Евгений Дроботун (201074@mail.ru)

БУРИМ АНТИВИРУС. Еще глубже!

WWW

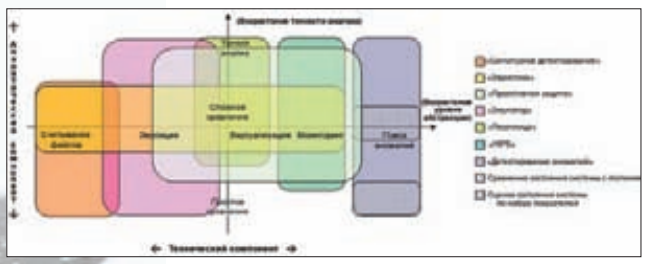
Про функции уведомления и функции обратного вызова (callback-функции) Windows можно почитать здесь: www.sww-it.ru/2010-02-21/362.

DVD

На диске можно найти утилиту Blacklight от F-Secure, описанную во врезке «Брутфорс PID».

РАССМАТРИВАЕМ СПОСОБЫ МОНИТОРИНГА СОБЫТИЙ И ПРОАКТИВНОЙ ЗАЩИТЫ В РАЗНЫХ АНТИВИРУСНЫХ ПРОГРАММАХ

Сегодня мы продолжим изучение способов противодействия антивирусам и посмотрим, чем еще, кроме компонентов для сигнатурного сканирования и анализа файлов, вооружают свои творения создатели антивирусных программ.



Наша любимая схема

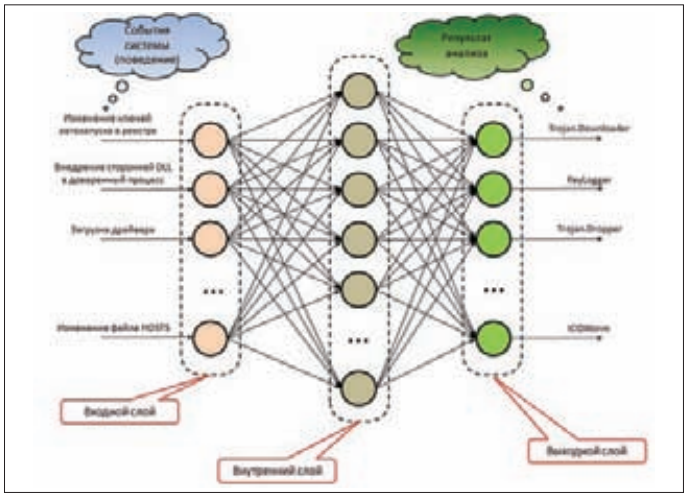


Рисунок 1. Простейший пример нейросети, анализирующей поведение системы

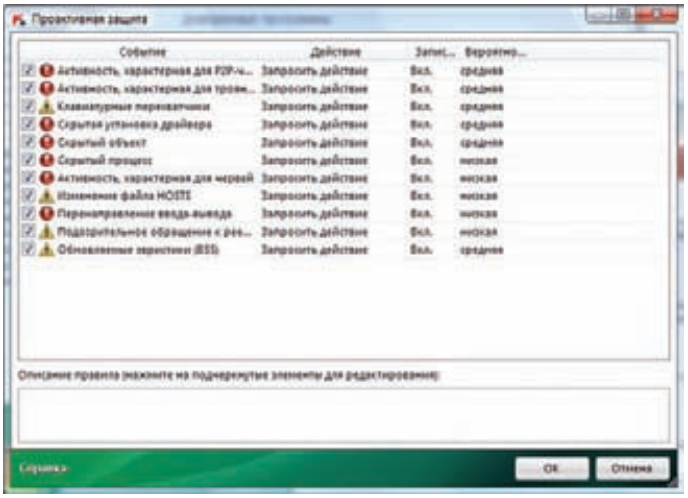


Рисунок 2. События, контролируемые проактивной защитой «Антивируса Касперского»)

Вот мы и добрались до таких понятий, как HIPS (Host Intrusion Prevention System), «проактивное детектирование», «поведенческий анализ», «анализ, основанный на мониторинге системных событий»... Эти понятия на самом деле вносят даже меньше ясности, чем какой-нибудь «эвристический анализ». Они могут подразумевать очень широкий спектр технологий и, в общем-то, никак не помогают понять, какая из них используется в конкретном продукте. Под этими понятиями может скрываться, например, примитивная защита нескольких ключей реестра, или уведомления о попытках доступа к определенным директориям, или анализ поведения программ, или какая-либо другая технология, основанная на мониторинге системных событий. Если мы снова обратимся к нашей любимой двухкомпонентной схеме, то увидим, что технический компонент — это мониторинг системных событий или анализ поведения системы, а аналитический может быть каким угодно, от пресечения единичных подозрительных событий до сложного анализа цепочек программных действий (то есть та же эвристика с набором правил и весовыми коэффициентами опасности или, может быть, даже что-то на основе нейросетей (смотри рисунок 1 и соответствующую врезку)).

В первую очередь нас будет интересовать технический компонент проактивной защиты, поскольку принципы анализа событий схожи с принципами анализа, описанными в прошлой статье, а отличия, если они есть, мы рассмотрим позже. Итак, приступим.

ЧТО БУДЕМ МОНИТОРИТЬ?

Для различных вредоносных программ характерно довольно много событий, причем их список неуклонно увеличивается по мере развития творческих способностей вирусописателей. Общую картину можно увидеть, к примеру, на рисунке 2. Здесь показаны события (вернее, даже совокупности событий), на которые реагирует «Антивирус Касперского». Конечно, некоторые пункты (а именно «Активность, характерная для P2P-червей», «Активность, характерная для троянских программ» и «Активность, характерная для червей») описаны очень уж обобщенно, и сейчас мы попытаемся всё это дело немного конкретизировать. Итак, в первую оч ередь речь идет о сетевой активности: подавляющее большинство нечисти обязательно пытается или что-то передать в сеть (пароли, явки, номера кредиток и т. д.) или, наоборот, что-то получить из сети (загрузить основную часть малвари или обновления, принять какие-нибудь команды управления, да мало ли еще что). Многие вредоносные программы пытаются проинжектить свой код

Рисунок 3. Так McAfee Antivirus Plus пытается контролировать изменения в реестре

БРУТФОРС PID

Некоторые утилиты (например, Blacklight от F-Secure или SpyDLLRemover) используют для поиска скрытых процессов брутфорс идентификаторов процессов (BPID). Метод достаточно прост и в то же время эффективен. Такие утилиты открывают процессы и перебирают все возможные PID. К примеру, Blacklight в цикле пытается вызвать OpenProcess для всех возможных значений идентификатора процесса из диапазона от 0x0 до 0x4E1C. Таким образом, можно получить список процессов, присутствующих в системе. Затем вызывается функция CreateToolhelp32Snapshot , которая выдает второй список процессов. Потом два эти списка сравниваются: если процесс присутствует в первом и отсутствует во втором, то он считается скрытым.

Рисунок 4. Контроль за изменениями в реестре с помощью RegNotifyChangeKeyValue. Выше красной пунктирной линии показана реализация в McAfee, ниже линии — в «Касперском». Как говорится, найди отличия

в доверенный процесс (обычно это explorer.exe или svchost.exe), получить привилегии отладчика и изменить системные компоненты (например, пропатчить ядро ОС в своих интересах) — в общем, как можно глубже внедриться в систему.

Думаю, с отслеживаемыми событиями все ясно, поэтому пойдем дальше.

СЛЕДИМ ЗА РЕЕСТРОМ

Любая уважающая себя малварь при заражении системы должна позаботиться о своем повторном запуске после перезагрузки. Она может выбрать для этого множество доступных мест, начиная от папки «Автозагрузка» и заканчивая кучей ключей автозапуска в реестре. Чаще всего, конечно же, малварь модифицирует ключи автозагрузки в реестре.

Некоторые вредоносные программы также пытаются изменить или дополнить какие-либо другие ключи реестра (к примеру, прописать свою DLL, изменить параметры Internet Explorer'а или установленной в системе антивирусной программы и т. д.). В общем, делаем вывод: за реестром надо следить, и следить тщательно. И все, даже самые захудалые антивирусные программы, гордо несущие в своем составе компонент проактивной защиты, просто обязаны это делать.

Самые распространенные способы контроля изменений в реестре — это перехват API-функций работы с реестром или использование специализированных функций контроля над реестром. Что касается первого способа, то для контроля над реестром обычно достаточно перехватить функции RegOpenKey, RegCreateKey, RegDeleteKey из advapi32.dll или более низкоуровневые NtOpenKey, NtCreateKey, NtDeleteKey и т. п. На рисунке 3 показано, как антивирус от McAfee перехватывает функции ядра для работы с реестром.

Следующий способ заключается в использовании специальной API-функции RegNotifyChangeKeyValue. Создатели Windows обучили эту функцию извещать программу, которая ее вызвала, об изменении определенного ключа реестра. Ее используют многие антивирусные программы (рисунок 4). В случае изменения реестра функция устанавливает в сигнальное состояние заранее созданный объект типа Event (событие), и антивирусу остается только отследить это событие и принять соответствующие меры. Помимо RegNotifyChangeKeyValue, в недрах Windows прячется еще одна небольшая функция CmRegisterCallBackEx, с помощью которой также можно контролировать реестр. Эта функция устанавливает так называемую callback-функцию (функцию уведомления), которая вызывается при изменениях в заранее определенных ветках реестра. Эта функция, в отличие от RegNotifyChangeKeyValue, может использоваться только в режиме ядра, и поэтому такой контроль за реестром осуществляется с помощью драйвера (к примеру, драйвер «Антивируса Касперского 2010» klif.sys импортирует функцию именно для этой цели (рисунок 5)). Как правило, антивирусы редко довольствуются только одним способом контроля реестра (особенно первым, который не так уж и надежен, тем более если перехват нужных функций делать в юзермоде) и используют комбинацию двух (McAfee, например, помимо перехвата использует RegNotifyChangeKeyValue), а то и всех трех способов.

ИНЖЕКТ В ЧУЖИЕ ПРОЦЕССЫ

Инжект в память чужого процесса можно считать классическим методом, который используют современные вирусы, так как он позволяет реализовывать вредоносные функции под доверенным процессом. Очень часто в роли этого доверенного процесса выступает explorer.exe, чуть реже — svchost.exe.

Прежде чем внедряться в нужный процесс, его необходимо найти. Если речь идет об explorer.exe, то самый простой и удобный способ для малвари — это задействовать

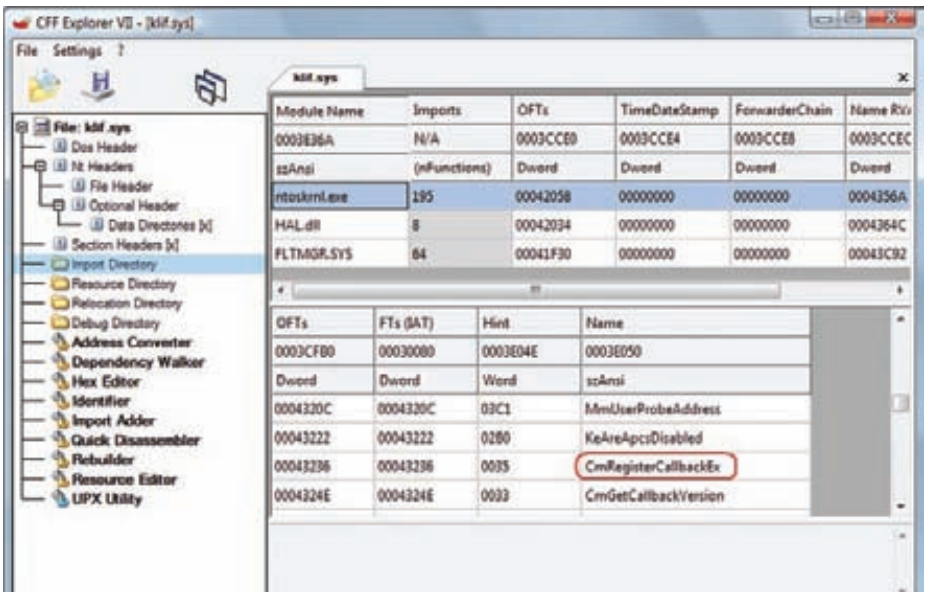


Рисунок 5. Функция CmRegisterCallBackEx в драйвере klif.sys из «Антивируса Касперского 2010»

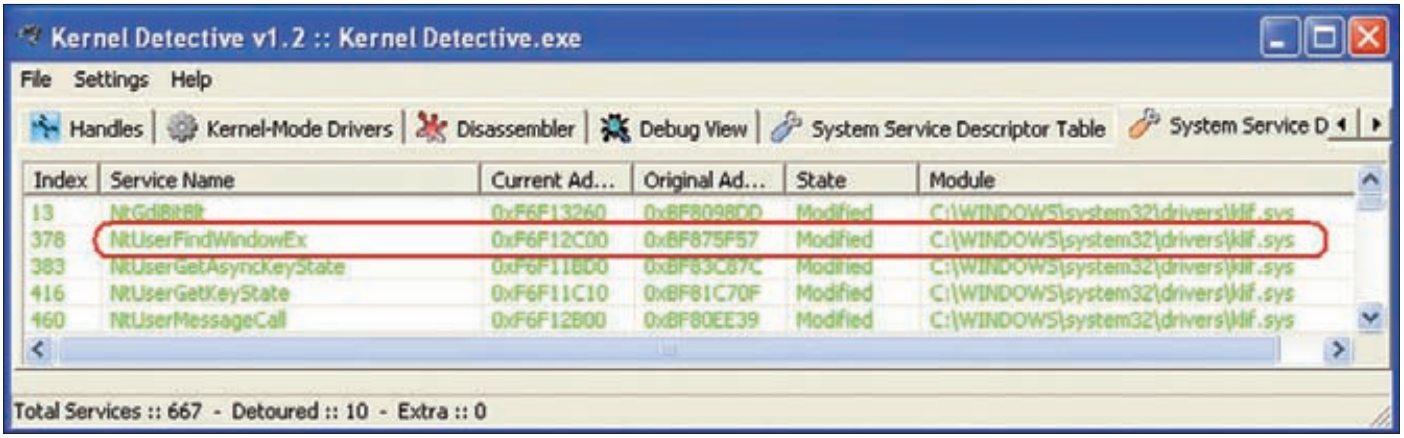


Рисунок 6. Перехват NtUserFindWindowEx «Антивирусом Касперского»

FindWindow (или FindWindowEx) с идентификатором окна progman. FindWindow, в свою очередь, базируется на NtUserFindWindowEx, и, перехватив ее (рисунок 6), можно отследить, какая программа ищет процесс explorer.exe.

Существует как минимум два очень распространенных способа внедрения постороннего кода в адресное пространство какого-либо процесса. Первый способ: получаем дескриптор процесса посредством вызова функции OpenProcess, выделяем в нем нужный кусочек памяти с атрибутом PAGE_EXECUTE_READWRITE (API-функция VirtualAllocEx), копируем туда все, что нам нужно, добываем дескриптор основного потока, замораживаем его (SuspendThread), получаем и запоминаем регистровый контекст (GetThreadContext), пишем в указатель команд адрес начала вредоносного кода, обновляем регистровый контекст (SetThreadContext) и размораживаем поток (ResumeThread), передавая управление на внедренный код, который после выполнения задуманного восстанавливает оригинальный указатель команд.

Второй способ: получаем дескриптор процесса посредством OpenProcess, выделяем в нем память так же, как и в первом случае(VirtualAllocEx), копируем туда внедряемый код через WriteProcessMemory и создаем удаленный поток с помощью API-функции CreateRemoteThread.

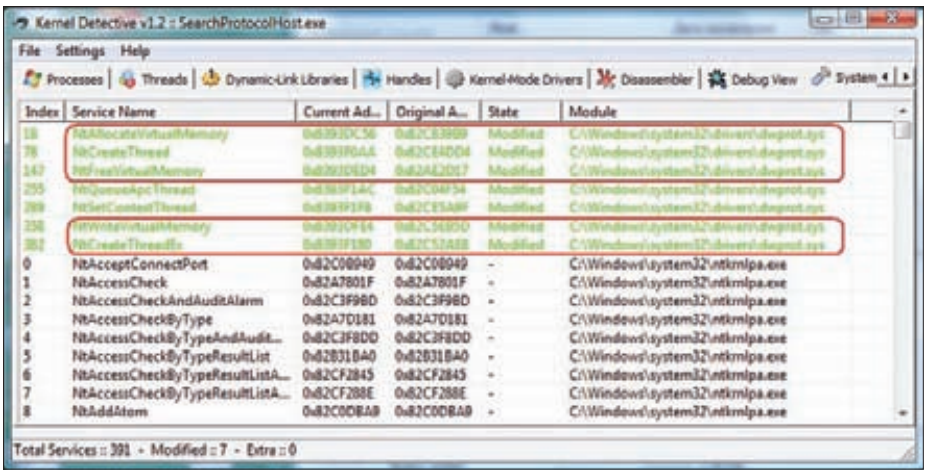


Рисунок 7. Контроль над операциями с памятью и созданием потоков в антивирусе DrWeb

Конечно, увидев последовательности вызовов OpenProcess\VirtualAllocEx\ WriteProcessMemory\SuspendThread\ GetThreadContext\ SetThreadContext\ ResumeThread или OpenProcess\VirtualAllocEx\

НЕЙРОСЕТИ НА АНТИВИРУСНОЙ СЛУЖБЕ

Искусственная нейронная сеть представляет собой математическую модель нейронной сети головного мозга человека (или какого-нибудь другого животного). Она состоит из простейших элементов — нейронов, которые связаны между собой (рисунок 1). Каждый нейрон выполняет несложную операцию: на основе поступающих от других нейронов сигналов, которые представлены в виде чисел от 0 до 1, и их весовых коэффициентов вычисляет выходной сигнал. Весовые коэффициенты, или веса связей между нейронами, представляют собой параметры, определяющие работу нейронной сети. Нейроны группируются в последовательность слоев, входные сигналы поступают на первый (входной) слой и последовательно проходят все слои до последнего (выходного). В нашем примере в качестве входных сигналов используются события, происходящие в системе (количество входов равно количеству анализируемых событий, при этом на каждый вход подается 1, если событие произошло, и 0, если события не было), каждый выход нейросети соответствует определенному типу вредоносной программы (значения на выходах могут меняться от 0 до 1, а в качестве решения выбирается тот выход, на котором в итоге зафиксировано самое большое значение).

Обучение нейронной сети может проводиться с учителем (на основе уже решенных задач) или без него (на основе реакции среды). Обучение с учителем (которое больше подходит для нашего случая) происходит при последовательном решении с помощью нейронной сети уже выполненных задач и сравнении полученного результата с уже известным: если они не совпадают, производится коррекция весовых коэффициентов связей между нейронами.

WriteProcessMemory\ CreateRemoteThread в файле, любой, даже самый захудалый файловый сканер, оснащенный хотя бы примитивным эвристическим анализатором, поднимет тревогу, ведь в нормальных программах такие последовательности вызовов API встречаются крайне редко. Но ведь шифрование файла, неявный вызов функций, получение адресов нужных API с помощью GetProcAddress по хешам их названий и прочие хитрости по сокрытию и запутыванию кода никто не отменял, и поэтому с этим надо что-то делать. Кроме того, при очень большом желании можно обойтись без WriteProcessMemory.

Поэтому проактивной защите приходится тяжело трудиться и контролировать и функции по работе с виртуальной памятью, и функции по созданию потоков. Конечно же, перехватывать эти функции нужно в ядре с помощью драйвера. Все функции по работе с виртуальной памятью основаны на функциях NtAllocateVirtualMemory, NtFreeVirtualMemory и NtWriteVirtualMemory, а API CreateRemoteThread — на NtCreateThread. И, например, DrWeb перехватывает их все с помощью своего драйвера dwprot.sys (рисунок 7).

Есть и другой способ отследить создание потоков — воспользоваться специальной функцией PsSetCreateThreadNotifyRoutine. Эта функция регистрирует функцию уведомления, которая вызывается в момент создания или уничтожения потока (рисунок 8). Поскольку функция уведомления вызывается в контексте потока, который инициировал создание нового потока, то определить инициатор и выяснить, насколько он его благонадежен, не составляет труда.

ЗА КАЖДЫМ ПРОЦЕССОМ НУЖЕН ГЛАЗ ДА ГЛАЗ

Я думаю, вряд ли кто-нибудь усомнится в необходимости контроля всех процессов, которые крутятся в памяти компьютера, и уж тем более о целесообразности выявления процессов, пытающихся скрыться от «всевидящего ока» Task Manager'а. Любой процесс может оказаться зловредным, а уж тот, который пытается скрыть свое присутствие в системе, и вовсе потенциальный враг. Поэтому все процессы необходимо строжайше учитывать

Рисунок 9. Использование PsSetCreateProcessNotifyRoutine в DrWeb для контроля над созданием процессов

и контролировать. Разработчики операционной системы Windows предусмотрели много механизмов для получения уведомлений о наступлении каких-либо событий. О применении некоторых из них в различных антивирусных программах я немного рассказал выше (такие механизмы основаны на использовании функций CmRegisterCallBackEx, RegNotifyChangeKeyValue и PsSetCreateThread NotifyRoutine).

Для регистрации функции уведомления о создании или завершения процесса предусмотрена функция PsSetCreateProcessNotifyRoutine (PsSetCreat eProcessNotifyRoutineEx начиная с Vista SP1), и большинство создателей антивирусных средств не считают зазорным ею пользоваться (к примеру, на рисунке 9 проиллюстрирована регистрация callback-функции в драйвере dwprot.sys от «Доктора Веба»). Операционная система вызывает зарегистрированный обработчик в двух случаях: когда процесс создается и когда процесс завершается. В первом из них функция уведомления вызывается, когда начальный поток уже создан, но его исполнение еще не началось. Во втором случае операционная система вызывает функцию уведомления перед завершением последнего потока в процессе. Что мы можем сделать, узнав, что какой-то процесс начал свою деятельность в операционной системе? Во-первых, мы можем просканировать всю память процесса, чтобы выяснить, нет ли в нем вредоносного кода (о том, как это сделать, мы уже знаем из прошлой статьи). Во-вторых, пробить этот процесс по списку доверенных и благонадежных. В-третьих, узнать, не предпринималась ли попытка создать процесс с отрицательным PID, что однозначно говорит о намерении скрыть такой процесс. Ну и в-четвертых, можно получить список процессов с помощью CreateToolhelp32Snapshot (или NtQuerySystemInformation на более низком уровне) и проверить, есть ли там

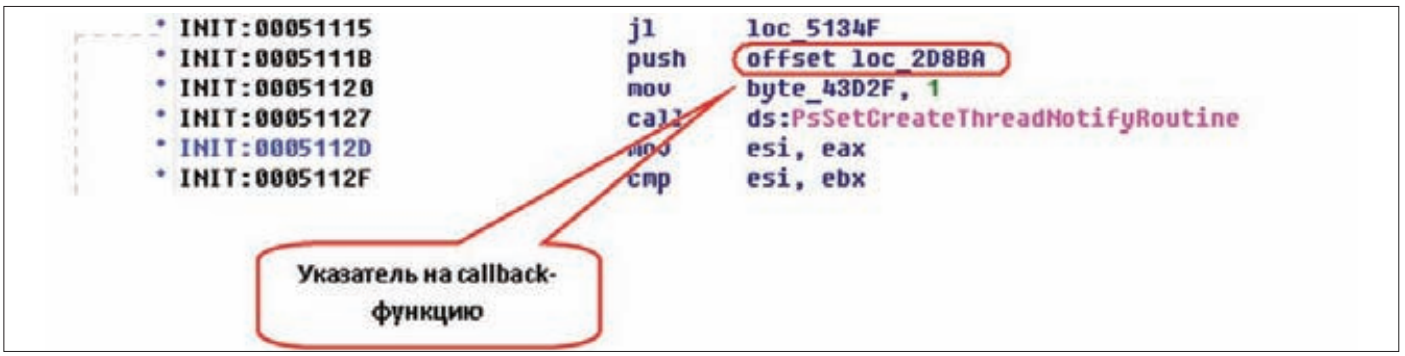


Рисунок 8. Настройка функции уведомления в «Антивирусе Касперского» на реагирование на создание потока

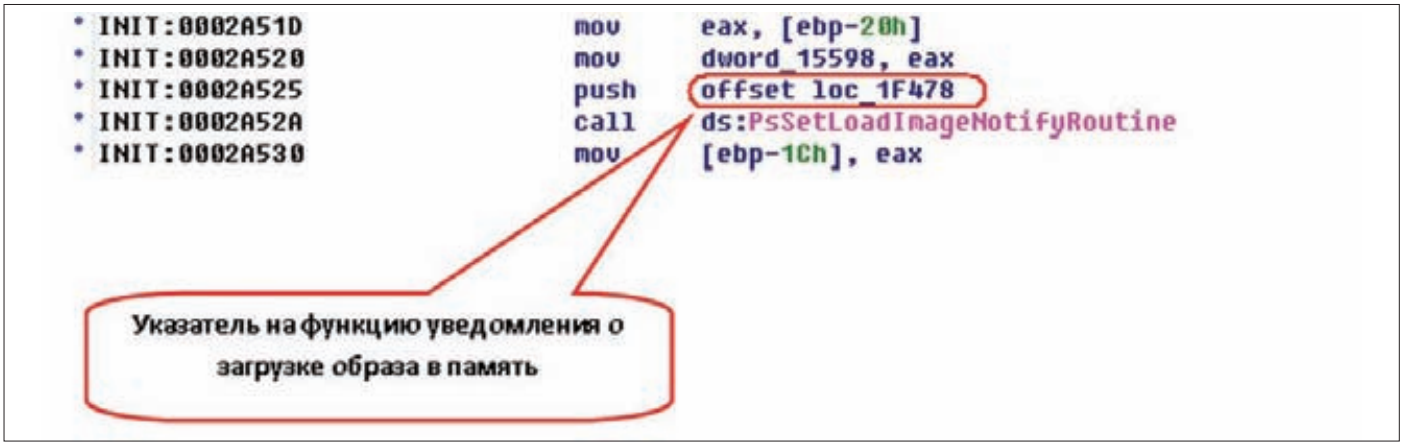


Рисунок 10. PsSetLoadImageNotifyRoutine в DrWeb контролирует загрузку драйверов

только что созданный процесс. Если его там нет, значит, ему есть что скрывать. При этом в большинстве случаев малварь скрывает запущенный процесс за счет перехвата функции ядра ZwQuerySystemInformation и фильтрации результатов работы этой функции. (Надеюсь, ты знаешь, чем API-функции с префиксом Zw отличаются от функций с префиксом Nt? Если нет, срочно читай Руссиновича.) Таким образом, обнаружив, что эта функция перехвачена кем-то неизвестным, срочно начинай бить тревогу.

Вообще, практически все более-менее серьезные вредоносные программы любят перехватывать различные функции, но это мы обсудим чуть позже.

ЗАГРУЗКА ДРАЙВЕРА

Думаю, ты знаешь, что драйвер служит не только для управления всякими устройствами, как было во времена старого доброго MS-DOS. Для многих программ драйвер — это путь к ядру системы, путь в ring0. Так же как любой солдат мечтает стать генералом, любая более или менее амбициозная малварь мечтает проникнуть в ядро системы и поставить ее на колени, а самый прямой и простой способ осуществить это — загрузка своего специально обученного вредоносного драйвера. В свою очередь, любой более или менее амбициозный авер просто обязан следить за всеми драйверами в системе для борьбы с этим явлением.

Первый способ, который напрашивается сам собой, — это контролировать загрузку драйверов путем перехвата API-функции NtLoadDriver. Он прост и, в общем-то, эффективен, им не брезгуют пользоваться очень многие производители антивирусных программ (к примеру, Comodo, F-Secure, «Лаборатория Касперского», которая включила этот механизм в свои антивирусы с шестой по девятую версию). Помимо этого, можно прибегнуть к уже знакомым нам функциям уведомления. Функция PsSetLoadImageNotifyRoutine регистрирует функцию уведомления, которая вызывается в момент загрузки образа или отображения образа в память. Операционная система вызывает зарегистрированную callback-функцию после отображения в память образа, исполняемого в пользовательском пространстве или в пространстве ядра (как раз то, что нам нужно, ведь драйвера как раз и грузятся в ядро), до начала исполнения образа (рисунок 10).

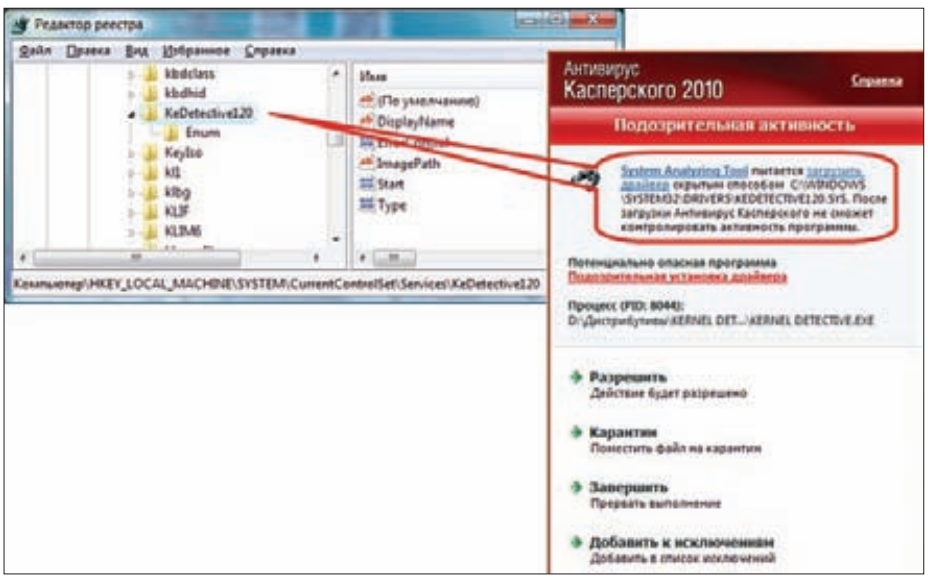


Рисунок 11. Реакция «Антивируса Касперского» на загрузку драйвера от Kernel Detective 1.2. Может, он и вправду вредоносный?..

Кроме того, некоторые антивирусы учитывают, что для загрузки драйвера его нужно прописать его в ветке реестра HKLM\System\CurrentControlSet\Services (рисунок 11). Достаточно следить за этой веткой, чтобы в случае появления там подозрительного раздела с параметром Type, значение которого равно 1, 2 или 8 (это SERVICE_KERNEL_DRIVER, SERVICE_FILE_SYSTEM_DRIVER или SERVICE_RECOGNIZER_DRIVER соответственно), своевременно поднять тревогу (рисунок 11). Надо сказать, что не все антивирусы оповещают пользователя о загрузке драйвера. Некоторые аверы однозначно и бесповоротно считают любой подозрительный загружаемый драйвер вредоносным и без спросу удаляют и сам драйвер, и программу, которая пыталась его загрузить, после чего бодро рапортуют, что малварь уничтожена.

ЧТО ДАЛЬШЕ?

Наш главный редактор уже многозначительно постукивает резиновой дубинкой по столу, намекая на то, что выделенное под статью место заканчивается. А ведь мне так хотелось рассказать про распознавание кейлоггеров, антируткит-технологии, «песочницу», подозрительную сетевую активность и целостность системных файлов, да и аналитический компонент проактивной защиты остался без внимания... Ну что же — жди продолжения, следующий номер не за горами.