

Ускорение

загрузки

Windows



Вадим Стеркин

Содержание

(щелкните для перехода к нужной странице)

Об авторе	4
О книге	4
Как измерить скорость загрузки Windows 7 и Vista	5
Длительность загрузки в журнале Diagnostics-Performance	5
Фильтр по событию	5
Подробный протокол загрузки.....	6
Уровни события 100	7
События диагностики	7
Создание отчета о скорости загрузки Windows 7 и устранение системных проблем	8
О создании пакета и роли сертификатов безопасности.....	8
Запуск диагностики	9
Интерпретация сведений о системе и ее загрузке	10
Три простых совета по ускорению загрузки Windows 7	15
Наведите порядок в автозагрузке	15
Проверьте SuperFetch и ReadyBoot	16
Дефрагментируйте диск – сейчас и по расписанию	17
Бонус: используйте режимы сна и гибернации!	18
Ускорение загрузки Windows 7 и Vista с помощью планировщика заданий	19
Программы в автозагрузке	19
Важное примечание о приоритете CPU и I/O.....	21
Создание простой задачи в планировщике заданий	21
Проверка работоспособности заданий планировщика	24
Экспорт и импорт заданий.....	24
Ускорение загрузки одним твиком реестра	25
В каких случаях изменение в реестре ускорит загрузку системы.....	25
Что входит в понятие «автозагрузка»	25
Влияние программ в автозагрузке на скорость запуска Windows.....	26
Microsoft против разработчиков программ	26
Твик, изменяющий приоритет CPU и I/O для программ в автозагрузке	27
Как проверить приоритет запущенных программ	28
Тест	29
Этапы загрузки Windows под микроскопом Windows Performance Toolkit	31
Загрузка и установка WPT	31
Подготовка к работе.....	31
Сбор данных	32

Файлы, используемые для анализа	33
Этапы загрузки Windows и их диагностика	34
Оптимизация ReadyBoot с помощью Windows Performance Toolkit	41
Эффективность простых советов на графике активности ReadyBoot	41
Ловкость рук и никакого мошенничества.....	42
Запуск оптимизации.....	42
Результаты оптимизации.....	44
Сторонний дефрагментатор – друг или враг?	44
Список литературы	45

Об авторе



Вадим Стеркин занимается развитием компьютерного портала OSZone.net, объединяющего сайт, форум и каталог программного обеспечения. Он отвечает на вопросы участников [форума OSZone](#) с 2002 года, а на сайте ведет раздел [автоматической установки Windows](#).

С 2006 года Вадим шесть раз подряд становился обладателем награды [Наиболее ценный специалист Microsoft](#) (MVP), присуждаемой за вклад в развитие технических сообществ. Вадим также пишет [статьи](#) о клиентских операционных системах Microsoft, которые помимо OSZone.net опубликованы на Microsoft Technet, в базе знаний Microsoft, журнале Windows IT Pro/RE.

В своем блоге outsidethebox.ms он простым и понятным языком делится опытом и секретами эффективной работы в Windows. Интересные записи выходят регулярно, и у блога уже сложилась постоянная аудитория читателей, отличающихся интеллектуальными комментариями.

О книге



Это второе издание книги, в которое вошли все мои материалы о диагностике и ускорении загрузки Windows, опубликованные в [блоге](#). В качестве бонуса сюда также включен рассказ об оптимизации ReadyBoot, который я нигде не публиковал. Эта книга – мой подарок подписчикам блога. Они получают обновленные издания первыми, и первыми же узнают о выходе моих новых статей и прочих вкусностях, которыми я потчую по адресу www.outsidethebox.ms.

Первое издание книги только из моего блога скачали 4,5 тысячи человек, а сколько загрузок было с других сайтов я понятия не имею :) Во втором издании я добавил главу об ускорении загрузки [одним твиком реестра](#), а также [примечание о приоритете программ](#), запускаемых из планировщика заданий.

Как измерить скорость загрузки Windows 7 и Vista

У вас Windows быстро загружается? А как вы это определяете, на глазок или секундомером? В Windows XP меряли «вагончиками», но в Windows 7 и Windows Vista их убрали. Из этой главы вы узнаете очень простой способ, позволяющий точно определить длительность загрузки вашей системы без дополнительных средств. Она открывает цикл материалов об измерении, диагностике и оптимизации скорости загрузки Windows 7, которые будут опубликованы в ближайшее время.



Новые операционные системы Microsoft собирают огромную массу информации о работе системы, сохраняя ее в журналах событий, которых тоже великое множество. На основе лишь одного события можно узнать интересные подробности о скорости загрузки.

Длительность загрузки в журнале Diagnostics-Performance

Windows 7 и Windows Vista отслеживают каждую загрузку системы и записывают отчет. Чтобы увидеть его, откройте **Пуск – Поиск - Журнал событий** и перейдите в раздел **Журналы приложений и служб – Microsoft – Windows – Diagnostics-Performance**. Там вы найдете один журнал, и он работает. Отчет о загрузке системы легко найти по коду события **100**.

Уровень	Дата и время	Источник	Код события	Кат
Ошибка	23.01.2011 2:29:06	Diagnostics-Performance	100	Кор
Критический	15.01.2011 23:45:34	Diagnostics-Performance	100	Кор
Критический	15.01.2011 23:29:44	Diagnostics-Performance	100	Кор
Критический	13.01.2011 3:04:49	Diagnostics-Performance	100	Кор
Критический	28.12.2010 1:51:05	Diagnostics-Performance	100	Кор
Критический	27.12.2010 0:33:54	Diagnostics-Performance	100	Кор
Критический	19.12.2010 23:43:05	Diagnostics-Performance	100	Кор

Событие 100, Diagnostics-Performance

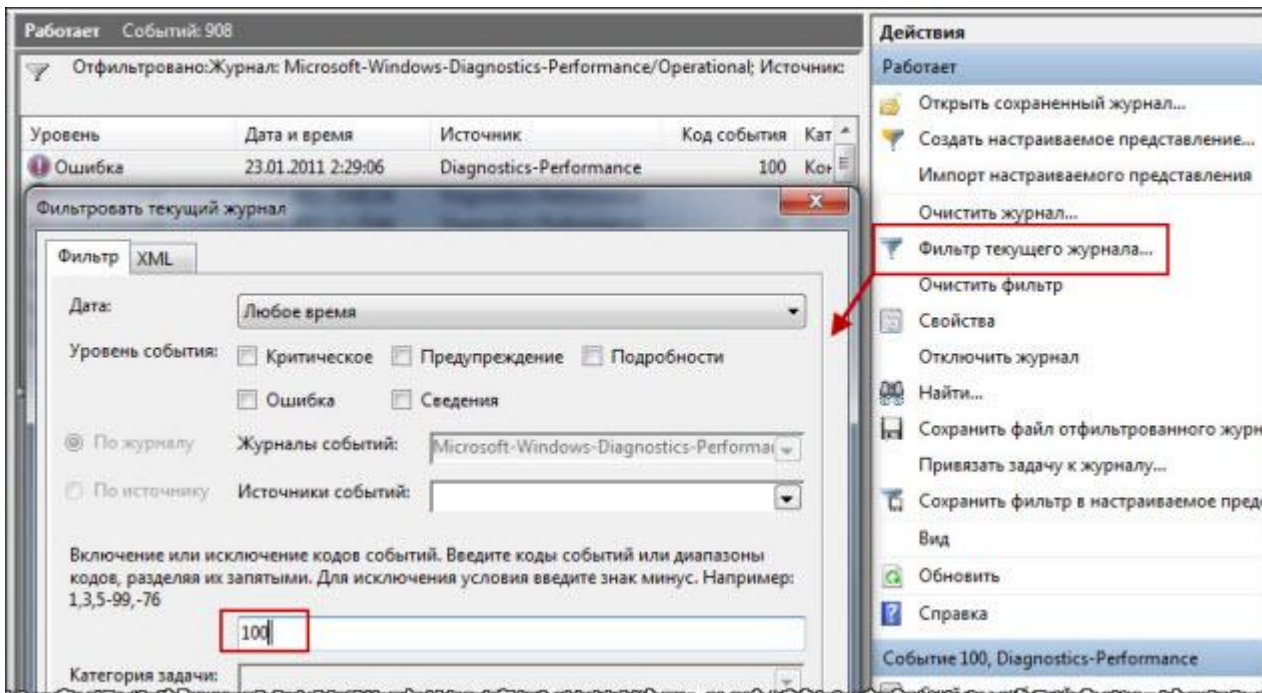
Общие | Подробности

Windows запущена:
Длительность загрузки : 97634ms
IsDegradation : false
Время события (UTC) : 2011-01-22T23:26:19.656000300Z

На рисунке желтым цветом выделено время загрузки в миллисекундах, поэтому **97634ms** означает **97 секунд**, т.е. около полутора минут. Это время определяется от самого начала загрузки Windows (сразу после завершения загрузки BIOS) и вплоть до полной загрузки рабочего стола, т.е. до прекращения активности процессов, участвующих в загрузке. **Из этого времени нужно вычесть 10 секунд**, чтобы получить актуальное время загрузки.

Фильтр по событию

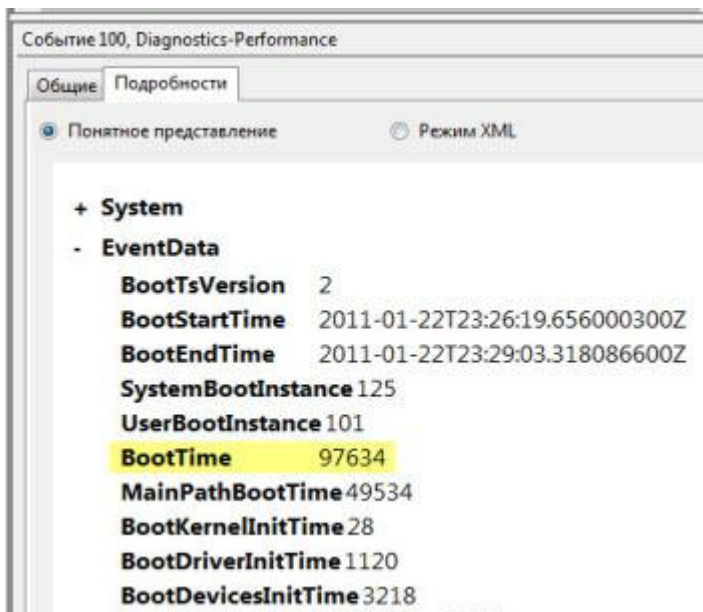
Некоторые загрузки занимают больше времени, некоторые меньше. Чтобы увидеть все события загрузки, отфильтруйте текущий журнал по коду события **100** ([подробнее о фильтрации](#)).



Увидев уровень сообщения **Ошибка** или **Критический**, не нужно впадать в панику, т.к. это вовсе не означает проблем с загрузкой системы, хотя и свидетельствует о том, что загрузку можно ускорить. Операционная система довольно придирчиво относится к времени загрузки, и чуть ниже вы узнаете, как она назначает уровень события.

Подробный протокол загрузки

На вкладке **Подробности** вы можете посмотреть остальную информацию о загрузке системы, как в текстовом виде, так и в формате XML.



Назначение некоторых параметров можно разгадать без труда, а иные вовсе не очевидны. Наиболее интересными являются:

- **BootTime** – общее время загрузки
- **BootUserProfileProcessingTime** – время загрузки профиля
- **BootPostBootTime** – время с момента появления рабочего стола до полного окончания загрузки
- **MainPathBootTime** – длительность основных системных этапов загрузки (**BootTime** минус **BootPostBootTime**)
- **BootNumStartupApps** – количество программ в автозагрузке

Изрядная часть остальных параметров отражает длительность различных этапов загрузки, соответствуя их названиям. Но для диагностики загрузки этого маловато, потому что нет информации о том, что конкретно происходит на каждом этапе.

Подробный рассказ об этапах загрузки Windows и их диагностике на примере отчетов утилиты **xbootmgr**, входящей в набор Windows Performance Analysis Tools, вас ожидает в следующих главах.

Уровни события 100

Давайте вернемся к вопросу об уровнях события **100** и посмотрим, от чего зависит критичность времени загрузки.

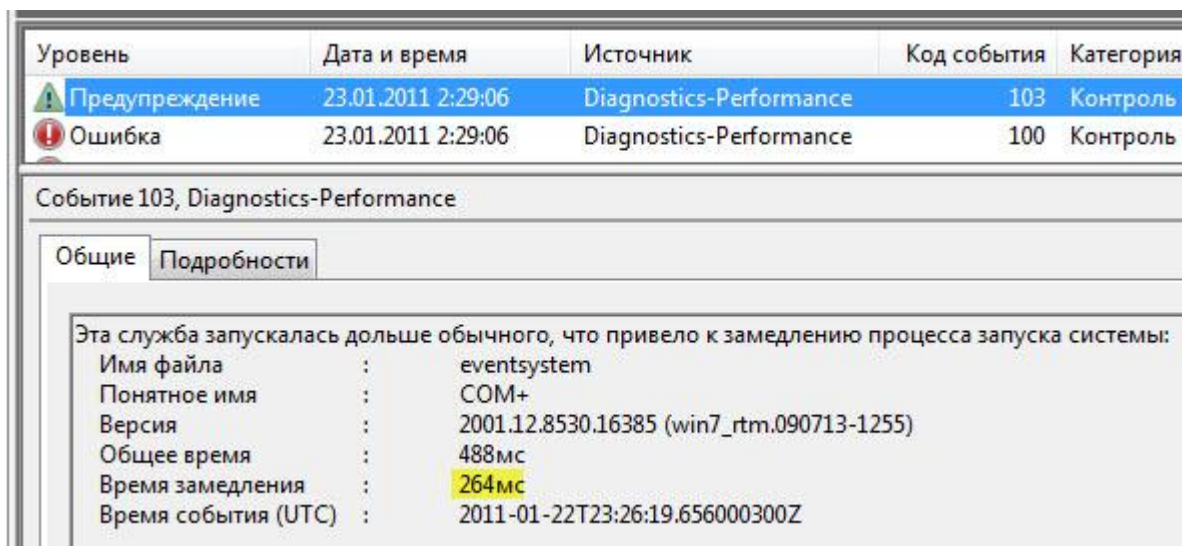
Уровень события 100	Условие (время в секундах)
Предупреждение	MainPathBootTime < 60 И BootPostBootTime < 30
Ошибка	60 < MainPathBootTime < 120 И 30 < BootPostBootTime < 60
Критический	MainPathBootTime > 120 ИЛИ BootPostBootTime > 60

Поскольку **BootTime = BootTime + BootPostBootTime**, можно сделать такие выводы.

- **Предупреждение** будет только в том случае, если быстро загружается как система (драйверы и службы), так и пользовательская среда (рабочий стол и программы в автозагрузке)
- Как минимум **Ошибка** будет появляться, если общая длительность загрузки (**BootTime**) больше 90 секунд
- **Критический** уровень вам гарантирован при общей длительности загрузки более двух минут

События диагностики

Нередко система сама предлагает диагностическую информацию о загрузке в соседних событиях с тем же временем и кодами **101 - 109**. Например, задержки по вине служб имеют код **103**. Но далеко не всегда эта информация полезна для диагностики загрузки.



Здесь нужно обращать внимание на время замедления, потому что замедление свыше 0,1 секунды уже дает основание к записи события. На рисунке видно, что служба COM замедлилась на 0,26 секунды, что вряд ли стоит рассматривать всерьез. Подробнее об этом способе диагностики рассказывается [в отдельной статье моего коллеги](#).

Выгоду можно извлечь даже из довольно скудной информации о длительности загрузки, если сопоставить ее с другими параметрами системы. Чтобы вам было интереснее, я подготовил диагностический пакет, основанный на PowerShell, который автоматизирует процесс сбора сведений и представляет их в наглядной форме. С помощью пакета можно также устранить основные системные проблемы, которые негативно влияют на скорость загрузки. На его примере мы дальше разберем факторы, влияющие на длительность загрузки системы, и сможем сделать выводы о том, как ее ускорить.

Создание отчета о скорости загрузки Windows 7 и устранение системных проблем

В состав Windows 7, входят [средства диагностики и устранения неполадок](#), основанные на PowerShell. Я предлагаю вам собственный пакет, который создает отчет о скорости загрузки Windows 7 и исправляет основные системные проблемы, которые могут негативно влиять на нее.

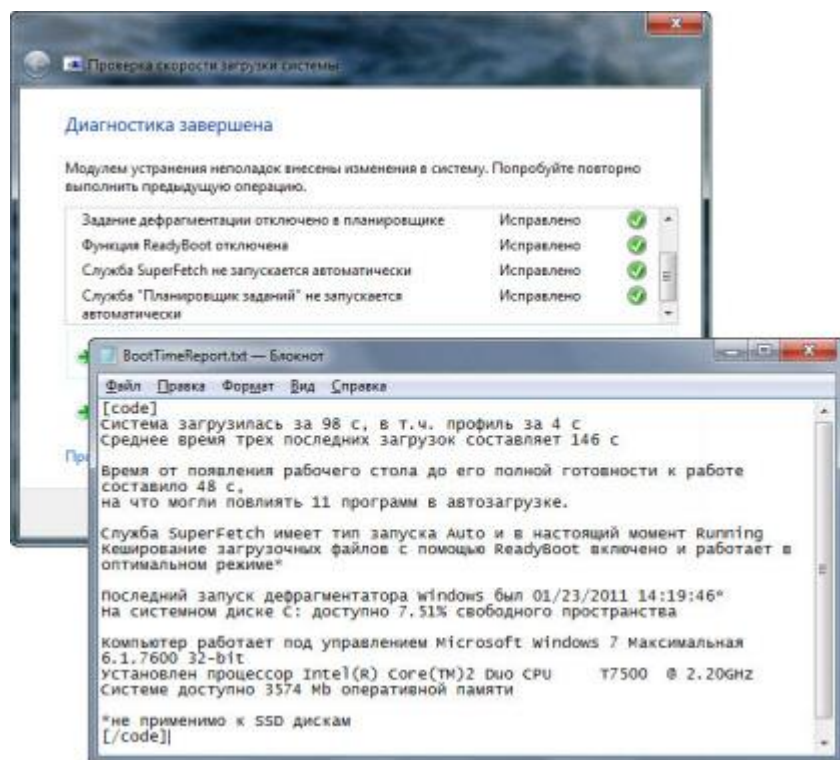
С помощью PowerShell можно получать разнообразные данные о системе, и в том числе очень легко обрабатывать сведения из журналов событий. Именно эти особенности я использовал для своего пакета, чтобы представить в удобном виде информацию о загрузке системы.



Входящий в пакет скрипт собирает информацию, а затем выводит в текстовом файле:

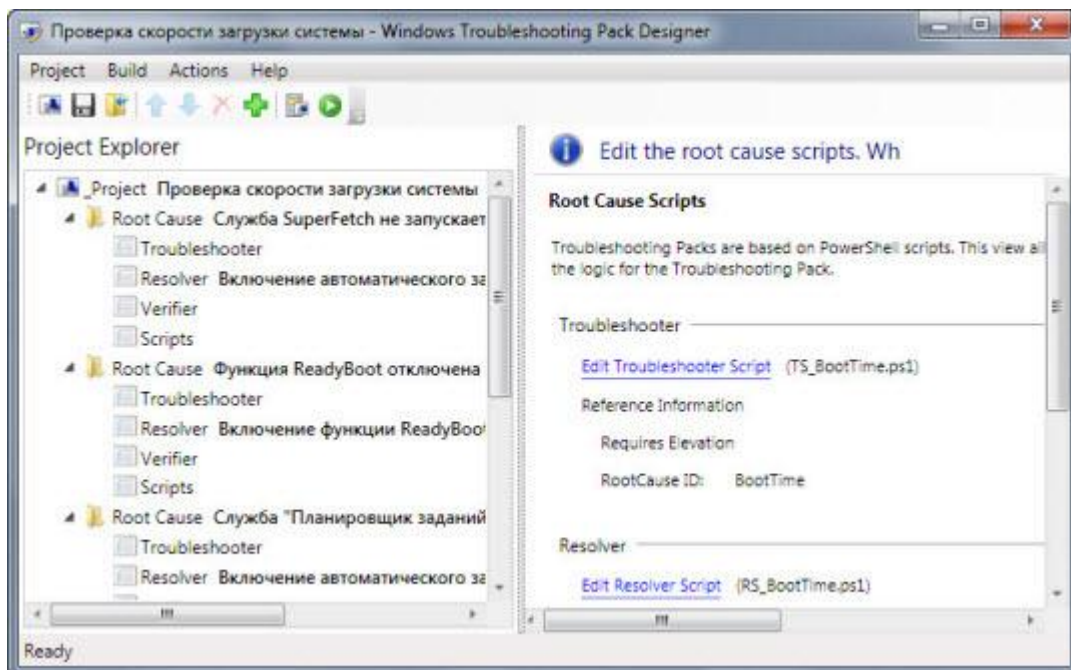
- основные сведения о последней загрузке
- состояние службы SuperFetch и ее функции ReadyBoot
- данные о работе дефрагментатора диска
- процент свободного места на системном разделе
- сведения об операционной системе, процессоре и памяти

Второе назначение пакета – это устранение ряда системных проблем, замедляющих загрузку. Не стоит ожидать от него магического ускорения, поскольку фактически речь идет о возврате к исходному состоянию настроек, которые пострадали из-за непонимания их роли в длительности загрузки системы.



О создании пакета и роли сертификатов безопасности

Я познакомился с PowerShell совсем недавно, но все оказалось проще, чем я думал. Не без помощи активного «гугления» я относительно быстро создал базовый скрипт по сбору информации (эквивалентный «батник» я бы в жизни не написал). Полезные советы по его оптимизации и тонкостям PowerShell мне дал Василий Гусев, Microsoft MVP и автор блога [PowerShell и другие скрипты](#). Он же заразил меня идеей создать диагностический пакет, хотя больше года назад я уже видел [его доклад на Платформе](#) на эту тему.



Создание пакета не таит в себе особых сложностей – это проще, чем создать файл ответов для автоустановки, если уже есть все скрипты.

Барьером оказалось то, что пакет должен быть обязательно подписан, а в системе должен быть установлен сертификат, удостоверяющий эту подпись. Иначе ничего не запустится! PowerShell – мощное средство для управления Windows, поэтому Microsoft предпринимает меры предосторожности.

Из бесплатных способов самым быстрым является создание самоподписанного сертификата. Вадимс Поданс, кстати тоже MVP PowerShell, прямо заявляет, что они являются злом (и делает это в [статье](#), описывающей создание такого сертификата средствами PowerShell :).

В первой версии пакета использовался самоподписанный сертификат. Текущая версия подписана моим личным сертификатом, который выдала компания [Digicert](#), удостоверив мою личность.

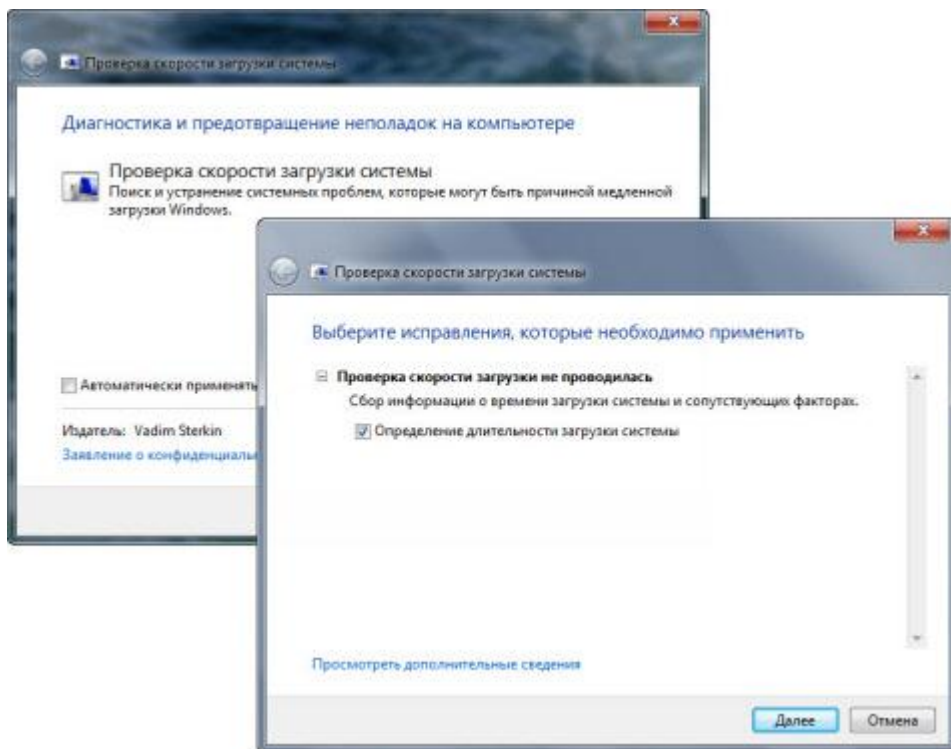
Запуск диагностики

Вам понадобятся права администратора.

1. Загрузите архив с [этой страницы](#) и распакуйте в любую папку.
2. Запустите файл **CheckBootSpeed.diagcab**.

Пакет может устранять проблемы в автоматическом режиме, но если вы хотите подойти к вопросу избирательно, щелкните **Дополнительно** в первом окне и снимите флажок **Автоматически применять исправления**.

Если у вас в порядке все системные настройки, связанные с загрузкой, вы увидите только предложение создать отчет о загрузке.



Через несколько секунд все будет готово. Вы увидите сводку диагностики, а отчет о загрузке и системе откроется в блокноте. Если диагностика исправила проблемы со службами, перезагрузите систему для вступления изменений в силу.

Текстовый файл с отчетом создается во временной папке, чтобы не засорять систему. Вы можете сохранить отчет где угодно, а также вставить на форуме или в блоге, для чего в тексте предусмотрен `тег code`.

Интерпретация сведений о системе и ее загрузке

Текстовый отчет состоит из пяти маленьких блоков, и подобраны они так, чтобы составить представление о загрузке конкретной системы.

Общее время загрузки

Первые две строки отчета содержат информацию о времени [полной загрузки системы](#), а также [профиля в память](#).

```
Система загрузилась за 98 с, в т.ч. профиль за 4 с
Среднее время трех последних загрузок составляет 146 с
```

Все значения округлены до целого числа секунд. Обратите внимание, как просто подсчитать среднее время последних трех загрузок. С журналом событий пришлось бы брать в руки калькулятор.

Загрузка рабочего стола

Далее идут сведения о финальном этапе загрузки, который проще всего ускорить. Длительность [полной загрузки рабочего стола](#) и [количество программ в автозагрузке](#).

```
Время от появления рабочего стола до его полной готовности к работе составило 48 с, на что могли повлиять 11 программ в автозагрузке.
```

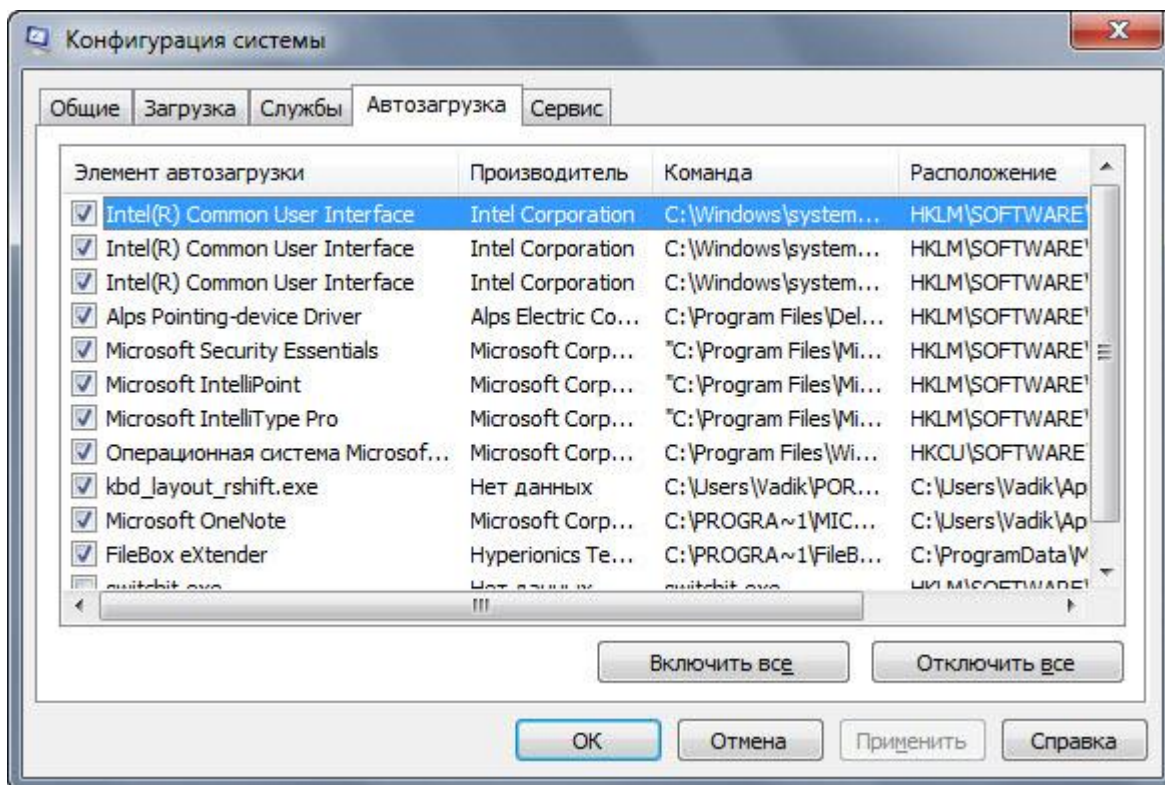
Полная готовность рабочего стола с моей точки зрения как пользователя означает окончание загрузки всех программ и моментальную реакцию на мои действия, без всяких «бубликов». Система смотрит на это аналогично, а об алгоритме мы поговорим в одной из следующих глав.



Строго говоря, на момент начала загрузки рабочего стола еще может продолжаться обработка процессов, запущенных на более раннем этапе.

Программы, стартующие при загрузке системы, играют в длительности этого этапа большую роль. Заметьте, что в моем примере он занял почти половину от общего времени загрузки системы!

Количество программ в автозагрузке соответствует тому, что вы можете увидеть на вкладке **Автозагрузка** утилиты **msconfig**.



ReadyBoot, SuperFetch и Prefetch

Функция **ReadyBoot** работает в рамках службы **SuperFetch**. После каждой загрузки она, дождавшись бездействия системы, анализирует предыдущий запуск и определяет, какие файлы использовались во время последней загрузки. В следующий раз ReadyBoot ускоряет загрузку Windows за счет кеширования файлов загрузки системы и программ автозагрузки в оперативной памяти, что быстрее, чем считывать файлы с жесткого диска.

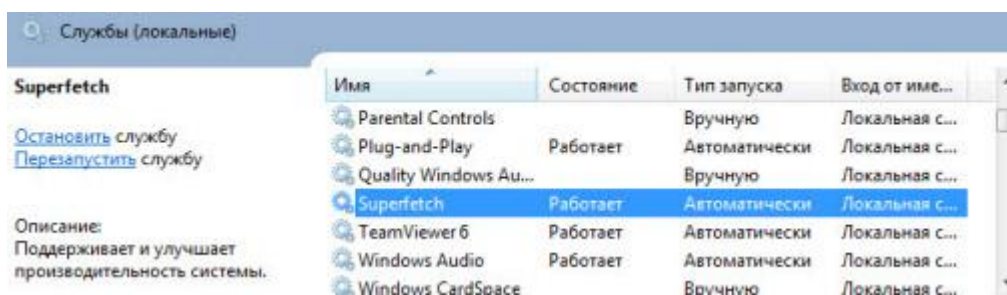


Функция **Prefetch**, появившаяся в Windows XP, служит для упреждающей загрузки данных в память. Служба **SuperFetch** в Windows 7 выступает в качестве интеллектуальной надстройки над Prefetch, анализирующей ваши сценарии использования системы.

От службы SuperFetch помимо ReadyBoot также зависят функции [ReadyBoost](#) и ReadyDrive, предназначенная для не слишком распространенных [гибридных дисков](#).

Служба SuperFetch имеет тип запуска Auto и в настоящий момент Running
Кеширование загрузочных файлов с помощью ReadyBoot включено и работает в оптимальном режиме*

Первая строка выводит статус службы SuperFetch. Очевидно, что если служба SuperFetch отключена, ReadyBoot не функционирует.

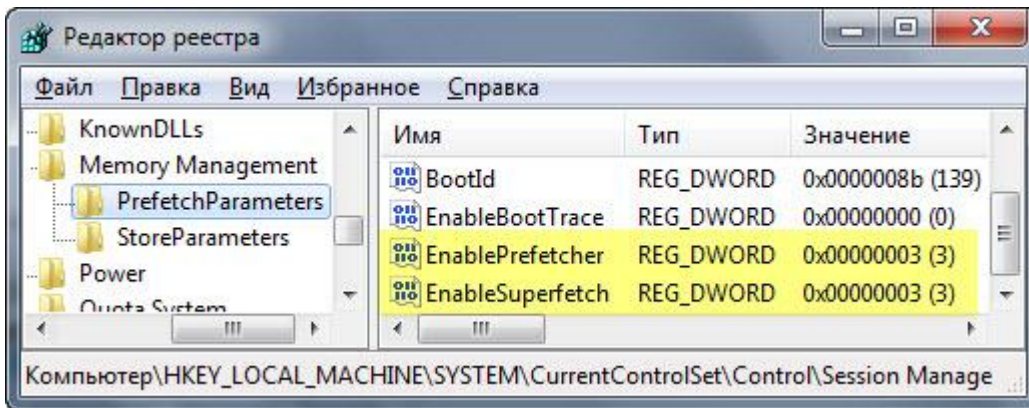


Вторая строка интерпретирует значение параметров **EnablePrefetcher** и **EnableSuperfetch** в разделе реестра

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Session Manager\Memory Management\PrefetchParameters

Функция ReadyBoot на жестких дисках считается включенной, если значения каждого параметра **2** или **3** в любых сочетаниях (на рисунке ниже значения по умолчанию):

- **2** – обрабатываются файлы загрузки (системные и программ в автозагрузке)
- **3** - наряду с ними кэшируются также программы, запускаемые во время работы системы



Упоминание об оптимальном режиме здесь тоже не случайно, и оно будет не у всех.

Если на компьютере установлено менее 1.7 Гб оперативной памяти (RAM), ReadyBoot сжимает кэшируемые файлы. В этом случае на их распаковку требуется какое-то время, пусть и совсем небольшое.

Звездочка (*) ссылается на важное примечание о том, что информация о ReadyBoot не является актуальной при загрузке с твердотельного накопителя (SSD). Когда Windows 7 определяет наличие SSD диска, служба SuperFetch отключает функцию ReadyBoot, поскольку в этом случае выигрыша в скорости загрузки нет.



Вы спросите, зачем так сложно, нельзя ли четко определить, работает ReadyBoot или нет? Windows 7 определяет наличие SSD на основе спецификаций ACS-2. Система опрашивает диск и интерпретирует полученный результат в соответствии с [таблицей](#), опубликованной на MSDN.

Накопитель считается твердотельным, если диск определяется как «не вращающийся» (non-rotational).

Microsoft включила утилиту для определения типа диска в состав набора Windows Logo Kit. Однако скриптом извлечь эту информацию невозможно. Впрочем, если вы знаете, как определить SSD с помощью PowerShell, пожалуйста, расскажите мне, и я обновлю пакет.

Работа дефрагментатора и свободное пространство на системном диске

Четвертый блок отчета выводит сведения [о работе встроенного дефрагментатора Windows](#), в т.ч. дату его последнего запуска, а также процент свободного пространства на системном диске.

```
Служба 'Дефрагментация диска' имеет тип запуска Manual
Дефрагментатор выполняет оптимизацию загрузочных файлов
Служба 'Планировщик заданий' имеет тип запуска Auto и в настоящий момент Running
Запланированная задача дефрагментации включена
Последний запуск дефрагментатора Windows был 01/23/2011 14:19:46*
```

Здесь звездочка также указывает на то, что для SSD дисков это не имеет значения. Windows 7 не выполняет дефрагментацию SSD дисков, поскольку они не фрагментируются. Поэтому, если система загружается с такого диска, сведения о работе дефрагментатора бесполезны в контексте ускорения загрузки. Однако фрагментированность обычного жесткого диска может негативно влиять на скорость загрузки системы.



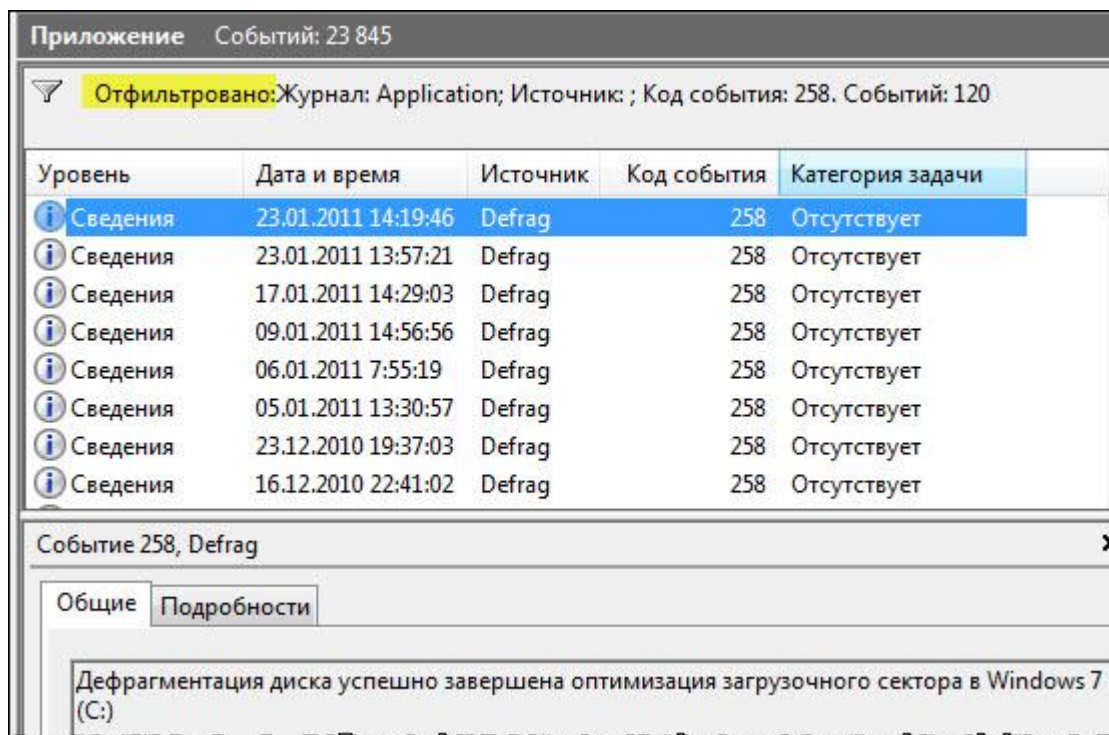
Для нормальной работы дефрагментатора требуется, чтобы служба «Дефрагментация диска» не была отключена (по умолчанию она имеет тип запуска «Вручную»). Оптимизация загрузки выполняется отдельной задачей по расписанию, однако этому может препятствовать параметр **Enable** со значением **N** в разделе реестра

`HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\Dfrg\BootOptimizeFunction`

По умолчанию он отсутствует, но может появляться в результате установки сторонних дефрагментаторов или неумелой настройки системы. В результате при попытке дефрагментировать загрузочные файлы будет возникать ошибка 0x89000018. Диагностический пакет обнаруживает и устраняет обе проблемы.

Windows поддерживает диск в порядке с помощью планировщика заданий, в котором для этого назначена специальная задача **ScheduledDefrag** в разделе **Microsoft - Windows – Defrag**. Если отключена эта задача или служба планировщика, дефрагментация не выполняется.

Дату последнего запуска дефрагментатора скрипт извлекает из журнала событий, как и в случае со временем загрузки. Однако здесь используется **Журналы Windows – Приложение**, куда записываются все действия дефрагментатора (событие **258** с источником **Defrag**).



Последнее событие дефрагментации вовсе необязательно относится к обработке системного диска. Если вы отфильтруете записи журнала, то заметите, что там есть отчеты о других разделах и даже съемных дисках. Дефрагментация загрузочных файлов выполняется отдельно, как видно на рисунке выше.

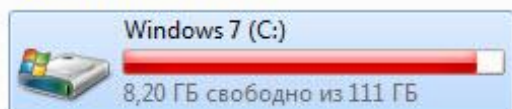
Если у вас дефрагментатор не работал даже пару недель, можно смело сказать, что резерв для оптимизации есть.

Безусловно, мой скрипт не способен определить сценарий, при котором вы пользуетесь сторонним дефрагментатором, а встроенный отключили. В этом случае вы сами несете ответственность за фрагментированность системного диска. Что же касается попеременного использования встроенного и стороннего дефрагментаторов, то я считаю, что в этом как минимум нет смысла. Ведь каждая программа по-своему раскладывает файлы на диске, и программы скорее будут мешать друг другу, чем помогать.

Также в этом блоке приводится информации о проценте свободного пространства на системном диске.

Системный раздел C: объемом 111,79 Гб имеет 7,51% свободного пространства

Жесткие диски (1)



Вообще-то, у меня в системе немного другая картина, но я создал небольшой недостаток места, чтобы проиллюстрировать один важный момент.

Чтобы обеспечить полную и эффективную дефрагментацию, на разделе должно быть не менее 15% свободного пространства.

В соответствии с [официальной документацией](#) и [высказываниями разработчиков](#) в противном случае выполняется только частичная дефрагментация, что происходит в том числе и при работе дефрагментатора по расписанию. Да, полную дефрагментацию можно форсировать параметром командной строки, но это уже ручная работа.

Мораль в том, что если вы хотите оптимизировать загрузку, не экономьте на спичках, т.е. на размере системного раздела, и поддерживайте на нем не менее 15% свободного пространства.

Сведения об аппаратной конфигурации

Завершают отчет три строки об операционной системе, процессоре и оперативной памяти.

```
Компьютер работает под управлением Microsoft Windows 7 Максимальная 6.1.7600 32-bit
Установлен процессор Intel(R) Core(TM)2 Duo CPU T7500 @ 2.20GHz
Системе доступно 3574 Мб оперативной памяти
```

Эту информацию вы можете посмотреть, например, в свойствах системы, нажав сочетание клавиш **WIN+PAUSE**.

Быстрое «железо» не просто положительно влияет на скорость загрузки системы, но и является одним из основных факторов в этом вопросе. В первую очередь это касается мощности процессора и скорости жесткого диска. Я оставил эту информацию напоследок не потому, что она не важна, а потому, что вас, скорее всего, интересует ускорение загрузки без обновления конфигурации :)

Используя диагностический пакет, вы можете не только устранить проблемы, тормозящие загрузку Windows, но и сделать выводы о том, за счет чего ее можно ускорить. В следующей главе вы найдете три простых совета, после применения которых ваша система будет загружаться быстрее!

Три простых совета по ускорению загрузки Windows 7

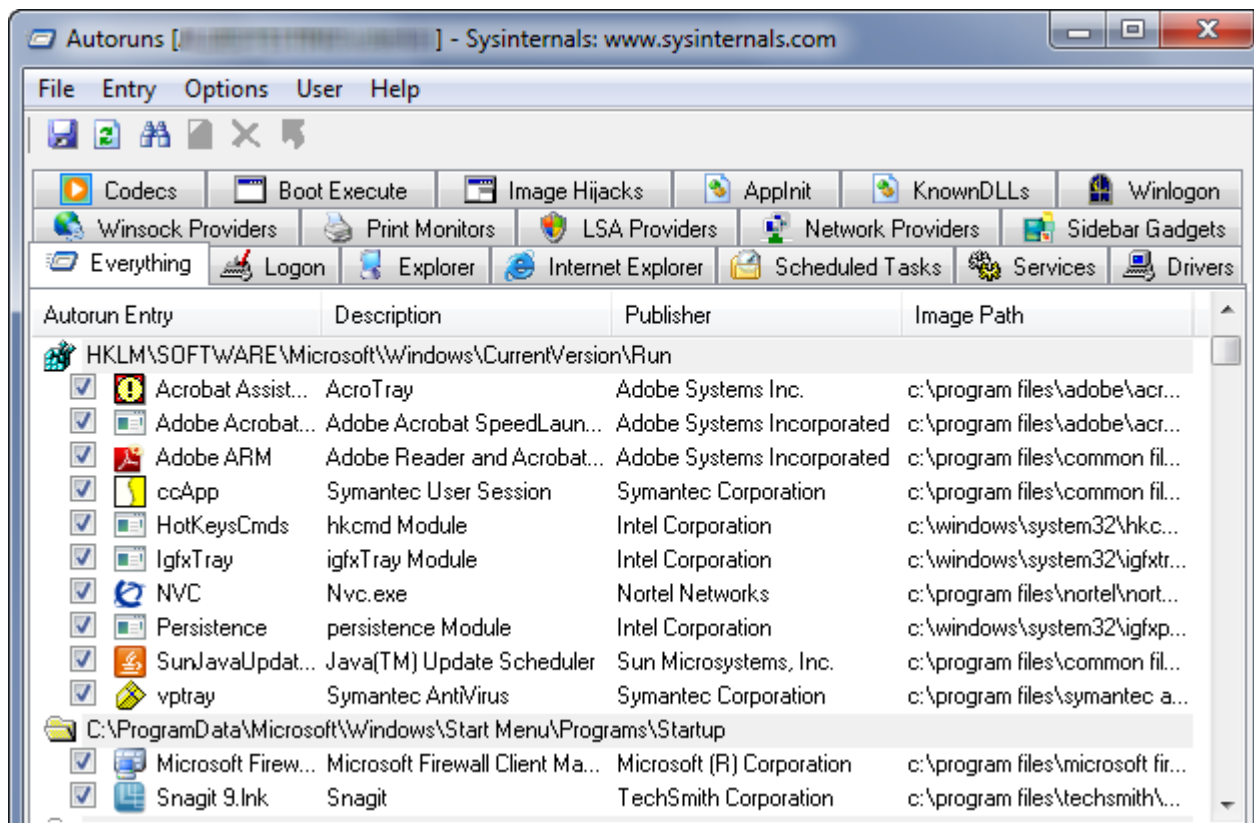
Ускорить загрузку любой системы очень просто, причем без сторонних программ и особых усилий. Воспользовавшись моими тремя простыми советами, вы лично в этом убедитесь. Их нужно выполнять в том порядке, в котором они перечислены, а почему – вы узнаете из этой главы.

Наведите порядок в автозагрузке

Если отбросить или устранить серьезные проблемы с загрузкой, сильнее всего замедляет ее большое количество программ, стартующих при запуске операционной системы. Стандартный краткий совет «отключите ненужные программы из автозагрузки» является поверхностным по двум причинам:

- Автоматический запуск многих программ удобен и позволяет сэкономить время. Отключив такие программы, действительно можно ускорить загрузку системы, но зато потом придется тратить время на их запуск вручную. Зачем менять шило на мыло?
- Зачастую под нож идут программы, назначение которых непонятно пользователям. При этом они могут играть важную роль в обеспечении безопасности системы. Например, крайне важно выполнять проверку обновлений для виртуальной машины Java, Apple QuickTime и Adobe Reader, [учитывая регулярно устраняемые в них уязвимости](#).

Конечно, универсальный совет по наведению порядка в автозагрузке дать невозможно, поэтому я поделюсь с вами подходом, который использую сам, когда меня просят «посмотреть комп». Я предпочитаю утилиту [AutoRuns](#).



Но при отсутствии оной подойдет и системная утилита **msconfig**.

Мой подход очень простой.

1. **Драйверы и защитные программы оставляем.** Понятно, что если отключить драйвер клавиатуры или мыши, устройства работать не будут. В общем случае, нужно оставлять записи, принадлежащие Microsoft и производителям оборудования (Intel, NVidia и т.д.). Антивирусы и фаерволы должны обеспечивать защиту системы с первой секунды, поэтому отключать их тоже нельзя.
2. **Неизвестное ищем в Интернете.** Если назначение какой-то программы неочевидно, можно быстро это выяснить поиском в сети по имени исполняемого файла (в столбце «Команда»). Следующие два пункта зависят исключительно от вашего умения анализировать результаты поиска и списка



предпочитаемых программ.

3. **Непонятное не трогаем.** Если поиск в сети не помог вам определить назначение программы, лучше оставить ее в покое. Впрочем, можно утолить жажду к экспериментам, воспользовавшись моим любимым способом (ниже). При этом риск создать себе проблему будет намного меньше, нежели при полном отключении программ.
4. **Ненужное отключаем.** Когда назначение программы **абсолютно понятно**, но вы ей почти не пользуетесь, надо убирать ее из автозагрузки. Подчеркну, что речь идет только о программах, с которыми можно взаимодействовать. Многие приложения в автозагрузке не имеют графического интерфейса и служат, например, для проверки обновлений безопасности «родительских» программ в фоновом режиме. Отключив их, вы создадите потенциальную уязвимость в защите системы.



У вас и так был порядок в автозагрузке, и мой подход не открыл для вас Америку? Тогда попробуйте **мой любимый способ** – распределение автозагрузки по времени с помощью планировщика заданий, чему [посвящена одна из следующих глав](#). Если потом экспортировать настроенные задания планировщика для любимых программ, даже после переустановки системы можно очень быстро оптимизировать автозагрузку.



Моментальный эффект ускорения загрузки вы увидите даже невооруженным глазом при следующем запуске системы. Здесь все просто: программ в автозагрузке стало меньше, следовательно, система загрузилась быстрее. Помимо количества, прирост скорости еще зависит и от «тяжести» программ, убранных из немедленной автозагрузки.

Этот совет подходит как владельцам жестких (HDD), так и твердотельных (SSD) дисков.

Проверьте SuperFetch и ReadyBoot

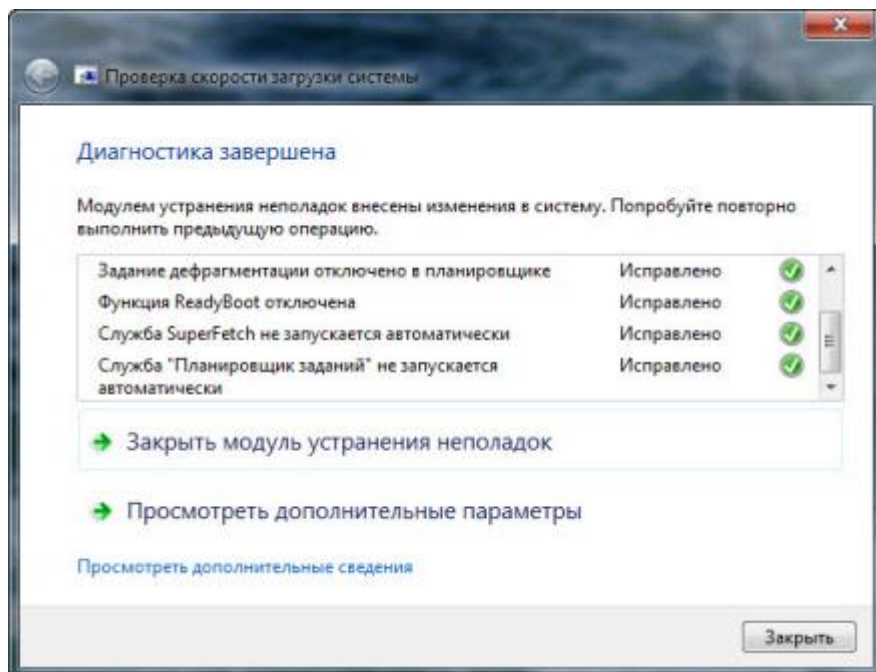


На твердотельных накопителях функция ReadyBoot [не работает](#), но на жестких дисках она способствует ускорению запуска Windows за счет размещения в оперативной памяти загрузочных файлов системы и приложений, стартующих вместе с ней. **Отложенный эффект** наведения порядка в автозагрузке (в том числе распределения по времени) связан именно с работой ReadyBoot и не столь заметен на глаз.

Одно из улучшений службы SuperFetch в Windows 7 выражается в том, что анализ и наполнение кэша выполняются спустя некоторое время после загрузки системы. Очевидно, это сделано для того, чтобы не мешать нам выполнять действия в системе сразу после ее запуска. Поэтому службе SuperFetch требуется время, чтобы отреагировать на изменения, которые вы сделали в автозагрузке программ.

Эффект от работы ReadyBoot проявится через несколько перезагрузок, но с учетом сказанного выше нет смысла перезагружать систему сразу же после входа. Либо дайте ей постоять немного, либо просто работайте в своем обычном режиме два-три дня, выполняя перезагрузку как обычно.

Проверить состояние SuperFetch и ReadyBoot можно с помощью [моего диагностического пакета](#).



Если ReadyBoot не работает, пакет поможет вам устранить неполадки (изменения вступают в силу после перезагрузки системы). Заодно можно исправить проблемы в работе дефрагментатора Windows, и дальше речь пойдет как раз о нем.

Дефрагментируйте диск – сейчас и по расписанию

Если ваша система установлена на SSD диск, любые советы по дефрагментации к ней [не относятся](#). Но вопрос фрагментации очень актуален для владельцев обычных жестких дисков, которые являются наиболее медленными компонентами в современных компьютерах.

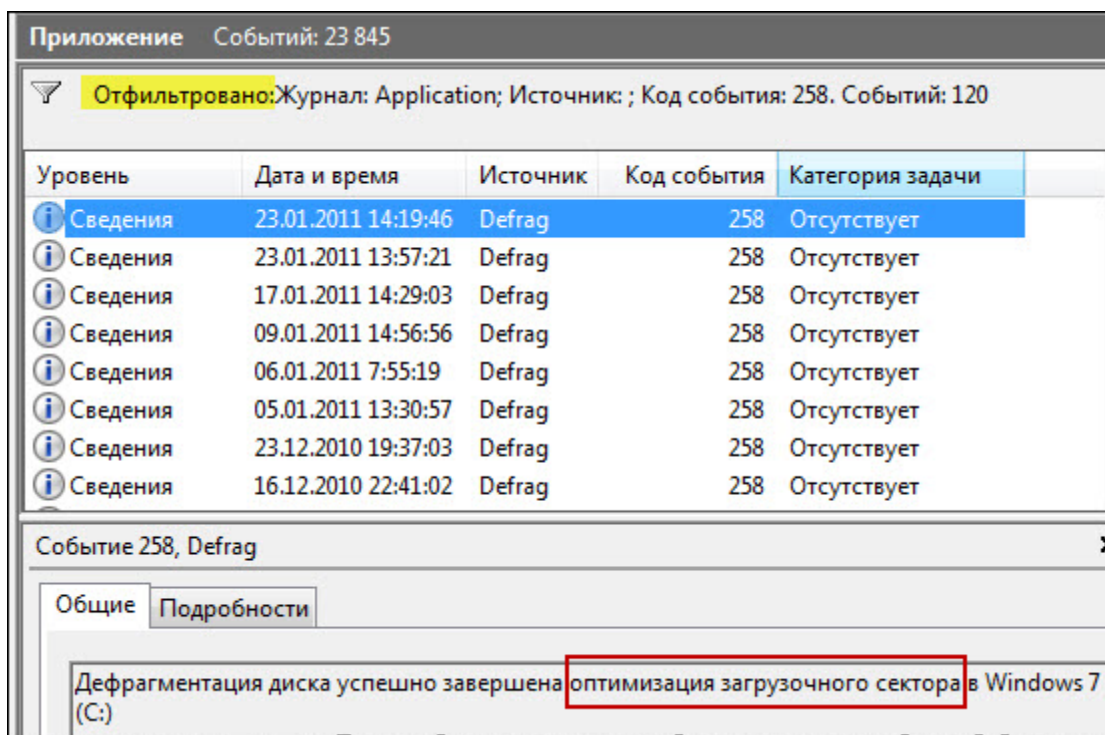


Итак, у вас работает ReadyBoot и вы уже перезагрузились пару раз после распределения автозагрузки. Самое время дефрагментировать загрузочные файлы на системном диске!

```
defrag C: /B /U
```

Любопытно, что данный параметр не документирован официально - его подсказал мой коллега на форуме, а позже я нашел упоминание в блоге разработчиков и дополнил свой [список параметров командной строки](#). По сути это просто быстрее, чем выполнять полную дефрагментацию, в рамках которой обрабатываются и файлы загрузки.

Дефрагментация загрузочных файлов дает **моментальный эффект**, а после пары перезагрузок он даже усилится за счет тренировки ReadyBoot. Однако эффект будет временным, если дефрагментация не выполняется по расписанию. Кстати, в этом случае система выполняет дефрагментацию загрузочных файлов отдельно, т.е. задействует как раз тот недокументированный параметр, о котором шла речь выше.



Упоминание о загрузочном секторе – это ошибка локализации. В английской системе это называется просто boot optimization, т.е. оптимизация загрузки.

Эффект от дефрагментации загрузочных файлов может свести на нет сторонний дефрагментатор, который по-своему размещает фрагменты файлов на диске.

Для эффективной работы дефрагментатора Windows необходимо соблюдение трех условий:

1. Включенная служба «Планировщик заданий».
2. Включенная задача **ScheduledDefrag** в планировщике.
3. Не менее 15% свободного пространства на диске.

Без первых двух условий дефрагментация по расписанию просто не выполняется, а третье необходимо для выполнения полной дефрагментации. Выявить и устранить проблемы в работе дефрагментатора Windows вам также поможет [мой диагностический пакет](#).

Но это не конец нашей истории – у меня есть еще один совет! Он не ускоряет загрузку системы, но позволяет намного быстрее приступать к работе с ней. Ведь разве не ради этого мы пытаемся сократить время загрузки? :)

Бонус: используйте режимы сна и гибернации!

Как бы вы не старались ускорить загрузку Windows, ее длительность всегда будет в разы превышать время выхода системы из гибернации и тем более сна. Я не буду останавливаться на рассказе о тонкостях этих режимов, потому что на OSZone [есть замечательная статья на данную тему](#).



Я активно использую оба режима на двух своих системах – после периода неактивности они уходят в сон, а вместо выключения я перевожу их в гибернацию. Перезагрузка требуется лишь при установке обновлений и некоторых программ, поэтому в среднем она происходит раз в три-четыре дня.

Посмотреть время выхода из сна и гибернации очень легко в уже знакомом вам [журнале событий Diagnostics-Performance](#). Оба режима записываются в событие с кодом **300**.

Отфильтровано: Журнал: Microsoft-Windows-Diagnostics-Performance/Operational; Исто

Уровень	Дата и время	Источник	Код события
❗ Ошибка	09.02.2011 21:21:18	Diagnostics-Performance	300
❗ Ошибка	09.02.2011 9:31:46	Diagnostics-Performance	300
❗ Ошибка	09.02.2011 1:38:40	Diagnostics-Performance	300
❗ Ошибка	08.02.2011 21:11:14	Diagnostics-Performance	300
❗ Ошибка	08.02.2011 9:25:53	Diagnostics-Performance	300

Событие 300, Diagnostics-Performance

Общие Подробности

ОС Windows возобновила работу после выхода из ждущего режима:

Продолжительность ждущего режима	:	22599мс
Время перехода в ждущий режим (UTC)	:	2011-02-08T06:39:34.645868000Z
Продолжительность выхода из ждущего режима	:	10020мс
Время выхода из ждущего режима (UTC)	:	2011-02-08T18:18:30.834680300Z
IsDegradation	:	false

По дате и времени на рисунке хорошо видно, что мой ноутбук выходит из гибернации дважды в день – с утра и после работы. На это требуется 10-12 секунд, сравните со скоростью полной загрузки системы! Из сна система вообще моментально выходит, и на нетбуке я пользуюсь этим режимом чаще, поскольку тот настроен на более быстрый уход в сон.

В мобильных системах сон и гибернация способствуют энергосбережению, и хотя в настольных системах это не имеет значения, ничто не мешает пользоваться этими режимами и в них. Попробуйте и посмотрите, как это удобно. Заодно вы узнаете, нет ли у вас в системе проблем с ними, что иногда случается (например, периферийные устройства отказываются «просыпаться»). Как и в случае с загрузкой, диагностику можно проводить с помощью Windows Performance Tools. Проблема зачастую лежит в области драйверов, но не всегда к устройству есть более подходящий драйвер.




Столкнетесь ли вы с проблемой, отчасти зависит от происхождения вашего компьютера. Очевидно, меньше всего им подвержены нетбуки и ноутбуки, продававшиеся с Windows 7 на борту. Особенно при использовании без мышей и других периферийных, поскольку производитель тестирует не только «железо», но и комплект драйверов. Настольные системы от OEM-сборщиков также с меньшей вероятностью подвержены проблемам. Если предустановленная система заменялась более функциональным изданием, вероятность бесперебойного сочетания драйверов становится ниже. Ну а самыми проблемными мне представляются «г-сборки» на «самосборе» :)

Воспользовавшись этими простыми советами, вы сможете намного быстрее приступать к работе после включения или перезагрузки системы. В следующей главе подробно рассматривается оптимизация автозагрузки операционной системы за счет распределения запускаемых программ по времени.

Ускорение загрузки Windows 7 и Vista с помощью планировщика заданий

Из этой главы вы узнаете, как с помощью планировщика заданий Windows 7 и Vista организовать отложенный запуск программ, которые находятся в автозагрузке операционной системы. Откладывая запуск приложений, в которых нет необходимости с первых минут работы ОС, можно добиться ускорения ее загрузки, сохраняя при этом удобство автоматического запуска.

Одним из важных критериев работы операционной системы является скорость ее загрузки. Однако далеко не всегда длительная загрузка свидетельствует о неполадках или недостаточной оптимизации ОС. Работая в Windows 7 и Vista, вы, возможно, наблюдали такую картину. Система загрузилась, видны рабочий стол с ярлыками и панель задач, но если навести курсор мыши на ярлык или кнопку Пуск, он превращается в «бублик» , и приходится ждать еще какое-то время, пока он не превратится в стрелку, позволяя выполнить желаемое действие.

Что же происходит в это время? Ответ вас, видимо, не удивит – скорее всего, происходит запуск приложений, находящихся в автозагрузке. И чем больше приложений запускается одновременно с системой, тем дольше она загружается с точки зрения конечного пользователя.

Программы в автозагрузке

Стандартный совет в таких случаях – «почистить автозагрузку», т. е. отключить запуск ненужных приложений, например, с помощью утилиты **msconfig**, запускаемой из меню Пуск – Поиск. Совет хороший, но что делать в том случае, если автоматический запуск все-таки имеет смысл?

Недостаток стандартной автозагрузки

Рассмотрим, например, Adobe Reader Speed Launcher (`reader_sl.exe`) – утилита прописывается в автозагрузку при установке Adobe Acrobat Reader. Запуск утилиты значительно ускоряет открытие Adobe Reader, в том числе и при просмотре PDF-файлов в браузере (обсуждение альтернативных программ для просмотра PDF-файлов выходит за рамки этого рассказа). Или возьмем `jusched.exe` – утилиту, следящую за обновлениями виртуальной машины Sun Java, которую нужно поддерживать в актуальном состоянии из соображений безопасности. Утилита также прописывается в автозагрузку, поскольку вручную выполнять обновление никто не будет.



Даже если вы считаете, что автоматический запуск этих утилит можно смело отключать, у вас в арсенале, скорее всего, найдутся программы, которые вы вполне осознанно запускаете автоматически. Но все ли они нужны вам с первой минуты работы в системе?

Вряд ли вы первым делом начинаете читать документы в формате PDF или беспокоиться об обновлении виртуальной машины. Возможно, вам сразу не требуется эмулятор виртуальных дисков или клиент обмена сообщениями.

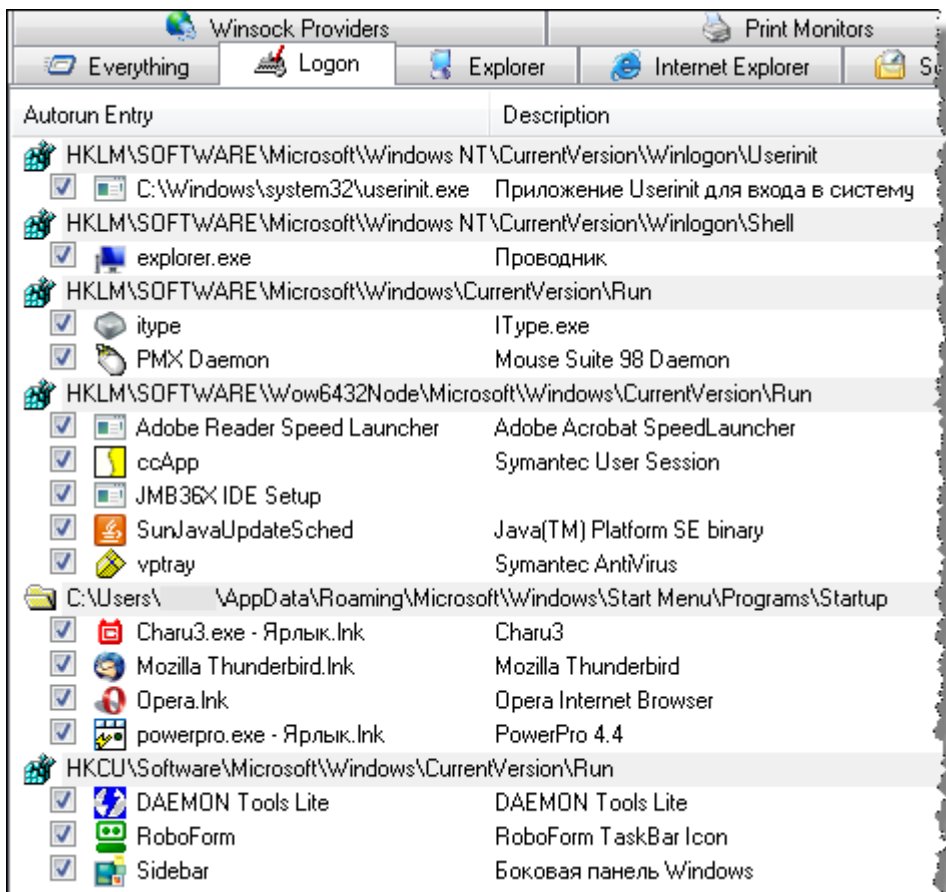
Список можно продолжать, но главный недостаток автозагрузки уже очевиден – все приложения запускаются одновременно с загрузкой системы или при входе в нее пользователя.

Было бы здорово, если бы существовала возможность немного распределить загрузку программ по времени. И такая возможность есть! **Планировщик заданий Windows 7 и Vista позволяет откладывать выполнение задач**, привязанных к запуску системы или входе пользователя. Дальше я продемонстрирую, как это можно сделать на реальном примере.

Оцените количество программ

Чтобы было интереснее, я предлагаю вам заглянуть в мою автозагрузку (на момент написания материала). Автоматический запуск приложений может выполняться как из папки Автозагрузка, так и из различных разделов реестра. Для просмотра автозагрузки можно воспользоваться упомянутой выше утилитой `msconfig`.

Мне, впрочем, больше нравится [AutoRuns](#) от Sysinternals – ее окно можно развернуть во весь экран, не говоря уже о более широком функционале. В AutoRuns вкладка **Logon** выполняет ту же функцию, что и вкладка **Автозагрузка** утилиты `msconfig`.



Как видите, у меня загружается полтора десятка различных приложений, начиная от вышеупомянутого Adobe Reader Speed Launcher и заканчивая почтовым клиентом Thunderbird. Прежде чем приступать к распределению загрузки программ по времени, нужно определиться, какие из них вам не нужны сразу же после запуска системы и в каком порядке вы бы хотели их запускать – отложенный запуск этих приложений будет реализован с помощью планировщика задач.

Определите самое важное и измерьте скорость

Я навожу порядок в автозагрузке, [пользуясь очень простым подходом](#). Применив его к своей автозагрузке, получаю:

- userinit.exe и explorer.exe – важнейшие системные приложения, их запуск необходим;
- ccapp.exe от Symantec обеспечивает автоматическую защиту – ее лучше иметь сразу;
- а вот vptray.exe нужен для доступа к панели управления антивирусом из области уведомлений (трея) – явно не первоочередная задача;
- почтовый клиент мне нужен сразу, поскольку я начинаю день с чтения почты;
- специализированные драйверы мыши и клавиатуры тоже нужны сразу

Но автоматическая загрузка практически всех остальных приложений и утилит хотя и нужна, но явно не с первых секунд работы системы. Вот их загрузку я и распределяю по времени – их автозапуск будет выполнять планировщик.

Если приложение имеет настройки автоматической загрузки в своем графическом интерфейсе, надежнее использовать их, чем msconfig или Autoruns. Некоторые приложения после запуска могут восстанавливать параметры автозагрузки, отключенные сторонними средствами.

Предварительное тестирование

Чтобы проверить, действительно ли распределение автозапуска приложений ускоряет загрузку вашей ОС, имеет смысл замерить время загрузки Windows 7 и Vista до и после отключения предполагаемых к распределению по времени программ. Это можно сделать с помощью [журнала событий](#) или [моего диагностического пакета](#).

Важное примечание о приоритете CPU и I/O

Когда в Windows 7 и Vista программа запускается из планировщика, ей принудительно назначается низкий приоритет процессорного времени (CPU) и операций ввода-вывода (I/O). Это сделано специально, поскольку задачи планировщика рассчитаны на выполнение в фоне.

Низкая нагрузка на процессор и жесткий диск оптимально подходит для фоновых задач, но не для всех программ. Так, программам для проверки обновлений или почтовому клиенту вряд ли нужно много ресурсов – их можно безболезненно запускать из планировщика. Но если программа требует повышенного внимания процессора или активно взаимодействует с диском, ее лучше не ограничивать.

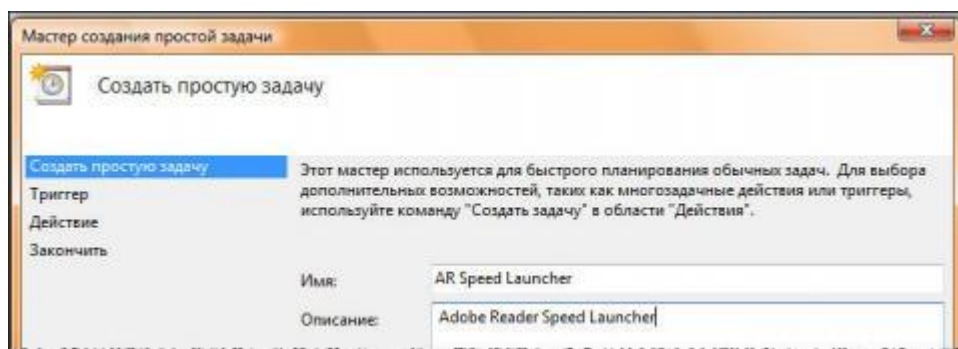
Это свойство планировщика можно обойти лишь частично. Например, нормальный приоритет CPU можно задать командным файлом с помощью команды **start**, имеющей параметр **/normal**. Однако в Windows не предусмотрено средств для изменения приоритета I/O.

О приоритетах CPU и I/O речь еще пойдет в следующей главе, а сейчас я предлагаю вам создать задачу в планировщике.

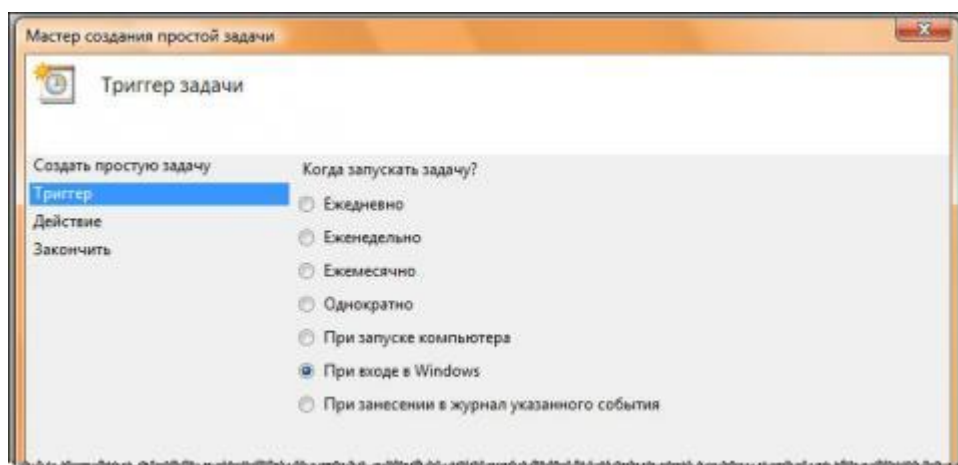
Создание простой задачи в планировщике заданий

Для примера я возьму все тот же Adobe Reader Speed Launcher – процедура будет фактически одинаковой для всех приложений. Откройте планировщик заданий (**Пуск – Поиск – taskschd.msc**).

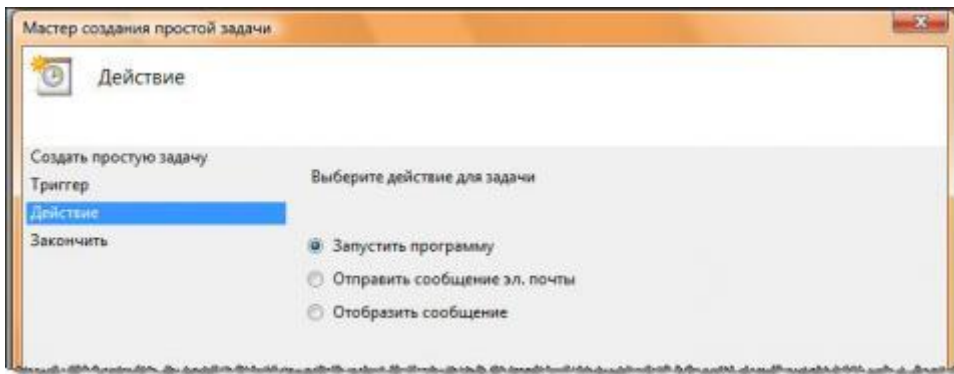
Я рекомендую разместить все задачи, связанные с автозагрузкой, в одной папке. Щелкните правой кнопкой мыши на узле **Библиотека планировщика заданий**, выберите из контекстного меню пункт **Создать папку** и укажите имя для нее – я назвал свою **StartUp**. В этой папке будут размещаться задачи автозагрузки приложений. Затем щелкните правой кнопкой мыши по созданной папке и выберите из контекстного меню пункт **Создать простую задачу**. Вы увидите первое окно мастера.



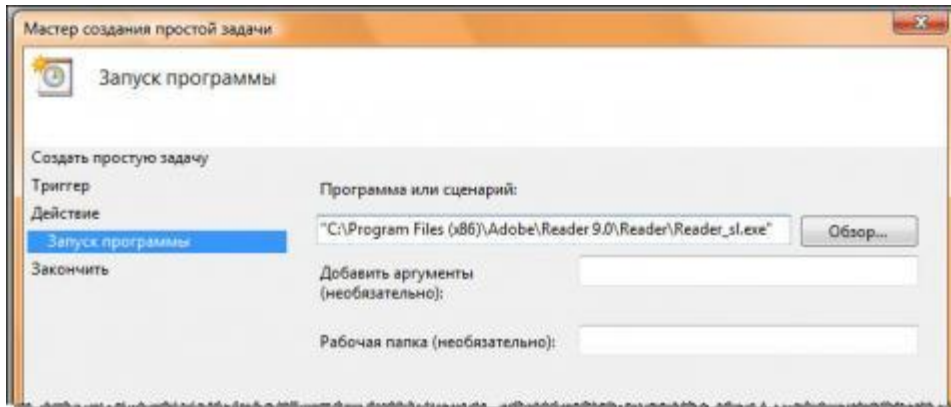
Укажите имя для задачи и нажмите кнопку **Далее**.



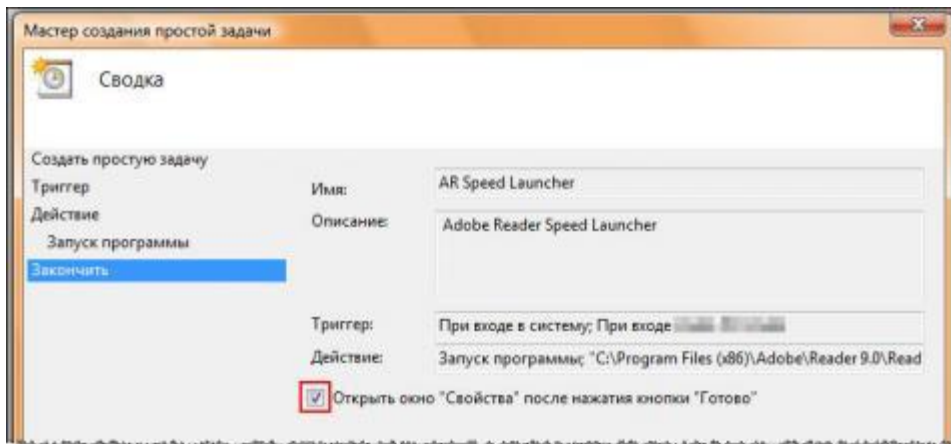
В качестве триггера задачи укажите **При входе в Windows** и нажмите кнопку **Далее**.



В качестве действия укажите **Запустить программу** и нажмите кнопку **Далее**.

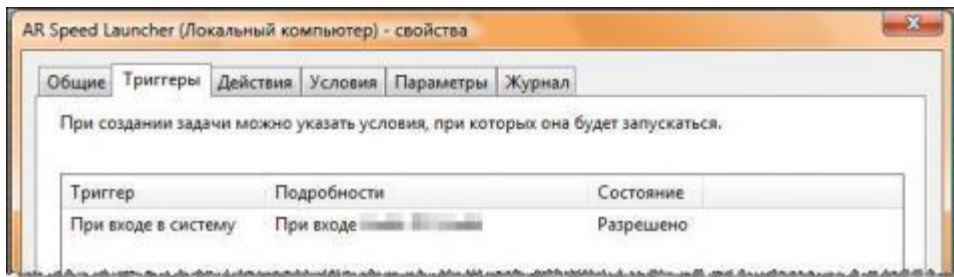


В окне **Запуск программы** вам нужно указать путь к исполняемому файлу программы. Используйте кнопку **Обзор** или введите путь вручную, не забывая заключать его в кавычки при наличии пробелов. Если вы используете [AutoRuns](#), можно скопировать путь к программе из информационной панели, расположенной внизу окна. Если программа использует аргументы командной строки, введите их в соответствующее поле. Например, у боковой панели Windows таким аргументом является /autorun. Нажмите кнопку **Далее**, чтобы перейти к сводке задачи.

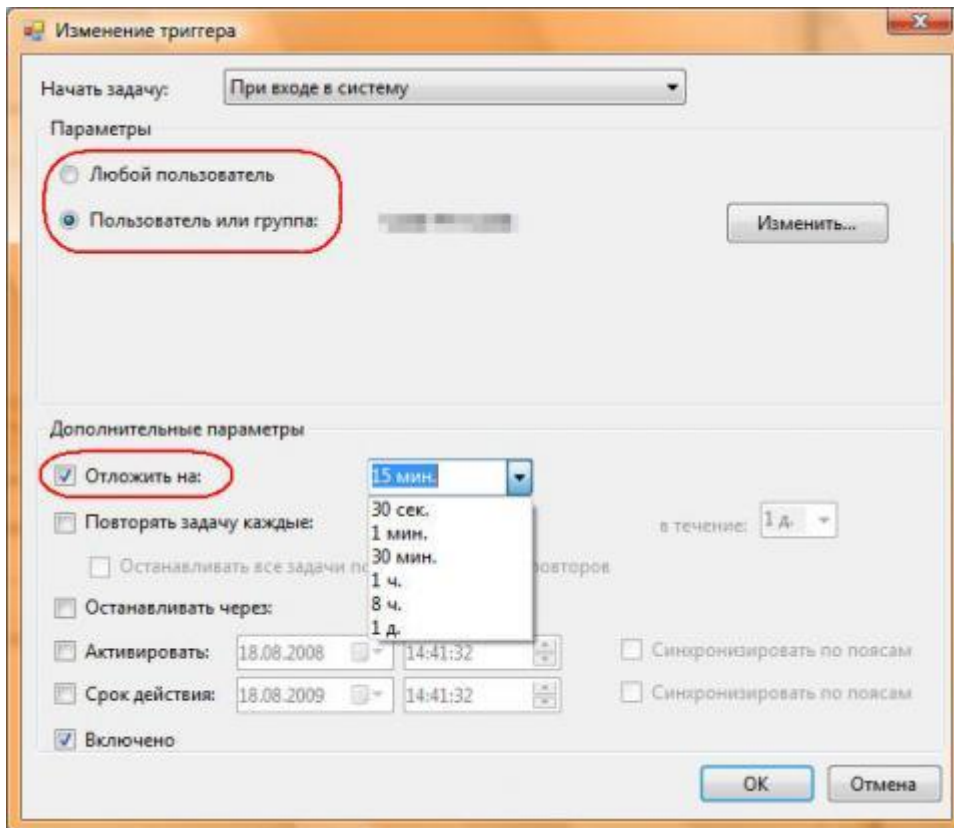


Убедитесь, что параметры задачи указаны правильно. Установите флажок, отвечающий за открытие свойств задачи, и нажмите кнопку **Готово**. Задача будет создана, и сразу же откроется окно ее свойств.

Теперь нужно настроить отложенный запуск задачи. Перейдите на вкладку **Триггеры**



и дважды щелкните по триггеру **При входе в систему** (либо выделите его и нажмите кнопку **Изменить**).



В окне **Изменение триггера** следует обратить внимание на два параметра, выделенные на рисунке.

Вы должны указать, будет задача запускаться только для конкретной учетной записи (по умолчанию – для вашей), или для всех пользователей, входящих в систему. Если с приложением работаете только вы, запускать его для всех пользователей, скорее всего, не имеет смысла.

1. Вы должны указать, на какой временной интервал будет отложен запуск программы после входа пользователя в систему. Здесь однозначных рекомендаций быть не может – все зависит от ваших предпочтений. Меня, например, вполне устраивает запуск менеджера буфера обмена через две минуты, а Skype – через 5 минут. Что же касается утилиты `jusched.exe`, то ее запуск я откладываю на 30 минут.
2. Если вам понадобится открыть программу, прежде чем сработает ее отложенный запуск, вы всегда можете это сделать – по умолчанию планировщик не запускает программу, если ее экземпляр уже запущен (это можно настроить в свойствах задачи на вкладке **Общие**). После того, как вы сконфигурируете параметры запуска, нажмите кнопку **ОК** – задача готова.

Аналогичным образом создавайте задания для других программ, запуск которых вы хотите отложить.

Ускорить процесс можно с помощью экспорта настроенной задачи, ее изменения и импорта с новым именем.

Об этом мы поговорим чуть ниже, а пока надо убедиться, что все работает.

Проверка работоспособности заданий планировщика

После перезагрузки компьютера вы сможете опытным путем убедиться, что созданные вами задания обрабатываются – если приложение запустилось в указанное время, значит все нормально. Такие утилиты, как reader_sl.exe и jusched.exe, не имеют графического интерфейса – используйте диспетчер задач (CTRL+SHIFT+ESC) для просмотра запущенных процессов. Вы также можете использовать вкладку **Журнал** в свойствах задачи (в Windows 7 журнал отключен по умолчанию). Наличие ошибок свидетельствует о неправильной работе задания.

Если вы замерили длительность запуска системы до распределения автозагрузки по времени, вы можете повторить замер и определить, насколько быстрее стала загружаться система. В моем случае отложенный запуск девяти приложений позволил сократить время загрузки примерно на одну минуту.

Экспорт и импорт заданий

Задания планировщика Windows 7 и Vista можно экспортировать для переноса на другой компьютер, с целью резервного копирования, а также для ускорения процесса распределения автозагрузки. Для экспорта задания щелкните по нему правой кнопкой мыши и выберите из контекстного меню пункт **Экспортировать**. Экспортируемая задача сохраняется в формате XML – такой файл можно открыть в любом текстовом редакторе (например, в блокноте). При импорте задач примите к сведению, что параметрами задачи являются, в том числе имя компьютера и имя учетной записи, которые могут отличаться на другом компьютере. В таком случае XML-файл можно отредактировать вручную перед импортом, указав правильные параметры, либо сделать это в графическом интерфейсе планировщика при импорте.



Клонирование заданий

Давайте посмотрим, как из одного задания отложенной автозагрузки сделать другое за несколько секунд. Откройте любое задание. Нас интересует интервал между запуском системы и программы, а также путь к программе и параметры командной строки.

Интервал определяется параметром **RandomDelay** в узле **Triggers**.

```
<Triggers>
  <TimeTrigger>
    <StartBoundary>2011-01-15T23:33:13.341902</StartBoundary>
    <Enabled>>true</Enabled>
    <RandomDelay>PT4M</RandomDelay>
  </TimeTrigger>
</Triggers>
```

В данном случае **4M** означает загрузку, отложенную на 4 минуты. Исправьте, например, на 5.

Запуск программы определяется в узле **Actions** параметрами **Command** и **Arguments**.

```
<Actions Context="Author">
  <Exec>
    <Command>"C:\Program Files\Skype\Phone\Skype.exe"</Command>
    <Arguments>/nosplash /minimized</Arguments>
  </Exec>
</Actions>
```

Для первого достаточно указать полный путь к программе в кавычках, а во втором задаются параметры командной строки, если они нужны.

Один из читателей блога спросил, можно ли достичь аналогичного эффекта без планировщика заданий. В принципе, можно обойтись обычным командным файлом. В нем можно прописать запуск программ и [паузы](#) между ними, а затем поместить файл в автозагрузку. Пример файла вы найдете [в моем комментарии](#) в блоге.

Ускорение загрузки одним твиком реестра

Можно ускорить загрузку Windows, изменив всего один параметр в реестре! Стоп... звучит слишком хорошо, чтобы быть правдой, не так ли? Однако это возможно, потому что когда-то Microsoft решила повоевать за автозагрузку с разработчиками программ. Я нарисовал для вас полную картину работы твика, чтобы ваши ожидания от него соответствовали действительности.

Эта глава не случайно идет после рассказов о ReadyBoot, дефрагментации и планировщике заданий. Хотя твик вполне рабочий, он не всегда ощутим и менее эффективен, чем описанные выше решения.

В каких случаях изменение в реестре ускорит загрузку системы

Действие твика таково, что наибольшее ускорение загрузки Windows будет наблюдаться в системах, где:

- много программ в автозагрузке
- слабый процессор
- медленный жесткий диск (например, 5400 rpm)

Таким образом, ускорение будет заметно на глаз на нетбуке или слабом ноутбуке с двумя-тремя десятками программ в автозагрузке.

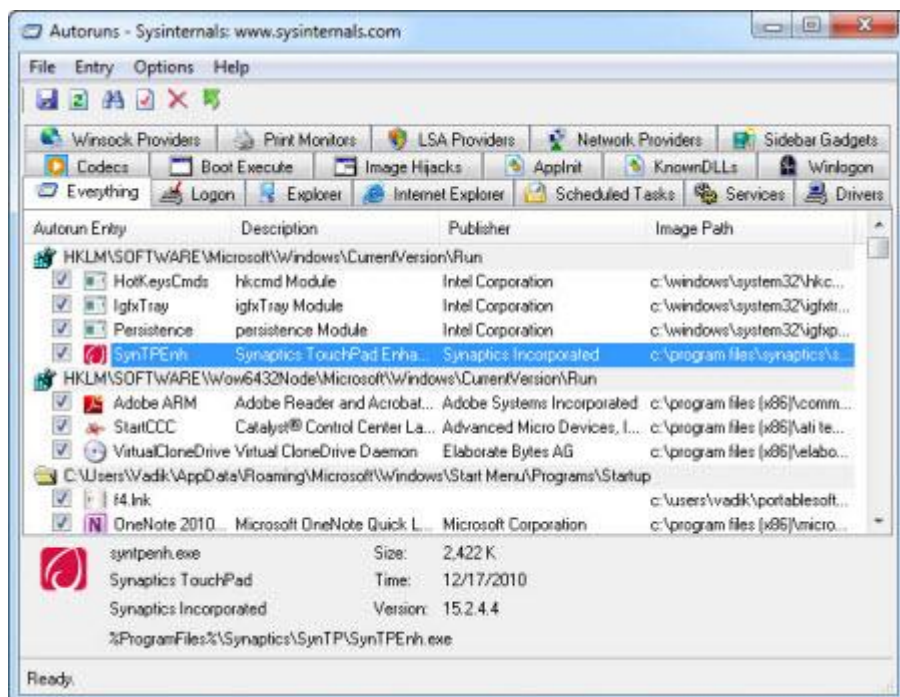
Если же у вас четырехъядерный процессор, SSD диск и минимум программ в автозапуске, вы вряд ли заметите ускорение загрузки (но возможен [эффект плацебо](#) :)

Что входит в понятие «автозагрузка»

Для автоматического запуска программ и скриптов в Windows, как правило, используются:

- **разделы реестра Run:**
HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Run
HKKEY_CURRENT_USER\Software\Microsoft\Windows\CurrentVersion\Run
- **папка «Автозагрузка»,** которую можно открыть [командой shell](#) – `shell:startup`
- **задания планировщика и скрипты групповой политики,** выполняющиеся при входе пользователя в систему

Программы из первых двух пунктов этого списка можно увидеть в утилите **msconfig**, а полную картину дает [Autoruns](#).



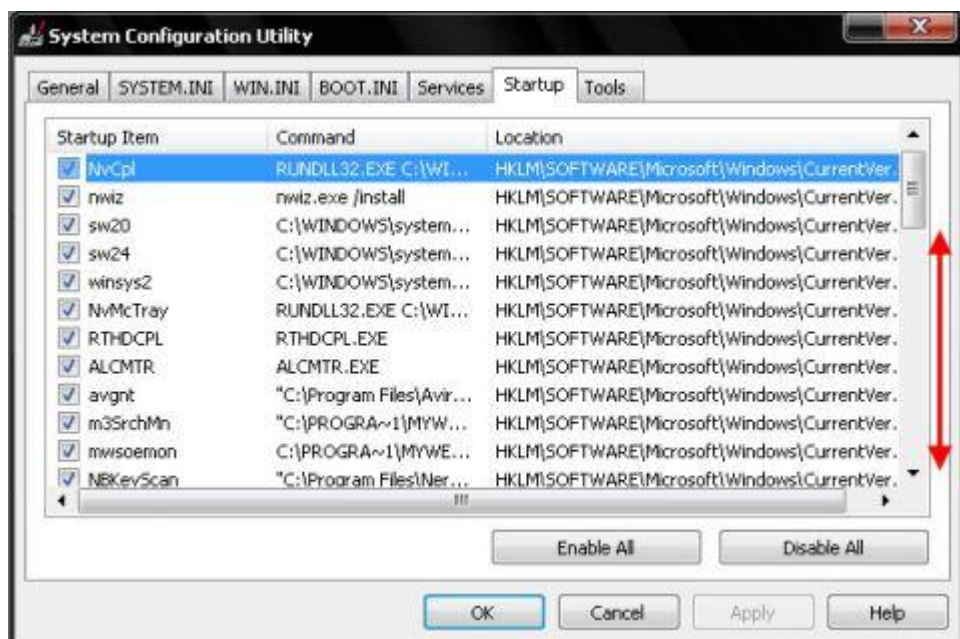
Вы можете узнать больше об автозагрузке из материалов моих коллег ([1](#) и [2](#)), а я продолжу свой рассказ.

Влияние программ в автозагрузке на скорость запуска Windows

Систему можно считать полностью загрузившейся, когда с рабочим столом или начальным экраном Windows 8 можно свободно взаимодействовать. Это значит, что система моментально откликается на запуск программ или вызов контекстных меню.

Если отбросить время, требуемое на загрузку драйверов, а также системных и сторонних служб, то программы в автозапуске становятся основным фактором, влияющим на длительность загрузки системы.

Многие программы стремятся прописаться в автозагрузку сразу при установке, а иногда это происходит без вашего ведома.



Доводилось видеть такую картину? Вряд ли у вас все так запущено, но у кого-нибудь из друзей, знакомых или родственников – вполне может быть. Все это тормозит загрузку системы!

Microsoft против разработчиков программ

Нет, Microsoft, конечно, не против разработчиков программ, ибо Windows без приложений никому не нужна. Однако Microsoft не радуется, что разработчики пишают свои программы в автозагрузку. Потому что пока они стартуют, в том числе и в фоне, мы не можем нормально запускать те программы, которые нам действительно нужны. Но при этом принято говорить, что Windows долго запускается, не так ли?

Создатели **Windows Vista**, видимо, чувствовали, что ОС получается тяжеловатой, в том числе и в плане загрузки системы. Поэтому в Microsoft решили уменьшить негативный эффект от программ в автозагрузке и сделать рабочий стол доступным немного быстрее.

Форсирование нормального приоритета потока для программ в автозагрузке

Поток (thread) является частью процесса и может выполняться с разным приоритетом. Потокам с высоким приоритетом требуется больше ресурсов, чем их коллегам с обычным или низким приоритетом.

В Windows Vista для размещенных в автозагрузке программ зафиксировали приоритет потока (thread priority) на обычном уровне (Normal) и заблокировали любые попытки повысить его. Эти ограничения действовали на протяжении некоторого периода времени после запуска системы (подробности чуть ниже).



Логика создателей Windows Vista была такова, что если этого не сделать, некоторые разработчики будут стремиться повысить приоритет своих программ, чтобы растолкать локтями тех, кто этого не сделал. Поэтому все программы в автозагрузке поставили в равные условия.

Понижение приоритета CPU и I/O для программ в автозагрузке

Все программы в автозагрузке Windows Vista поместили в «коробку» (“box”, в терминологии Performance Team, отвечающей за производительность системы).

На практике это означало, что по умолчанию на протяжении 60 секунд после запуска системы для всех приложений в автозагрузке задавался приоритет:



- **CPU ниже обычного** (below normal), чтобы уменьшить нагрузку на процессор
- **I/O очень низкий** (very low), чтобы снизить интенсивность обращений к диску

Таким образом, каждая программа в автозагрузке меньше нагружала систему во время ее запуска. А по истечении заданного периода времени приоритет программы восстанавливался на уровень, определенный ее создателями.

Чем кончилась война

Microsoft закопала топор войны за автозагрузку с выходом Windows 7. Трудно сказать, что послужило основной причиной. Возможно, было много нареканий от корпоративных клиентов, недовольных медленным выполнением скриптов групповой политики.

С другой стороны, за три года, прошедших с выпуска Windows Vista, улучшились аппаратные конфигурации компьютеров на рынке. Появились новые, более мощные процессоры, а твердотельные накопители стали проникать в массы. Это снизило полезный эффект от помещения в коробку автоматически запускаемых программ.

Так или иначе, в Windows 7 и Windows 8 приоритет CPU и I/O для программ в автозагрузке не понижается, а возможность увеличить приоритет потока не блокируется. Однако топор войны не утоплен, а именно закопан! И ниже я расскажу, как его выкопать.

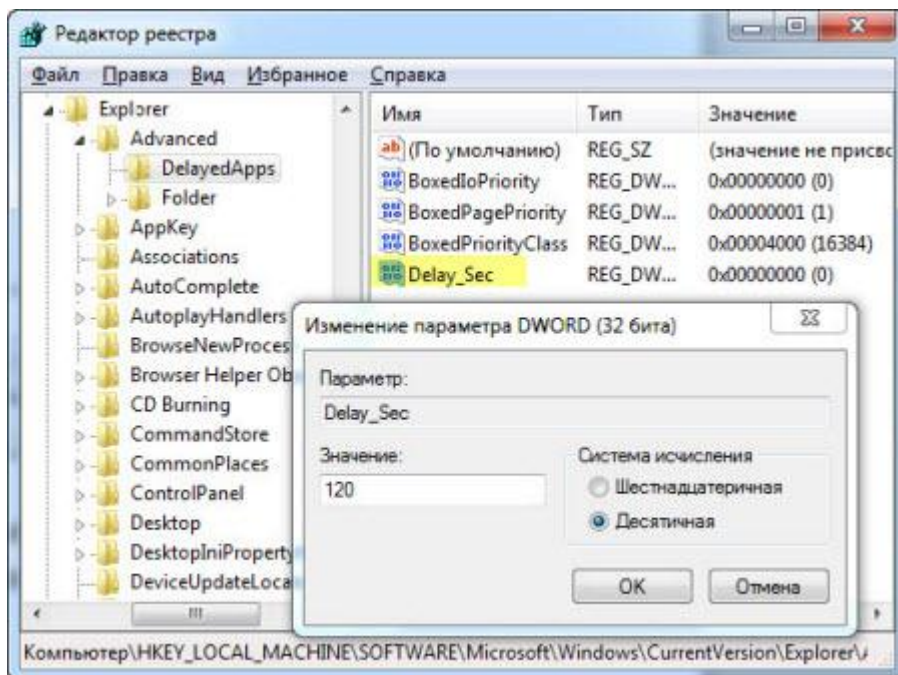
Твик, изменяющий приоритет CPU и I/O для программ в автозагрузке

Описанные выше возможности остались в Windows 7 и Windows 8. Их просто отключили, изменив низкоприоритетный период с **60** секунд до **0**. Вы можете установить любой интервал по своему усмотрению!

1. Перейдите в раздел реестра:

```
HKKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\Explorer\Advanced\Delayed Apps
```

2. [Станьте владельцем этого раздела](#) (по умолчанию им владеет системная учетная запись TrustedInstaller).
3. Задайте желаемое значение для параметра **Delay_Sec** в секундах (в десятичном формате). Например, установите 120 секунд.



4. Верните исходного владельца раздела реестра, как описано по ссылке в шаге 2.

Вот и все! Теперь на протяжении заданного интервала времени программы из автозагрузки будут иметь низкий приоритет CPU и I/O.

Как проверить приоритет запущенных программ

С помощью утилиты [Process Explorer](#) вы можете быстро посмотреть, с каким приоритетом запущены процессы.

1. Щелкните правой кнопкой мыши на любом столбце и выберите **Select Columns**.
2. На вкладке **Process Performance** установите флажок **Base Priority** (в списке процессов заголовок столбца называется **Priority**).
3. На вкладке **Process I/O** установите флажок **Priority**.

На рисунке ниже вы видите программы, отсортированные по приоритету CPU.

Process	PID	Priority	I/O Priority
System Idle Process	0	0	
AcroRd32.exe	2144	6	Very Low
chrome.exe	752	6	Very Low
EXCEL.EXE	1432	6	Very Low
firefox.exe	2056	6	Very Low
ieexplore.exe	2064	6	Very Low
ieexplore.exe	3268	6	Very Low
msseces.exe	1620	6	Very Low
notepad.exe	2176	6	Very Low
RtHDVCpl.exe	1984	6	Very Low
Skype.exe	2112	6	Very Low
Syn Toshiba.exe	316	6	Very Low
Syn TPEnh.exe	1484	6	Very Low
WINWORD.EXE	2100	6	Very Low
wmpayer.exe	2124	6	Very Low
amsvc.exe	1720	8	
audiodg.exe	1112	8	Normal
explorer.exe	1524	8	Normal
FlashUtil10v_ActiveX.exe	2020	8	Normal

CPU Usage: 22.55% Commit Charge: 25.88% Processes: 51 Physical Use

У процессов от AcroRd32.exe до wmpayer.exe:

- приоритет I/O очень низкий (**Very Low**)
- приоритет CPU ниже обычного (**6**)

Обычный приоритет CPU имеет значение 8, в чем можно убедиться, щелкнув правой кнопкой мыши по процессу и выбрав в меню пункт **Set Priority**.

По истечении заданного интервала времени вы увидите, что приоритет процессов вернулся в нормальное русло.

Тест

Материал был бы неполным без подтверждения теории практикой. К счастью, у меня все системы относительно быстрые (так, во всех в качестве системного диска используется SSD), а в автозагрузке чистота и порядок.

Самым слабым оказался ноутбук мамы, где установлен процессор Core 2 Duo и жесткий диск 7200 rpm. В автозагрузке находились:

- Google Updater
- драйверы Synaptics и Realtek, а также какая-то утилита Toshiba
- графический интерфейс антивируса Microsoft Security Essentials

Поэтому перегруженный автозапуск программ тоже пришлось эмулировать, и я добавил туда:

- браузеры IE, Chrome и Firefox
- Microsoft Office Word и Excel
- Adobe Reader, Windows Media Player и Skype



Поскольку состав программ в автозапуске изменился, нужно было обеспечить чистоту эксперимента:

1. Сделать три контрольных перезагрузки [для тренировки ReadyBoot](#).
2. [Дефрагментировать загрузочные файлы](#).

Затем я выполнил по 5 перезагрузок системы в обычном режиме и с включенным ограничением приоритета (время в миллисекундах взято из [события 100](#)).

Номер запуска	Обычная автозагрузка	Автозагрузка с ограниченным приоритетом
1	66754	62025
2	64380	59823
3	69242	63399
4	60904	59787
5	72725	60839
Среднее	66801	61175

Как видите, среднее время пяти загрузок составило **66,8** секунды при обычной автозагрузке против **61,2** секунды при ограниченном приоритете программ в автозапуске. Другими словами, в протестированной конфигурации железа и программ **среднее время готовности рабочего стола сократилось на 9%**, благодаря внесенному в реестр изменению.

Если вы уже применили к своей системе советы из предыдущих глав, ваша система должна загружаться быстрее. Однако достижению оптимального результата могут мешать скрытые проблемы запуска драйверов, служб и приложений. Журнал событий может пролить свет на проблему, но далеко не всегда.

Следующая глава познакомит вас с Windows Performance Toolkit – профессиональным набором для диагностики загрузки Windows. Он дает возможность изучить процесс загрузки системы с потрясающим уровнем детализации.

Этапы загрузки Windows под микроскопом Windows Performance Toolkit

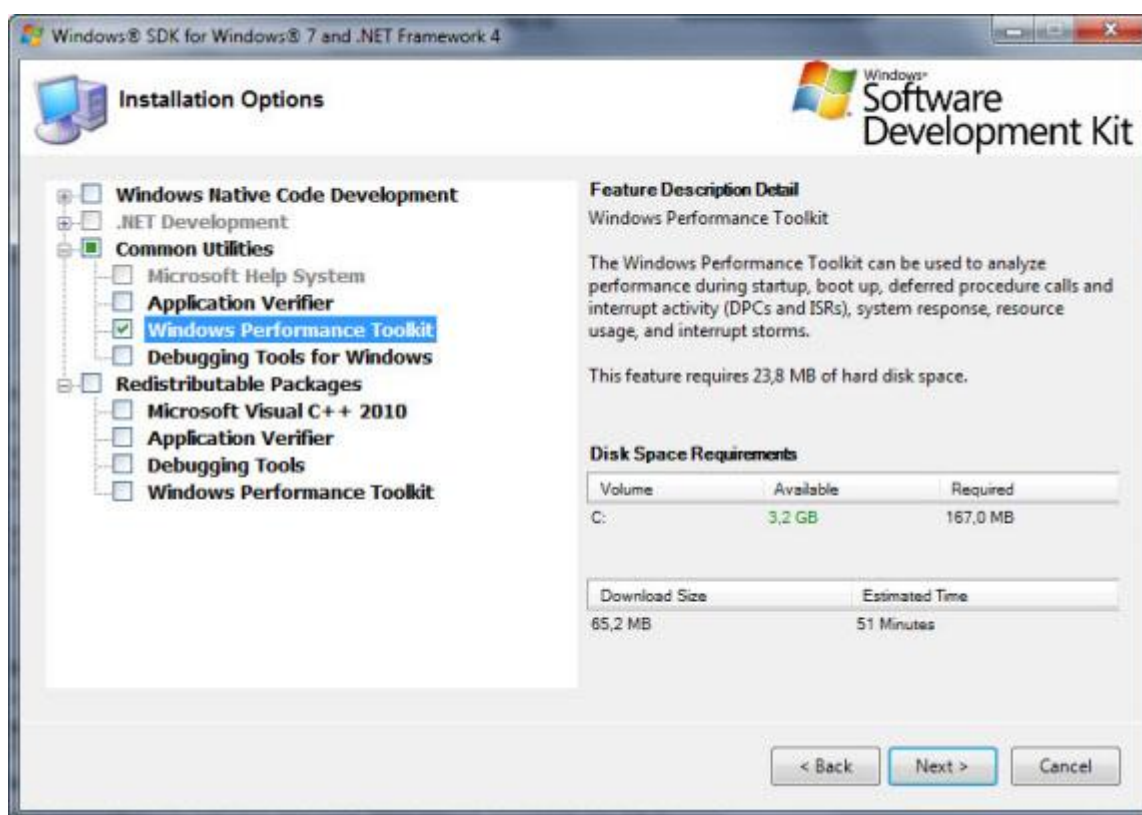


Составить полное представление о загрузке Windows 7 и Vista можно с помощью набора Windows Performance Toolkit. Утилиты командной строки **xbootmgr** и **xperf** позволяют создать подробный отчет о запуске системы и представить его в графическом и текстовом виде.

Я уверен, что после прочтения предыдущих глав и применения полученных знаний на практике ваша система стала загружаться быстрее. Однако эти простые способы не позволяют выявить скрытые факторы или проблемы, замедляющие загрузку Windows. Теперь настало время познакомиться поближе со всеми этапами загрузки и провести их детальный анализ с помощью Windows Performance Toolkit (WPT).

Загрузка и установка WPT

Поскольку набор Windows Performance Toolkit 4.7 для Windows 7 и Vista недоступен для отдельной загрузки, этому вопросу придется уделить чуть больше внимания, чем обычно. Самый простой и экономичный с точки зрения трафика способ – это скачать его с помощью [веб-установщика Windows 7 SDK](#). Пройдя все приветственные окна установщика, вы увидите страницу с выбором компонентов.



Здесь нужно снять все флажки кроме одного, показанного на рисунке и продолжить установку.

Вы, наверное, обратили внимание, что WPT присутствует в двух узлах. Если вам нужны пакеты MSI, их можно скачать, отметив флажок в узле **Redistributable Packages**. После установки вы найдете пакеты для различных архитектур в папке Program Files\Microsoft SDKs\Windows\v7.1\Redist. Если вы выбрали только этот вариант (*вместо* рекомендуемого выше), запустите установку двойным щелчком по MSI-пакету.

Подготовка к работе

Следуя трем простым правилам, вы застрахуете себя от возможных проблем, обеспечите правильную работу всех команд и точно измерите длительность загрузки.

1. Прежде чем выполнить первую команду, [создайте точку восстановления системы](#) и убедитесь, что у вас есть под рукой установочный диск / флэшка или [диск восстановления](#). Предупреждение вовсе не дежурное, ибо случаи неадекватного поведения WPT были отмечены у нас на форуме, да и сам я их видел.
2. [Включите автоматический вход в систему](#), чтобы задержка на ввод пароля не влияла на измерения.

3. Все команды выполняйте в командной строке, [запущенной от имени администратора](#). Там же можно добавить в меню пункт для ее запуска в нужной папке – пригодится.

При анализе могут создаваться файлы большого размера, поэтому убедитесь, что на разделе есть несколько гигабайт свободного пространства. Все логи загрузки лучше хранить в одной папке, допустим, **C:\Trace**. Откройте командную строку с полными правами и введите:

```
md c:\Trace
```

Здесь и далее я буду использовать пути применительно к этой папке и стандартной установке WPT в 32-разрядной Windows 7. При необходимости изменяйте пути на свои.

Сбор данных

Закройте все программы и сохраните все документы. Процесс сбора данных **о загрузке системы** запускается одной командой:

```
xbootmgr -trace boot -traceFlags BASE+CSWITCH+DRIVERS+POWER -resultPath C:\Trace
```

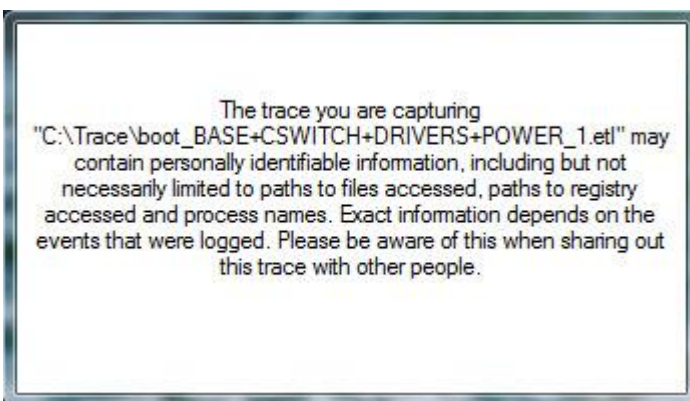
Аналогичные команды можно использовать для диагностики **гибернации**:

```
xbootmgr -trace hibernate -traceFlags BASE+CSWITCH+DRIVERS+POWER -resultPath C:\Trace
```

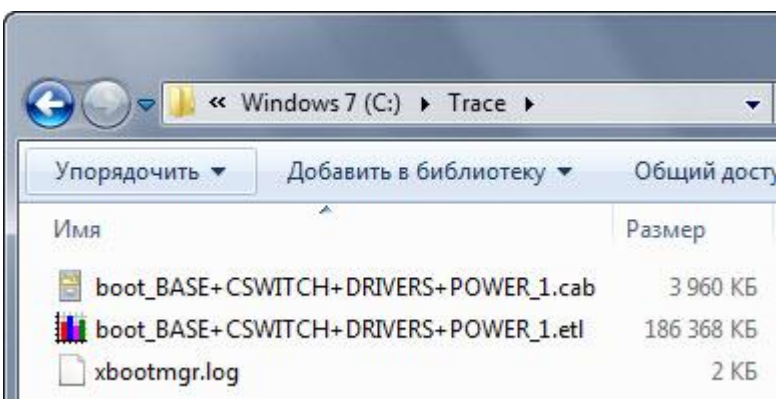
и сна:

```
xbootmgr -trace standby -traceFlags BASE+CSWITCH+DRIVERS+POWER -resultPath C:\Trace
```

Вернемся к загрузке, однако. Компьютер будет перезагружен. Если после входа в систему вы увидите запрос UAC от xbootmgr, разрешите утилите продолжить работу. Через две минуты вы увидите примерно такое окно.



Когда оно исчезнет, в папке **C:\Trace** должно быть три файла, как показано на рисунке ниже.



Если вы вместо файла **boot_BASE+CSWITCH+DRIVERS+POWER_1.etl** видите там два других файла с расширением ETL, это может означать, что утилита еще работает, над их объединением в один – подождите несколько минут. При отсутствии изменений выполните в командной строке

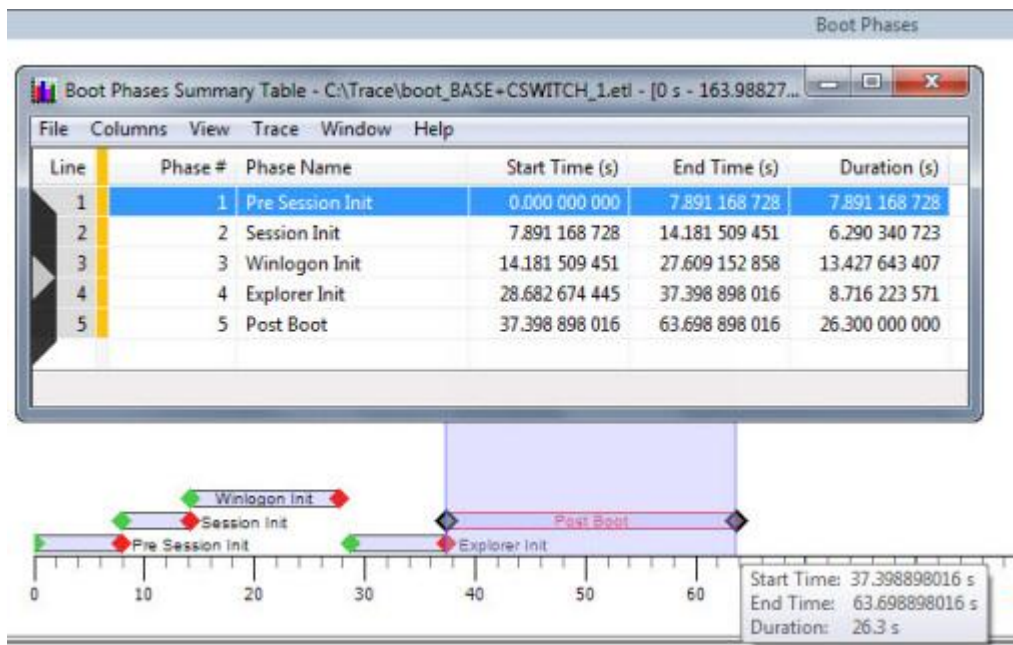
```
xperf -stop
```

и перезагрузите систему. После чего попробуйте заново запустить сбор данных.

Файлы, используемые для анализа

Я думаю, что вы уже успели дважды щелкнуть файл **boot_BASE+CSWITCH+DRIVERS+POWER_1.etl** и полюбоваться красивыми графиками и диаграммами. В левой панели графики можно отображать и скрывать, а также переходить к ним двойным щелчком мыши.

График **Boot Phases** отражает длительность основных этапов загрузки, которые мы будем рассматривать подробнее дальше. На нем видно, что последний этап, **Post Boot** занял 26 секунд (Duration), а общее время загрузки составило 64 секунды (End Time).



Для определения длительности основных этапов загрузки можно выделять их мышью, как показано в нижней части рисунка. Можно также щелкнуть на графике правой кнопкой мыши и выбрать из меню пункт **Summary Table**, чтобы получить отчет в табличном режиме (верхняя часть рисунка).

Для удаленной диагностики по почте или в форуме можно создать текстовый отчет в виде XML-файла. Выполните команды:

```
cd c:\trace  
xperf /t ti -i boot_BASE+CSWITCH+DRIVERS+POWER_1.etl -o summary_boot.xml -a boot
```

Первая переходит в папку с логами, а вторая - создает требуемый XML-файл. Для его просмотра отлично подойдет Internet Explorer!

```
- <results timeFormat="msec">  
- <boot>  
+ <processSummary numProcesses="94" numUnexpectedLonglived="60"  
  numUnexpectedShortlived="23" numUnexpectedVeryShortlived="9">  
- <timing bootDoneViaExplorer="35431" bootDoneViaPostBoot="106431"  
  osLoaderDuration="2962" postBootRequiredIdleTime="10000" postBootDisturbance="61000"  
  pnpBootStartStartTime="45" pnpBootStartEndTime="3916" pnpBootStartDuration="3871"  
  pnpSystemStartStartTime="6073" pnpSystemStartEndTime="6797"  
  pnpSystemStartDuration="724">  
+ <interval name="PreSMSS" startTime="0" endTime="6884" duration="6884">  
+ <interval name="SMSSInit" startTime="6884" endTime="13191" duration="6306">  
+ <interval name="WinlogonInit" startTime="13191" endTime="23662" duration="10471">  
+ <interval name="ExplorerInit" startTime="23662" endTime="35431" duration="11769">  
+ <interval name="PostExplorerPeriod" startTime="35431" endTime="106431"  
  duration="71000">  
+ <interval name="TraceTail" startTime="106431" endTime="187060" duration="80629">  
</timing>  
+ <services autoStartStartTime="14336" autoStartEndTime="48732"  
  autoStartDuration="34395">  
+ <pnp>  
<groupPolicy />  
</boot>  
</results>
```

Сложите узлы, как показано на рисунке, чтобы лучше видеть общую картину. В узле **timing** указано время в миллисекундах, и там можно увидеть длительность двух больших, условно говоря, частей загрузки (выделены **зеленым**, а также показаны выше в графическом представлении):

- **bootDoneViaExplorer** – время загрузки операционной системы вплоть до появления рабочего стола, в этом примере 37 секунд
- **bootDoneViaPostBoot** – полное время загрузки системы, рабочего стола и всех программ в автозагрузке, в этом примере 64 секунды (из этой цифры следует вычитать 10 секунд, в течение которых определяется полное бездействие системы)

Время первой части складывается из основных этапов загрузки операционной системы (обведены **синим** на рисунке выше), вплоть до начала загрузки рабочего стола. В уже знакомом вам событии **100** [журнала Diagnostics-Performance](#) длительность этого этапа записывается в параметре **MainPathBootTime**.

Разница между этими двумя частями – это время от начала загрузки рабочего стола, до его полной готовности. В событии **100** – это **BootPostBootTime**.

Этапы загрузки Windows и их диагностика

Для анализа загрузки нужно представлять, не только в какой последовательности эти этапы идут, но и что происходит на каждом из них. К сожалению, официальная документация не дает ответа на этот вопрос, но мне удалось найти на MSFN [замечательное руководство](#) на английском, которое содержит необходимую и достаточную информацию по этому вопросу. Далее я предлагаю вам его в своем изложении, с дополнениями и в сопровождении собственных примеров диагностики.

На рисунке ниже представлены три основных этапа загрузки, причем главный из них состоит из четырех фаз.



Давайте рассмотрим все этапы подробно.

Этап OSLoader

Этап **OSLoader** следует сразу после инициализации BIOS. Визуально он начинается после заставки и диагностических экранов BIOS, а заканчивается примерно с появлением экрана «Запуск Windows».

На этапе **OSLoader**:

- загрузчик Windows (**winload.exe**) загружает основные системные драйверы, которые необходимы для считывания минимально необходимого набора данных с диска
- затем загрузчик инициализирует систему до момента, с которого становится возможной загрузка ядра
- когда ядро начинает загружаться, **winload.exe** помещает в оперативную память системный раздел реестра и дополнительные драйверы, помеченные в качестве BOOT_START

Длительность этапа отражает значение параметра **osLoaderDuration** в узле **timing** XML-файла. Обычно, она находится в пределах 2-3 секунд.

Этап MainPathBoot

Визуально этап **MainPathBoot** начинается с экрана «Загрузка Windows».



Он завершается при появлении рабочего стола. Если не настроен автоматический вход в систему, длительность этого этапа увеличивается за счет времени, которое требуется для ввода пароля.

Во время этапа **MainPathBoot** происходит основная работа по загрузке операционной системы:

- инициализируется ядро
- происходит определение устройств Plug and Play (PnP)
- запускаются службы
- выполняется вход в систему
- инициализируется Explorer, т.е. система готовится к загрузке рабочего стола

Этап состоит из четырех фаз, каждая из которых обладает собственными характеристиками и может по-своему влиять на длительность загрузки системы.

Фаза PreSMSS

Визуально фаза **PreSMSS** начинается примерно с экрана «Загрузка Windows», но ее окончание невозможно определить на глаз.

Фаза **PreSMSS** (в графическом представлении WPT она обозначена как **Pre Session Init**) начинается с инициализации ядра. Во время нее:

- ядро инициализирует структуры данных и компоненты, а затем запускает диспетчер PnP
- диспетчер PnP в свою очередь инициализирует драйверы **BOOT_START**, которые были загружены с помощью **winload.exe** на этапе **OSLoader**
- когда диспетчер PnP обнаруживает устройство, он загружает необходимый драйвер и выполняет его инициализацию

Диагностика

Если фаза занимает много времени, ищите в XML-файле в узле **<PNP>** драйвер, который долго загружается. Диагностику в графическом режиме я покажу на примере следующей фазы.

Фаза SMSSInit

Визуально начало фазы **SMSSInit** невозможно определить. Ее частью является пустой экран, который отображается между заставкой и экраном входа в систему, чье появление сигнализирует о завершении фазы.

Фаза **SMSSInit** (в графическом представлении WPT она обозначена как **Session Init**) начинается с того, что ядро передает контроль диспетчеру сессий (**smss.exe**). Во время этой фазы система:

- инициализирует реестр
- загружает и запускает устройства и вторую волну драйверов, которые не помечены как **BOOT_START**

- запускает процессы подсистемы

Фаза завершается с передачей контроля процессу **winlogon.exe**.

Диагностика

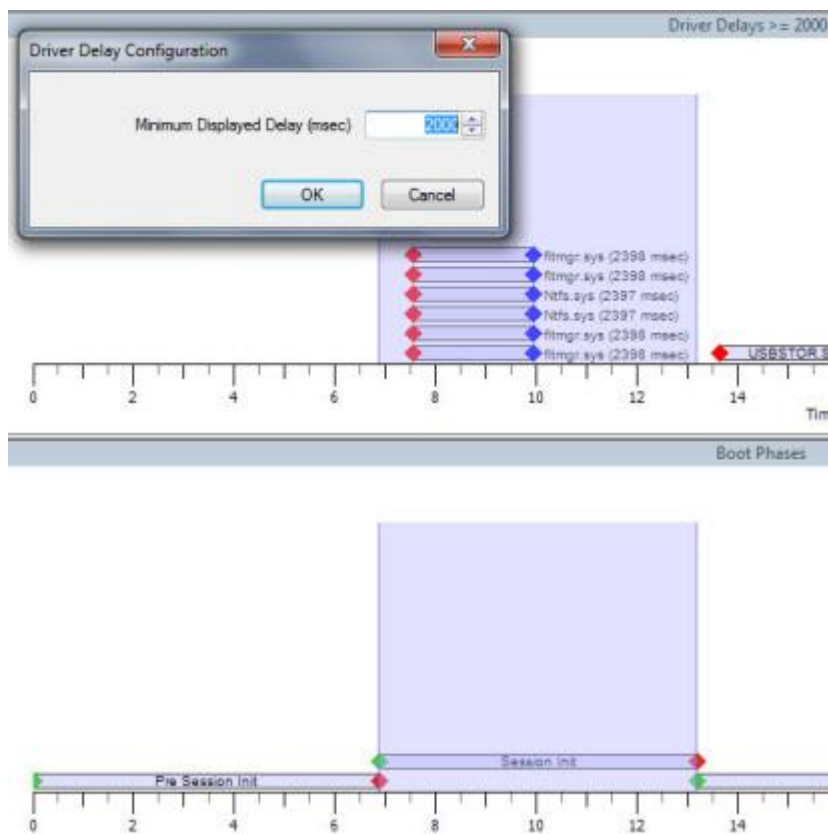
Наиболее распространенной причиной задержек в этой фазе является драйвер видеокарты. Он инициализируется сначала во время системной сессии, а затем во время пользовательской. При этом инициализация во время пользовательской сессии занимает меньше времени, потому что в течение системной параллельно выполняется запуск других задач.



Сократив время запуска драйвера видеокарты, можно уменьшить длительность загрузки системы. Таким образом, если фаза **SMSSInit** затягивается, попробуйте обновить драйвер видеокарты.

Более точную диагностику можно провести с помощью графиков **Boot Phases** и **Driver Delays** примерно так:

1. На графике **Boot Phases** выделите фазу **Session Init** и выберите из контекстного меню команду **Clone Selection**. Выбранный период будет выделен на всех активных графиках.
2. На графике **Driver Delays** щелкните правой кнопкой мыши и выберите из меню команду **Set Delay Threshold**. Она позволяет отфильтровать драйверы по времени задержки. Введите, например 2000, чтобы отобразить драйверы, загружавшиеся дольше двух секунд.



Вы увидите все драйверы, загружавшиеся в фазе **Session Init** дольше заданного времени. У меня вся фаза занимает 6 секунд, и двухсекундная задержка драйверов является нормальной. Но если у вас проблемы в этой фазе, с помощью фильтра вы сразу увидите, какой драйвер их вызывает.

Фаза WinLogonInit

Визуально фаза **WinLogonInit** начинается перед появлением экрана приветствия, а завершается перед появлением рабочего стола.

Фаза **WinLogonInit** начинается сразу после запуска **winlogon.exe**. Во время этой фазы:

- отображается экран приветствия
- диспетчер управления службами запускает сервисы
- происходит запуск сценариев групповой политики

Фаза завершается запуском оболочки Windows - процесса **explorer.exe**.

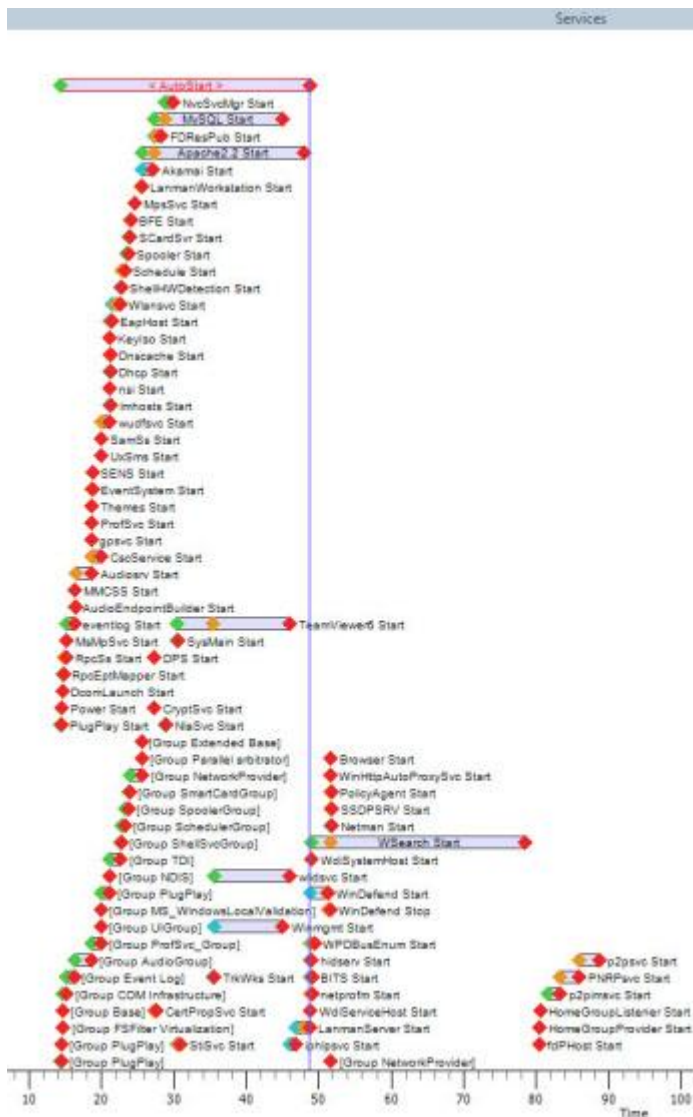
Диагностика



Во время фазы **WinLogonInit** выполняется множество параллельных операций. На многих системах она характеризуется нагрузкой на процессор и большим количеством операций ввода-вывода (I/O). Длительность фазы во многом зависит от поведения служб.

Чтобы обеспечить плавную загрузку системы, службы могут объявлять зависимости или использовать порядковые группы загрузки. Windows обрабатывает группы загрузки в последовательном порядке. Поэтому задержка даже одной службы в ранней группе может затягивать загрузку следующей группы служб и тормозить весь процесс загрузки.

Для выявления проблемной службы удобнее всего использовать графические возможности WPT. Откройте ETL-файл двойным щелчком мыши и прокрутите отчеты вниз до графика запуска служб.

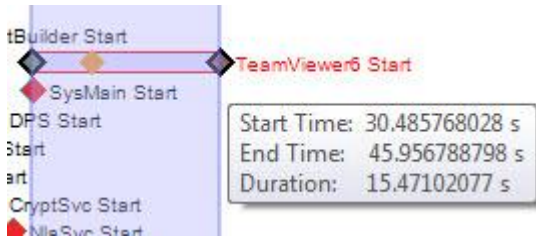


Зачастую проблема вызвана не системными, а сторонними службами. На рисунке хорошо видно, что среди автоматически стартующих служб дольше всего загружаются три:

- Apache 2.2
- MySQL
- TeamViewer

При этом **Apache** блокирует загрузку следующей группы служб (очевидно, в ее отсутствие это сделала бы служба **TeamViewer**). Поскольку ни одна из этих служб не является системной, проблему легко решить. Можно в оснастке «Службы» изменить тип ее запуска на отложенный и посмотреть, будет ли она быстрее запускаться на более позднем этапе. Если это не дает эффекта, можно вовсе отключить службу и запускать ее вручную при необходимости. Во второй волне служб, имеющих отложенный тип запуска, видна задержка **WSearch**, отвечающей за поиск Windows, но я не стал ее трогать пока.

Чтобы увидеть время запуска каждой службы, щелкните точку начала запуска и растяните диапазон до ее конца. Для изменения масштаба графика крутите колесо мыши, удерживая нажатой клавишу **CTRL**.



Отключение трех вышеперечисленных служб позволило сократить общее время загрузки почти на 40 секунд! Обратите внимание, что группа автоматического запуска служб теперь стартовала намного быстрее (смотреть нужно относительно шкалы времени, т.к. масштаб графиков разный).



Wsearch все равно запускается дольше других служб, но уже всего 8 секунд вместо 30, что не дает мне достаточно оснований к ней придраться.

Если задержку вызывает антивирусная программа, отложенный запуск службы может понизить уровень защиты, а ручной запуск или отключение службы могут нарушить работу программы. В этом случае можно лишь посоветовать обновить антивирус до последней версии. Если это не дает эффекта, вам придется сделать выбор между любимой программой и длительностью загрузки.

Фаза ExplorerInit

Визуально фаза **ExplorerInit** начинается перед загрузкой рабочего стола, но ее окончание определить на глаз невозможно.

В фазе **ExplorerInit**:

- сначала запускается процесс **explorer.exe**
- затем система создает процесс диспетчера окон рабочего стола (DWM)
- DWM инициализирует рабочий стол и отображает его

Инициализация DWM и рабочего стола происходит на переднем плане, но в это же время в фоне диспетчер управления службами (SCM) запускает службы, а диспетчер памяти кеширует данные. Поэтому на многих системах эта фаза сопровождается нагрузкой на процессор, и нередко задержки при загрузке на этом этапе можно отнести на счет слабости аппаратных ресурсов.

Диагностика

В течение фазы **ExplorerInit** ресурсы процессора могут потреблять программы, работающие в качестве служб (например, защитные программы или серверы приложений). Они запускаются либо в этой фазе, либо продолжают свою загрузку, будучи запущенными в более ранних фазах. С другой стороны, некоторые службы (например, с отложенным запуском) могут быть еще не запущены на момент окончания фазы **ExplorerInit**.

Этап PostBoot

Этап **PostBoot** начинается после появления рабочего стола и завершается после того, как будет определено бездействие системы.



На этапе **PostBoot** рабочий стол уже загружен, и с ним можно взаимодействовать. Но при этом параллельно в фоне выполняется различная активность. Например, продолжается запуск служб и программ автозагрузки, что может сопровождаться появлением их значков в области уведомлений.

Средства WPT определяют бездействие системы по следующему алгоритму. Каждые 100 мс проверяется наличие активности в системе. Если бездействие системы составляет не менее 80% (за исключением низкоприоритетных процессов и дисковой активности), считается, что в этом интервале система бездействует. Проверка продолжается до тех пор, пока не наберется 10 секунд бездействия. Поэтому, определяя общее время загрузки системы, вычитайте из значения **bootDoneViaPostBoot** 10000 мс, т.е. 10 секунд.

Диагностика

На этапе **PostBoot** запускаются приложения, находящиеся в автозагрузке. Чтобы сократить длительность этого этапа, нужно [навести там порядок](#). В графическом представлении WPT используйте график **Process Lifetimes**, чтобы увидеть все процессы, которые запускаются или продолжают запуск на данном этапе.

Безусловно, диагностика загрузки с помощью WPT требует навыка, и с наскоку разобраться в этом вопросе непросто. Но от вас и не требуется профессиональных знаний, поскольку текстовый отчет в XML файле вкупе с полным графическим представлением всех этапов загрузки позволяет быстро определить причину задержек при запуске Windows.

WPT можно использовать не только для диагностики загрузки, но и для ее оптимизации. В заключительной главе книги вы найдете рассказ о том, как натренировать ReadyBoot и выжать из системы максимум.

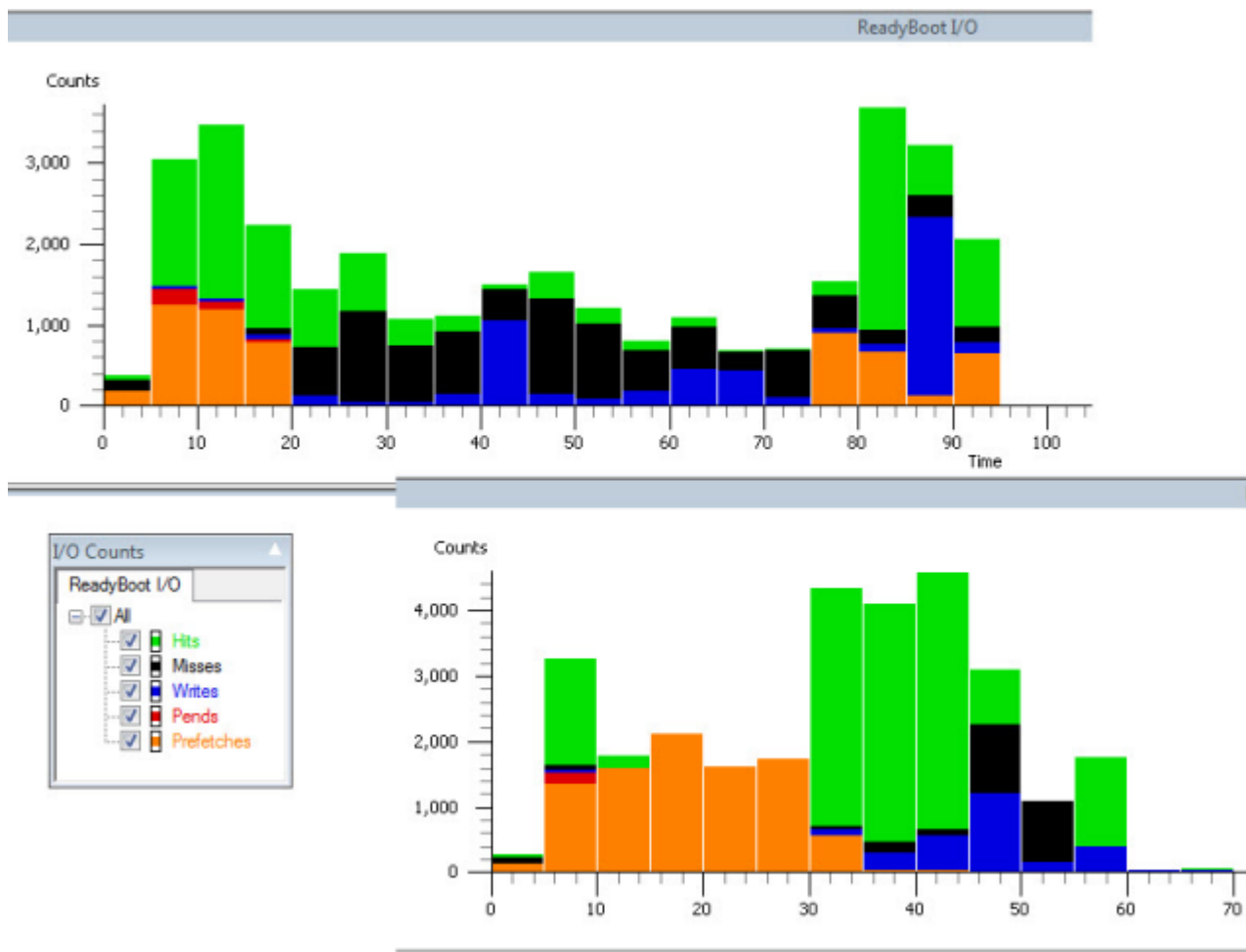
Оптимизация ReadyBoot с помощью Windows Performance Toolkit

Спортсмены много и упорно тренируются, чтобы достичь высоких результатов. При загрузке с жесткого диска скорость зависит от того, насколько оптимизирована работа функции ReadyBoot. Ее нужно потренировать, чтобы Windows 7 загружалась быстрее.

Я не случайно упомянул жесткий диск, поскольку ReadyBoot не используется на твердотельных накопителях (SSD). Это руководство применимо только к жестким дискам, **не делайте этого на SSD**. А вот на жестких дисках тренировка ReadyBoot оказывается весьма действенной. Если вы последовали моим [трем простым советам по ускорению загрузки](#), вы уже потренировали ReadyBoot. С помощью Windows Performance Toolkit это легко проверить.

Эффективность простых советов на графике активности ReadyBoot

Сначала я отследил загрузку с помощью **xbootmgr** и увидел, что ее можно ускорить за счет оптимизации ReadyBoot. После применения трех советов к своей системе я создал еще один отчет и сравнил графики **ReadyBoot I/O**.



На рисунке показана эффективность ReadyBoot до и после применения простых советов. Зеленым цветом обозначены попадания, т.е. когда из кэша ReadyBoot загружались нужные данные, а черным – промахи. Масштаб графиков немного отличается, поэтому сопоставлять их надо по шкале времени.

Сокращение промахов и увеличение попаданий ReadyBoot положительно сказалось на скорости загрузки – с 97 секунд оно сократилось до 69.

Как видите, даже без специальных программ можно добиться отличных результатов.

Ловкость рук и никакого мошенничества

WPT появился не на пустом месте – в Windows XP ему предшествовала утилита BootVis, которая зачастую подавалась под соусом магического ускорения загрузки. Это «чудо» тиражировалось по всему Интернету и, в конце концов, Microsoft это надоело. Утилиту убрали из свободного доступа, но ее до сих пор можно найти в каталогах программ.

Я привел эти графики не только в подтверждение действенности моих советов. Нужно понимать, что метод тренировки ReadyBoot, который я продемонстрирую ниже, не является секретным или магическим. Он основан на тех же принципах, просто процесс форсируется с помощью WPT.

Тренировка занимает некоторое время – в ее ходе производится шесть перезагрузок, и после каждой создается отчет, плюс однажды выполняется дефрагментация загрузочных файлов. У меня все это заняло около часа. Учитывая, что я пользуюсь гибернацией и в среднем перезагружаюсь два раза в неделю, особой практической выгоды я из этого не извлек. Да, мне удалось сбросить еще 6-7 секунд, но по сравнению с затратами на тренировку это мизер. Конечно, если вы запустите тренировку на системе, еще не оптимизированной простыми советами, выигрыш в скорости будет более значительный.



Эффект от оптимизации ReadyBoot не является постоянным. По мере работы системы и установки программ, набор служб и приложений в автозагрузке может изменяться, что сбивает прицел ReadyBoot. С другой стороны, использование стороннего дефрагментатора способно разом похоронить все достижения тренировки.

Поэтому я оставляю решение о том, стоит ли этим заниматься, целиком на ваше усмотрение. Впрочем, если вы хотите выжать из системы максимум и уже успели познакомиться с WPT, вам не понадобится долго вникать в ситуацию.

Запуск оптимизации

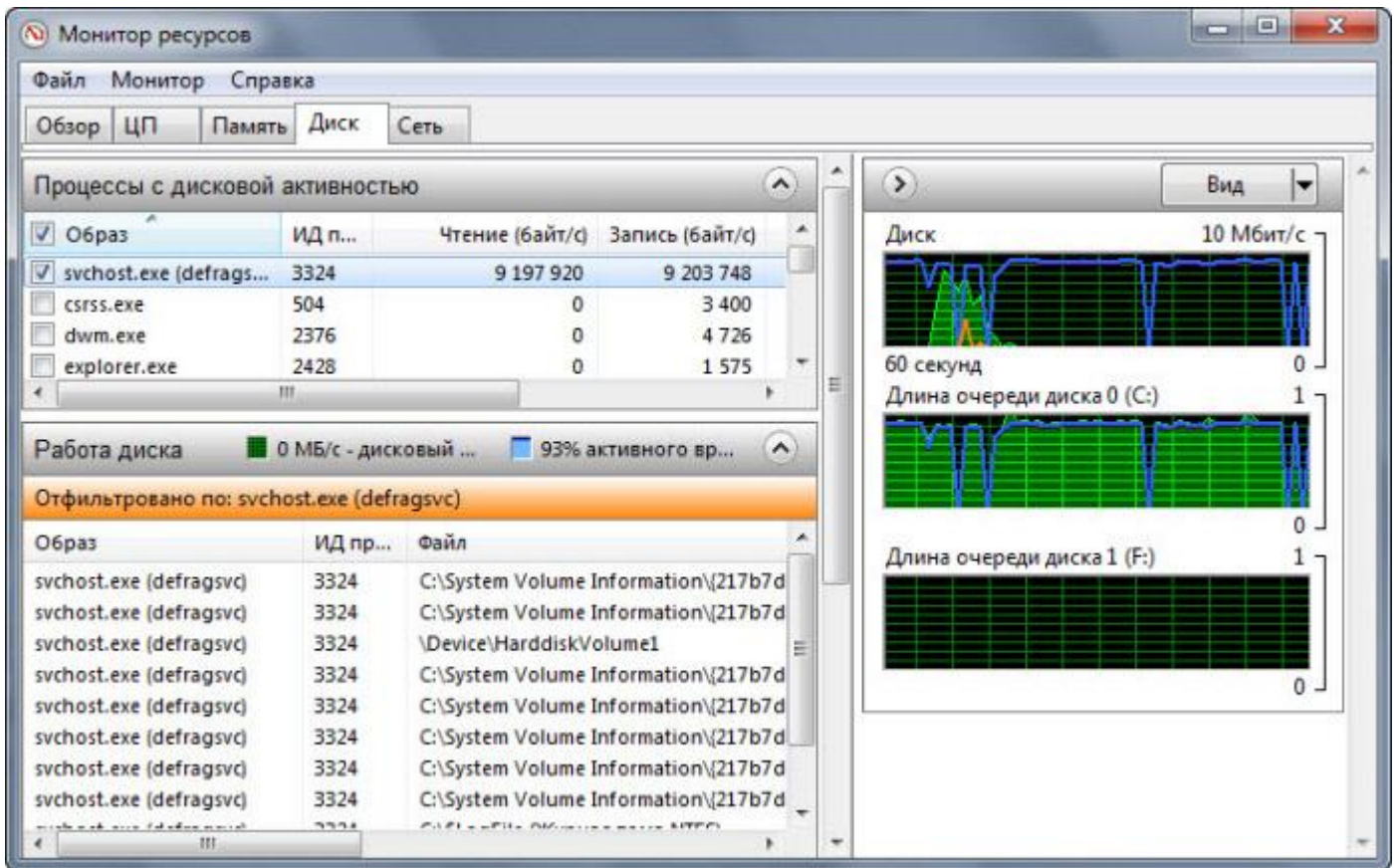
Перед тем как приступить к оптимизации ReadyBoot, [запустите мой диагностический пакет](#) и убедитесь, что у вас нет системных проблем. Иначе тренировка может сорваться или просто не будет иметь смысла. И я не буду повторяться о том, [что нужно сделать до начала работы с WPT](#).

Я также дам одну рекомендацию, которая мне очень несвойственна, но сэкономит вам время. Если вы отключите контроль учетных записей на период тренировки, вам не придется одобрять запрос на запуск **xbootmgr** при каждой перезагрузке. Только не забудьте потом включить UAC :)

Все чудо заключается в запуске с полными правами одной команды:

```
xbootmgr -trace boot -prepSystem -verboseReadyBoot -resultPath C:\Trace
```

Теперь можно идти варить кофе. Первая перезагрузка служит для сбора начальных данных, после второй выполняется дефрагментация диска. Монитор ресурсов иллюстрирует этот процесс очень хорошо.



Четыре следующие перезагрузки служат для тренировки ReadyBoot. По окончании процесса в папке C:\trace должны быть отчеты о действиях при каждой перезагрузке с именем и финальный отчет.

The screenshot shows a Windows Explorer window displaying the contents of the 'C:\Windows7\Trace' directory. The files are listed in a table with columns for Name, Size, and Date Modified.

Имя	Размер	Дата изменения
bootPrep_BASE+CSWITCH_1.etl	72 704 КБ	05.03.2011 13:36
bootPrep_BASE+CSWITCH_1.cab	1 820 КБ	05.03.2011 13:36
bootPrep_BASE+CSWITCH_2.etl	72 704 КБ	05.03.2011 13:40
bootPrep_BASE+CSWITCH_2.cab	3 169 КБ	05.03.2011 13:40
bootPrep_BASE+CSWITCH_3.etl	75 776 КБ	05.03.2011 14:28
bootPrep_BASE+CSWITCH_3.cab	4 535 КБ	05.03.2011 14:28
bootPrep_BASE+CSWITCH_4.etl	77 824 КБ	05.03.2011 14:32
bootPrep_BASE+CSWITCH_4.cab	5 945 КБ	05.03.2011 14:33
bootPrep_BASE+CSWITCH_5.etl	79 872 КБ	05.03.2011 14:36
bootPrep_BASE+CSWITCH_5.cab	7 163 КБ	05.03.2011 14:37
bootPrep_BASE+CSWITCH_6.etl	81 920 КБ	05.03.2011 14:41
bootPrep_BASE+CSWITCH_6.cab	8 064 КБ	05.03.2011 14:41
boot_BASE+CSWITCH_1.etl	82 944 КБ	05.03.2011 14:45
boot_BASE+CSWITCH_1.cab	8 618 КБ	05.03.2011 14:45
xbootmgr.log	13 КБ	05.03.2011 14:45

Результаты оптимизации

Чтобы сравнить выигрыш во времени от тренировки ReadyBoot, откройте файлы:

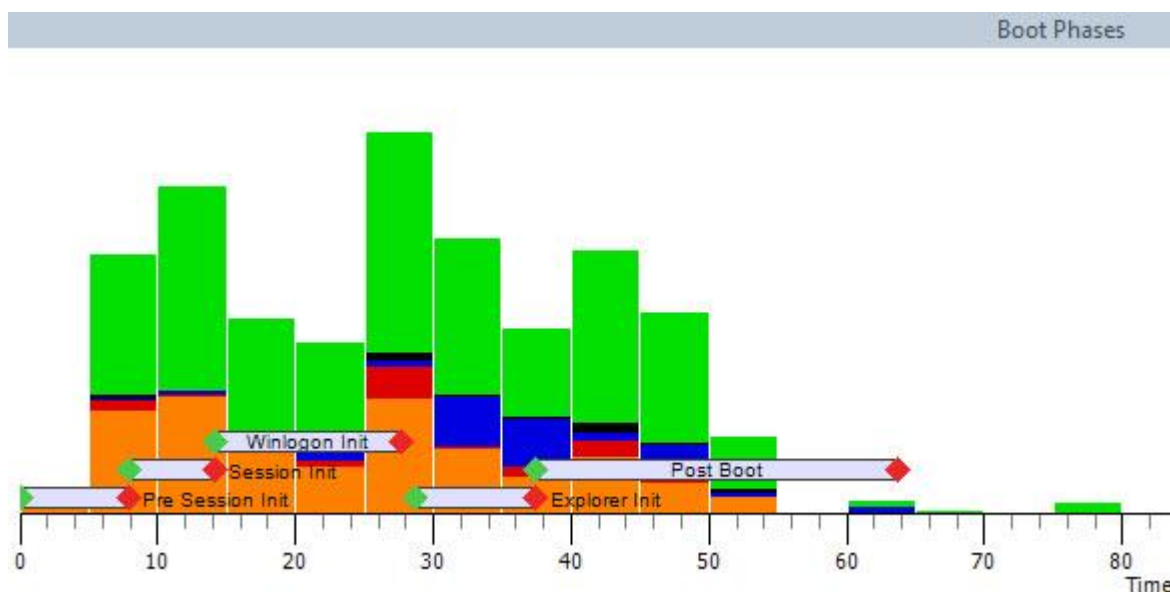
- **boot_BASE+CSWITCH_1.etl** (первый отчет до тренировки)
- **bootPrep_BASE+CSWITCH_1.etl** (финальный отчет после тренировки)

Длительность загрузки можно посмотреть на графике **Boot Phases**, о чем [я уже рассказывал](#) в предыдущей главе.

После тренировки ReadyBoot почти не промахивается. Наложив график ReadyBoot на этапы **Boot Phases**, я увидел интересную особенность.

Чтобы совместить графики, щелкните один из них правой кнопкой мыши и выберите из меню пункт **Overlay Graph**, затем нужный график.

Обратите внимание, что попадания ReadyBoot продолжаются уже после формального завершения загрузки (окончание этапа **PostBoot**).



В это время у меня уже начинают запускаться из планировщика первые отложенные программы, и они попадают в поле зрения ReadyBoot.

Сторонний дефрагментатор – друг или враг?

Как видите, оптимизация ReadyBoot с помощью WPT оказалась совсем несложным делом. Функция ReadyBoot будет поддерживать высокую скорость загрузки, когда у вас в порядке служба SuperFetch, а планировщик заданий обеспечивает дефрагментацию загрузочных файлов по расписанию силами Windows. Однако использование стороннего дефрагментатора сводит на нет все достижения тренировки, поскольку он раскладывает фрагменты файлов по-своему.



Его совместимость с ReadyBoot вы легко можете определить с помощью WPT – достаточно после нескольких полных дефрагментаций создать отчет и взглянуть на график **ReadyBoot I/O**. Если выяснится, что ReadyBoot у вас бьет мимо цели, стоит задуматься над тем, нужен ли сторонний дефрагментатор, не умеющий оптимизировать загрузку Windows.

В блоге я [приглашал читателей к эксперименту](#) по сравнению оптимизации загрузки разными дефрагментаторами, и [вот что получилось в итоге](#).

На этом рассказ о диагностике и ускорении загрузки завершен. Я надеюсь, вам удалось устранить проблемы, если они были, а также получилось значительно уменьшить время запуска Windows. Если вам понравилась книга и результаты применения моих советов, [напишите мне](#) – я буду рад услышать ваше мнение.

Список литературы

В процессе работы над книгой использовались следующие материалы:

- [Diagnostics-Performance log Event 100 - Critical, Error, or Warning - when and why?](http://social.technet.microsoft.com/Forums/en-US/w7itproperf/thread/48005f5d-5f66-439a-af51-3a2ebb894e31/#d14d81a8-2948-421c-a7ac-850c85343e10) [http://social.technet.microsoft.com/Forums/en-US/w7itproperf/thread/48005f5d-5f66-439a-af51-3a2ebb894e31/#d14d81a8-2948-421c-a7ac-850c85343e10]
- [Performance Testing Guide for Windows](http://www.microsoft.com/whdc/system/sysperf/Win7Perf.mspix) [http://www.microsoft.com/whdc/system/sysperf/Win7Perf.mspix]
- [Windows Performance Analyzer \(MSDN\)](http://msdn.microsoft.com/en-us/library/ff191077(VS.85).aspx) [http://msdn.microsoft.com/en-us/library/ff191077(VS.85).aspx]
- [Trace Windows 7 boot/shutdown/hibernate/standby/resume issues](http://www.msfm.org/board/topic/140247-trace-windows-7-bootshutdownhibernatestandbyresume-issues/) [http://www.msfm.org/board/topic/140247-trace-windows-7-bootshutdownhibernatestandbyresume-issues/]
- [Внутреннее устройство ядра Windows Vista: часть 2](http://technet.microsoft.com/ru-ru/magazine/2007.03.vistakernel.aspx?pr=blog) (Марк Руссинович) [http://technet.microsoft.com/ru-ru/magazine/2007.03.vistakernel.aspx?pr=blog]