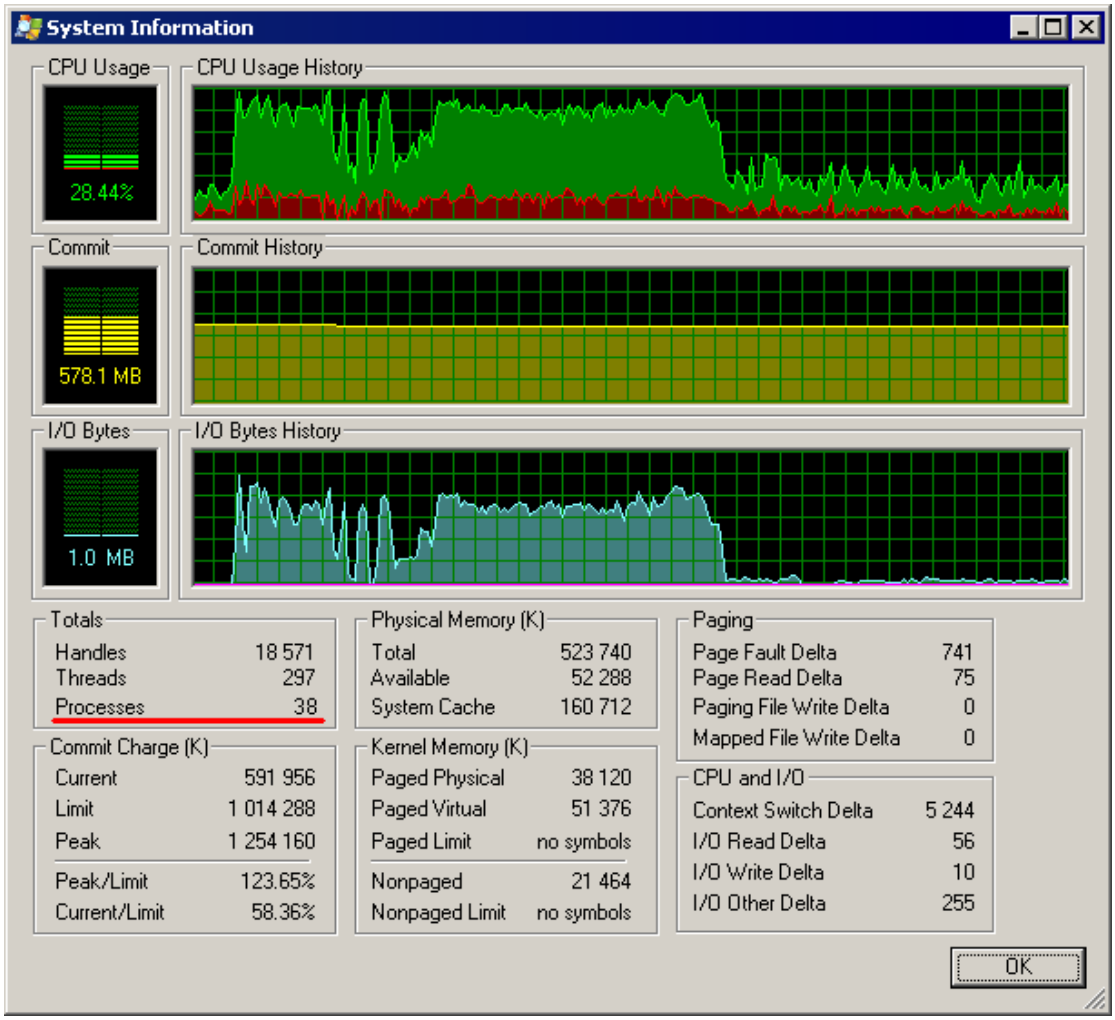

Рисунок 5 счетчики производительности в окне "System Information" — еще один источник данных о количестве процессов в системе

Расхождение в показаниях указывает на наличие Rootkit'a, который в данном случае действительно имеет место быть (причем, ни KAV, ни DrWEB, ни NOD32 с самыми свежими базами его так и не обнаружили, хотя он был выловлен в дикой Сети более полугода назад — вот и верь после этого антивирусам). Но хватит лирики. Ближе к делу. Как же поймать этот неуловимый rootkit? Исходя из самых общих рассуждений: раз rootkit поселился в системе, то, вероятно, он нашел способ получать управление при загрузке и с определенной степенью вероятности один из элементов автозапуска (о которых мы говорили выше) и есть наш rootkit, так что просмотрим представленный список еще раз. И внимательнее! Исключение составляют rootkit'ы, проникающие через дыры в системе и обитающие исключительно в оперативной памяти, умирая сразу же после перезагрузки, кстати, говоря, это довольно распространенное исключение и количество таких rootkit'ов неуклонно растет, особенно на серверах, которые не перезагружаются неделями или даже месяцами (личный рекорд моего домашнего сервера — полгода без перезагрузки). У Руссиновича на этот счет есть специальная утилита Rootkit Revealer (о ней мы поговорим чуть позже), которая может помочь. А может… и не помочь. К сожалению, очень трудно порекомендовать рецепт универсального противоядия против всех rootkit'ов и остается только радоваться, что 90% - 96% малвари ловится буквально голыми руками.

Наведя курсор на процесс и вызвав контекстное меню, мы можем: а) убить процесс (иногда это требует, чтобы Process Explorer был запущен с правами Администратора); б) приостановить процесс, вогнав его в сон — кстати, не совсем безопасная операция, многие честные программы после пробуждения отказываются работать, а зловредное программное

обеспечение зачастую держит "теневой" процесс-монитор, отслеживающий остановку/смерть основного и автоматически перезапускающий его вновь, что немедленно разоблачает вредоносную сущность последнего.

Еще можно перезапустить процесс, вызывать JIT отладчик (если он установлен в системе и если вы разбираетесь в ассемблере), повысить/понизить приоритет процесса, найти главное окно приложения, автоматически переключившись на него, поискать имя процесса на WEB'е и… наконец, вызывать, пункт "Properties" (свойства), в котором, кстати говоря, и собрано большое количество инструментов, полезных для поисков малвари (см. рис. 6).

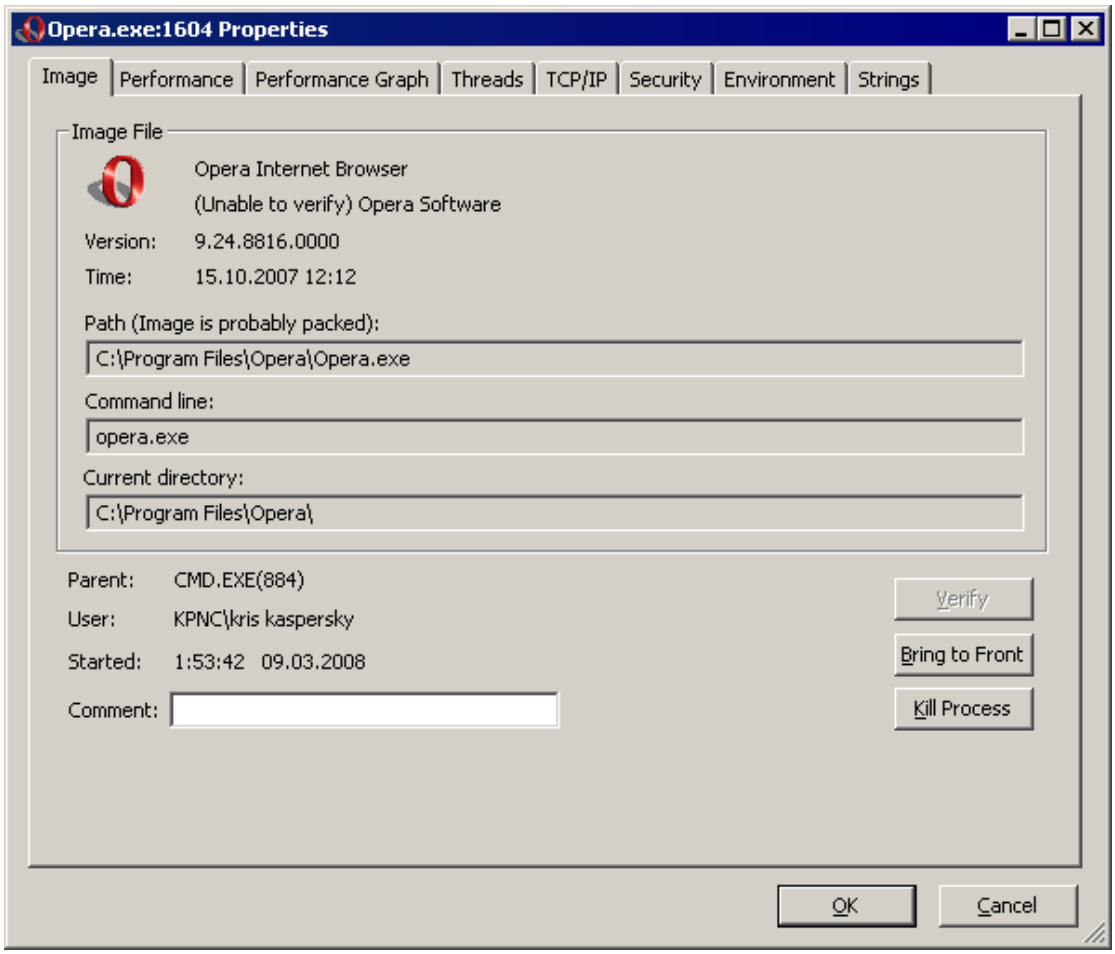

Рисунок 6 вкладка "Image" – основные свойства процесса

На первой (слева) вкладке "Image" перечислены основные свойства образа исполняемого файла: название (взятое из ресурсов — если оно там есть), путь к исполняемому файлу, процесс-родитель, пользователь, запустивший этот файл на выполнение и т. д. Если же всех этих параметров нет, значит, у Process Explorer'а не хватает прав для их получения — запустите его под Администратором, а в некоторых случаях и пользователем типа System, для чего можно воспользоваться системным планировщиком, вызываемым утилитой at.exe, входящей в штатный комплект поставки. Если же и под System'ом данные о файле получить не удается — кто-то очень сильно постарался, чтобы скрыть их от нас. Кто бы это мог быть? Явно не пингвин (кстати, пингвины очень больно кусаются), а нечто более воинственное типа малвари.

Другой тонкий момент. Допустим, обнаружили мы подозрительный файл, определив путь запуска, заглянули туда FAR'ом, а там… нет ничего такого!!! Вариантов два: либо это rootkit, скрывающий свое присутствие на диске и тогда попытка создания exe файла с заданным именем в заданном месте вернет ошибку. Либо же… активный процесс не может удалить себя (система блокирует файл от записи и удаления), но разрешает его переименование, следовательно, малварь могла переименовать себя во что-то другое, причем, переименование действует в пределах всего дискового тома, а не только текущего каталога!!! Другими словами, подлинный путь запуска зловредной программы определить непросто! А без этого мы не можем

ни удалить ее с диска, ни отослать на www.virustotal.com для проверки! К сожалению, универсального решения данной проблемы нет. Отсутствие файла по указанному пути — верный признак малвари, но вот где эту малварь искать — хороший вопрос!

Вкладка "Performance" содержит данные о памяти, потоках, и операциях ввода/вывода, выполняемых процессом, что позволяет (приблизительно) определить его род занятий. В частности, интенсивный ввод/вывод программы, замаскированной под русификатор наводит на размышления и эти размышления отнюдь не в пользу программы.

"Performance Graph" – основные данные счетчиков производительности, представленные в графической форме. Нас, главным образом, интересует нижняя диаграмма, отображающая ввод/вывод, без которого не обходится практически ни одна малварь, сканирующая диск в поисках конфиденциальной информации или рассылающая спам от нашего имени, или… что бы она ни делала, без ввода/вывода ей не обойтись.

Вкладка "Threads" перечисляет потоки процесса вместе с базовыми адресами их создания, позволяя "убивать" или "замораживать" ненужные нам потоки. Малварь активно использует потоки для внедрения в адресное пространство "доверенных" процессор, таких, например, как Opera.exe, FireFox.exe, чтобы получить беспрепятственный доступ в Сеть и не создавать еще один процесс, привлекающий к себе внимание. Теоретически, можно было бы запомнить сколько потоков имеет тот или иной процесс, а при появлении новых — карать их нещадно, но к сожалению, многие приложения создают потоки динамически, по мере надобности. Тоже самое относится к драйверам (в том числе и драйверам звуковых карт), поскольку создание потока в контексте процесса — один из немногих способ взаимодействия ядерного кода драйвера с прикладным кодом и API (прямой вызов диалоговых окон из драйвера невозможен и ему волей-неволей приходится пробивать "тоннель" в адресное пространство прикладных процессов, впрочем, мы отклонились от темы).

Потоки, созданные малварью, в 96% — 99% случаев располагаются в области динамической памяти (так же называемой кучей), в то время как легальные потоки — в тех же 99% дислоцируются внутри страничных образов динамических библиотек или самого исполняемого файла. Главное — это правильно определить стартовый адрес потока, чего Process Explorer делать не умеет и хотя автор этой статьи указал Марку Руссиновичу на данную ошибку больше года назад, выслав демонстрационный "вирусоподобный" код вместе с кодом для определения подлинных стартовых адресов, Руссинович оставил все как есть, чем существенно затруднил поиск вредоносного программного обеспечения.

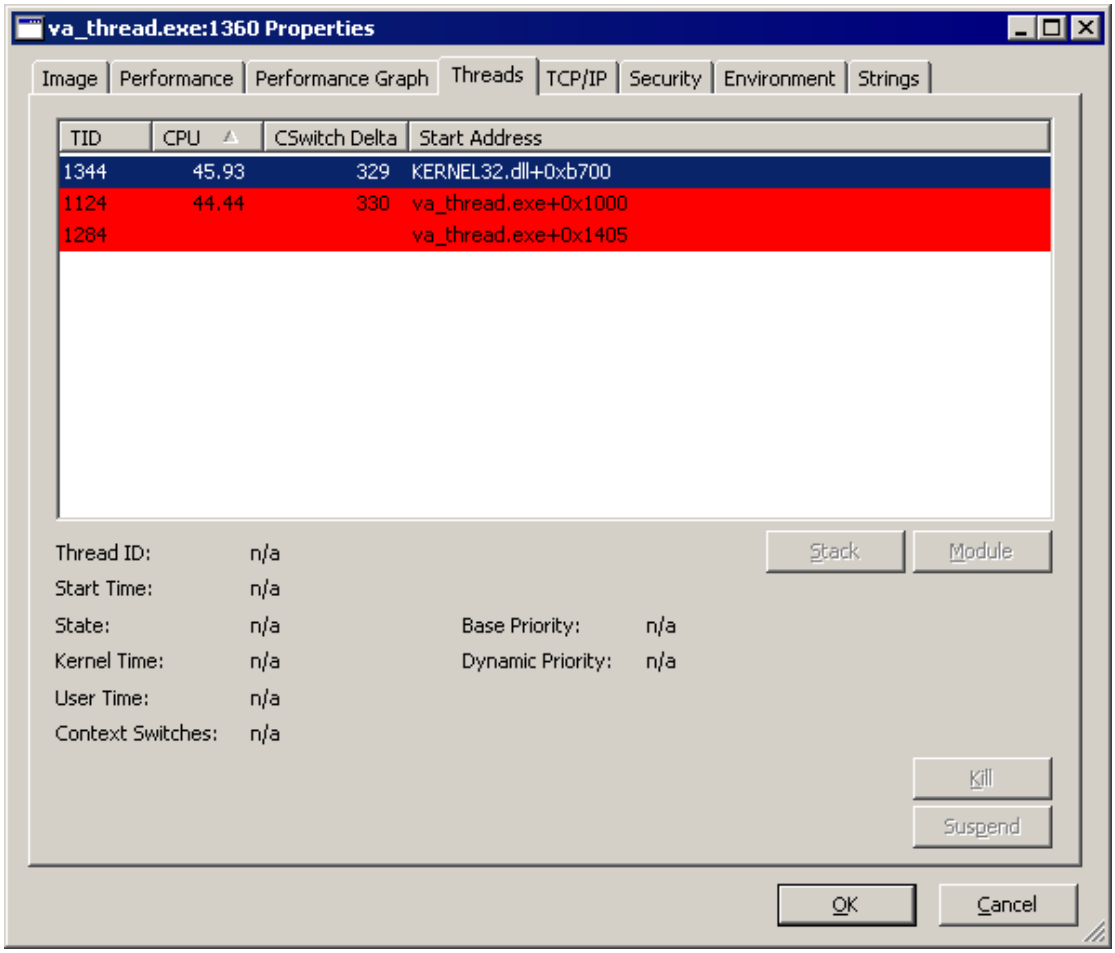

Рисунок 7 Process Explorer до сих пор неправильно определяет истинные стартовые адреса потоков

Но кое-какие зацепки у нас все-таки есть. Потоки, созданные с помощью функции CreateRemoteThread (а в большинстве случаев используются именно она), согласно Руссиновичу, имеют стартовый адрес KERNEL32.DLL+0xB700 (где 0xB700 – смещение кода, лежащего внутри API-функции CreateRemoteThread, и, естественно, слегка отличающее в зависимости от версии Windows). Самое противное, что многие "честные" программы так же используют CreateRemoteThread, причем, на вполне легальных основаниях. Так в чем же все-таки разница?! Наводим курсор на поток и нажимаем кнопку "Stack" – у честных программ там будет куча вызовов, ведущая к RPCRT и другим библиотекам, а вот у малвари стек обычно пуст. Описанный метод, конечно, не самый надежный, но… имея под рукой только Process Explorer без всяких отладчиков, лучшего, пожалуй, и не придумаешь. Прибив зловредный поток кнопкой "kill" или заморозив его по "Suspend" смотрим на реакцию приложения — отсутствие реакции с нормальным продолжением работоспособности укрепляет нашу уверенность, что мы имеем дело именно с малварью.

Вкладка "TCP/IP" содержит список активных сетевых соединений, легко "палящий" примитивные зловредные программы. Действительно, если мы установили игрушку типа "Тетрис", а она внезапно слушает какой-то порт, ожидая подключений — это же back-door, причем явный. Естественно, продвинутая малварь не даст себя так просто обнаружить и остается утешаться только тем, что ее количество относительно невелико.

Вкладки "Security" и "Environment" мы пропускаем, как не содержащие ничего интересного, а вот на String'ах задержимся надолго. Казалось бы, что можно найти в списке ANSI строк?! Ничего интересного… А вот и нет! Во-первых, список берется не из исполняемого файла (в случае малвари, как правило, упакованного), а из его распакованного образа в памяти. Практически никто из современных хакеров не утруждает себя шифрованием строк в runtime и эти самые строки содержат не только имена API-функций, с которыми работает программа, не только ключи реестра, но и различные высказывания (как правило, оскорбительного характера), адреса серверов на которые следует отсылать добытую информацию — да много еще чего!!!

Строки — верный ключ к выявлению малвари. Исключение составляет, пожалуй лишь зловредные программы, упакованные сложными протекторами с динамической распаковкой, но таких не так уж и много.

Rootkit Revealer

Никому ненужная игрушка, которая реально ничего не ловит, а только создает видимость спокойствия. Марк Руссинович утверждает, что Rootkit Revealer обнаруживает все rootkit'ы, выложенные на www.rootkit.com, что не соответствует истине, поскольку на www.rootkit.com имеется множество rootkit'ов успешно борющихся с Rootkit Revealer'ом теми или иными способами, о существовании которых Руссинович наверняка знает, но делает вид, что все идет по плану. К тому же на www.rootkit.com выкладываются преимущественно "сырые" proof-of-concept программы, полноценные версии которых распространяются на коммерческой основе или же используются для целенаправленных атак.

Что делает Rootkit Revealer? Он сравнивает результаты сканирования компьютера через высококровные API-функции и через низкоуровневые native-API. Rootkit'ы, работающие "посередине", естественно, обнаруживаются за счет разницы в "показаниях", однако, существует множество rootkit'ов, которые трудно поймать даже на уровне ядра, не говоря уже о том, что львиная доля малвари даже не пытается маскироваться, а потому Rootkit Revealer _принципиально_ не в состоянии ее обнаружить.

Тем не менее, учитывая бесплатность данной утилиты, высокую скорость сканирования (с поддержкой удаленного сканирования по сети) — почему бы ее и не попробовать. А вдруг случиться чудо и она поможет?!

заключение

Список достойных утилит для поиска малвари этим не ограничивается, их можно найти как на сайте Марка Руссиновича (www.sysinternals.com, автоматически переправляющего нас на сайт Microsoft), так и взять из других источников. В конце концов, всякая утилита — это всего лишь инструмент, возможности которого напрямую зависят от мастерства его обладателя. Опытный администратор обнаружит малварь и голыми руками, начинающего не спасет и набор дорогостоящих защитных комплексов.

>>> врезка удаленная проверка системы

Администратор должен заботится не только о здоровье сервера, но и всего "зоопарка" вверенных ему машин (основных рассадников малвари), но скакать по всем этажам, отвлекая пользователей от текущих дел — это же какую выдержку нужно иметь! Спортивная подготовка так же не помешает. Если же всего этого нет — вероятно, стоит задействовать возможности удаленного администрирования, выполняя "медосмотр" узлов локальной сети, не покидая пределов своего любимого кресла и попивая кофе (чай, квас) из огромной черной кружки с надписью "root".

Существует множество путей решения задачи. Во-первых, это штатный telnet-сервис, входящий в комплект поставки начиная с W2K, а, быть может, и раньше. Telnet (с некоторыми ограничениями) поддерживает работу консольных приложений, но это еще ничего. Хуже другое — он должен быть _изначально_ запущен на всех рабочих станциях (что отличается от его состояния по умолчанию). Тоже самое относится к многочисленным утилитам сторонних разработчиков, многие из которых поддерживают и графические приложения (неожиданное появление которых на экране страшно нервирует пользователей) и ни одна из известных мне утилит не позволяет запустить заданную программу на _всех_ узлах локальной сети, не делая никаких дополнительных телодвижений.

PsExec от Марка Руссиновича (входящая в комплект PsTools) — относится к тем приятным исключениям, которые поддерживают удаленный запуск программ на всех (или заданных) узлах, при необходимости автоматически копируя программу на удаленный узел.

Однако, в ее использовании немало скрытых подводных камней. Запускающий ее пользователь должен принадлежать к группе администраторов и, что самое важное, на удаленной машине необходимо заранее создать пользователя из группы администраторов с тем же самым именем и паролем, в противном случае удаленный запуск не получится. Так же необходимо держать запущенной службу удаленного доступа к реестру (the Remote Registry service) и открыть на брандмауэре все порты, ответственные за RPC-доступ (после памятных

атак червя MSBlast эти порты очень часто закрываются, хотя Windows Firewall в конфигурации по умолчанию держит их открытыми).

Другой подводный камень намного более серьезен. Это даже не камень, а прямо айсберг какой-то (чей собрат по всей видимости потопил "Титаник", а все потому, что никто не хочет думать головой). Попытка удаленного запуска утилиты autorunsc.exe (консольной версии утилиты autoruns.exe) "подвешивает" консоль, не давая никакого полезного выхлопа. Пляски с бубном (типа перенаправления вывода и все такое) не помогают! "Диспетчер задач" показывает, что процесс висит, но… не выполняется, то есть очень даже хорошо выполняется! После того, как компания Microsoft приобрела фирму Sysinternals, она обязала Марка Руссиновича добавить ко всем утилитам NAG-screen с текстом лицензионного соглашения (EULA), вспыхивающий при первом запуске.

Рисунок 8 NAG-Screen, вспыхивающий при первом запуске всех утилит Руссиновича, в том числе и консольных

Экран, конечно, графический с кнопочками "согласен"/"не согласен". Они-то и подвешивают "консоль" при удаленном запуске и чтобы все прошло успешно, _все_ утилиты Марка Руссиновича хотя бы однажды должны быть запущены на _каждой_ машине от имени Администратора (точнее, пользователя, входящего в группу администраторов на данной машине). Естественно, это ужасно напрягает, своя все преимущества удаленного запуска на нет. К счастью, если добавить специальный ключик в реестр (что можно сделать и удаленно), никакого NAG-screen'а больше не будет. Короче, открываем (удаленно, конечно) ветку: HKCU\Software\Sysinternals\<имя_программы>, где <имя программы> – название утилиты, например, в нашем случае — "AutoRuns", создаем параметр EulaAccepted типа DWORD и устанавливаем его в 1.

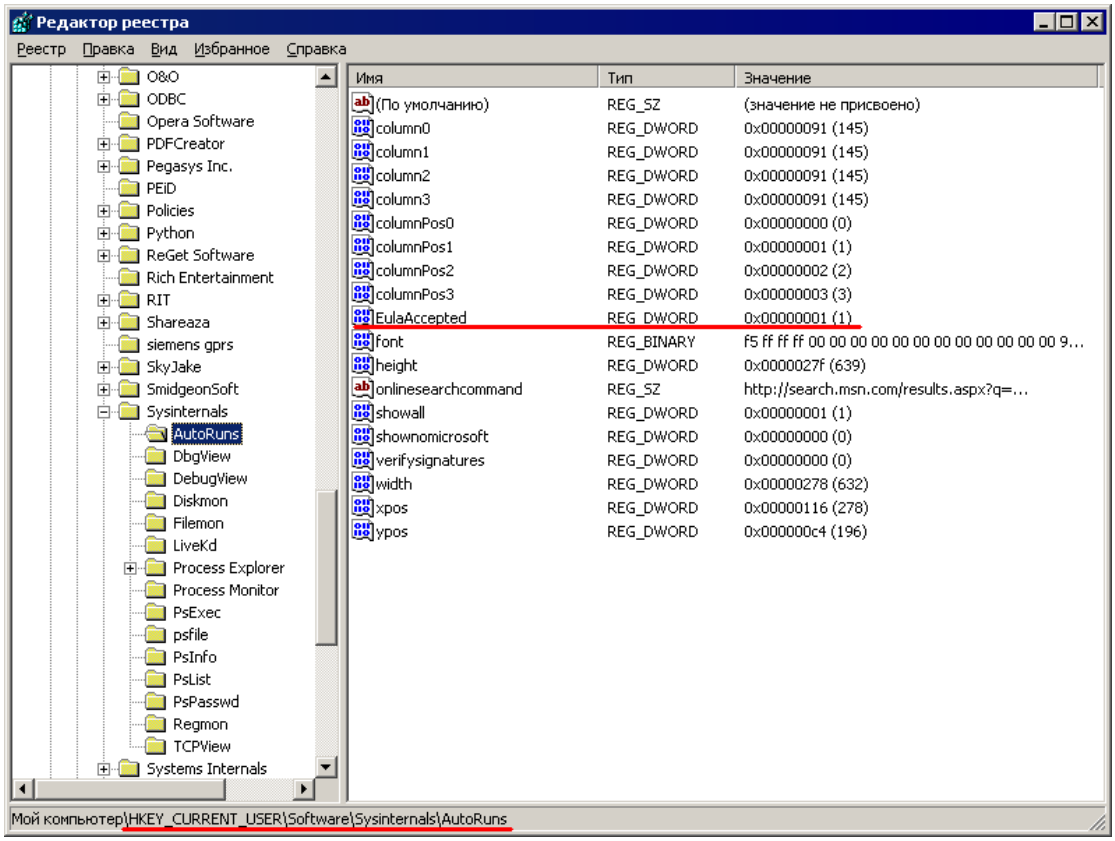

Рисунок 9 обход NAG-Screen'а при помощи удаленного редактирования реестра

Все! Теперь можно запускать autorunsc.exe на удаленной машине с помощью утилиты psexec.exe и все будет работать (ключ -c – копировать программу на удаленный узел перед запуском, автоматически удаляя ее после завершения работы, если вместо имени узла — в данном случае \\S2K3VM\ – указать "*", то программа будет запущена на всех узлах локальной сети, естественно, за исключением текущего! ну разве не красота?!).

```
$L:\>psexec.exe \\S2K3VM\ -c autorunsc.exe -b

PsExec v1.94 - Execute processes remotely
Copyright (C) 2001-2008 Mark Russinovich
Sysinternals - www.sysinternals.com

Sysinternals Autoruns v9.13 - Autostart program viewer
Copyright (C) 2002-2008 Mark Russinovich and Bryce Cogswell
Sysinternals - www.sysinternals.com

HKLM\System\CurrentControlSet\Control\Session Manager\BootExecute
       autocheck autochk *
              autocheck autochk *
              Auto Check Utility
              Microsoft Corporation
              5.02.3790.1830
              c:\windows\system32\autochk.exe
              d18fa3530aa4124a9d64f97162b1e3df                              (MD5)
              08052f92587247180835b8ad13ddd79b33587864                      (SHA-1)
              d49e8d367e1d9f1b9b0a05ef6a3d1188a83eb540086b8dbe78ea1688447 (SHA-256)
autorunsc.exe exited on S2K3VM\ with error code 0.
```

Листинг 2 пример удаленного запуска утилиты autorunsc.exe на узле \\S2K3VM

>>> *врезка полезные ссылки*

Для удобства все ссылки на программы, упомянутые в статье, собраны в одну врезку, представленную ниже:

- **AutoRuns for Windows v9.13:**

- консольная и графическая версии утилиты AutoRuns (490 KB), показывающей _большинство_ объектов автозагрузки, а так же предоставляющая дополнительный сервис по их версификации и выявлению новых объектов: http://technet.microsoft.com/en-us/sysinternals/bb963902.aspx;

- **Process Explorer v11.11:**
  - усовершенствованный аналог штатного "Диспетчера Задач" (1,6 Мбайт): http://technet.microsoft.com/en-us/sysinternals/bb896653.aspx;
- **PsTools**:
  - набор утилит для удаленного запуска программ (1 Мбайт): http://technet.microsoft.com/en-us/sysinternals/bb897553.aspx;
- **Rootkit Revealer**:
  - забавная игрушка, иногда ловит rootkit'ы, но чаще — нет (231 Кбайт): http://technet.microsoft.com/en-us/sysinternals/bb897445.aspx;