скрытые ключи автозапуска в системном реестре

крис касперски ака мыщъх, no-email

существует множество широко известных ключей автозапуска, в которые прописываются вирусы, черви, трояны и другие программы, пытающиеся внедриться в атакуемую систему, и которые проверяют антивирусы, брандмауэры и прочие сторожевые механизмы, бьющие хакеров еще на излете. чтобы выжить в этом суровом мире, полном ужасных защитных монстров, приходится извращаться не по-детски и разрабатывать методики поиска малоизвестных ключей автозапуска, неподвластные никаким анализаторам реестра.

введение

Для поддержания своей жизнедеятельности малварь должна хотя бы изредка получать управление. В идеале — при каждой загрузке операционной системы, что достигается, например, через прописывание пути к исполняемому файлу в следующей ветви системного реестра HKCU\SOFTWARE\Microsoft\Windows\CurrentVersion\Run, однако, это очень плохой способ, поскольку появление нового элемента в разделе \Run редко остается незамеченным. О нем знают как защитные механизмы (антивирусы, персональные брандмауэры, разные диагностические утилиты), так и продвинутые пользователи.

Хакеры и разработчики антивирусов ведут непрекращающиеся археологические раскопки реестра, ковыряя его с двух сторон, причем, каждая сторона стремиться найти как можно больше ключей прямо или косвенно связанных с автозапуском. Необязательно юзать общесистемные ключи, относящееся непосредственно к Windows. Для решения поставленной задачи вполне достаточно прицепиться к часто запускаемому приложению, например, к IE, стартуя вместе с его запуском.

Реестр содержит тысячи (если не сотни тысяч!) подходящих ключей, но только несколько десятков из них контролируются защитными механизмами. На хакерских форумах начинающие кодокопатели часто спрашивают полный список ключей автозапуска или перечень новых ключей, появившихся в XP, Висле или другой операционной системе. Мало того, что их вопрос обычно остается безответным (обвинения в ламеризме — не в счет), так еще и все существующие списки (которые, кстати говоря, можно запросто выковырять из любой антивирусной программы — они там, как правило, лежат открытым текстом) представляют интерес в основном для разработчиков защит. Использовать их для внедрения малвари — сплошное пионерство и долго такая малварь не проживет.

Мы не будем приводить готовых ключей (ну, разве что в качестве примера). Напротив, мы покажем как находить их самостоятельно, а так же рассмотрим основные ошибки начинающих хакеров, совершаемые с завидным постоянством. Это только кажется, что внедриться в систему просто — на самом деле тут слишком много подводных камней и скрытых граблей.

Статья так же будет интересна и тем, кто озабочен стерильностью своей системы, но не знает где обычно хакеры ныкают вирусы и на какие ветви реестра следует обращать внимание в первую очередь.

>>> врезка беззаконная законность

Вот мыщъх уже слышит (не)справедливые упреки и уворачивается от самонаводящихся тухлых помидоров, летящих косяками и накрывающих его нору под громкие возгласы: да как вообще можно писать такие статьи?! это же незаконно и вообще все такое!!! ну на счет "всего такого" мыщъх не специалист, поэтому действует строго в рамках закона, который по этому поводу говорит, что сама по себе модификация ключей системного реестра еще не есть преступление, даже если она осуществляется без ведома владельца машины. если бы это было незаконно, пришлось бы сажать всех разработчиков инсталляторов, ведь никто из пользователей не может с уверенностью сказать какие именно ключи реестра они изменяют.

Установка программы, ворующей пароли (или другую конфиденциальную информацию), действительно, является серьезным правонарушением, но технология автозагрузки здесь _совсем_ не причем. Более того, сокрытие подобных "хакерских" технологий только вредит, поскольку не позволяет конечным пользователям эффективно отражать атаки зловредных программ.

план перехвата ассоциации расширений

Ассоциация расширений? А разве существует такая? Скорее уж ассоцирование расширений с обрабатывающими их приложениями. Это технически верно, но уж слишком длинно для заголовка. Впрочем, не будем придираться к формулировкам, а лучше сразу определимся с перехватом.

Запустим "Редактор Реестра" и откроем раздел **HKEY_CLASSES_ROOT**, образованный (как известно) путем слияния данных из двух источников: **HKLM\SOFTWARE\Classes** и **HKCU\SOFTWARE\Classes**. Первый носит глобальный характер, распространяющийся на всех пользователей, зарегистрированных в системе, и требует для своей модификации прав администратора (которых у малвари чаще всего нет). Второй — относится только к текущему пользователю, прав администратора он не требует, но и не затрагивает всех остальных.

Отсюда правило: *сначала мы пытаемся модифицировать HKLM\SOFTWARE\Classes, и, если обламываемся, либо повышаем свои привилегии до уровня администратора (используя тот или иной хак), либо переключаемся на HKCU\SOFTWARE\Classes, будучи готовым к тому, что прав на его модификацию у нас может и не быть* (однако, в живой природе такие жестоко ущемленные пользователи практически не встречаются и значительная часть их них вообще постоянно сидит под администратором, а потом удивляются откуда вирусы берутся в таких количествах).

В [HKLM|HKCU]\SOFTWARE\Classes помимо прочей полезной информации хранится список зарегистрированных расширений и указания по их обработке при запуске файлов через стандартную оболочку типа "Проводника" или команду "start", набираемую в командной строке. Файловые менеджеры FAR и Total Commander так же используют список зарегистрированных расширений, путем вызова API-функций **ShellExecute/ShellExecuteEx** и/или команды **start** (точное поведение зависит от текущих настроек).

За этими ветвями практически никто не следит. "Практически" — потому что ассоциированные приложения могут проверять целостность ассоциаций при каждом запуске и ругаться матом, если "их" расширение оказалось сопоставлено с посторонним приложением. Так же, некоторые защитные системы могут контролировать ассоциации стандартных системных расширений типа .exe, однако, в _общем_ _случае_ перехват ассоциаций остается никем незамеченным, особенно, если действовать по плану. А есть ли у нас план?! Конечно же есть!!! (Ну не могли же мы все скурить за это время). Причем, не какой-то там отстойный подзаборный план, а первосортный гибрид!!!

Прежде всего, необходимо выбрать расширение, которое мы собрались атаковать. Это должно быть довольно распространенное расширение, открываемое пользователями по меньшей мере несколько раз в день, например, ".txt" или ".bmp". Нет, ".txt" все-таки лучше. Вот и будем его терзать — грызть зубами, бить хвостом и рвать когтями по полной программе. Кстати, на счет программ…

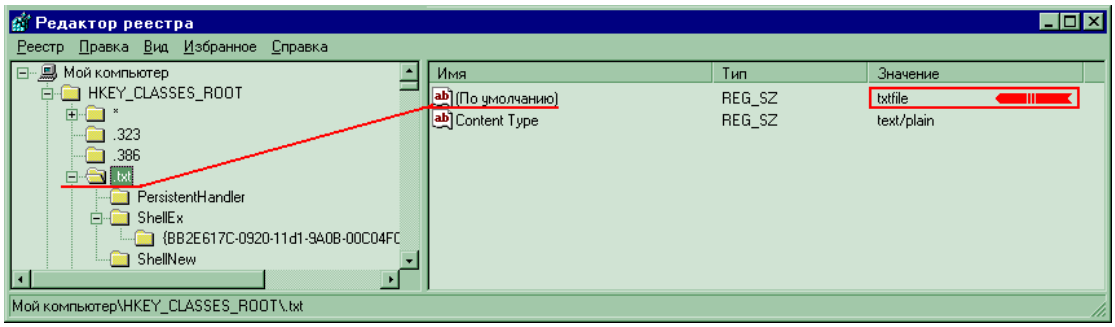

Рисунок 1 тип файла, соответствующий расширению .txt

…открываем HKEY_CLASSES_ROOT\.txt (см. рис. 1), смотрим на значение по умолчанию и видим, что там в данном случае находится ключ "txtfile", описывающий тип файла, а путь к сопоставленному с ним приложением хранится в **HKEY_CLASSES_ROOT\txtfile\shell\open\command** (см. рис. 2), где обозначена строка "%SystemRoot%\system32\NOTEPAD.EXE %1". Магическое число "%1" представляет собой первый аргумент командной строки, содержащий имя открываемого файла, ну а "**%SystemRoot%\system32\NOTEPAD.EXE**" — соответственно, всем известный "Блокнот".

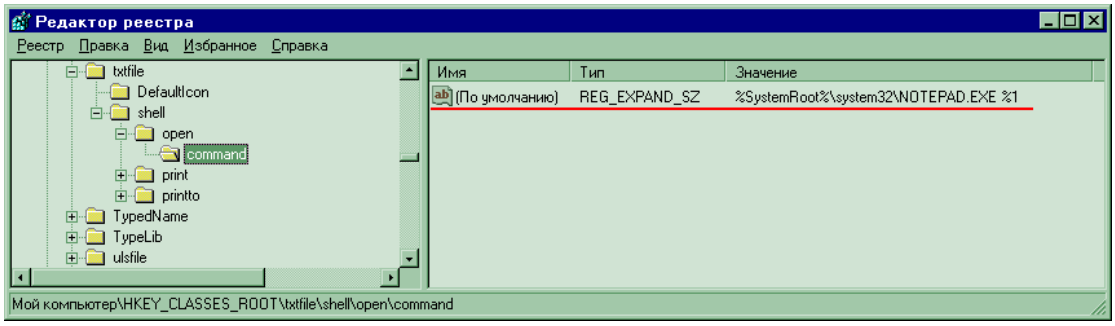

Рисунок 2 приложение, ассоциированное с текстовыми файлами

Поставим перед собой задачу — перехватить обработчик ассоциаций, запуская свой собственный исполняемый файл (не обязательно вредоносный) при открытии текстовых файлов так, чтобы факт перехвата остался незамеченным.

Самое простое, что только можно сделать — это заменить "%SystemRoot%\system32\NOTEPAD.EXE %1" на "my_own_path\my_own_malware-file.exe %1", заставив систему вызывать "my_own_malware-file.exe" вместо "Блокнота". Естественно, "Блокнот" при этом придется вызывать нам самим, не забыв передать ему первый аргумент командой строки (некоторые хакеры забывают, в результате чего "Блокнот" открывает пустой файл, высаживания пользователя на измену).

Кстати говоря, это не обязательно должен быть именно "Блокнот". Достаточно многие пользователи используют нестандартные редакторы, поэтому, мы _всегда_ должны считывать исходное содержимое \txtfile\shell\open\command, перемещая его куда-нибудь внутрь реестра или своего собственного .ini-файла — туда, где его никто не найдет. Жестко прошивать вызовов "Блокнота" внутри my_own_malware-file.exe категорически не допустимо!!! Если у вас текстовые файлы открываются блокнотом, это еще значит, что аналогичным образом обстоят дела и у других пользователей! Это — грубейшая ошибка!!!

Остается устранить одну мелкую недоработку и малварь готова к внедрению в промышленною эксплуатацию. Что это за недоработка? Сейчас увидим! Щелкаем по "Моему Компьютеру" (ну, в смысле по вашему), в меню "Сервис" выбираем пункт "Свойства Папки", в открывшемся диалоговом окне переходим ко вкладке "Типы файлов", находим среди расширений "TXT", нажимаем кнопку "Дополнительно", в "Действиях" выбираем "Open" (см. рис. 3) и... тайное сразу же становится явным! Система честно сообщает, что текстовые файлы она открывает с помощью my_own_malware-file.exe, что не может не вызывать у продвинутых пользователей серьезных подозрений, переходящих в полную уверенность, что их поимели. Нет, нам это не подходит и чтобы малварь жила и процветала нужно заныкать ее получше.

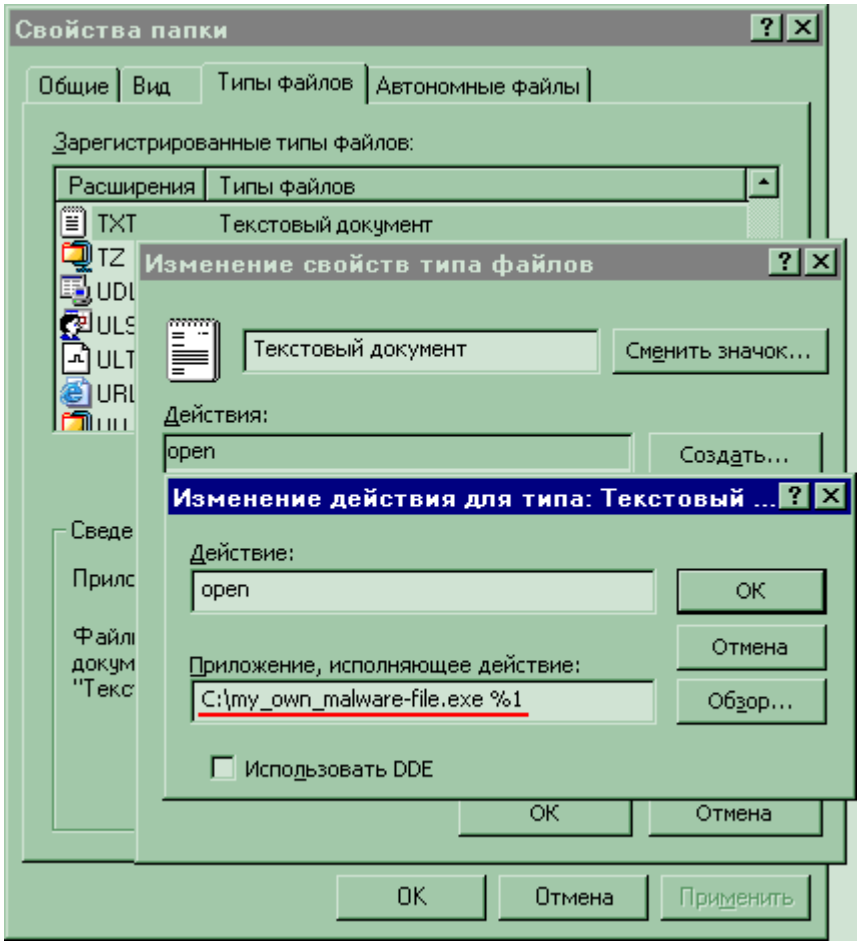

Рисунок 3 перехват и его разоблачение

Идея – берем "NOTEPAD.EXE" (или то имя, что ассоциировано с текстовыми файлами у данного пользователя) и заменяем одну латинскую букву на соответствующую ей русскую, аналогичную по начертанию, например, букву "A", после чего переименовываем "my_own_malware-file.exe" в "NOTEPAD.EXE" (где "A" — русская) и осуществляем перехват по вышеописанной методике.

Даже если у пользователя установлена противная защитная программа, следящая за расширениями, ругающиеся при их изменении, то жертва никак не отреагирует на предупреждение об опасности, посчитав, что это просто глюк. Ну посудите сами: "внимание! ассоциация расширения .txt-файлов изменилась с NOTEPAD.EXE на NOTEPAD.EXE". Типа масло масленнное.

Впрочем, учитывая, что некоторые шрифты слегка по разному отображают "одноименные" латинские и русские символы, наш обман в принципе может быть и разоблачен, поэтому, крайне желательно, чтобы заменяемая буква встречалась в имени файла только однажды — иначе разница в начертаниях будет намного сильнее бросаться в глаза. Другая проблема — пользователь, просматривающий каталог Windows может очень удивиться, увидев два "одинаковых" файла, поэтому малварь лучше убрать в другой каталог.

Естественно, на файлы, открываемые в FAR'e по <F4> это _никак_ не подействует, поэтому, если заведомо известно, что жертва активно пользуется FAR'ом, следует искать другой путь.

Хорошая идея — подменить стандартный обработчик exe-файлов, которые открываются пользователем намного чаще, чем файлы документов все вместе взятые (на самом деле — это _плохая_ идея, поскольку за ассоциациями исполняемых файлов следят достаточно многие сторожевые программы, но… кто не рискует…)

Открываем "**HKEY_CLASSES_ROOT\.exe**", видим, что значение по умолчанию установлено в "**exefile**", лезем в "HKEY_CLASSES_ROOT**exefile**\shell\open\command", где находится ""%1" %*", что переводится на русский язык как: запустить выбранный файл (имя которого передано в первом аргументе командной строки — "%1"), передав ему все аргументы,

какие только есть – "%*". Если заменить "%1" на путь/имя нашего файла-перехватчика, то он будет запускаться всякий раз, когда пользователь щелкает по иконке исполняемого файла или нажимает на <Enter> в FAR'e (на некоторых хакерских форумах высказывается мнение, что такая замена носит рекурсивный характер, то есть, при файл-перехватчик перехватывает запуски _всех_ исполняемых файлов, включая самого себя, в результате чего мы зацикливаемся в бесконечном рекурсивном спуске. на самом деле, подобное утверждение совершенно безосновательно: система анализирует ассоциации только однажды и потому рекурсии не возникает).

>>> врезка классы в ассоциациях

При перехвате зарегистрированных расширений необходимо быть готовым ко встрече с "странностями". Странность первая (и легко преодолимая): значение по умолчанию перехватываемого расширения содержит пустую строку, вместо ожидаемого типа файла. Если это так — смотрим в подраздел ShellEx и ищем там длинную строку циферок (стандартный уникальный идентификатор). Например, в моем случае это: **`HKEY_CLASSES_ROOT\.cdr\ShellEx\{BB2E617C-0920-11d1-9A0B-00C04FC2D6C1}`**.

Смотрим какое значение по умолчанию содержит ветка {BB2E617C-0920-11d1-9A0B-00C04FC2D6C1}. Ага! Еще один идентификатор — {1AEB1360-5AFC-11d0-B806-00C04FD706EC}. Ищем его в реестре и… ведь находим, черт возьми!!! Да ведь не возьмет! Ему такое добро ни даром ни с доплатой не нужно.

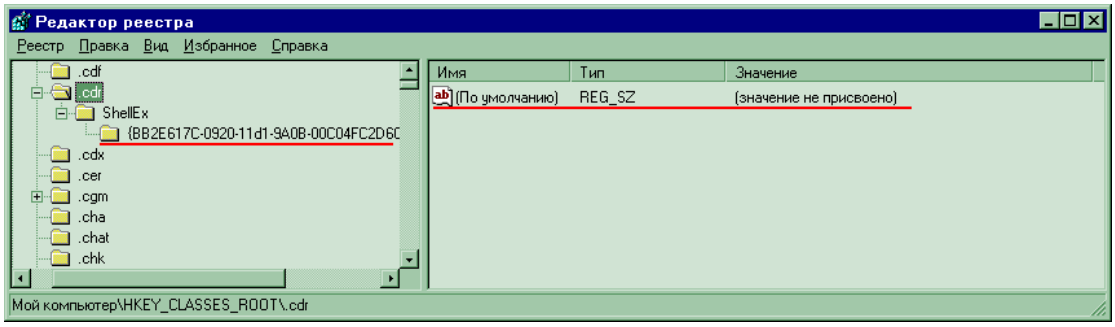

Рисунок 4 пример расширения со странностями

Оказывается, что за таинственным идентификатором прятался OLE-компонент "Извлечение миниатюр графических фильтров Office", название которого содержится в параметре {1AEB1360-5AFC-11d0-B806-00C04FD706EC} ключа по умолчанию (см. рис. 5).

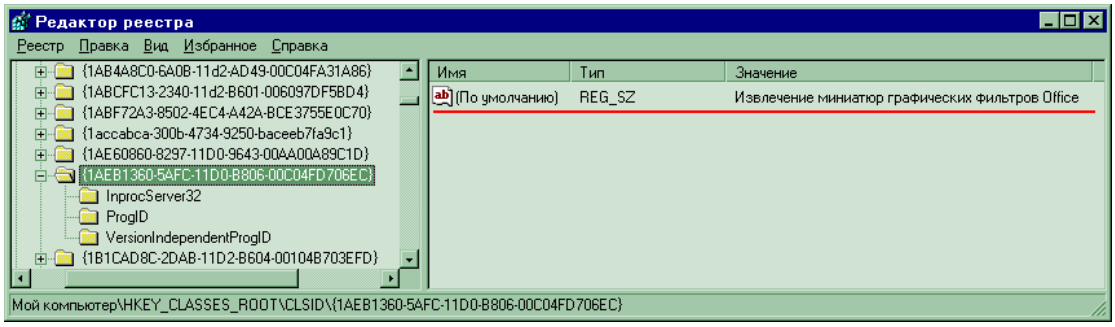

Рисунок 5 докапываемся до обработчика

Спустившись на один уровень вглубь, мы откопаем ветку \InprocServer32, параметр по умолчанию которой задет путь к динамической библиотеке используемой для отображения содержимого документов непосредственно в проводнике (см. рис. 6), причем не только .cdr файлов, но и многих других, отображаемых при помощи C:\WINNT\System32\thumbvw.dll.

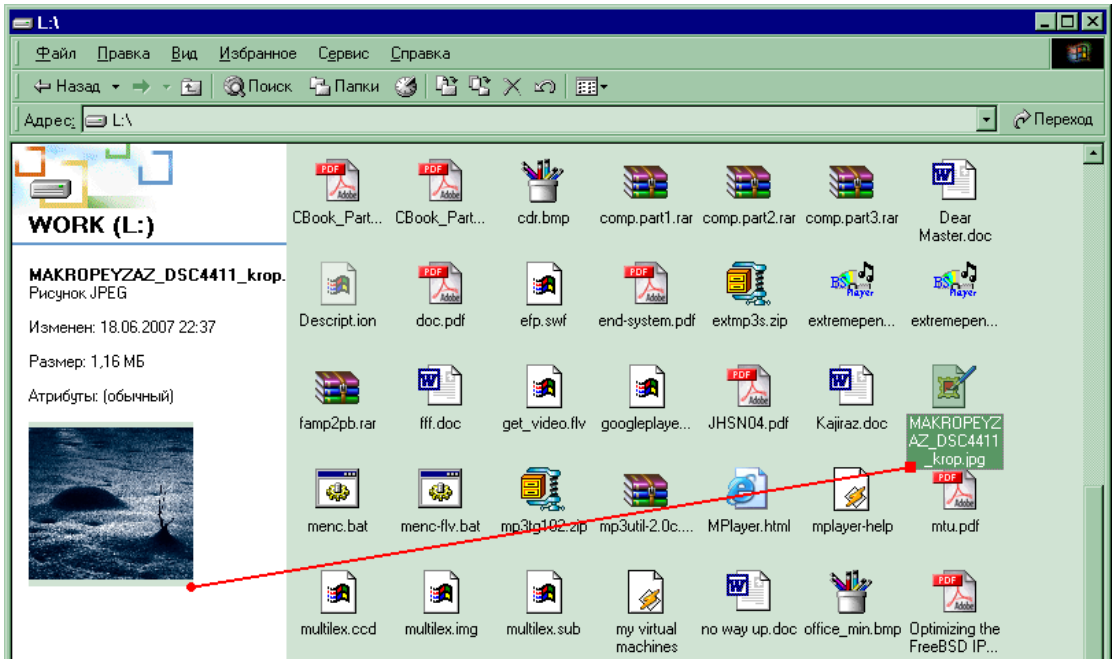

Рисунок 6 отображение миниатюр в "Проводнике"

Ситуация проясняется. У мыщъх'а .cdr-файлы не ассоциированы ни с каким приложением (именно потому, значение по умолчанию содержит пустую строку), однако, установленный пакет Microsoft Office позволяет просматривать их в виде миниатюр в "Проводнике".

Какие преимущества это нам дает? А вот какие — можно не только подменять существующие обработчики чужих файлов, но и устанавливать свои — тогда при выделении данного файла в "Проводнике" система попытается отобразить его "миниатюру", вызывав нашу динамическую библиотеку. Ассоциация типов при этом _никак_ не изменится и ни одна известная мыщъх'у защитная программа не сможет отследить этот эквилибристический трюк.

Единственный минус данного решения в том, что он не работает, если пользователь предпочитает жить в FAR'e или командной строке, однако, большинство потенциальных жертв запускают файлы исключительно из "Проводника" и даже закоренелые поклонники командной строки прибегают к нему время от времени.

Странность вторая — значение по умолчанию пусто как голова с бодуна, а подраздел \ShellEx отсутствует. Напрочь. (Ну, или не содержит ничего интересного). Если так — открываем \PersistentHandler и в случае успешного завершения операции извлекаем его значение по умолчанию, которое может выглядеть, например, так: HKEY_CLASSES_ROOT\.dbg\PersistentHandler\{098f2470-bae0-11cd-b579-08002b30bfeb}.

Открываем HKEY_CLASSES_ROOT\CLSID**{098F2470-BAE0-11CD-B579-08002B30BFEB}**\InprocServer32 и извлекаем оттуда имя динамической библиотеки или очередной идентификатор (если он там, конечно, есть) и поступаем с ним исходя из контекста ситуации и нашей сексуальной ориентации. Перехват динамических библиотек, реализующий OLE/COM/ActiveX-сервера довольно сложная задача и начинающим лучше не браться за ее решение — глюков не оберешься! Лучше выбрать другой тип расширений для перехвата, благо недостатка в них испытывать не приходится.

Странность третья — значение по умолчанию не содержит ничего и никаких подразделов не наблюдается. Это значит, что несмотря на факт регистрации данного расширения, ему не сопоставлено никаких программ и не предусмотрено никаких специальных указаний для "Проводника" (типа контекстного меню, миниатюр и т. д.). Такие расширения следует пропускать.

!исследуем реестр сами!

Перехват ассоциаций — хоть и привлекательный, но все-таки не единственно возможный трюк, позволяющий малвари внедряться в систему. При желании можно пойти _совсем_ иными путем, о котором не знает _никто_ кроме нас.

Как искать ключи, ответственные за автозагрузку? Да очень просто — запускаем "Редактор Реестра", вбиваем в поиск ".exe" или ".dll" и ищем все ветки, откуда вызываются исполняемые файлы и динамические библиотеки. Вот, например, Adobe Acrobat содержит следующую замечательную ветвь HKLM\SOFTWARE\Adobe\Adobe Acrobat\PrintMe, в которой зарыт если не клад, то что-то на него похожее:

- URL2KPMInst: https://www.printme.com/support/adobe/PrintMeDriverforWindows2000.exe
- URL9XPMInst: https://www.printme.com/support/adobe/PrintMeDriverforWindows9x.exe
- URLNTPMInst: https://www.printme.com/support/adobe/PrintMeDriverforWindowsNT.exe
- URLXPPMInst: https://www.printme.com/support/adobe/PrintMeDriverforWindowsXP.exe

Не нужно быть Пострадамусом, и даже Штирлицем быть не нужно, чтобы понять, что здесь лежат ссылки на динамически загружаемые драйвера для печати pdf-документов под разные системы. Так почему бы их не изменить?! Ну драйвера печати — это еще туда-сюда (скорее всего они уже установлены), но куча программ содержит URL'ы к серверам обновлений, которые проверяются при каждом запуске или через определенные промежутки времени.

Конечно, это не совсем автозагрузка, точнее даже сказать совсем не автозагрузка, но для внедрения малвари в систему она очень даже пригодна! В частности, панель Google хранит путь к файлу обновлений в ветке HKL\SOFTWARE\Google\Common\Google Updater\path: C:\Program Files\Google\Common\Google Updater\GoogleUpdaterService.exe, позволяя нам заменять GoogleUpdaterService.exe на свою собственную программу, делающую все, что задумано и передающую управление настоящей GoogleUpdaterService.exe. И никто ничего не заподозрит!!! Отыскать внедренную таким образом малварь практически невозможно! Для этого необходимо знать назначение _всех_ ключей реестра, а это невозможно, поскольку существует бесчисленное множество программ с никем не стандартизированными настройками.

Чисто теоретически можно периодически сканировать реестр на предмет изменений в ключах, содержащих "*.exe" и "*.dll", однако, во-первых, мыщъх'у не известно ни одного готового сканера, который бы делал это, а, во-вторых, даже если такой сканер и появится он станет заложником большого количества ложных срабатываний. Например, стоит скопировать в FAR'e исполняемый файл из одной папки в другую, как его имя попадет в специальную ветку реестра, хранящую историю строки редактирования. Сканер, будучи тупой машиной, не может отсеивать ложные срабатывания и для полноценного анализа нам понадобиться человек, причем не абы какой человек, а весьма продвинутый пользователь, у которого малварь просто не водится.

Другая теоретическая возможность — запрет на модификацию всех потенциально опасных ветвей реестра. Что ж, операционные системы семейства NT имеют гибкую политику доступа, но… полный запрет на запись в реестр приведет к развалу системы, следовательно, нужно тщательно проанализировать _все_ ключи, отделить опасные от безопасных и… ходить топиться! Потому что времени на это уйдет… Было бы просто замечательно, если бы разработчики программ думали головой и сразу сортировали свои настройки по степени потенциально опасности, устанавливая им надлежащие права доступа. Так ведь нет! Они ленятся и валят все в кучу! А пользователи потом страдают, находясь под угрозой атаки!

Впрочем, угроза атаки не так уж и велика. Несмотря на то, что реестр содержит просто кладезь ветвей, пригодных для внедрения, их автоматизированный поиск невозможен, поскольку, как уже говорилось, тупая машина не в состоянии определить назначение ключей по их содержимому. Это может сделать только человек! Следовательно, места для внедрения необходимо искать еще на этапе написания малвари. Но тут мы сталкиваемся с тем, что многообразие используемых программ (и их версий) приводит к сужению круга потенциальных жертв. Впрочем, если брать широко распространенные программы (такие, например, как Adobe Acrobat Reader), то круг окажется не таким уж и узким.

заключение

Хакер отличается от не-хакера прежде всего стремлением идти непроторенным путем, искать новые решения, вместо использования давно известных и хорошо обкатанных. Хакерство — это творчество. Хакерство — это дерзость. И мыщъх верит, что после прочтения этой статьи, исследовательского запала под хакерскими хвостами основательно прибавится, в результате чего кое-кому придется, схватившись за голову, в спешке совершенствовать защитные алгоритмы, потому что старые перестанут справляться с растущим потоком плодотворной софт-вари.