Линух+Виндовс+* HOWTO

rgo amty@yandex.ru

Содержание

1 Введение

Значит так, версия этого документа 0.2. Поэтому я настоятельно рекомендую все утверждения тщательно проверять, особенно это относится к утверждениям насчёт GRUB. То что написано про лило, сислинух, winnt-loader, и вся "теоретическая" часть, почти наверняка не содержит оишбок. Но гарантировать я не могу. Поэтому: доверяйте но проверяйте. Если вы найдёте неточность, или вам кажется что здесь чего-то не хватает, пишите `amty@yandex.ru`.

Если вас абсолютно не интересует как и что делает загрузчик, вы можете срезу перейти к 3.

2 Загрузка с харда

Жёсткий диск снаружи это такая коробочка. Довольно тяжёлая. Это потому что она — почти сплошное железо. Снаружи коробка, а внутри металлический диск[1]. Изнутри компьютера, хард выглядит как последовательнось секторов.

Но такая последовательность при ближайшем рассмотрении не очень удобна. И для того чтобы избавится от этих неудобств, умные люди придумали файловые системы, и таблицы разделов.

Старые системы работали не с одномерным массивом секторов, а с трехмерным, где каждый сектор адресовался тремя числами: номером цилиндра, номером головки, номером сектора. И благодаря всяким там ... которые гонятся за обратной совместимостью, до сих пор биос не может отделаться от такой адресации. Уже давно контроллер жёсткого диска сам решает как отображать линейные адреса секторов в трехмерные, но программа работающая с биосом вынуждена указывать ему трёхмерные адреса секторов :(. Но не надо плакать, современные биосы поддерживают packet адресацию которая позволяет избавиться от лимита 504M. То есть если вашему биосу меньше 5-7 лет то он всяко может дотянутся гораздо дальше.

2.1 Файловая система

Файловая система, это способ записи информации на носитель, который позволяет:

- именовать связанные куски хранящейся информации (файлы) символическим именем;
- добавлять новые файлы не затирая старых;
- удалять файлы;
- создавать файлы специального типа содержащие другие файлы (директории, их также обзывают папками или каталогами).

Наиболее распространённой файловой системой пожалуй является fat. Собственно распространённость является единственным бонусом использования этой fs: я не знаю оси которая не умела бы общаться с 12-ти и 16-ти битными версиями этой fs, и все оси с которыми я сталкивался последние лет пять, понимали 32-х битную версию. Её основным недостатком являются убогие атрибуты файлов. Из-за этого на неё невозможно поставить полностью функциональную *nix систему. Вот ещё насколько примеров файловых систем:

- ext2 — native файловая система линукса. Она специально ориентирована на работу в линукс, поддерживая все основные требования UNIX к файловой системе.
- iso9660 — файловая система, которая используется для хранения информации на CD-ROM носителях. Сама по себе очень убогая система, ибо при разработке, одним из основных требований была совместимость с DOS, то есть 8.3 имена (8 символов на имя и три на расширение), и опять же минимум атрибутов файлов. Отличием этой файловой системы от других, является проблематичность создания новых файлов, так как она ориентирована на использование

[1] Вы можете попробовать его разобрать и посмотреть. Но предупреждаю, я не знаю как собрать его обратно. Даже и не спрашивайте.

в read режиме. На данный момент существует масса расширений этой файловой системы, которые позволяют использовать более длинные имена, или снимают ограничение на степень вложенности директорий.

- tar — это на самом деле *nix команда, предназначенная изначально для сохранения дерева директорий/файлов на магнитную ленту. Всё действо tar заключается, по сути, в сериализации дерева директорий. Обычно используется вместе с каким-нибудь компрессором. Сейчас в качестве компрессоров широко используется gzip и bzip2.

- виртуальная файловая система. Это на самом деле не файловая система которая пишется в файл или раздел. Это уровень абстракции оси, который объединяет всякие разные файловые системы в нечто единое, сглаживая все различия. И, по большому счёту, ей здесь не место, но... Но я её сюда вписал :).

Большинство fs очень плохо относятся к изменению размеров, и подвержены фрагментации. Одни в большей степени другие в меньшей, но так или иначе все.

2.2 Таблицы разделов

Таблица разделов, это способ иметь несколько "логических" дисков, на одном физическом. И такое деление очень часто используется, для разных целей:

- Для облегчения задач администрирования. Например, админ системы может иметь три раздела: на одном основные системные файлы, без которых загрузка невозможна, на другом всё остальное необходимое для полноценного функционирования системы, и на третьем домашние директории пользователей. В такой ситуации, если с системой что-то случится (например, фс похерится), то больше вероятность что система останется бутабельной, и меньше придётся доставать из бэкапа.

- Для повышения производительности. Пример: в линуксе возможно создать отдельный раздел, со специальной файловой системой под названием swap, которая используется для сохранения/подгрузки страниц виртуальной памяти, в случае нехватки физической оперативки. Здесь специальная файловая система позволяет избежать борьбы демона kswapd (ответственного за все эти приколы) с кэшированием, которое используется для оптимизации обращений к жёсткому диску.

- Для установки нескольких осей на один писюк. Это собственно то ради чего я всё затеял. Как я уже говорил *nix на файловой системе fat — это кал. Винду вообще невозможно поставить на ext2fs. Более того подчастую навозможно сосуществование двух осей на одной и той же fs.

- Сюда можете вписать ещё два десятка полезностей таблицы разделов.

Таблиц разделов на самом деле до хрена. Поменьше пожалуй чем файловых систем, но... до хрена. На писюке традиционно используется таблица разделов доса. Но FreeBSD например использует хоть и похожую схему, но не полностью совместимую (поправте если наврал где-то, я бздю в глаза не видел — позор на мою седую голову). Идея всех таблиц разделов примерно такая: в начале диска (в первых секторах) записать всю управляющую инфу про то какие разделы на диске есть, где они начинаются и где кончаются. Я не буду вдаватся в подробности того как куда и что надо записать чтобы получить таблицу разделов (даже для обычного писюка, уж не говорю про Мас'овскую), но немного про терминологию сказать надо. Я буду говорить про досовую таблицу. Если кого-либо из любителей фри это задевает, я предлагаю переписать эту секцию. Хоть целиком. Я с радостью приму вашу помощь.

Раздел[2], логический диск[3], том[4] это всё одно и тоже (просто названия разных осей, соответственно линукс, дос и виндовс). Разделы бывают трёх типов:

[2] от *англ.* partition
[3] logical drive
[4] volume

1. primary[5] — это раздел описанный в таблице разделов (здесь немного путаница с терминологией — в данной ситуации под таблицей разделов я понимаю запись о разделах которая находится в самом начале диска). Таких разделов не может быть больше четырёх.

2. extended — это специального вида primary раздел, который содержит в себе пачку logical партиций.

3. logical — а это раздел, который расположен (и описан) в extended партиции. Не путать с логическим диском. Логический диск — это партиция которую видит дос.

В то время как количество primary разделов на харде жёстко ограничено, количество logical в extended ограничено только лимитами ядра системы, и размером харда...[6]. Поскольку каждая logical партиция предваряется записью в которой она полностью описывается, и кроме того в этой записи хранится указатель на следующую logical партицию.

Первый сектор каждого раздела, содержит загрузочную запись. Загрузочная запись это специальная программа которая может быть использована для загрузки оси (ну или для того чтобы спрятать вирус). Есть еще одна загрузочная запись: MBR (Master Boot Record) — главная загрузочная запись. Эта запись располагается в самом первом секторе жесткого диска. И именно её использует тупой биос, который на самом деле на знает как грузить оси. И я не видел биоса, который умел бы грузиться с другой загрузочной записи. Еслиб все биосы умели бы это, я бы не писал этот кал.

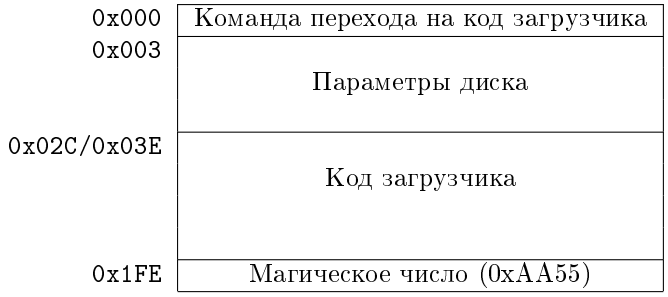
Все эти сложности BIOS'овой таблицы вызваны, как всегда историческими причинами. Изначально таблица разделов поддерживала только четыре primary партиции, но потом этого показалось мало и были придуманы extended.

Теперь несколько примеров (я их позаимствовал из документации по лило, и если вы пользуетесь этим загрузчиком, то скорее всего вы можете найти их у себя на харде).

2.2.1 Флоп

Это даже не пример таблицы разделов. А пример отдельного раздела. Ибо я не знаю способа (кроме написания злых прог на C, или ещё каких языках программирования), использовать что-то более сложное на флопе. Содержимое флопа — это в действительности пример раздела.

Загрузочный сектор флопа выглядит так:



Секция с параметрами диска на самом деле не нужна. То есть, если её нету то загрузка с дискеты возможна, просто дос не будет при её отсутствии понимать такую дискету. Даже если она форматирована на стандартные 1440 байт (80 дорожек, 18 секторов на дорожку).

[5] что-то я не припомню перевод

[6] нет скорее только лимитами системы :)

2.2.2 Хард

С хардом всё естественно сложнее. Но не намного. Таблица разделов может выглядеть так:

Таблица разделов	/dev/hda
Раздел 1	/dev/hda1
Раздел 2	/dev/hda2

Но этого мало! Любой из нарисованных здесь разделов может быть типа extended, со всеми вытекающими отсюда:

Таблица разделов	/dev/hda
Раздел 1	/dev/hda1
Раздел 2	/dev/hda2
Extended раздел	/dev/hda3
Таблица разделов Extended раздела	
Раздел 3	/dev/hda5
Таблица разделов Extended раздела	
Раздел 4	/dev/hda6

Здесь как вы можете видеть третий раздел — extended, и в нём ещё два раздела. И напоследок, MBR:

0x000	Код загрузчика
0x1BE	Таблица разделов
0x1FE	Магическое число (0xAA55)

Здесь абсолютно не нужна таблица параметров диска, потому что биос сам умеет узнавать у харда всю конфигурацию, а дос узнаёт всё от биоса.

Да, чуть не забыл, магическое число — это сигнатура, по наличию которой биос судит о корректности загрузочной записи.

3 Загрузчики

Теперь после небольшого въезжалова в тему, я разберу несколько способов грузить оси (linux & DOS & windows only :( ). Вообще, все загрузчики, которые я знаю, поддеживают два способа загрузки оси (и это должно греть вам душу: не всё ещё потеряно):

1. Chain-loading (цепочковая загрузка что ли :)?). Простейший способ: загружаем другой бутсектор (другую загрузочную запись), и передаём ей управление. Точно также как это делает BIOS. Типа “я не знаю как грузить оси, пускай native загрузчик разбирается”. Я уже говорил, что на каждом разделе есть загрузочная запись, но не все имеет смысл использовать. Ибо, например, далеко не каждый загрузчик может встать в бутсектор логического раздела.

2. Собственно загрузка оси. То есть копирование ядра оси в оперативку, разархивирование если оно пожато, передача параметров в ядро и передача управления ядру. Возможна также дополнительная настройка системы, например перевод процессора в защищённый режим. Каждое

ядро хочет чтобы его грузили по своему. Я не спец в разных ядрах, но, например, протокол загрузки ядра линукс менялся, по-моему где-то в районе 2.0 версии (ломает искать). Ядро валяется на диске как обычный файл, но поскольку большинство загрузчиков ничего не понимает в файловой системе, то они один раз выясняют геометрическое положение ядра на харде и при загрузке исходят из этого. Именно поэтому, при компиляции и установке нового ядра (файла) наобходимо обновлять загрузчик.

Далее, все загрузчики делятся на две (как минимум) части:

1. Пользовательский интерфейс. Программа которая записывает бутсектор.
2. Собственно загрузочная запись (1 stage). Это 512 байт которые записываются в бутсектор. Не буду ручатся за все загрузчики, но я знаю точно что, например, lilo, GRUB, syslinux, состоят из большего количества частей. Вот ещё не перечисленные:
3. Второй сектор раздела. (1.5 stage) (Или может вторые 512b?). Пользуется как продолжение программы загрузчика.
4. Некий набор секторов (2 stage), которые реально являются кусками файла некоторой файловой системы. Так, например, делает GRUB, который подгружает эти сектора (для этого естественно при установке загрузчика, в него надо вписать адреса этих секторов, и этим занимается программа инсталлятор 1)), и в результате умеет работать с файловыми системами.

Размер загрузчика очень ограничен, и поэтому в него сложно запихать поддержку всего железа с которым ему приходится работать. Из-за этого загрузчик вынужден использовать прерывания BIOS, например, для чтения секторов жёсткого диска. Здесь может возникнуть проблема: если биос старый и не поддерживает больших дисков, то загрузчик не в состоянии прочитать сектор который очень далеко от начала диска. Наиболее известные из таких ограничений: биос не умеет работать с диском на котором больше 1024 цилиндров, биос не тянет больше одного (или двух, или четырёх) жестких дисков.

Но все современные загрузчики умеёт работать с пакетной адресацией биоса, и соответственно, если и биос тоже умеет с ней работать, то таким загрузчиком можно грузить всё что угодно, откуда угодно.

При дальнейшем описании загрузчиков, я опускаю некоторые детали, в частности, возможности lilo и GRUB модифицировать таблицу разделов (устанавливать/сбрасывать флаги партиций, типа hidden, модифицировать тип партиции). Но, на мой взгляд, там всё очень просто. Понадобиться прочитаете ;)

3.1 lilo

lilo — это наиболее распространённый загрузчик для linux. lilo умеет напрямую грузить только линукс, но за счёт технологии chain-loading, грузит и всё остальное. Основным и единственным файлом конфигурации лило является файл `/etc/lilo.conf`. Я не буду вдаваться в подробности конфигурирования, но в директории `/usr/doc/lilo-<version>/samples` можно найти готовые примеры этого файла. Более того. В директории `/usr/doc/lilo-<version>/doc` есть файл user.tex Там есть все примеры, которые я привёл. Есть описалово того как ставить дос и линукс на один хард. Причём в различных вариантах. И не надо пугаться слова дос. В данной ситуации оно означает и виндовс тоже (как впрочем это часто случается в *nix world). Я до кучи приведу перевод одного кусочка из начала `user.tex`, мне кажется это в тему здесь.

> Для нетерпеливых: существует скрипт для быстрой установки лило. См. раздел 4.1.2. (мануала по лило (rgo))
>
> Но... постойте... вот несколько простых правил, которые помогут вам избежать большинства проблем с lilo:

- **Не паникуйте.** Если что-то не работает, попробуйте найти, что неправильно. Постарайтесь проверить свои предположения и, только после этого, попытайтесь исправить проблему.
- Прочитайте документацию. Особенно если система не соответствует тому, чему она должна соответствовать, на ваш взгляд.
- Убедитесь, что у вас есть загрузочная дискета, что вы умеете ею пользоватся (и в частности запустить `/sbin/lilo` ;) (rgo)), и что эта дискета аптудэйт.
- Запускайте `/sbin/lilo` **всегда**, когда вы меняете ядро или файлы конфинурации лило. Если вы сомневаетесь запустите лило. Вы не сможете запустить `/sbin/lilo` слишком много раз.
- если вы проводите деструктивный апгрейд (это ещё что такое? (rgo)), и/или удаляете ваши линуховые партиции de-install lilo **до этого**, если лило стоит в MBR.
- Не надо перехерачивать скрипты установки. Всегда перед установкой лило проверяйте `/etc/lilo.conf`, который они творят.
- Если вы пользуете большой диск. Будьте готовы к проблемам: быть может вам понадобиться пользовать опцию `linear`.

lilo.conf имеет следующую структуру:

1. глобальные опции, которые определяют интерфейс загрузчика (см. описалово опции install), общие опции для секций описывающих загрузку конкретных осей.
2. Описание пунктов меню загрузчика. Каждому пункту должна соответствовать одна ось (ну точнее здесь должно быть определено что, и как грузить, и как пункт меню должен называться)

Описание пункта меню завершается либо EOF (конец файла для не программеров), либо началом описаня следующего (“image” или “other” опции) Приведу также основные опции lilo.conf

boot=<boot-device> — указывает в какой загрузочный сектор писать загрузчик (1 stage). По дефолту лило пишет загрузочный сектор раздела смонтированного в корень виртуальной файловой системы.

image=<pathname> — такая запись в lilo.conf означает начало секции описывающую загрузку линукса. Таких секций может быть сколько вам угодно. pathname — это путь в **текущей** виртуальной файловой системе к файлу ядра линукса.

other=<device> — это начало секции описывающей начало загрузки не-линукса (методом chain-loading). device — это устройство на котором находится бутсектор, который будет грузится.

lba32, linear — это способ обойти исторические ограничения биоса на максимальный объём диска. Если один из загрузочных секторов которые вы пользуете находится очень далеко от начала диска, то добавьте опцию lba32 (linear устарела, но пока работает), и может быть всё станет Ок. Но эти опции могут конфликтовать с compact, на некоторых системах.

compact — указание загрузчику пытаться грузить в память по нескольку секторов за раз, это повышает скорость загрузки. Особенно существенно если вы грузитесь в флопа.

timeout=<time> — время в десятых секунды, в течение которого лило будет ждать прежде чем грузануть дефолтовую ось.

label=<name> — эта опция может быть использована только в описании пункта меню. Она указывает как будет называтся этот пункт.

Кроме приведённых есть опции установки пароля, графического интерфейса лило, модификации таблицы разделов. Но меня ломает всё это переводить.

3.1.1 Опции специфичные для линуха.

Большинство этих опций соответствуют параметрам ядра, про которые вы можете почитать в `kernel-parameters.txt` в документации к ядру. Вы можете эти опции пользовать как глобально, так и для каждого ядра в отдельности.

root=<device> — устройство которое ядро будет монтировать в корень виртуальной файловой системы.

vga=<val> — графический режим в котором будет находится консоль (скажу на всякий случай: разрешение в котором будет работать графическая оболочка X, gnome, kde, icewm никак не связано с этим параметром). val может принимать следующие значения:

- ask — при загрузке спросить режим у пользователя, полезно для определения какие режимы биос тянет.
- normal — стандартный текстовый режим 80x25,
- extended или ext — текстовый режим 80x50,
- число, номер режима, в записи языка C (в десятичной, восьмеричной или шестнадцатиричной системе счисления). Это пожалуй самая главная опция :). Номера можно разделить на две группы, первая это номера соответствующие текстовым режимам работы видеокарты, и вторые это номера соответствующие графическим режимам видяхи. Для графических режимов, необходима поддержка консоли на фреймбуфере в ядре. Если вы пользуете дефолтовое ядро, то скорее всего оно эту поддежку обеспечивает. Если же ядро — самокомпиль, то... всё в ваших руках ;). Если вы не вылезаете из иксов, то вас вообще не должна интересовать эта опция, хотя лого при загрузке это прикольно. (оффтоп: а вы пробовали на несколько-процессорной машине грузить линух с vga=0x319? Нет? Попробуйте. Меня такая глупая радость разобрала, что я чуть со стула не упал)

  0x0100-0x017f — стандартные BIOS VGA режимы, это идентификаторы (+ 0x100), которые используются в функции 00 прерывания биос 0x10. То есть если вы хотите грузануть линух в 640x480:16, то если в биосе это 0x12, то ядру надо передать vga=0x212.

  0x0200-0x08ff — тоже самое с BIOS VESA SVGA режимами. Только биосовый номер режима надо увеличить на 0x200.

  более подробно всё описано в доках ядра:

  ```
  Documentation/i386/boot.txt|
  Documentation/svga.txt
  Documentation/fb/vesafb.txt
  ```

  Вот до кучи, кину список VESA SVGA режимов (BIOS'вые номера уже увеличены на 0x200 для использования с 'vga='):

	640x480	800x600	1024x768	1280x1024
256	0x301	0x303	0x305	0x307
32k	0x310	0x313	0x316	0x319
64k	0x311	0x314	0x317	0x31A
16M	0x312	0x315	0x318	0x31B

read-only — указывает что, корневая файловая система должна быть смонтирована read only. Лучше если вы проставите эту опцию.

append=<string> — добавить параметры ядра. лило не знает всех параметров ядра, и те которые он не знает можно указывать здесь. Пример: `append="/dev/hdc=ide-cd console=ttyS1,9600"`.

- есть ещё опции, и вы их можете найти в `man lilo.conf`

3.1.2 Другие оси

Здесь по-моему нет никаких интересных опций.

3.1.3 Скрипты установки

Если вы ставите lilo из сорцов, то посмотрите на скрипт `QuickInst`. Он, интерактивном режиме, пытается найти все разделы с которых можно грузиться и прописывает файл `/etc/lilo.conf`.

3.2 GRUB

FIXME: Я опять забыл скачать доки по груб. GRUB — это наиболее мощный загрузчик из тех что я видел. GRUB понимает файловые системы, и имеет навороченный boot-prompt. Собственно это всё что я про него помню. По-моему, он в первую очередь загрузчик фри, но линух он грузит безо всякого chain-loading. То есть это значит, что вам не надо иметь ещё один загрузчик для линуха.

GRUB использует свою схему именования устройств и разделов. Имя устройства выглядит примерно так: "(hd0)"; имя раздела — (hd0, 0). hd — указывает что это жёсткий диск, возможно использовать также

fd — флоповод

hd — хард

(hd0) — это master жёсткий диск висящий на первом ide контроллере, (hd1) — slave, там же. Разделы нумеруются по порядку включая extended, но в отличие от линуха не с 1, а с 0.

grub.conf выглядит примерно также как и lilo.conf. Вот основные опции grub.conf:

default=<number> — номер дефолтового пункта меню

timeout=<number> — время в секундах, ожидания перед загрузкой дефолтового пункта

title — имя пункта меню, является знаком, что начинается описание нового пункта.

root <device> — работает для линукса (FIXME: и фри?).

kernel <filename> <options> . filename — путь к ядру, либо вида: `/boot/vmlinuz-2.6.11`, относительно root, либо `(hd0, 2)/boot/vmlinuz-2.6.11`, относительно `/dev/hda3`. options — это опции передающиеся ядру. В отличие от лило, здесь надо указывать все опции, ибо (по моему) нет способа сказать лилошное-глобальное `vga=0x314`

rootnoverify <device> — аналог root, для не линукс,

chainloader <device> +1 — FIXME: что ето?

Груб, на мой взгляд, имеет один дополнительный бонус (по сравнению с лило): его не надо переустанавливать если ядро поменяно, достаточно присвоить новому ядру имя старого (уже установленного):

```
mv arch/boot/i386/bzImage ${old-kernel}
```

Есть ещё один бонус — возможность read-only доступа к файловым системам fat, ext2, ext3 прямо из командной строки загрузчика. То есть, если вы часто переустанавливаете ядро, имеете пару установок DOS/Windows, и ещё десяток *nix, то груб для вас. Например, после неудачного эксперимента система не бутабельна, но вы помните что в какой то доке было написано как можно всё-таки грузанутся, читаете доку и грузитесь; или просто говорите, что-нибудь типа:

```
root=(hd0,2)
kernel (hd1)/kernels/bare.i vga=0x314 init=/bin/bash
boot
```

Путь к ядру я написал, из предположения наличия первого диска дистра слаки в cdrom'е висящем slave'ом на первом ide контроллере (во завернул :)).

3.3 loadlin

loadlin — досовая программа, которая грузит линух из доса. То есть запустившись, и считав ядро в память loadlin.exe, кладёт большой член на дос, и бутит ядро. loadlin ни фига не записывается ни в какой бутсектор, и поэтому прост до фига. Но надо имет рабочую установку доса. Если у вас win95, win98, winMe (по моему и winMe тоже), то loadlin почти идеал (зная как часто надо переустанавливать эти системы, я боюсь вас достанет устанавливать loadlin, прописывая процедуру загрузки в autoexec.bat и config.sys). А ежели у вас winXP, то проще попробовать что-нить другое. (FIXME: loadlin. И эти доки тоже :( )

3.4 syslinux

syslinux — это набор программ позволяющих создавать загрузочные флопики и CDROM'ы. Его можно использовать и в качестве основного загрузчика с харда, но это не удобно. Я опишу вкатце как этим пользоваться. Более подробно можно прочитать в доках по syslinux:

```
/usr/share/syslinux/syslinux.doc
/usr/share/syslinux/isolinux.doc
/usr/share/syslinux/pxelinux.doc
```

`pxelinux.doc` — рассказывает как syslinux можно применить к загрузке компа по сети, но поскольку я с этим не сталкивался, то ничего сказать пока не могу.

3.4.1 флоп

Первым делом надо отформатить флопик, создать на нём файловую систему fat, и записать на неё файлы `bzImage`, `initrd.img`. Потом в корне флопика создаём файл `SYSLINUX.CFG`, в котором указываем что и как надо грузить. Опции файла `SYSLINUX.CFG`:

DEFAULT <kernel> <options> — дефолтовая загрузка.

APPEND <options> — глобальные опции ядра которые будут передаваться любому загружаемому ядру в любом случае (ну почти см. `APPEND -`, как при ручной загрузке из командной строки, так и автоматической.

LABEL <name> — начало описания пункта меню.

KERNEL <image> — имя файла ядра или файла с образом бутсектора который надо грузить

APPEND - — при использовании в описании пункта меню отменяет глобальные опции установленные `APPEND`.

TIMEOUT <time> — таймаут в десятых секунды

FONT file.psf — загрузить в видяху шрифт из file.psf

F1 filename

F2 filename

...

F9 filename

F0 filename — позволяет привязать к нажатию одной из функциональных клавиш отображения файла на экране. `F0` означает `F10`, а `F11` и `F12` использовать невозможно.

есть и еще опции, найдёте сами.

Если есть файлик initrd, то надо указать опцию ядра `inird=<file>`, где `file` — это файл с пожатым gzip'ом образом файловой системы. Если не знаете как такой файлик сделать, ищите в `Bootdisk-HOWTO`

Если надо грузить систему отличную от линукс, делаем следующее:

в линукс:

```
dd if=/dev/device of=other.bss bs=512 count=1
```

`/dev/device` — это девайсина на которой валяется бутсектор интересующей нас системы.

в досе:

```
copybs a: a:dos.bss \#FIXME: надо на это посмотреть
```

Обратите внимание, расширение файла образа бутсектора `bss`. И это важно. Оно таким и должно быть.

Когда на дискете есть всё что нужно, осталось сказать

```
syslinux [-s] /dev/fd0
```

или, в досе

```
syslinux [-s] a:
```

Опция `-s` означает safe, slow and stupid (то есть безопасный, тормозной и тупой как бревно): syslinux поставит именно такой бутсектор. Это позволяет избежать проблем с некоторыми биосами.

Все прочие замороки создания загрузочного диска я не включаю в этот документ. Если я попытаюсь это сделать, то этот howto превратится в книгу. Книгу про то как устоен линух. Но если вам это интересно см. `Bootdisk-HOWTO`.

3.4.2 CD-ROM

С сидюком, всё не сложнее чем с флопом. Но как сказано в доках по syslinux: биосы бажные, и здесь могут быть проблемы с загрузкой. Я опишу процесс создания образа сидюка на примере `mkisofs`. Как его записать, или как сделать тоже самое, например, в `Nero Burning Rom` я **не знаю**. Эти вопросы не ко мне. Итак. Создаём дмректорию в которую кидаем всё что должно быть на уважающем себя сидюке, ну всякие архивы, музыку, и т.п. Дальше создаём ещё одну директорию, например `isolinux`. Копирум в `isolinux` файл `isolinux.bin` (в моей системе `/usr/share/syslinux/isolinux.bin`), `isolinux.cfg` (создаётся также как и `SYSLINUX.CFG` для флопа). Копируем туда всякие файлы типа ядра, initrd, бутскторов. И

```
mkisofs -o <isoimage> \
      -b isolinux/isolinux.bin -c isolinux/boot.cat \
      -no-emul-boot -boot-load-size 4 -boot-info-table \
      <root-of-iso-tree>
```

Для незнакомых с `mkisofs` поясню. `-b` это файл и el-torito бут-образом. `-c` — файл бут-каталога el-torito. isoimage — имя файла в который будет сохранён образ полученной файловой системы. Всё записываем и имеем бутабельный CD-ROM.

3.5 Booteasy

В глаза не видел...

3.6 WinNT загрузчик

Загрузчик winnt вполне даже загрузчик (я просто когда-то в этом очень сумневался :)). И его вполне можно пользовать для chain-loading. Но, для этого, сначала надо выковырять в файл линуховый бутсектор (ну или бутсектор какой другой оси). Как его выковырять в виндовс, я не знаю (точнее знаю: можно воспользоваться дос-версией syslinux). Но поскольку мы предполагаем что вы ставите линух, то проблем не будет. Итак. Если у вас уже установлен winnt и вы хотите установить линух, то вам собственно для начала надо поставить линух :). Но, не надо ставить лило в MBR. Будьте скромнее и поставте его например в бутсектор корневого раздела линуха. Факт: линух после этого грузится не будет. Поэтому в процессе установки вам надо сделать загрузочный диск линуха. После установки, грузитесь в линух с дискеты, заходите как root и

```
# dd if=/dev/hda2 of=/bootsect.lnx bs=512 count=1
```

вместо `/dev/hda2` надо подставить имя раздела на котором располагается бутсектор лило. Если вы не помните, можете посмотреть опцию boot файла `/etc/lilo.conf`.

теперь надо перекинуть файл /bootsect.lnx на c: проще всего это сделать через дискету:

```
# mcopy /bootsect.lnx a:
```

Все. Можно ребутаться в винду.

В винде, надо открыть блокнотом скрытый системный файл `C:\boot.ini`. И добавить в конец строку:

```
C:/bootsect.lnx=''Linux''
```

И, конечно, очень важно не забыть скопировать bootsect.lnx в корень диска C. Теперь совсем всё.

3.7 BootMagic

Это коммерческий загрузчик. Что он из себя представляет я не знаю.

4 Другие способы

Есть и другие способы иметь две оси на одном харде. Можно эмулировать одну ось из другой. Более тормозно чем native, но люди говорят что можно.

4.1 VMWare

это коммерческий эмулятор писюка. Спокойно может запустить линух из под винды.

4.2 Wine

Это попытка (небезуспешная между прочим) реализации win32api в линуксе. До хрена виндовых приложний запускается из под wine и даже не подозревает что win32api совсем не native M$ апи.

4.3 cygwin

Это что-то типа wine. Только здесь идея запускать линуксовые приложения под виндой. На мой взгляд (человека из *nix) если бы не некоторые особенности виндовс, было бы совсем здорово. Главное отличие от wine что cygwin-приложения более native в виндовс, чем виндовские в wine.

5 FAQ

5.1 Q: Я поставил виндовс и мне не загрузится под линуксом

A: если вас есть аптудэйт загрузочная дискета линукса, то грузитесь с неё, и вызывайте из командной строки лило:

```
lilo
```

Если загрузочной дискеты нету, но есть диск с дистрибутивом, то надо загрузится с него, смонтировать корневую файловую систему установки линукса, где стоит лило (и находится `/etc/lilo.conf`). Некоторые дистрибутивы это делают за вас (например RedHat9.0). в некоторых надо вручную:

```
mount /dev/hda3 /mnt
```

вместо `/dev/hda3` подставьте имя раздела на который поставлен линух. Потом:

```
chroot /mnt
```

вместо mnt надо подставить директорию на которую вы (ну или дистр) смонтировали корневой раздел.

```
lilo
```

Всё. С GRUB тоже самое, только вместо команды lilo надо выполнить команду установки груба (FIXME: какую команду?)

5.2 Q: Я поставил линух и теперь мне не загрузиться в виндовс

A: добавьте следующее в /etc/lilo/conf:

```
other=/dev/hda1 # вместо /dev/hda1 подставьте имя раздела на крый установлен
                # виндовс
        label=Windoz
```

и выполните команду:

```
lilo
```

6 See Also

6.1 HOWTOs

- Linux+DOS+Win95+OS2 процесс установки указанных систем на один хард
- Linux+FreeBSD
- Linux+NT-Loader — как пользоваться winnt загрузчиком
- Linux+Solaris
- Linux+Win95
- Linux+Win9x+Grub-HOWTO
- Linux+WinNT
- Linux+Windows-HOWTO
- Bootdisk-HOWTO — как создать загрузочный диск линуха.

6.2 Другие доки.

- /usr/share/doc/lilo-<version>/doc/user.dvi, если вы нашли user.tex, но там нету user.dvi:

```
su root
cd /usr/share/doc/lilo-<version>/doc/

PATH=/usr/share/texmf/bin:$PATH make
exit
```

  для просмотра пользуйте программу xdvi

- man lilo.conf, man lilo.
- info grub
- /usr/src/linux/Documentation/i386/boot.txt
- /usr/src/linux/Documentation/svga.txt
- /usr/src/linux/Documentation/fb/vesafb.txt
- /usr/share/syslinux/*doc

Те что указаны путями файловой системы, вам может быть придётся искать самостоятельно, ибо пути могут различатся:

/usr/share/doc
/usr/doc
/usr/local/doc
/usr/local/share/doc