exploit review
(4й выпуск)

крис касперски ака мыщъх, no-email

OpenSSL: подделка цифровой подписи RSA

brief: в конце августа 2006 года Daniel Bleichenbacher выступил с докладом на конференции криптоаналетиков, где показал, что при стечении определенных обстоятельств цифровая подпись RSA может быть подделана в прямом смысле слова — одной бумагой и карандашом (даже без использования кластера суперкомпьютеров). виною тому не сам алгоритм RSA, а ошибки его реализации. одной из таких реализаций оказался знаменитый проект OpenSSL, которой вплоть до версии 0.9.8c использовал RSA ключи с экспонентой 3, удаляющие padding-поля PKCS #1 до генерации хэш-суммы, которая позволяла удаленному атакующему подделать PKCS #1 сигнатуру, подписанную RSA-ключом и препятствующую корректной проверке различных цифровых сертификатов, использующих PKCS.
PKCS расшифровывается как Public-Key Cryptography Standards (Крипиграфические Стандарты на Публичный Ключ) и подробно описан в RFC-3447 (ftp://ftp.rfc-editor.org/in-notes/rfc3447.txt), технические детали атаки (со всеми математическими вкладками) лежат на: www.imc.org/ietf-openpgp/mail-archive/msg14307.html;

targets: уязвимости подвержены RSA-ключи с экспонентой 3, которые достаточно широко используется не только в OpenSSL, (уязвимы все версии вплоть до 0.9.7j и 0.9.8b входящих в состав практически всех LINUX'ов, xBSD и MAC OS), но так же воплощенных в кремнии и железе: маршрутизаторах CICSO, IBM Hardware Management Console и т. д.

exploit: отсутствует;

solution скачать последнюю OpenSSL версию с официального сайта проекта (www.openssl.org/source, ftp.openssl.org/source) или установить патч, взятый оттуда же www.openssl.org/news/patch-CVE-2006-4339.txt.

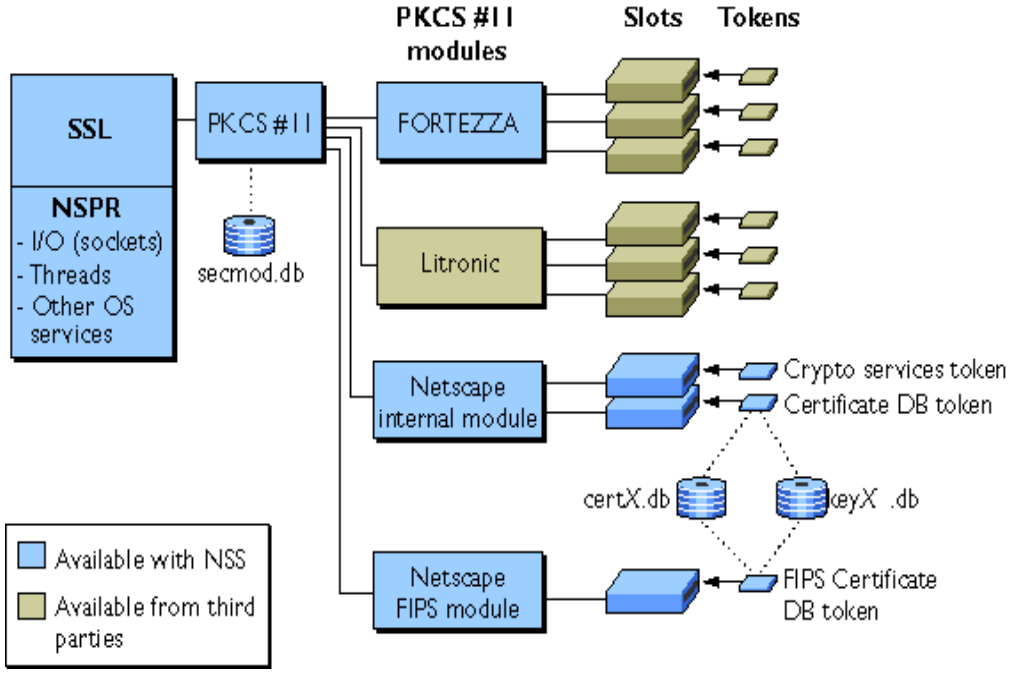

Рисунок 1 место PKCS в общей иерархии криптосистемы

ICQ под угрозой

brief: за минувший месяц в популярном интернет-пейджере ICQ обнаружились две дыры. первая, связанная с ICQ toolbar, позволяет злоумышленнику: читать содержимое файла RSS, вызывать toolbar-методы (RefreshRSS, OpenFeed, MarkAsRead, OpenRSSDialog, CloseRSSFrame, SetRSSNotificationFlag, OpenRSSNewDialog...), просматривать содержимое текущей WEB-страницы и воровать cookies, содержащие конфиденциальную информацию. это делается путем копирования файла "options2.html", лежащего в папке с ICQ toolbar'ом и размещении его на любом из "хакерских" сайтов после надлежащей модификации. технические детали можно узнать на www.securityfocus.com/archive/1/445515/30/0/threaded;
другая, намного более опасная, дыра связана с отсутствием контроля длины некоторых полей в IM-сообщениях, отправляемых получателю напрямую, без использования сервера. переполнение буфера происходит в функции MCRegEx__Search(), которая вызывает функцию memset() для очистки буфера, выделенного из динамической памяти (кучи), не проверяя при этом фактический размер блока, результатом чего становится отказ в обслуживании (технические детали доступны по ссылке www.coresecurity.com/index.php5?action=item&id=1509). выполнение shell-кода, как это сообщалось на http://www.securityfocus.com/bid/19897/discuss на самом деле невозможно.
обе уязвимости были обнаружены командой Core Team (www.coresecurity.com).

targets: ICQ Toolbar 1.2, 1.3/ICQ Pro 2003 b, ICQ 99a 2.21;

exploit: не требуется;

solution отключить ICQ toolbar и установить свежую заплатку от производителя в соответствии со своей версией ICQ: www.securityfocus.com/bid/19897/solution.

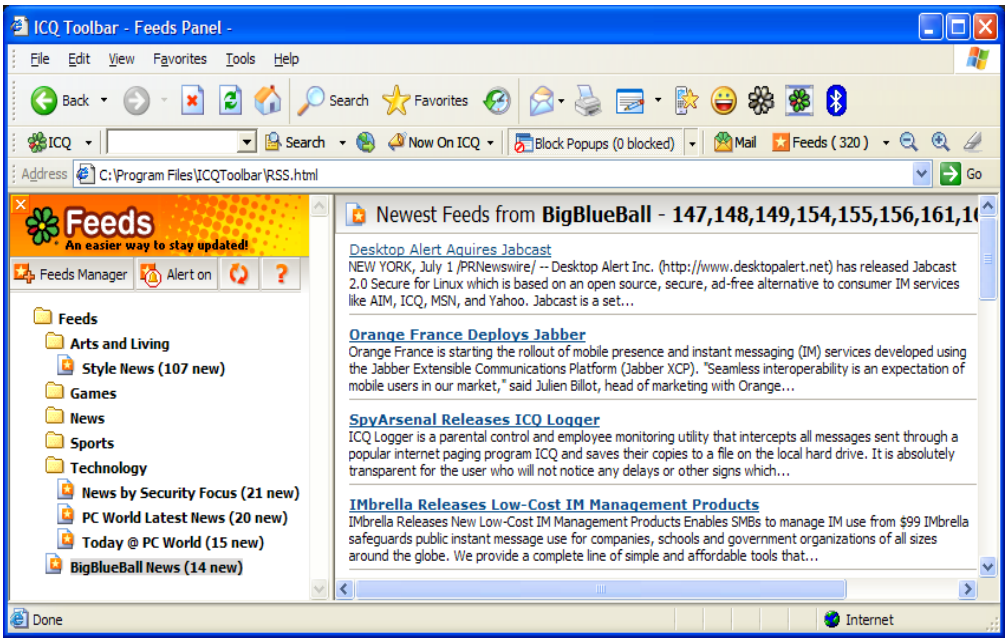

Рисунок 2 ICQ toolbar – удобная мишень для атаки

множественные уязвимости в горящем лисе

brief: FireFox и родственные ему продукты (SeaMonkey, Camino, Thunderbird…) теряет статус безопасного браузера и свежие дыры обнаруживаются одна за другой, позволяя злоумышленнику: выполнять зловредный shell-код в контексте уязвимого приложения, вызывать крэш браузера, запускать JavaScript с повышенными привилегиями вплоть до передачи управления на машинный код, воровать секретную информацию и т. д. отсутствие атак объясняется относительно невысокой распространенностью FireFox'a,

однако, это лишь уменьшает вероятность атаки, но не исключает ее полностью, к тому же интерес хакеров к FireFox'у растет пропорционального квадрату его популярности и скоро в Сети появится странички, изготовленные специально для атаки на FireFox.
бессмысленно разбирать каждую из обнаруженных дыр (на это не хватит никакого бумажного пространства), поэтому мы решили ограничится одним лишь перечнем ссылок, который лежит на www.securityfocus.com/bid/18228/references;

targets: **Mozilla Camino** 0.7, 0.8, .8.3, 0.8.4, 1.0, 1.0.1/**Mozilla Firefox** 0.8, 0.9, 0.9 rc, 0.9.1, 0.9.2, 0.9.3, 0.10, 0.10.1, 1.0, 1.0.1, 1.0.2, 1.0.3, 1.0.4, 1.0.5, 1.5.0.1, 1.0.6, 1.0.7, 1.0.8, 1.5, 1.5.0.2, 1.5.3, 1.5 beta 1, 1.5 beta 2/**Mozilla SeaMonkey** 1.0, 1.0 dev, 1.0.1/**Mozilla Thunderbird** 0.6, 0.7, 0.7.1, 0.7.2, 0.7.3, 0.8, 0.9, 1.0, 1.0.1, 1.0.2, 1.0.5, 1.0.6, 1.0.7, 1.0.8, 1.5, 1.5 Beta 2, 1.5.1, 1.5 .2;

exploit: для реализации большинства уязвимостей, никакого специального написанного exploit'a _не_ требуется и атака осуществляется непосредственно из самого браузера;

solution задействовать режим автоматического обновления и почаще просматривать новостной канал на сайте производителя (модуль автоматического обновления скачивает только стабильные версии продуктов, а заплатки зачастую наносятся поверх текущих), однако, обновленные версии нарушают работу некоторых расширений (extensions), заставляя нас выбирать между безопасностью и комфортной работой.
как вариант, можно пересесть на браузеры Opera, Lynx или Links ошибки которых, можно свободно пересчитать по пальцам одной руки.

Рисунок 3 все дело в шляпе, то есть в лисе (горящем)

full disclose
удаленное управление в Intel Centrino PRO Wireless Network с ядерными привилегиями

brief: 3 сентября 2006 года в 14:37:48 по восточному стандартному времени хакер **johnny cache** описал принципиально новую атаку на драйвера устройств беспроводной связи Intel Centrino PRO, открывающую новую страницу в книге переполняющихся буферов: lists.immunitysec.com/pipermail/dailydave/2006-September/003459.html.
используя ошибки синхронизации — race condition (вполне типичные для драйверов и хорошо известные каждому пользователю по голубому экрану смерти с ругательством IRQL_NOT_LESS_OR_EQUAL) ему удалось не только получить отказ в обслуживании, но и воздействовать на регистр EIP с передачей управление на shell-код, исполняющийся в режиме ядра, то есть с на самом высоком уровне привилегий, который только возможен. данная атака заслуживает всестороннего изучения, вот почему она была вынесена в отельную статью;

targets: в настоящее время под угрозой находится следующие устройства: Intel PRO/Wireless 2915ABG 9, 2915ABG 10, 2200BG 9, 2200BG 8, 2200BG 10, однако, список уязвимых драйверов все еще составляется, кроме того, аналогичные ошибки синхронизации встречаются в драйверах DSL-модемах, ИК адаптеров, Голубых Зубьев и др. устройств, обрабатывающих асинхронные запросы;

exploit: не требуется;

solution отключите беспроводные устройства (задействуйте их только при острой необходимости) или установите пакет обновления от Intel: support.intel.com/support/wireless/wlan/sb/CS-023065.htm (размер которого составляет порядка 100 Мбайт)

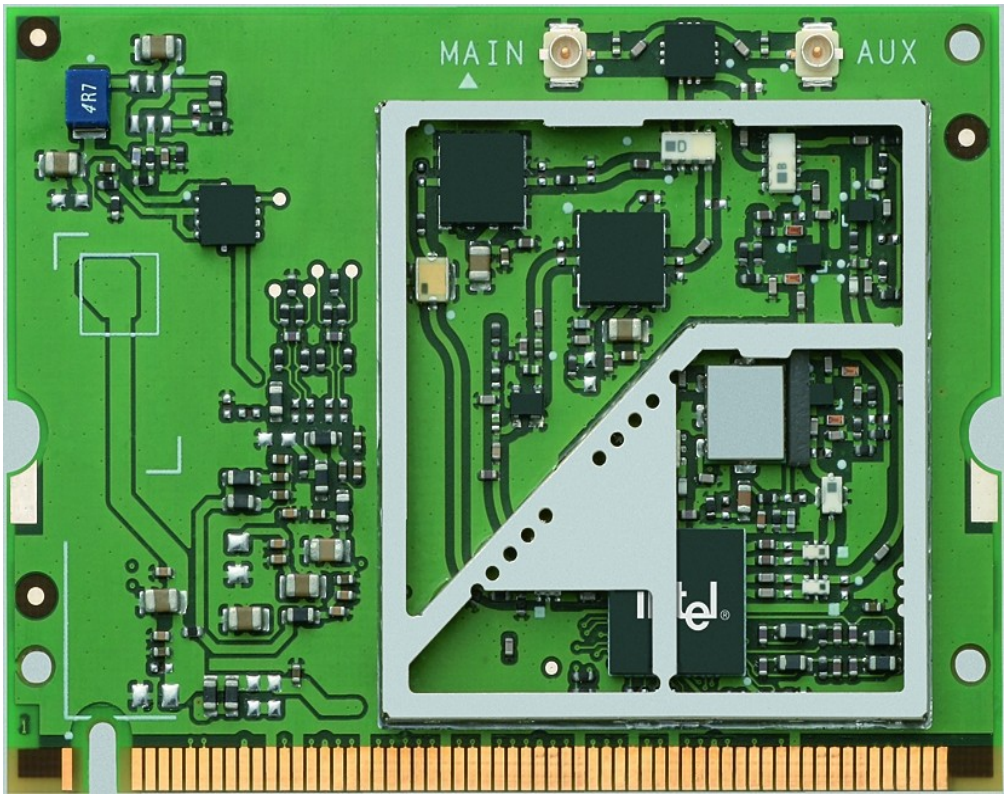

Рисунок 4 карта беспроводного доступа на основе Intel PRO/Wireless 2915ABG 9

details: ритуал вызова бага, появление которого сопровождалось голубым экраном смерти, в общих чертах выглядел так:

1) johnny cache установил на жертву, снабженную картой беспроводного доступа Intel Centrino PRO, утилиту **netcat** (входит практически в любой LINUX-дистрибутив, бесплатный порт для Windows можно нарыть на www.vulnwatch.org/netcat), заставив ее слушать 2048 порт (в принципе, можно обойтись и без этого, прослушка лишь увеличивает вероятность успешной атаки);
2) атакующий узел начал бомбардировать жертву UDP-пакетами размером 1400 байт, заполненных для наглядности CCh байтами, и поступающими на 2048-порт с интервалом в 400 микросекунд между ними;
3) одновременно с этим, атакующий направил шторм disassociation-запросов ("disassociation requests") с интервалом 4000 микросекунд. disassociation request – это одно из 6 командных сообщений беспроводного протокола, черновое описание которого лежит на tools.ietf.org/wg/netlmm/draft-giaretta-netlmm-dt-protocol-00.txt;
4) BSOD не заставил себя долго ждать и система рухнула. эксперименты показали, что вероятность возникновения голубого экрана смерти зависит _только_ от скважности пакетов, но отнюдь не от их содержимого, как и должно быть в ситуации с ошибкой синхронизации.

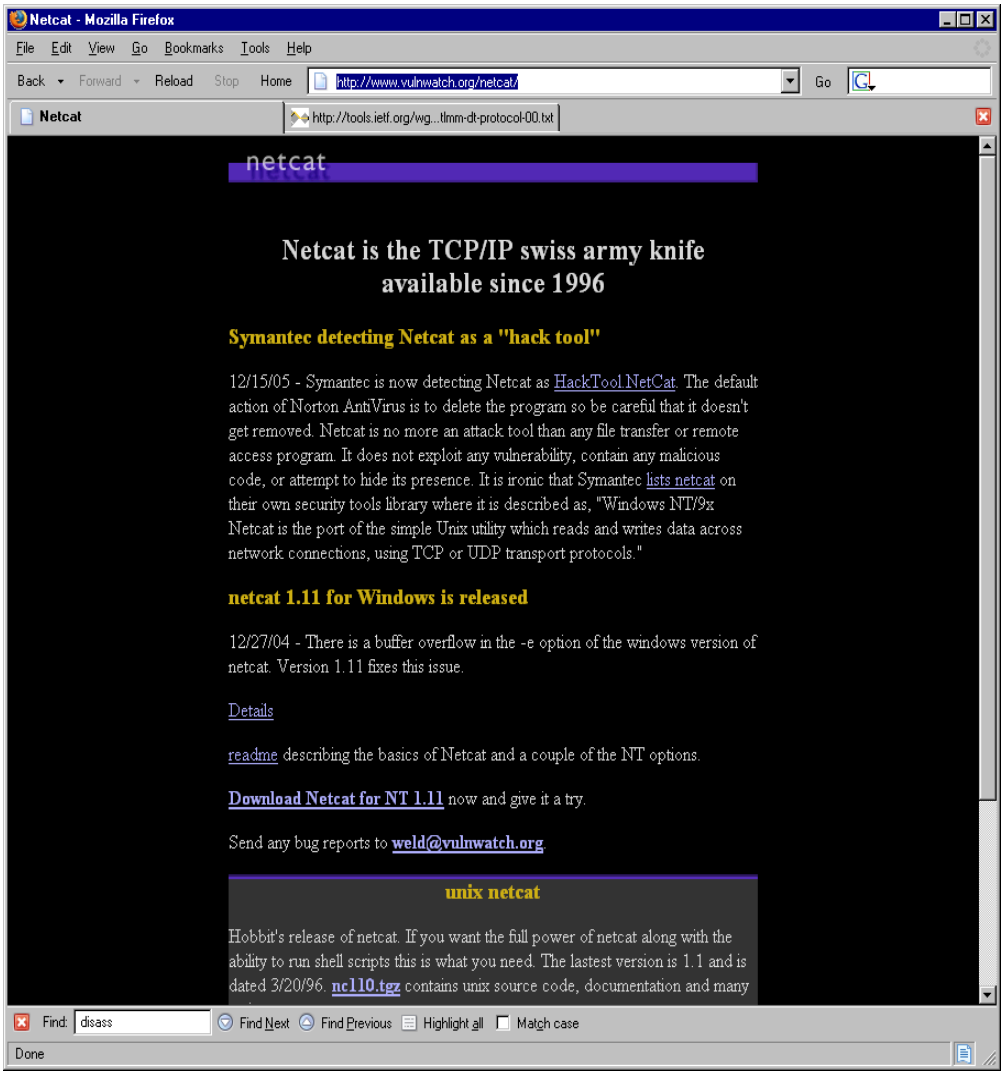

Рисунок 5 сетевые коты водятся и под NT

johnny cache приложил к своему посту ссылки на дампы памяти Windows XP: www.802.11mercenary.net/~johnycsh/prone_to_deletion/dd/crash2.zip (BSOD без подмены EIP или, выражаясь его терминологией: "unsuccessfully attempt to gain EIP") и www.802.11mercenary.net/~johnycsh/prone_to_deletion/dd/crash3.zip (BSOD с успешной подменой EIP).

готовой атакующей программы с shell-кодом на борту предоставлено не было, но ее сможет написать каждый желающий, однако, прежде, чем погружаться в анализ дампов (занимающих в своей совокупности свыше 200 Мбайт), мы должны рассмотреть механизм обработки прерываний в NT, иначе ни хрена не будет понятно.

прерываниями — называются события, поступающие от оборудования, генерируемые процессором или операционной системой в определенные моменты времени, например, когда DMA-контроллер завершает передачу данных или таймер говорит "щелк". обработка прерываний происходит в соответствии с их приоритетом: прерывания с более высоким приоритетом прерывают менее приоритетные прерывания, возвращая им управление после того, как они будут обработаны. аппаратные контроллеры прерываний обеспечивают до 256 уровней приоритетов, однако, NT не поддерживает их, предпочитая использовать свою собственную систему свою собственную систему приоритетов, известную под аббревиатурой **IRQL** (Interrupt Request Levels – Уровни Запроса Прерываний).

в NT существует всего существует 32 уровня (см. рис. 6), пронумерованных целыми числами от 0 до 31. Уровень 0 имеет минимальный приоритет, 31 — максимальный. Нормальное выполнение потока происходит на нулевом уровне, называемого пассивным (PASSIVE) и его может прерывать любое асинхронное событие, возникающее в системе. при этом операционная система повышает текущий IRQL до уровня возникшего прерывания и передает управление его ISR (Interrupt Service Routine – процедура обработки прерывания), предварительно сохранив состояние текущего обработчика.

приоритеты с номерами 1 и 2 отданы под программные прерывания (например, возникающие при ошибке обращения к странице памяти, вытесненной на диск), а все остальные — обслуживают аппаратные прерывания от периферийных устройств, причем, прерывания от таймера имеет приоритет 28.

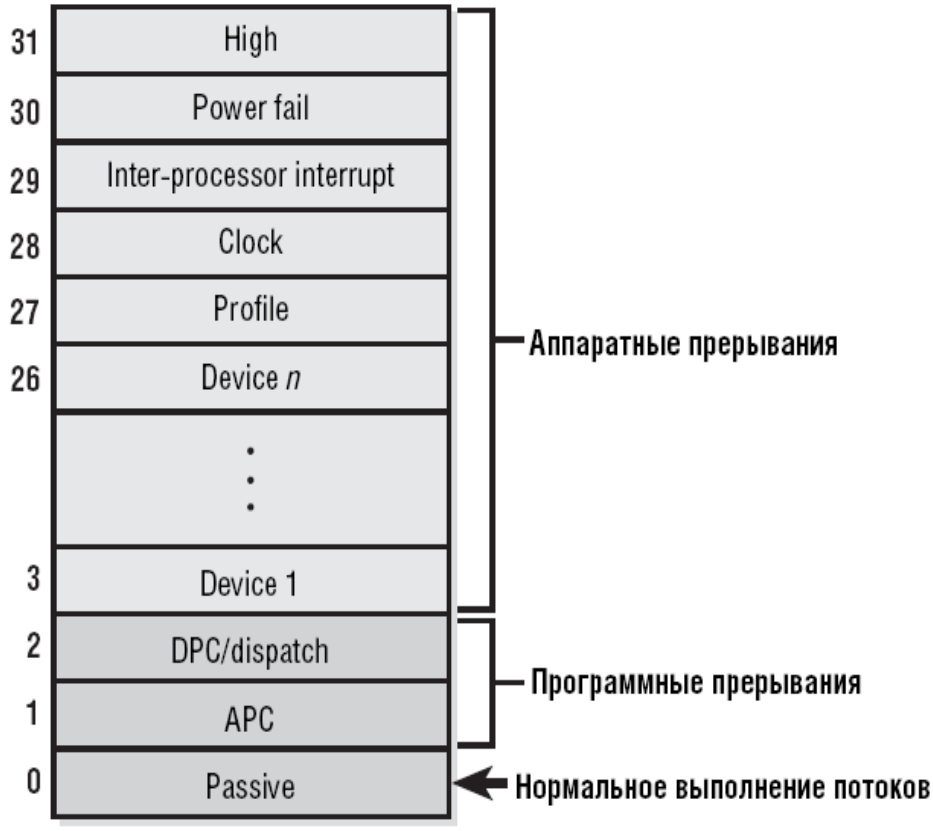

Рисунок 6 приоритеты прерываний в NT

покажем как происходит обработка прерываний, поступающих от устройств: допустим, поток A работает на уровне IRQL равном PASSIVE_LEVEL (см. рис. 7). Устройство Device 1 возбуждает аппаратное прерывание с уровнем DIRQL (Device IRQL т. е. IRQL с номером от 3 до 31 включительно). ось прерывает выполнение Потока A, повышает IRQL до DIRQL и передает управление на ISR устройства Device 1. обработчик прерывания обращается к устройству Device 1, делает с ним все, что оно требует, ставит в очередь отложенную процедуру `DpcForISR` для последующей обработки и понижает IRQL до прежнего уровня. отложенные процедуры (Deferred Procedure Calls или, сокращено, DPCs) выполняются на IRQL равном 2 (DISPATCH_LEVEL) и потому не могут начать свою работу вплоть до выхода из ISR.

прерывания, возникающие во время выполнения ISR маскируются. если прерывание возникнет во время выполнения `DpcForISR`, операционная система прервет ее работу, передаст управление ISR, который поставит в очередь еще одну отложенную процедуру, и вновь возвратится в `DpcForISR`. таким образом, сколько бы прерываний ни возникало, отложенные процедуры обрабатываются последовательно, в порядке очереди.

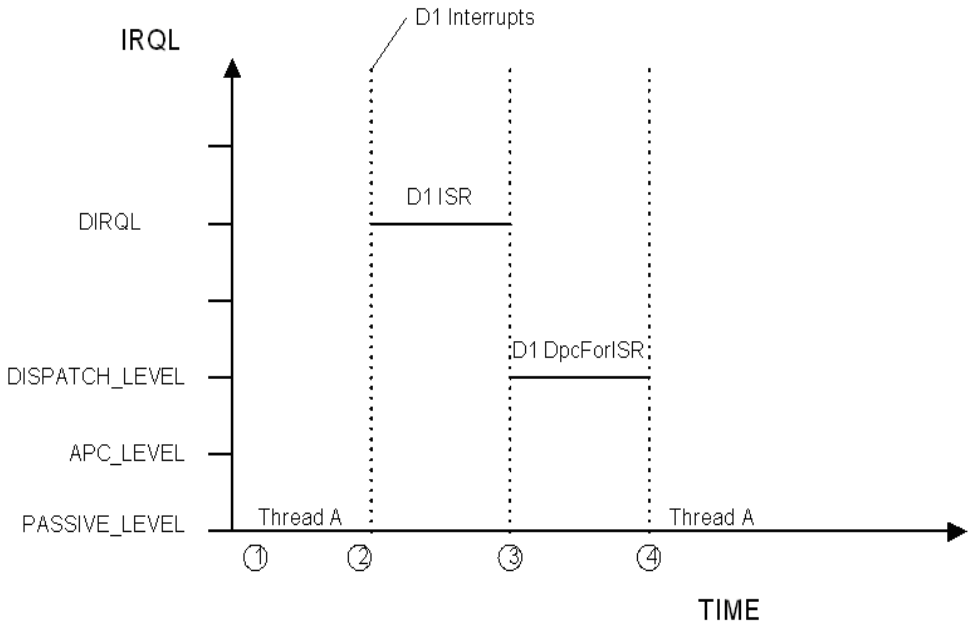

Рисунок 7 обработка прерываний на однопроцессорной машине

на двухпроцессорных машинах картина несколько усложняется (см. рис. 8). допустим, что во время обработки отложенной процедуры `DpcForIsr` выполняющейся на процессоре 0, устройство Device 1 вновь сгенерировало сигнал прерывания, посланный процессору 1 (процессор 0 еще не успел завершить обработку ISR и не понизил IRQL). ось повышает IRQL процессора 1 до DIRQL и передает управление на IRS устройства Device 1, ставя отложенную процедуру `DpcForIsr` в очередь.

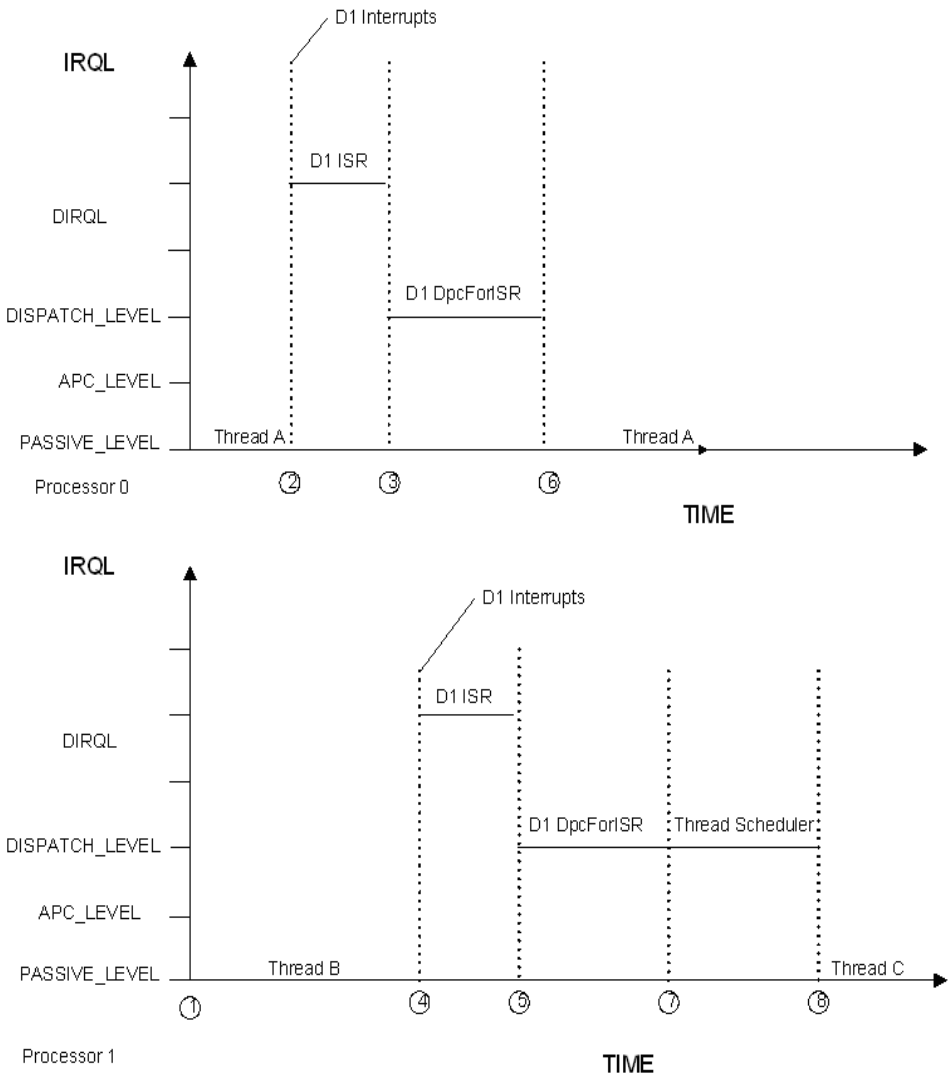

Рисунок 8 обработка прерываний на двухпроцессорных машинах

когда ISR на обоих процессорах завершаются, система понижает IRQL и начинается выполнение отложенной процедуры `DpcForIsr`, стоящей как в очереди процессора 0, так и в очереди процессора 1. Да-да!!! Процедура `DpcForIsr` исполнятся сразу на _обоих_ процессорах _одновременно_ отвечая за обработку двух прерываний от одного устройства! Как вам это нравится?! В такой ситуации очень легко превратить совместно используемые данные в мешанину, возвратив неожиданный результат или завесив систему (см. рис 9), вызвав BSOD.

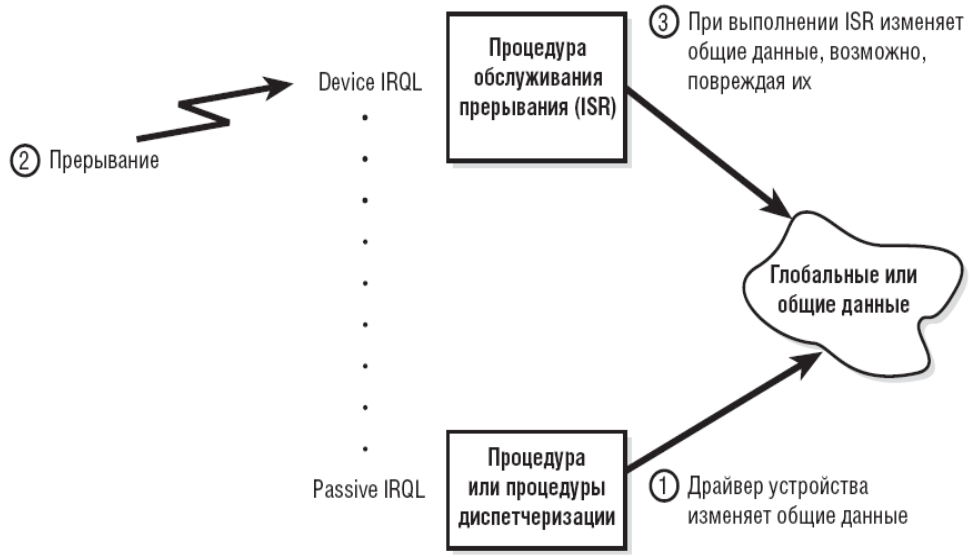

Рисунок 9 "срыв" синхронизации и его последствия

а теперь разберемся с дампами памяти, полученными johnny cache. распаковав архив crash2.zip, мы увидим три файла: crash2.txt — краткое описание содержимого, crash2.pcap — трафик, собранный sniffer'ом, и MEMORY.DMP — полный дамп памяти. вот он-то нам и нужен! для его анализа необходимо иметь либо DDK для XP (DDK для W2K не подходит), либо последнюю версию Microsoft Debugging Tools, которую можно бесплатно скачать с www.microsoft.com/whdc/devtools/debugging/default.mspx.

Microsoft предоставляет множество утилит для разбора аварийных дампов, но мы, как настоящие хакеры, будем пользоваться консольным отладчиком i386kd, который рулит, а все остальные графические поделки отдыхают. командная строка для запуска выглядит приблизительно так:

```
i386kd -z L:\dumps\crash2\memory.dmp -logo my_out
```

Листинг 1 командная строка для загрузки дампа в отладчик

здесь: "L:\dumps\crash2\memory2.dmp" — путь к дампу, "my_out" – имя файла, в который будет записываться лог (если, конечно, он нам нужен). еще можно указать ключ `"-y SRV*D:\sym*http://msdl.microsoft.com/download/symbols"`, чтобы отладчик динамически подгружал необходимую символьную информацию из сети, однако, в нашем случае она будет излишней и без нее вполне можно обойтись.

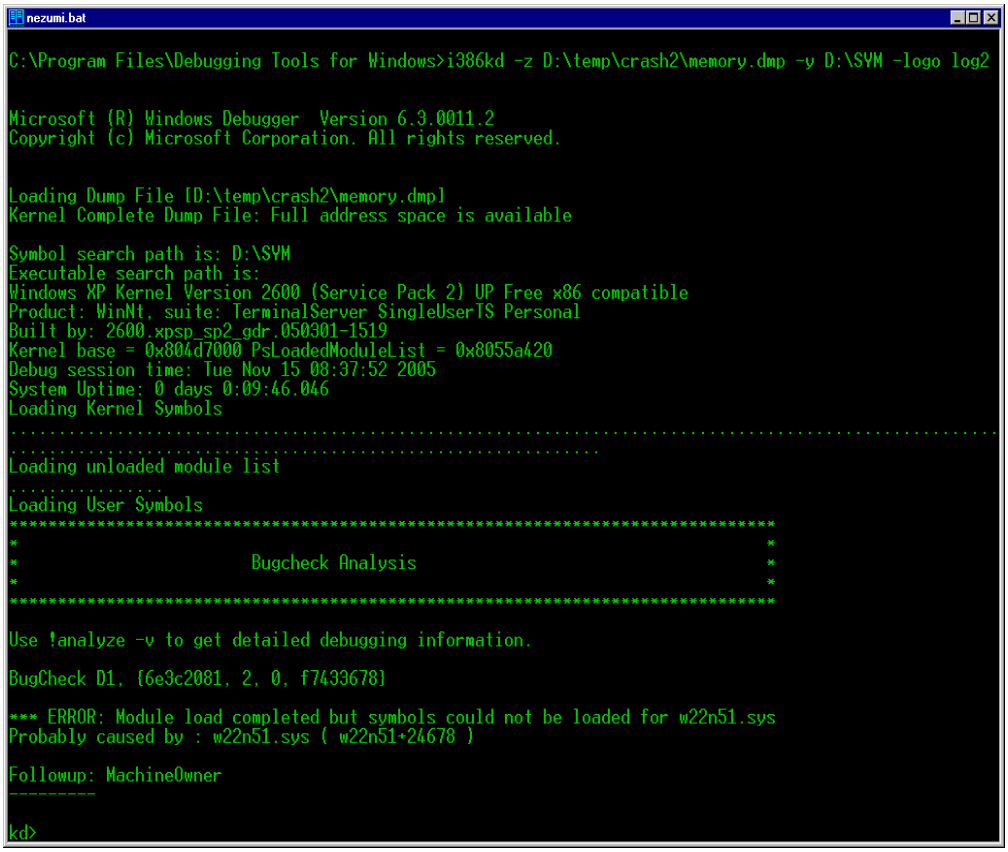

Рисунок 10 анализ дампа памяти в отладчике i386kd

поехали! отладчик заглатывает дамп сообщая, что он был отрыгнут операционной системой Windows XP Kernel Version 2600 (Service Pack 2) и что причиной аварии стал BugCheck-код D1h со следующими параметрами {6e3c2081, 2, 0, f7433678}, вероятным виновникам вызова которого явился драйвер w22n51.sys, управляющий (по нашим данным) беспроводной сетевой картой Intel Centrino. Далее следует совет "use !analyze -v to get detailed debugging information" (наберите "!analyze -v" для получения детальной отладочной информации) и приглашение к вводу, отмеченное лениво мерцающим курсором с "kd>".

набираем, как нас просят "!analyze -v", и получаем следующий отчет, приведенный ниже с несущественными сокращениями:

```
kd> !analyze -v
DRIVER_IRQL_NOT_LESS_OR_EQUAL (d1)
Arguments:
Arg1: 6e3c2081, memory referenced
Arg2: 00000002, IRQL
Arg3: 00000000, value 0 = read operation, 1 = write operation
Arg4: f7433678, address which referenced memory

Debugging Details:
------------------
READ_ADDRESS: 6e3c2081
CURRENT_IRQL: 2

FAULTING_IP:
w22n51+24678
f7433678 8a11  mov     dl,[ecx]        ds:0023:6e3c2081=??

STACK_TEXT:
8054e9e0 65537365 2f3c746e 656d616e 200a0d3e w22n51+0x24678
7479426c 00000000 00000000 00000000 00000000 0x65537365

IMAGE_NAME:  w22n51.sys
```

Листинг 2 анализ дампа памяти, вызвавшего BSOD, но не оказавшего воздействия на регистр EIP

вместо бессловесного hex-значения BugCheck кода мы получили его наименование — DRIVER_IRQL_NOT_LESS_OR_EQUAL означающее, что драйвер осуществил попытку выполнить операцию, недозволенную на данном уровне IRQL, в данном случае равном 2, что соответствует DISPATCH_LEVEL, на котором выполняются отложенные процедуры, о проблемах синхронизации которых мы уже говорили. конкретно, драйвер пытается прочесть ячейку 6E3C2081h, находящуюся в странице памяти, вытесненной на диск (на уровне DISPATCH_LEVEL подкачка не работает). это очевидная ошибка драйвера, указывающая на серьезное разрушение структур, обрабатываемых им данных (в данном случае — фреймов пакетов). тем не менее, EIP находится в пределах драйвера w22n51.sys и в стеке нет никаких следов присутствия CCh-байт, которыми "заряжены" атакующие пакеты. так что все, что мы имеем — это тривиальный отказ в обслуживании.

идем дальше и загружаем в отладчик memory3.dmp, который, по утверждению johnny cache, оказал убийственное воздействие на регистр EIP. что ж, посмотрим-посмотрим:

```
i386kd -z L:\dumps\crash2\memory2.dmp -logo out3
```

Листинг 3 загрузка дампа памяти в отладчик

после команды "`!analyze -v`" отладчик выдает следующий результат, приводимый здесь с традиционными сокращениями:

```
kd> !analyze -v
DRIVER_IRQL_NOT_LESS_OR_EQUAL (d1)
Arguments:
Arg1: 5c01abf7, memory referenced
Arg2: 00000002, IRQL
Arg3: 00000001, value 0 = read operation, 1 = write operation
Arg4: cccccccf, address which referenced memory

Debugging Details:
------------------
WRITE_ADDRESS: 5c01abf7
CURRENT_IRQL:  2
FAULTING_IP:
+ffffffffcccccccf
cccccccf 01963b10ffd6  add [esi+0xd6ff103b],edx

LAST_CONTROL_TRANSFER:  from ff103b96 to cccccccf

STACK_TEXT:
cccccccc ff103b96 01c486d6 00000000 00000000 0xcccccccf
01c486d6 00000000 00000000 00000000 00000000 0xff103b96
```

Листинг 4 анализ дампа памяти, оказавшего воздействие на регистр EIP

на первый взгляд ничего не изменилось — все тот же противный DRIVER_IRQL_NOT_LESS_OR_EQUAL, возникающий на уровне DISPATCH_LEVEL, но… присмотритесь к значению регистра EIP, вылетевшему далеко за пределы драйвера, и указывающего на инструкцию "`add [esi+0xd6ff103b], edx`" случайно очутившуюся по адресу CCCCCCCFh, который очень сильно смахивает на начинку атакующего пакета. но почему же тогда EIP равен не CCCCCCCCh, а CCCCCCCFh, которого в пакете не было?!

все просто! страница с адресом CCCCCCCCh случайно оказалась в оперативной памяти и не была вытеснена на диск, поэтому, как только пакет-камикадзе передвинул регистр EIP на адрес CCCCCCCCh, процессор начал выполнение кода, каждый раз увеличивая EIP на размер успешно выполненной команды и споткнулся лишь тогда, когда встретил инструкцию "`add [esi+0xd6ff103b], edx`", обратившуюся к ячейке 5C01ABF7h, которой не было в оперативной памяти, а подкачка на уровне DISPATCH_LEVEL, как уже говорилось, ни хрена не работает, вот операционная система и сказала "мяу".

чтобы подкрепить наше предположение фактами, травой и грибами, находясь в отладчике, дадим команду "u CCCCCCCC", чтобы дизассемблировать код по данному адресу:

```
kb>u cccccccc
cccccccc d6                  ??? ; setalc Set AL to Carry Flag
cccccccd 86c4                xchg   ah,al
cccccccf 01963b10ffd6        add    [esi+0xd6ff103b],edx
ccccccd5 86c4                xchg   ah,al
ccccccd7 0100                add    [eax],eax
ccccccd9 0000                add    [eax],al
ccccccdb 0000                add    [eax],al
ccccccdd 0000                add    [eax],al
```

Листинг 5 дизассемблерный листинг окрестностей аварии

первой идет неизвестная отладчику i386kd недокументированная команда SETALC (Set Al to Carry Flag), а за ней — XCHG AH,AL. обе эти команды выполняются успешно, но как только процессор доходит до команды "ADD [ESI+0XD6FF103B],EDX" расположенной по адресу CCCCCCCFh и обращающейся к недоступной ячейке 5C01ABF7h, возникает исключение и операционная система показывает голубой экран.

таким образом, воздействие на регистр EIP, путем направленного шквала пакетов, все-таки возможно! поскольку, отправляемые UDP пакеты находятся в стеке, в них легко внедрить shell-код, главное — определить какое именно двойное слово из начинки пакета попадает в EIP, что легче всего выяснить экспериментально, путем замены части CCh на FFh, например, или тщательного изучения дизассемблерных текстов драйвера. отладчик здесь — плохой помощник, поскольку он изменяет временные промежутки (тайминги), непредсказуемым образом воздействуя на race condition. кстати, на двухпроцессорных машинах (равно как машинах, оснащенных двуядерными процессорами) ошибки синхронизации проявляются намного чаще, что открывает невиданные ранее просторы для удаленных атак.