вторжение в ядро вислы — спроси меня как!

крис касперски ака мыщъх, no-email

пресс-релизы, распространяемые империей зла, пугают нас чудовищными имениями в ядре вислы, направленными на борьбу с малварью и фактически ставящей хакеров на грань выживания. кое-кто даже поговаривает, что вирусы скоро вымрут как мамонты. ну с вирусами ладно — покончить с ними нам обещают с завидной регулярностью, но все никак не кончат. хуже всего то, что висла накладывает серьезные ограничения на хаченье ядра, и на грани выживания оказываются совсем не вирусы, а… хакеры и легальные разработчики проактивных защитных средств. но… на самом деле, все эти слухи _очень_ сильно преувеличены и внедриться в ядро вполне реально.

введение

Чем отличается Висла от XP? Если отбросить интерфейс, разработанный для блондинок и прочих пионерок, мы получим ядро, доставшееся в наследство от Windows Server 2003 и претерпевшее ряд существенных изменений, вылившихся в принципиально новую драйверную модель (и потому драйвер, написанный с учетом особенностей Висты, под XP работать просто не будет, и разработчики к своей огромной радости вынуждены поддерживать сразу две независимых ветви драйверов — для Висты и XP, они-то наивные думали, что этот кошмар кончился еще со смертью 9x).

Но в контексте безопасности (о котором мы, собственно, и говорим), изменения 32-разрядной версии Висты настолько невелики, что о них не стоит даже и говорить. Ну закрыли доступ к псевдоустройству \Device\PhysicalMemory с прикладного уровня — так это еще в Server 2003 SP1 было сделано. Толку с того? \Device\PhysicalMemory редко использовался хакерами, и в основном пострадали легальные программисты.

Заблокировали прямую секторную запись на неразмонированный том. И, между прочим, правило сделали. Кроме Рутковской ее все равно никто не использовал. Но даже сейчас остается куча путей для обхода блокировки, которые Microsoft не закрыла и судя по всему закрывать и не собирается, так что хакеры продолжают пребывать в нирване, собираясь нанести очередную серию атак. Вот только покурят сначала.

Другое дело — x86-64 версия Вислы, ставящая всех разработчиков раком и снабженная нереально крутыми защитными механизмами, такими, что почитав мануал, можно даже не курить, потому что крыша уже сорвана и назад уже не воротится.

Первое и главное — все драйвера в обаятельном порядке должны быть снабжены цифровой подписью и загрузить драйвер даже обладая правами администратора у нас не получится!!! К 32-битной версии Вислы (и в 64-битной версии под Itanium) это _не_ относится (см. рис. 1) и мы по-прежнему можем загружать неподписанные драйвера функцией **ZwLoadDriver** или любым другим способом (например, через реестр).

Microsoft Windows

Kernel Enhancements for Windows Vista and Windows Server Longhorn

Draft Version 0.9 - May 2006

Digital Signatures

Digital signatures allow an administrator or an end user who is installing Windows-based software to know whether a legitimate publisher has provided the software package.

Windows Vista relies on digital signatures on kernel-mode code to increase the safety and stability of the Microsoft Windows platform and enable new customer experiences with next generation premium content:

- → Drivers must be signed for devices that stream protected content. This includes audio drivers that use Protected User Mode Audio (PUMA) and Protected Audio Path (PAP), and video device drivers that handle protected video path-output protection management (PVP-OPM) commands.
- → Unsigned kernel-mode software does not load and does not run on x64-based systems.

  Even users with administrator privileges cannot load unsigned kernel-mode code on x64-based systems. This applies for any software module that loads in kernel mode, including device drivers, filter drivers, and kernel services.

The mandatory kernel-mode code signing policy applies to all kernel-mode software on x64-based systems that are running Windows Vista. However, Microsoft encourages publishers to digitally sign all software, including device drivers for both 32-bit and 64-bit platforms. Windows Vista performs kernel-mode signature verification on x86-based systems to support protected media content. Kernel-mode driver signatures are not mandatory for 32-bit systems.

Рисунок 1 32-битная версия Вислы не требует обязательной цифровой подписи для драйверов

Второе — x86-64 версия Вислы препятствует модификации ядра, предотвращая сплайсинг (т. е. перехват системных вызовов), подмену обработчиков в таблице прерываний и т. д. За это отвечает механизм **PatchGuard**, который хакеры взломали еще до того, как Висла попала на прилавок и атаки на ядро все еще продолжаются, а вот легальные разработчики сильно страдают, поскольку без модификации ядра невозможно написать ни нормальный антивирус, ни брандмауэр, ни другой продукт подобного типа.

К счастью, огромное количество драйверов, написанных под W2K/XP, модифицируют ядро в целях производственной необходимости, и по соображениям обратной совместимости, x86-версия Вислы (и 64-битная Висла под Itanium) _никак_ не препятствуют модификации ядра (см. рис. 2), хоть Microsoft и грозится, что скоро все будет не так. Но это навряд ли, поскольку, операционная система, на которой не запускаются любимые (и, к тому же, легально купленные!) приложения идет в топку.

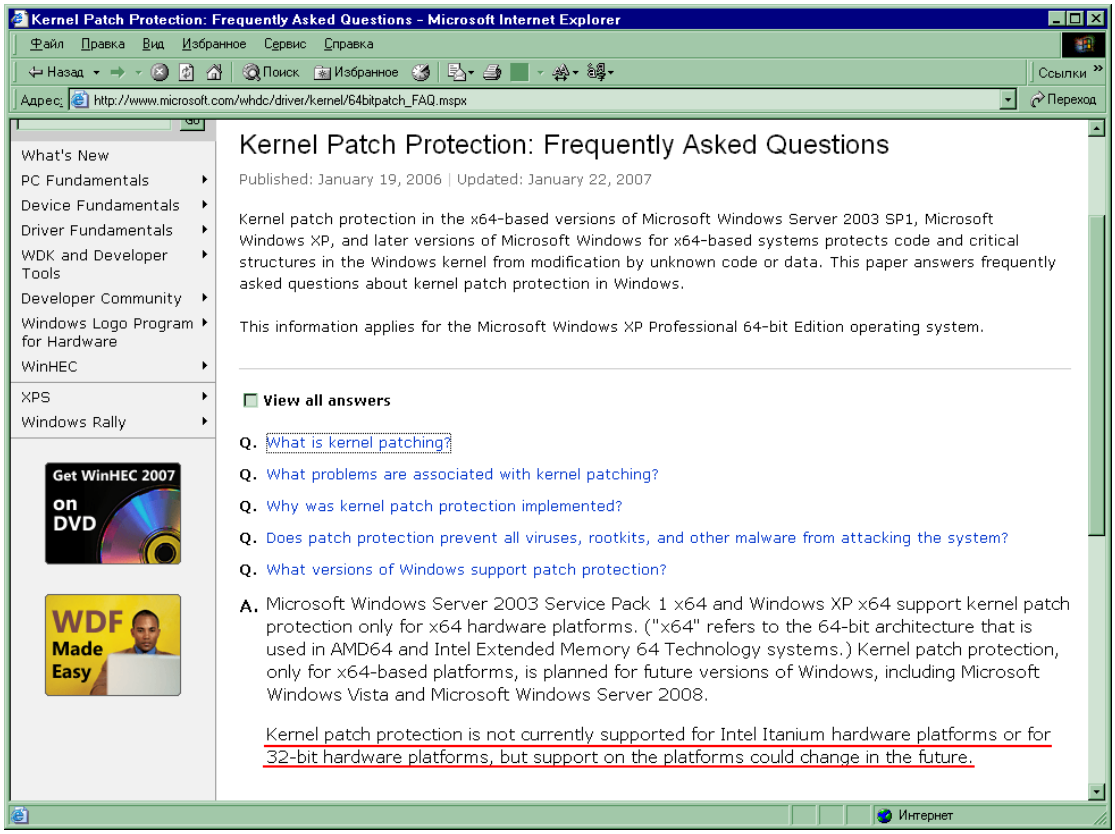

Рисунок 2 32-битная версия Вислы не проверяет целостность ядра (во всяком случае пока)

Есть и другие менее существенные изменения, но для нас они не представляют ровным счетом никакого интереса. Главным образом мы будем говорить об атаках на ядро x86-64 версии Вислы/Longhorn 2008, поскольку 32-битные версии атакуются по прежней схеме.

>>> врезка основные "хакерские" отличия Вислы от XP

- **x86-версия и 64-битная версия под Itanium:**
  - заблокирован доступ к псевдоустройству \Device\PhysicalMemory с прикладного уровня:
    - защита взломана хакером по кличке crazylord (см. Phrack 3Bh, "Playing with Windows /dev/(k)mem");
    - и вообще, это не слишком большая потеря для кодокопателей;
  - заблокирован прямой доступ к неразмонированным томам:
    - обходится через документированный интерфейс SPTI, недокументированные IOCTL-коды SCSIOP_ATA_PASSTHROUGH и IOCTL_IDE_PASS_THROUGH, а так же драйверами сторонних производителей: ASPI, RawDisk и др.;
- **x86-64-версия**:
  - требование цифровой подписи для _всех_ драйверов:
    - обходится утилитой Atsiv от Linchpin Labs;
    - не страхует от уязвимых драйверов (см. "пурпурная пилюля");
    - отсутствует механизм отзыва цифровой подписи для драйверов начального уровня;
    - работает только в паре с Defender, требуя своевременного обновления сигнатур;
  - запрет на модификацию ядра, обеспечиваемый механизмом Patch-Guard:
    - Patch-Guard не блокирует запись в ядро, а лишь проверяет его целостность с некоторой периодичностью, вполне достаточной для осуществления атаки и отлома самого Patch-Guard'a;

        - обходится кучей методик, например, генерации подложных байт, в результате чего контрольная сумма модифицированного участка не изменяется;
    - блокировка прямого доступ к неразмонированным томам:
        - обходится по одной из методик, описанных выше.

немного истории или опыт — сын ошибок трудных или блокировка записи на диск как ответ MS Рутковской

Экспериментируя с бета-версией Вислы, Жанна Рутковская предложила не слишком изящный, но все-таки пригодный для "промышленных" атак метод обхода цифровой подписи драйверов, вошедший в состав знаменитой "Голубой Пилюли" (Blue Pill), и продемонстрированный ею на хакерских конференциях SyScan (Сингапур) и BlackHat (Лас-Вегас) не по-детски жарким летом 2006 года.

Рисунок 3 Жанна Рутковская демонстрирует "Голубую Пилюлю" на конференции Black Hat

Полный текст презентации (на английском языке, в PowerPoint-формате) лежит на http://invisiblethings.org/papers/joanna%20rutkowska%20-%20subverting%20vista%20kernel.ppt, а в двух словах суть идеи можно обрисовать так: если "скушать" всю доступную память (например,

при помощи функции VirtuallAlloc), то ядро (в конфигурации по умолчанию!) начнет вытеснять драйвера на диск в файл подкачки, до которого можно "дотянуться" открыв диск как логическое устройство функцией CreateFile (требует прав администратора) и модифицировав выгруженные драйвера через WriteFile, внедрив в них shell-код, отключающий проверку цифровой подписи, например, после чего вредоносный драйвер может быть загружен обычный путем, то есть через ZwLoadDriver.

Рисунок 4 "кушаем" память, чтобы вытеснить ядро на диск

Поначалу, Microsoft послала Жанну с ее "атакой" в /dev/nul, поскольку, с правами администратора еще не такое можно сделать, однако, Жанна продолжала упорствовать, раздавая интервью и давая советы разработчикам ядра, что им глупым делать (см. блог Жанны на http://theinvisiblethings.blogspot.com/2006/10/vista-rc2-vs-pagefile-attack-and-some.html).

Конкретно, она предлагала следующее:

1. заблокировать посекторный доступ к диску из пользовательского режима;
2. зашифровать файл подкачки или хотя бы проверять его целостность, как на NetBSD;
3. запретить выгрузку ядерных компонентов на диск, пожертвовав ~80Мбайтами памяти;

Кстати говоря, последний механизм _уже_ реализован в Висле, начиная, кажется еще со времен NT 4.0, и переписывать ядро не нужно! Достаточно запустить "Редактор Реестра", зайти в ветку HKLM\SYSTEM\CurrentControlSet\Control\Session Manager\MemoryManagement, найти там параметр DisablePagingExecutive (по умолчанию равный нулю) и установить его в единицу. Вот и все! После перезагрузки мы получим невыгружаемое ядро. Кстати говоря, это полезно сделать уже хотя бы для того, чтобы избежать проблем с некоторыми багистрыми драйверами, разработчики которых забыли, что на уровне обработки прерываний диспетчер подкачки не работает и если драйвер попытается вызывать код, выгруженный на диск, система рухнет в голубой экран, но, это, впрочем, не относится к нашим баранам.

К большому удивлению окружающих, Microsoft пошла по первому пути, заблокировав прямую запись на неразмонтированный дисковый том (а системный том размонтировать нельзя, иначе как потом работать?!). Это вызывало шквал негодования в лагере поклонников

Рутковской, сочинивших кучу историй из серии, а как же писать утилиты для восстановления ошибочного удаленных файлов, например?!

```
-> file \pagefile.sys: inode = 0xc95c
-> file \pagefile.sys: attr DATA found ar
-> run list:
 0> vcn 0: lcn = 2085104, len = 338672
searching... 0.0%
-> pattern found in sector 16682008
WriteFile failed (err = 0x5)
Error while writing to disk!
```

Рисунок 5 под Vista RC2 "Голубая Пилюля" Рутковской уже не работает, выдавая сообщение об ошибке "Write File failed (err = 0x5)"

На самом деле, Microsoft выбрала правильное решение, исправив древний баг, на который до сих пор просто как-то не обращали внимания. Прямая запись на неразмонированный том чревата полным разрушением последнего. Допустим, прикладная программа модифицирует запись MFT для восстановления ошибочного удаленного файла, а в это же самое время операционная система перемещает MFT на другое место или использует освободившуюся запись для размещения нового файла. В результате происходит хаос.

Доказательством того, что Microsoft действительно исправила баг, а вовсе не кинулась в бой с Рутковской, служит тот факт, что открытие дискового устройства функцией CreateFile происходит вполне успешно, а вот функция WriteFile клеит ласты независимо от того, какой сектор она обновляет — принадлежащий или не принадлежащий файлу подкачки (исключение составляют первые 8 Кбайт тома, что соответствует 16 секторам, — в них запись по прежнему разрешена).

Несистемный том размонтируется в два приема: вызываем CreateFile и передаем полученный обработчик функции DeviceIoControl, вызываемой с флагами FSCTL_LOCK_VOLUME или FSCTL_DISMOUNT_VOLUME, однако, если даже файл подкачки расположен на другом томе размонтировать его все равно не удастся. Тупик? Вовсе нет. Существует множество других методов низкоуровневой работы с диском (как документированных, так и нет).

Прежде всего это интерфейс **SPTI** (SCSI Pass-Through Interface), присутствующий во всех NT-подобных системах и позволяющий посылать дисковым устройствам SCSI-команды, преобразуемые операционной системой в "нативные" команды данного устройства, в роли которого может выступать хоть "флешка", воткнутая в USB, хоть IDE-винт. Достаточно слегка покурить MSDN – поиск по IOCTL-командам: SCSI_PASS_THROUGH, IOCTL_SCSI_PASS_THROUGH и IOCTL_SCSI_PASS_THROUGH_DIRECT выдает много интересного, в том числе и готовые демонстрационные примеры (включенные в DDK). Эксперимент показывает, что низкоуровневая запись на системный том через SPTI до сих пор не заблокирована (хоть и требует прав администратора).

Существует так же большое количество недокументированных IOCTL-кодов. Например, следующие команды предназначены для "прямой" работы с IDE-дисками: SCSIOP_ATA_PASSTHROUGH/IOCTL_IDE_PASS_THROUGH и они так же _не_ заблокированы (во всяком случае пока… ну а там мы что-нибудь придумаем).

Так же хочется вспомнить про **ASPI**-драйвер от компании Adaptec, через который работают многие программы пишущие или копирующие CD/DVD. Это очень глючный драйвер и при определенных обстоятельствах он дает прямой доступ не только к оптическим приводам, но и к жестким дискам, причем, _без_ прав администратора (впрочем, и без всяких гарантий, что он вообще заработает).

Рисунок 6 RawDisk – сторонний драйвер для Вислы, обеспечивающий возможность прямой записи на любой диск, включая системный

Наконец, можно не париться, а воспользоваться нормальным драйвером **RawDisk** от компании ELDOS (http://www.eldos.com/rawdisk), позволяющим писать куда угодно в любой версии Вислы. Естественно, если мы точим вирус, а не утилиту для восстановления ошибочного удаленных файлов, этот драйвер придется всюду таскать за собой и к тому же платить за него деньги, поскольку он не бесплатен.

>>> вставка обход блокировки прямого доступа к диску

- использование штатного механизма SPTI (требует наличия прав администратора) через документированные IOCTL-команды: IOCTL_SCSI_PASS_THROUGH_DIRECT; SCSI_PASS_THROUGH и IOCTL_SCSI_PASS_THROUGH;
- использование недокументированных IOCTL-кодов (так же требует прав администратора): SCSIOP_ATA_PASSTHROUGH и IOCTL_IDE_PASS_THROUGH;
- использование драйвера ASPI от компании Adaptec (_не_ требует прав администратора, но работает крайне нестабильно);
- использование прочих драйверов сторонних производителей (например, RawDisk от компании ELDOS);

подпись драйверов

Механизм подписи драйверов, реализованный компаний Microsoft еще в Windows 2000, изначально не предназначался для защиты системы от вторжения и просто информировал

администратора о потенциальных проблемах, предотвращая установку драйверов, написанных мелкими фирмами, настолько мелкими, что даже не позаботившимися о получении подписи.

В x86-64 версиях Вислы подпись драйверов стала обязательной. Ну и что с того? Механизм подписи драйверов — есть, а вот процедуры отзыва подписи — нет. Представим себе, что сотрудник некоторой компании (занимающейся разработкой драйверов) крадет и выкладывает в открытый доступ секретный ключ, после чего драйвера может подписывать кто угодно и Microsoft _ничего_ не может сделать.

Ну… почти ничего. Антивирусная пародия с гордым названием Defender (если только она не выключена пользователем за ненадобностью) получает из сети свежие сигнатуры и потому _потенциально_ способа пресечь загрузку драйверов, подписанных украденной цифровой подписью, однако, это воздействует только на драйвера, загружаемые после WINLOAD.EXE, а первичные драйвера, загружаемые на ранней стадии загрузки операционной системы, ни Defender, ни какой либо другой механизм прищемить не в состоянии. Во всяком случае пока. Но даже потом… что делать с легальными драйверами, подписанными украденной подписью? Если они перестанут работать, система рухнет, что само по себе является нехилой распределенной атакой. Просто крадем секретный ключ крупной компании и выкладываем его в сеть. Айда фирма получает другой секретный ключ, заново подписывает _все_ ранее выпущенные драйвера, а пользователи их качают.

Но это все ерунда. Дизассемблирование показывает, что проверка цифровой подписи осуществляется всего в двух местах — в файлах NTOSKRNL.EXE и WINLOAD.EXE, которые можно хакнуть прямо на диске. Доступ к этим файлам заблокирован, но… ни фига! Переименуем файл NTOSKRNL.EXE в NTOSKRNL.HCK (система это разрешает), тут же скопируем его в NTOSKRNL.EXE, который уже и отпачтим. Тоже самое сделаем и с WINLOAD.EXE. Естественно, для этого придется предварительно усмирить Windows Resource Protection (она же WRP), устанавливающую права доступа к системным файлам в TrustedInstaller, что повыше администратора будет, однако, весь фокус в том, что обладая правами администратора, можно заполучить и TrustedInstaller. Подробнее о том как это сделать рассказывается в отчете корпорации Symantec: http://www.symantec.com/avcenter/reference/Windows_Vista_Kernel_Mode_Security.pdf

А теперь самое интересное! Лаборатория Linchpin Labs (занимающаяся разработкой системных утилит для Windows) совершила первую _реальную_ атакую на Вислу, доказав полную практическую бесполезность процедуры цифровой подписи. Зарегистрировав "левую" фирму она получила секретный ключ, которым подписала драйвер **Atsiv**, позволяющий загружать неподписанные драйвера, и не просто загружать, а еще и скрывать их присутствие в системе, что достигается путем исключения драйвера из списка PsLoadedModuleslist (точнее _не_ включения драйвера в список PsLoadedModuleslist, поскольку **Atsiv** использует собственный загрузчик), то есть **Atsiv** фактически представляет собой самый настоящий rootkit, образующий огромную дыру в безопасности. Сам-то он подписан, а вот загружаемые им драйвера — нет.

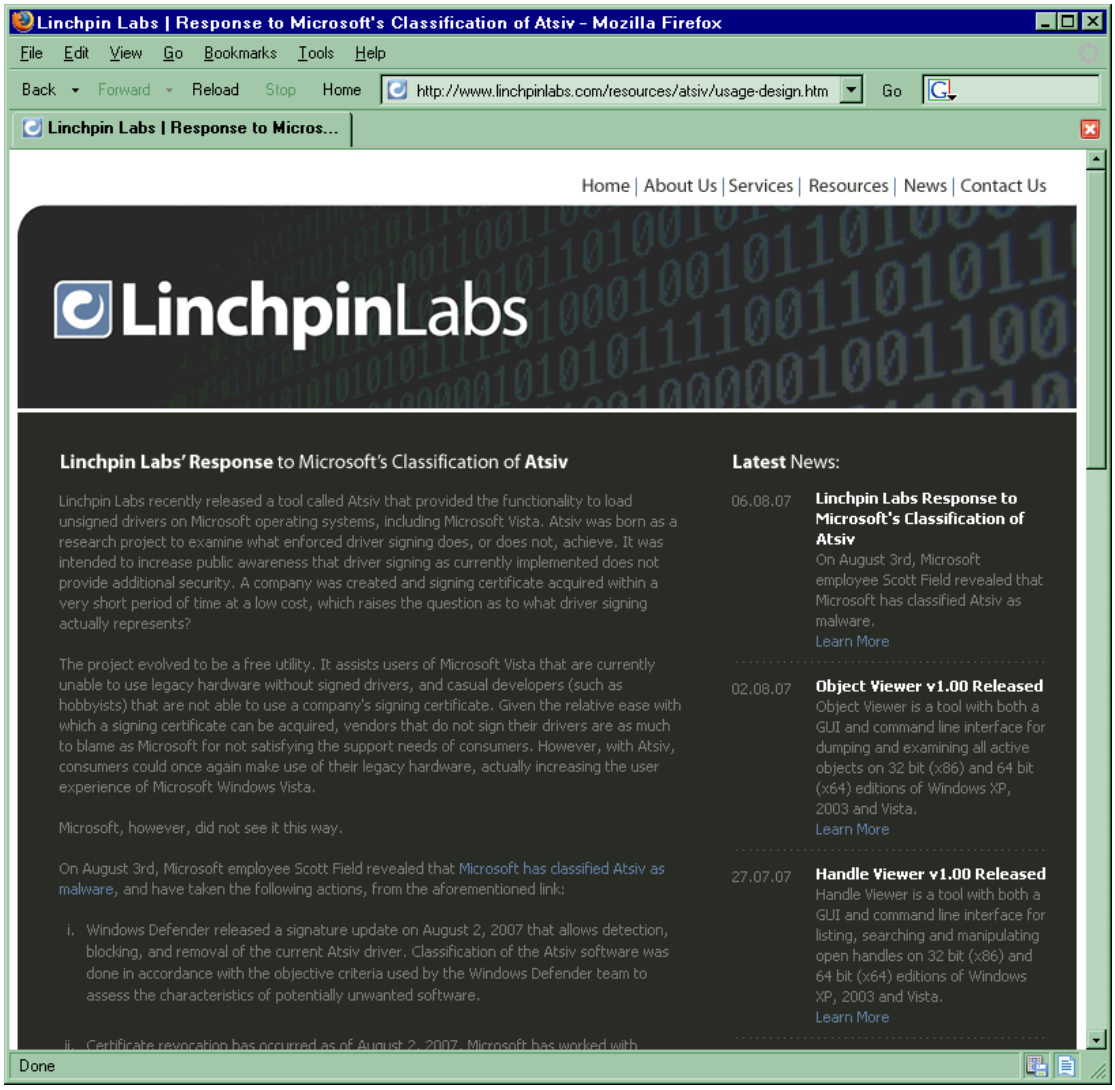

Рисунок 7 отсюда можно (было) бесплатно скачать драйвер для x86-64 Вислы, позволяющий загружать неподписанные драйвера

Получение секретного ключа — чисто формальная процедура, и "нотариально" заверить драйвер — не проблема. Лабораторию Linchpin Labs вообще сильно удивило насколько просто делаются такие вещи. Но ведь иначе и быть не может! Драйвера создаются тысячами фирм и если каждую фирму подвергать жестокой проверки на предмет откуда она взялась и чем собирается заниматься, то Висла сильно рискует остаться без свежих драйверов. Разработчики забьют на нее и пересядут на Linux. А без драйверов Висла никому не нужна. Даже даром.

Реакция Microsoft последовала незамедлительно и в базе Defender'a появилась новая сигнатура, блокирующая запуск **Atsiv'a** (при условии, что пользователь держит Defender включенным и своевременно обновляет базу сигнатур), а сам **Atsiv** внезапно исчез с сайта разработчиков (http://www.linchpinlabs.com/resources/atsiv/usage-design.htm), хотя до этого он распространялся бесплатно. То есть, Microsoft как бы разрулила ситуацию.

"Как бы" потому что остается без ответа вопрос, поставленный лабораторий Linchpin Labs: "A signed file uniquely identifies the company that developed that file, but when companies can be created and registered in jurisdictions known for protecting the privacy of company founders and directors, you have to ask what does driver signing actually represent? Signed drivers can be signed by an arbitrary legally registered company. Absent any control over what the driver actually is or does, this provides no real additional security, other than removing author anonymity" ("*Подписанный файл однозначно идентифицирует компанию, разработавшую его, но когда компании создаются и регистрируются таким образом, что истинные основатели и директоры фирмы остаются в тени, спрашивается: что же в действительности представляет собой цифровая подпись?*

Отсутствие реального контроля за поведением драйвера не обеспечивает никакой реальной безопасности, за исключением невозможности распространения анонимных драйверов").

В общем, цифровая подпись представляет лишь видимость защиты, и с rootkit'ами приходится бороться дедовскими методами — путем антивирусной проверки с постоянно обновляемой базой сигнатур. Если у нас нет антивируса, мы можем "подцепить" малварь, снабженную цифровой подписью, но если антивирус есть — какой смысл в цифровой подписи? Правильно, никакого смысла в ней нет, только лишняя головная боль легальным разработчикам.

>>> вставка основные дефекты цифровой подписи

- нет возможности отзыва подписи драйверов (кроме как через Defender);
- утилита Atsiv от лаборатории Linchpin Labs легко грузит неподписанные драйвера;
- проверка цифровой подписи может быть отключена патчем файлов NTOSKRNL.EXE и WINLOAD.EXE на диске;

пурпурная пилюля и другие драги в ассортименте

Никакой программный продукт не застрахован от ошибок и потому драйвера представляют собой весьма соблазнительный объект для хакерских атак, причем такие атаки начались задолго до появления Вислы и заморочек с цифровой подписью.

Все очень просто — драйвер работает в режиме ядра и взаимодействует с прикладными программами через те или иные механизмы, зачастую даже не требуя прав администратора. Допустим, в драйвере есть ошибка типа переполнения буфера, позволяющая "впрыснуть" shell-код в ядерное пространство, повысить уровень своих привилегий или выполнить руками драйвера действие, которое (при нормальном развитии событий) недоступно с прикладного режима (например, осуществить запить в порт ввода/вывода).

Учитывая, что многие драйвера создаются вовсе не для управления какими-то устройствами, а как раз для выполнения действий, необходимых разработчику, но недоступных с прикладного уровня (такие драйвера часто называется "псевдодрайвера"), угроза атаки становится вполне реальной. Теоретически, разработчик псевдодрайвера должен предотвратить его "неавторизованное" использование посторонними программами, практически же… нас окружает куча дырявых драйверов, часть из которых уже работает в Висле.

Рисунок 8 Пурпурная Пилюля от Alex Ionescu

В середине августа 2007 года, Alex Ionescu обнаружил ошибку в видео-драйвере AMD ATI ATIDSMXX.SYS, позволяющую локальному пользователь читать/писать ядерную память с прикладного уровня, впрыскивая shell-код или отключая механизм проверки цифровой подписи. Для демонстрации атаки он написал "Пурпурную Пилюлю" (Purple Pill), и выложил ее в открытый доступ 7 августа, а через 78 часов по соображениям хакерской этики убрал обратно, но 39 человек уже успели ее скачать.

```
C:\tmp\amd-hotstuff\atdcm64a.sys:
        Signers:
                ATI Technologies, Inc
                VeriSign Class 3 Code Signing 2004 CA
                Class 3 Public Primary Certification Authority
        Signing date:   03:20 03/02/2007
        Publisher:      ATI Technologies Inc.
        Description:    ATI DSM Dynamic Driver
        Product:        ATI DSM Dynamic Driver
        Version:        3.0.502.0
        File version:   3.0.502.0 built by: WinDDK
```

Рисунок 9 дырявый драйвер от AMD/ATI

Дыра в уязвимом драйвере была оперативно исправлена, но… как заставить пользователей скачать обновления?! Microsoft, перебрав несколько вариантов, решила ничего не предпринимать, поскольку уязвимые драйвера установлены на миллионах машин (включая ноутбуки) и если "забанить" их в Defender'е пользователи просто завопят и судьба Вислы будет предрешена. Кому нужна система, которая в _обязательном_ порядке требует установки обновлений, принудительно переводя монитор в позорный VGA-режим?!

I've made **Purple Pill** available here. It also comes with a simple u
which prints something on the debug buffer, sets up some dummy
an unload routine.

Please send me any feedback if the tool doesn't work for you as

Рисунок 10 Пурпурная Пилюля в открытом доступе

Дыра в AMD/ATI драйвере первая (под Вислу), но уж точно не последняя и пилюли разных цветов будут появляться и дальше, поскольку, намного проще (и дешевле) использовать брешь в чужом драйвере, чем регистрировать левую фирму для подписи своего собственного. Естественно, наибольший интерес представляют драйвера, входящие в штатный комплект поставки Вислы и работающие с сетевым стеком, позволяя осуществлять не только локальные, но и удаленные атаки.

Все что нам нужно — это отладчик и дизассемблер. Список наиболее "соблазнительных" драйверов с краткими пояснениями приведен ниже (примечание: для упрощения анализа рекомендуется загрузить отладочные символы с серверов Microsoft, которые распространяются бесплатно):

- NETIO.SYS:
  - реализует IPv4/IPv6 сетевой стек;
- HTTP.SYS
  - обрабатывает HTTP-запросы;
- MRXSMB10.SYS
  - отвечает за поддержку SMB версии 1;
- MRXSMB20.SYS
  - отвечает за поддержку SMB версии 2;
- MRXDAV.sys:
  - обрабатывает WebDAV;
- MSRPC.sys
  - реализует механизм удаленного вызова процедур MS RPC

>>> вставка основные вектора атаки на драйвера

- ошибки переполнения буферов и др.;
- документированные и недокументированные IOCTL-коды;
- возможность неавторизованного использования драйвера чужими приложениями;

заключение

Всякий раз, выпуская очередную операционную систему, Microsoft обещает положить конец вирусам, червям и хакерам, но… всякий раз исполнение обещанного переносится на

неопределенный срок. Новоявленные защитные механизмы (как правило, позаимствованные из мира UNIX) взламываются задолго до того, как система успевает поступить на рынок. Противостояние меча и щита продолжается…

Разумеется, мы рассмотрели лишь некоторые, наиболее интересные атаки на ядро Вислы, оставив за бортом огромный пласт материала (включая механизм Patch-Guard, заслуживающий отдельной статьи), однако, какие бы пакеты обновлений ни выходили, хакеры прорвали стратегически рубежи обороны ядра и вышли на оперативный простор. Естественно, Microsoft это дело так не оставит и будет нам всячески противостоять ("нам" — это и хакерам, и легальным разработчикам), так что читайте журнал "Хакер", чтобы быть в курсе последних событий, которые мы будем освещать.

Рисунок 11 логотип Вислы

>>> вставка что в имени твоем?

В переводе с английского "Vista" означает "перспектива", "перспективы, возможности, виды (на будущее)", что выглядит очень странно в свете решения Microsoft "похоронить" эту перспективу через два года, заменив ее новой системой. Существует мнение, что Висла окажется чем-то вроде Windows Me – плохо продуманной, сделанной на скорую руку системой, выброшенной на рынок лишь затем, чтобы заполнить образовавшийся вакуум хоть чем ни будь.

>>> врезка полезные ссылки

- **"Invisible Things":**
  - официальный сайт Жанный Рутковской (на английском языке) с кучей интересных материалов и ворохом хакерских утилит: http://theinvisiblethings.org;
- **official blog of the "Invisible Things":**

- официальный блок Жанны Рутковской (на английском языке): http://theinvisiblethings.blogspot.com/;

- **Subverting Vista Kernel:**
  - текст презентации, посвященной "Голубой Пилюле" (на английском языке): http://invisiblethings.org/papers/joanna%20rutkowska%20-%20subverting%20vista%20kernel.ppt;
- **Assessment of Windows Vista Kernel-Mode Security:**
  - анализ ядра Вислы с точки зрения безопасности, выполненный сотрудниками исследовательского центра корпорации Symantec (на английском языке): www.symantec.com/avcenter/reference/Windows_Vista_Kernel_Mode_Security.pdf;
- **Atsiv от Linchpin Labs:**
  - утилита командной строки, позволяющая загружать неподписанные драйвера: http://www.linchpinlabs.com/resources/atsiv/usage-design.htm;
- **Driver Signing on Vista 64-bit - Using the Process against Itself:**
  - анализ утилиты **Atsiv** от корпорации Symantec (на английском языке): http://www.symantec.com/enterprise/security_response/weblog/2007/07/driver_signing_on_vista_64bit.html;
- **64-bit Driver Signing on Windows Vista - 'Computer Says No':**
  - объяснение от корпорации Symantec почему цифровая подпись драйверов плохая идея, ничуть не увеличивающая безопасность (на английском языке): http://www.symantec.com/enterprise/security_response/weblog/2007/08/64bit_driver_signing_on_window.html;
- **Purple Pill: What Happened:**
  - описание принципов работы "Пурпурной Пилюли" (на английском языке): http://www.alex-ionescu.com/?cat=2;
- **AMD ATI ATIDSMXX.SYS Driver Local Privilege Escalation Vulnerability:**
  - описание бага в AMD/ATI драйвере не Security Focus'e (на английском языке): **http://www.securityfocus.com/bid/25265/;**
- **Exploiting Windows Device Drivers**:
  - обзорная статья, посвященная проблемам поиска дырявых драйверов: http://www.piotrbania.com/all/articles/ewdd.pdf;
- http://ati.amd.com/products/wp/ATIWDDMWhitepaperFinalV38.pdf