Создание операционной системы на базе ядра linux. С нул…

НАСТРОЙКА LINUX*

Создание операционной системы на базе ядра linux. С нуля

ИЗ ПЕСОЧНИЦЫ

unnx 16 августа 2011 в 15:57 117k

Рано или поздно каждый пользователь Линукса задумывается над созданием собственного дистрибутива. Некоторые аргументируют это тем, что можно «все настроить под себя». Другие сетуют на то, что среди уже представленных дистрибутивов в Ветке нет идеального. А у них, якобы, есть суперконцептуальные идеи для собственной системы. Зачем я всю эту психологию затеял? Для того, чтобы сразу перекрыть кислород играющимся с Линуксом новичкам, которым делать нечего. Если уж задумались над созданием ОС, думайте до конца. Итак,

Я хочу создать ОС на базе Linux.
Сразу предупреждаю: был бы XVIII век, всех тех, кто для основы своей будущей системы выбирает другой развитый дистрибутив (и, не дай Бог, популярный...) ждала бы виселица. Пост именно про создание системы с нуля, а значит, всякие Slax и Linux Mint мы трогать не будем.

Шаг 1. Выбор носителя
Вариантов немного: либо ваша ОС запускается с LiveCD, либо с жесткого диска, либо с флеш-устройства. Сразу оговорюсь: не скажу в посте ни слова про жесткий диск, потому что гораздо удобнее создавать гибкий дистрибутив из серии «все свое ношу с собой», либо залоченный дистрибутив на оптическом диске. Если вы научитесь создавать LiveCD или LiveUSB систему, с установкой на жесткий диск проблем не будет.

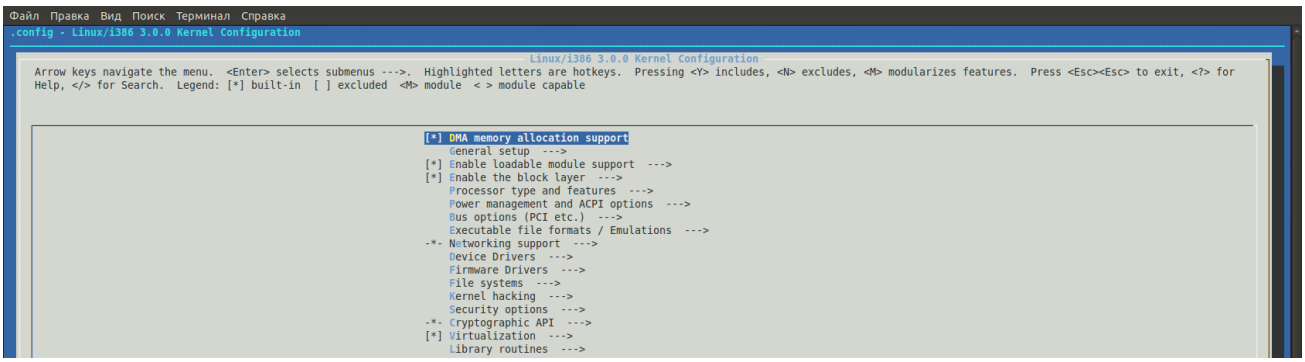
На всякий случай, приготовьте чистую флешку, CD-диск, и установите, наконец, Virtualbox.

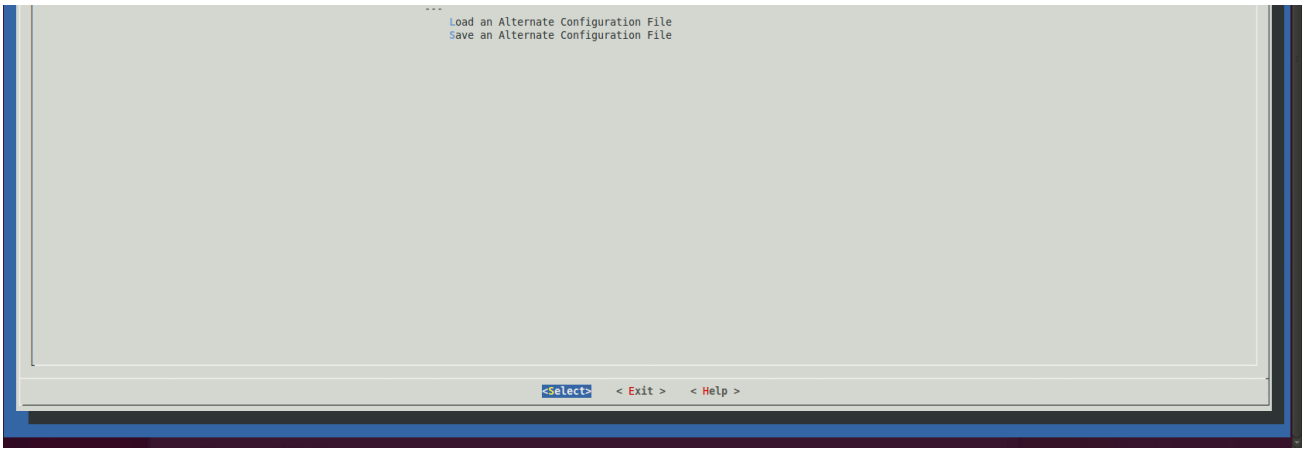
Шаг 2. Компиляция ядра
По поводу выхода третьего ядра Linux, этот шаг воодушевляет на дальнейшие разработки… Итак, нам нужны исходники ядра. Каждый пользователь знает, что их можно достать на сайте kernel.org. Ни в коем случае, слышите?, никогда не прикручивайте к своей системе постороннее ядро, скомпилированное не вами!

Поскольку лень моя зашкаливала, я создал папку /linuxkernel и распаковал туда архив с исходниками. Залогинившись под рутом, я сделал следующее:

```
cd /linuxkernel
make menuconfig
```

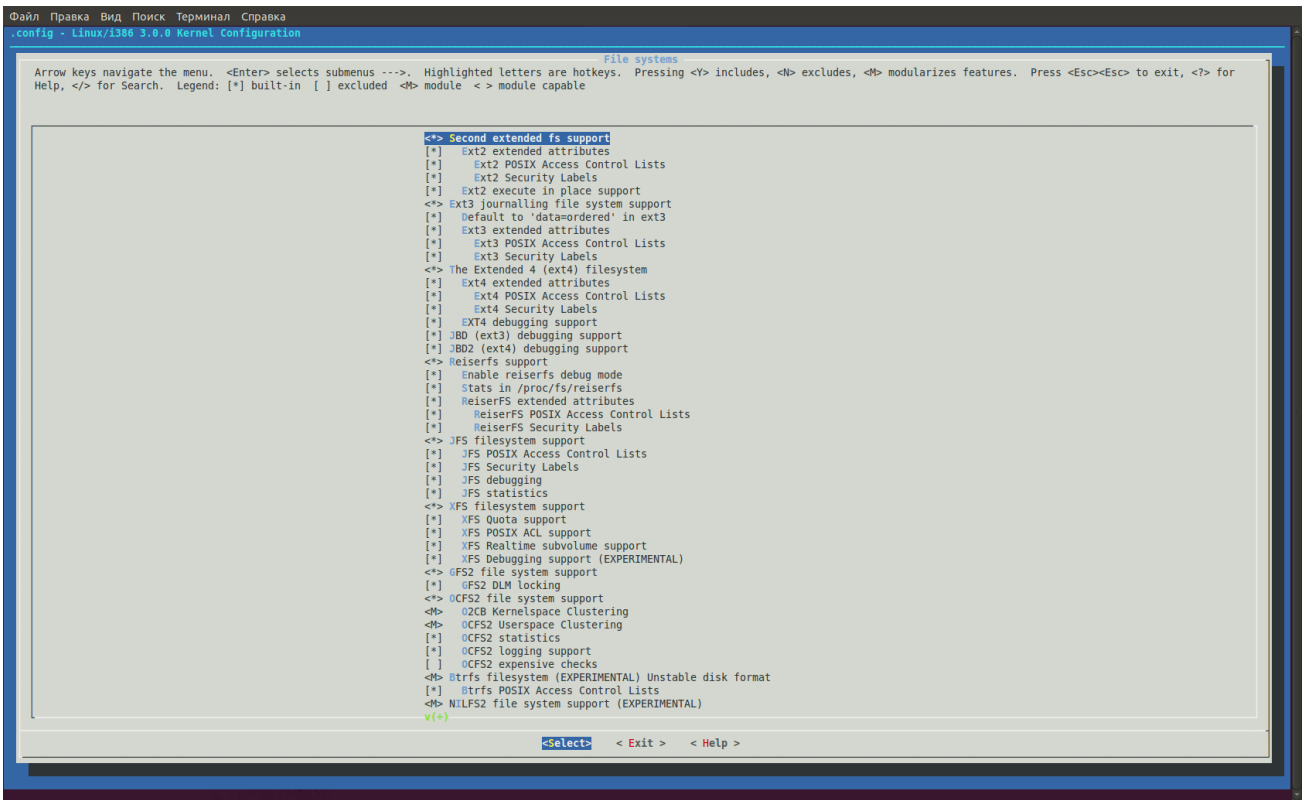
В принципе, ядро можно конфигурировать тремя способами: make config (диалоговая конфигурация), make menuconfig (псевдографическая конфигурация через ncurses), а также make xconfig (графическая конфигурация). Суть в том, что make config испортит вам настроение надолго, т.к. он задаст все возможные вопросы по всем аспектам всех тем. Проблема с make xconfig встречается не у всех, но вот у меня встречалась и встречается. Если приспичило сделать через X, разбирайтесь сами. Оптимальный вариант — make menuconfig. Эта штука откроет вам псевдографический интерфейс, через который вы сможете настроить ядро на свой лад. Штука требует библиотеки ncurses, которая легко устанавливается.





В принципе, если ваш мозг хоть сколько понимает Линукс, вы разберетесь с конфигурированием. Процесс это интересный, вариантов действительно много, а справка, хоть и на английском языке, но все же радует своей доступностью и простотой.

Однако, направить вас все же придется. Перейдите по адресу File Systems ---> и поставьте необходимые звездочки. Буква M означает, что поддержка того или иного драйвера осуществляется с помощью подключения к ядру внешнего модуля (ненавижу их!). Нам понадобится также поддержка isofs, для чтения дисков. File Systems ---> CD-ROM/DVD Filesystems ---> ISO 9660 CDROM file system support. Можно еще поддержать древнедосовские системы.



Чмошные разработчики Mandriva забыли разрешить File systems ---> DOS/FAT/NT Filesystems ---> NTFS write support, и на одном из их дистрибутивов я мучился с доступом к древневиндовскому разделу.

Посмотрите Processor type and features ---> Processor family, мне порекомендовали выбрать Pentium-MMX.

Еще поройтесь в Device Drivers, полезно. Можете шутки ради понавыбирать там все и скомпилировать ядро весом > 50 Мб.

Далее. Ядро после загрузки себя должно загружать, собственно, систему. Либо из скомпилированных в себе файлов (используются во встраиваемых системах), либо из CPIO архива, сжатого чем-нибудь, либо из Initrd. Здесь вам не DOS, здесь не получится сразу сослаться на какой-нибудь init'овый файл в корневом каталоге диска или флешки. ~~На самом деле получится, не слушайте дядю Анникса!~~ Неправильно это, хоть в Интернете по этому поводу уже нехилая полемика ведется. В своей системе мы будем использовать initrd, т.к. это удобно, и не вызовет нецензурных выражений от сторонних разработчиков, в отличие от CPIO архива.

Ах, да, скомпилируйте ядро командой

```
make bzImage
```

Если у вас x86, найдете его по адресу /linuxkernel/arch/x86/boot/bzImage.

Для суровых челябинских программистов можно использовать Кросс-компайлинг…

Создание Ramdisk.

Теперь нам нужен initrd с установленной там простейшей оболочкой. Мы будем использовать busybox, потому что эта няша может все. Способ мы украдем у Роберто де Лео, создателя Movix (я бы даже уважать его начал, если бы не запредельная любовь к Perl):

```
dd if=/dev/zero of=/dev/ram0 bs=1k count=5000 - Создаем Ramdisk в
оперативной памяти нашего компьютера.
mke2fs -m0 /dev/ram0 5000 - Форматируем Ramdisk в системе Ext2
```

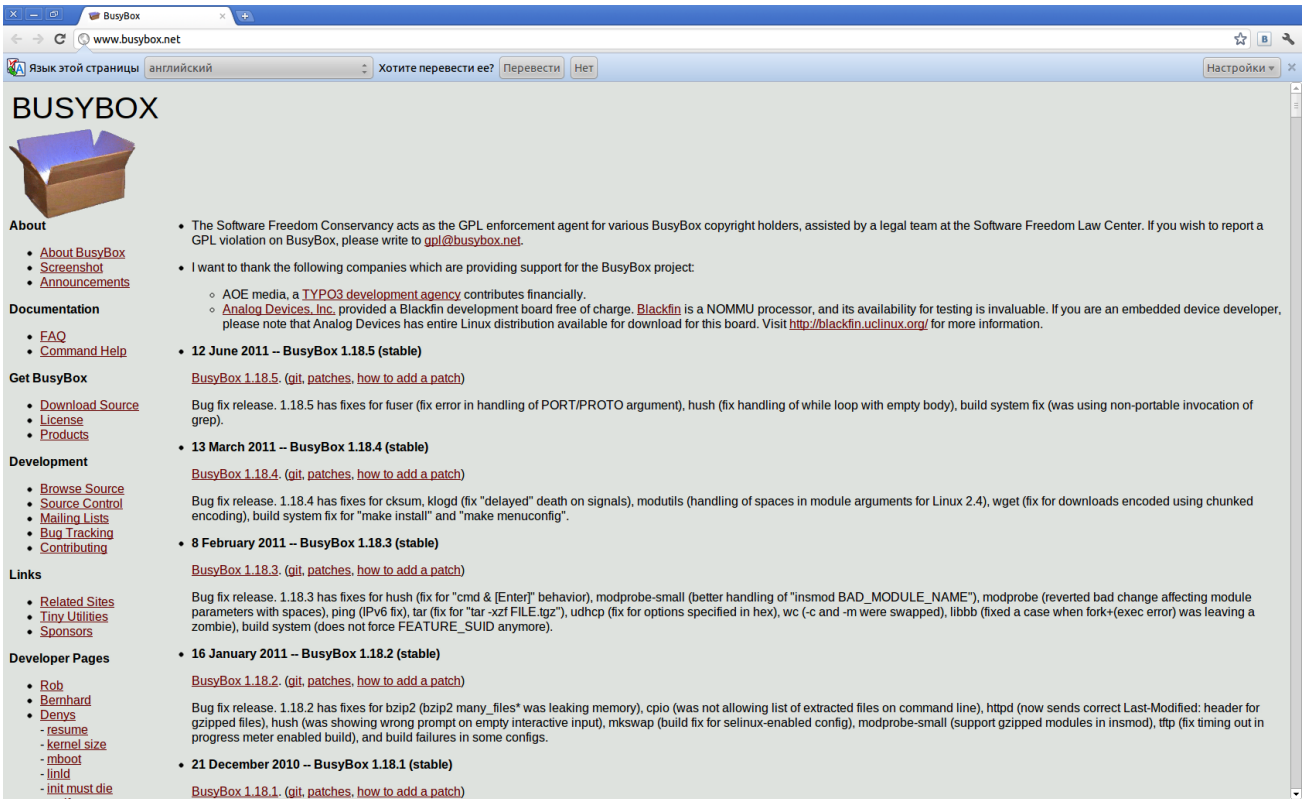
```
mkdir /distro - Создаем папку
mount /dev/ram0 /distro - Монтируем в папку /distro
```

Все, теперь у нас есть Ramdisk, емкостью в 5 Мб. Можно и больше, только не нужно. В отличие от Томаса Матеджисека, я не собираюсь пичкать initrd модулями в Squashfs, сжатыми LZMA. Все, что необходимо, будет скомпилировано вместе с ядром. Да, это не очень логично и правильно, но мороки в сто раз меньше. А специально для тех, кто осуждает такой подход, можно разрешить опцию модульности в ядре: Enable loadable module support.

В нашем Ramdisk'e, смонтированном в /distro, есть такая папка, lost+found. Это потому, что мы отформатировали его в ext2. Ни в коем случае нельзя ее удалять, хоть она здесь вряд ли поможет, образ-то фиксированный. Нам бы busybox сначала поставить…

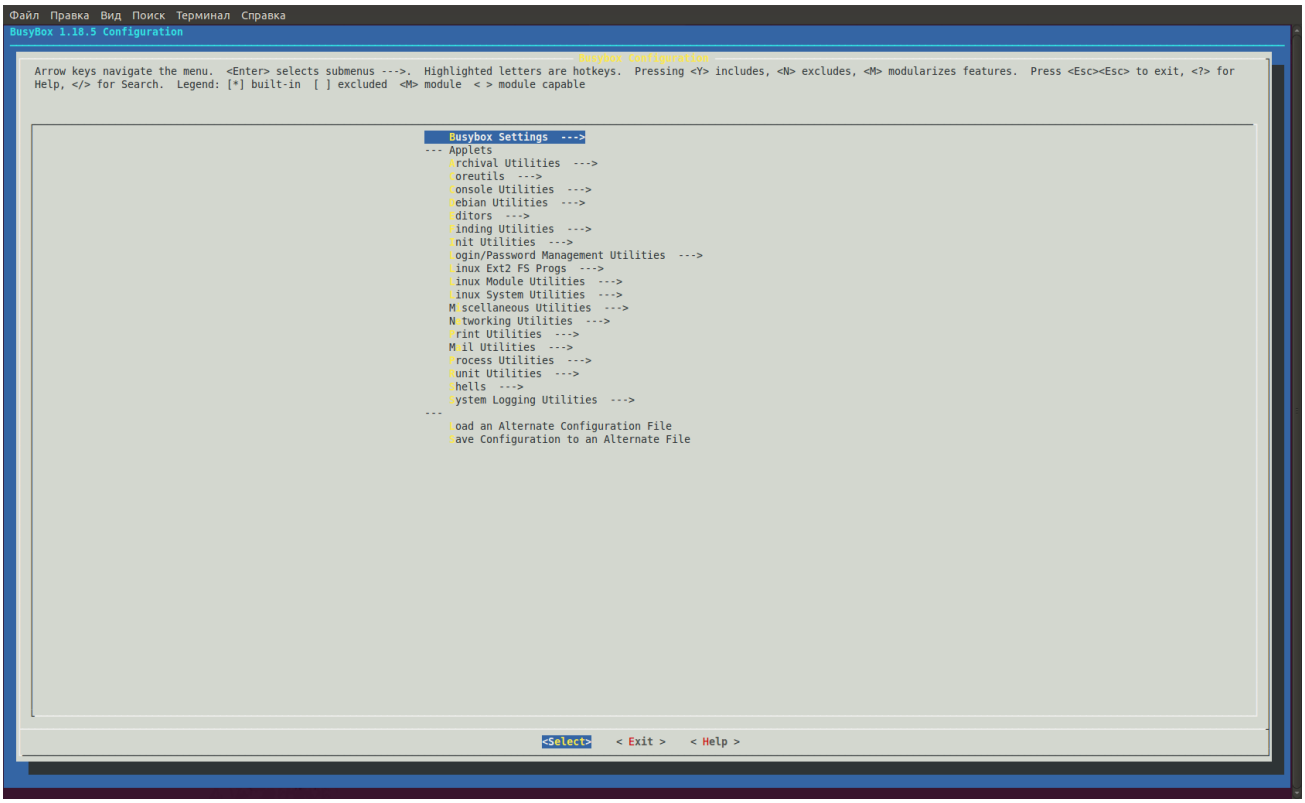
Установка Busybox

Вот почему у таких классных проектов такие отстойные сайты? Хотя… это уже не суть важно, если исходники скачаны и успешно распакованы в папку /busybox.



Сконфигурировать busybox можно так же:

```
cd /busybox
make menuconfig
```



Если вы еще не поняли, что это, объясню. Busybox заменяет тонны UNIX приложений, хранящихся в папках /bin, /sbin, /usr/bin, /usr/sbin. Вместо этого, создается только одно приложение: /bin/busybox, а на него создается куча ссылок в указанных выше папках. Установим busybox следующей командой:

```
make CONFIG_PREFIX=/distro install
```

Еще Busybox создаст файлы /sbin/init и зачем-то /linuxrc, чтобы ваша система корректно запустилась. Но не все необходимые папки были созданы. Так что завершаем все руками и создаем:

```
/distro/etc
/distro/lib
/distro/dev
/distro/mnt
distro/proc
/distro/root
/distro/tmp
```

```
/distro/root
```

Если что забыл — вспомните, т.к. директории эти забыть сложно.

Все бы хорошо, вот только busybox для работы требует библиотеки, которые нужно скопировать в наш дистрибутив. Очень легко узнать, какие:

```
ldd /distro/bin/busybox
```

Программа покажет нам библиотеки, требуемые для нашей оболочки. Сразу говорю: linux gate создается ядром и скопирован быть не может.

При копировании библиотек можно отсекать отладочную информацию (так Роберто советует):

```
objcopy --strip-debug откуда куда
```

Делаем из Линукса Линукс

Надо создать несколько системных текстовых файлов:

Нам нужен /etc/inittab. Удивлю вас: в начале жизни система даже не знает, что такое Root. У нас даже пользователь безымянный, но вот файл общесистемных низкоуровневых фич (ОНФ) должен присутствовать. Пилотное содержание файла следующее:

```
::sysinit:/etc/rc.d/rc.S

# Запустить оболочку в консоли.
::respawn:-/bin/sh

# Перезагрузка по нажатии на Ctrl+Alt+Del.
::ctrlaltdel:/sbin/reboot

# Команды, выполняемые перед выключением и перезагрузкой.
::shutdown:/sbin/swapoff -a >/dev/null 2>&1
::shutdown:/bin/umount -a -r >/dev/null 2>&1
```

Следующий файл — /etc/fstab. Это таблица, в которой описано, что и куда монтировать при загрузке. ~~Вещь бесполезная!~~ Нам нужно обязательно

смонтировать proc, иначе вообще ничего работать не будет, так что в файле пишем:

```
none /proc proc defaults 0 0
```

Для mount нужен также файл /etc/mtab. Создайте его и оставьте пустым.

Но mount сделает все необходимое только тогда, когда мы явно его об этом попросим. А просить мы будем в том самом первозагрузочном файле /etc/rc.d/rc.S (rc.d — папка). Вежливо попросим:

```
#!/bin/ash

/bin/mount -av -t nonfs
```

Еще нам необходим файл профиля (b)(a)sh, тут вообще раздолье для фантазии. Создаем файл /etc/profile и заполняем следующим:

```
PATH="$PATH:/bin:/sbin:/usr/bin:/usr/sbin:"
LESS=-MM
TERM=linux
HOME=/root
PS1='> '
PS2='> '
ignoreeof=10
export PATH DISPLAY LESS TERM PS1 PS2 HOME ignoreeof
```

Понадобится также файл /etc/shell, в котором указано, что есть оболочка:

```
/bin/sh
/bin/ash
/bin/bash
```

Вот собственно и все. Можно записывать наш Ramdisk в файл.

```
mkdir /os - папка для "готового".
umount /dev/ram0 - размонтируем кусочек оперативной памяти.
dd if=/dev/ram0 of=/os/initrd bs=1k count=5000 - создаем файл.
gzip /os/initrd - сжимаем файл initrd
```

Создание загрузочной флешки

«Финишная прямая» нашей маленькой разработки. Берем флешку, вставляем, форматируем в vfat (можно и в ext, но не забывайте, что еще не все пользователи Windows застрелились).

На флешке создаем папку boot, в ней папки initrd и kernel.

Из папки /os копируем сжатый Ramdisk в папку boot/initrd на флешке, называем «main.gz». Из папки с исходниками ядра копируем bzImage в папку boot/kernel на флешке, называем «main.lk». Достаем файлы загрузчика Syslinux (в Интернете, либо из другого дистрибутива: тут не принципиально), а именно syslinux.bin, syslinux.boot, syslinux.cfg. Копируем их в корневой каталог нашей флешки. В файле syslinux.cfg пишем что-то подобное:

```
default mm
prompt 1
timeout 100
label mm
kernel /boot/kernel/main.lk
append initrd=/boot/initrd/main.gz load_ramdisk=1 ramdisk_size=5000
rw root=/dev/ram0
label mc
kernel /boot/kernel/main.lk
append initrd=/boot/initrd/custom.gz load_ramdisk=1
ramdisk_size=5000 rw root=/dev/ram0
label cm
kernel /boot/kernel/custom.lk
append initrd=/boot/initrd/main.gz load_ramdisk=1 ramdisk_size=5000
rw root=/dev/ram0
label cc
kernel /boot/kernel/custom.lk
append initrd=/boot/initrd/custom.gz load_ramdisk=1
ramdisk_size=5000 rw root=/dev/ram0
label hd
localboot 0x80
```

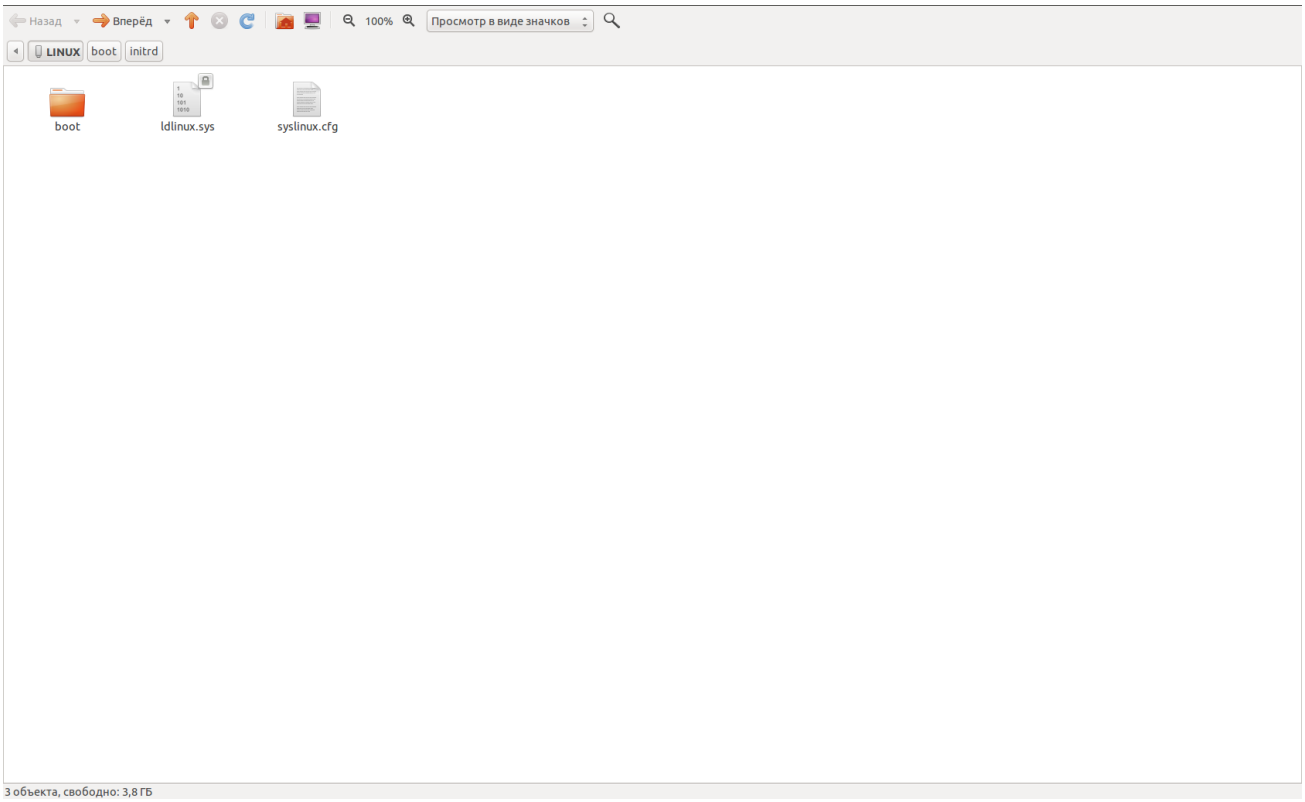
Тем самым мы поддержали кастомные initrd и ядро, которые, эксперимента ради, можно подключить к нашему дистрибутиву.

Узнаем, каким девайсом в системе является наша флешка (можно запустить mount без параметров и посмотреть). Это либо /dev/sdb1, либо /dev/sdc1, либо /dev/sdd1. Стоит отмонтировать флешку перед началом установки.

Устанавливаем syslinux (если пакета в системе нет, apt-get install syslinux):

```
syslinux -d путь_к_устройству
```

В корневом каталоге флешки должен появиться файл ldlinux.sys. Если он есть, значит syslinux.bin, syslinux.boot больше не нужны.



Как настроить BIOS на загрузку из флешки, я вам рассказывать не буду — это легко. Скажу только, что очень удобно создать папку /boot/initrd/init, в которую можно будет смонтировать /boot/initrd/main, для последующей работы с ним. Только не забудьте разжимать и сжимать его gzip'ом.

Ну вот и все.

Как-бы я только что рассказал вам, как создать с нуля систему на Linux. Легко, не правда ли? Далее вы можете редактировать скрипт /sbin/init, ведь у вас еще много

работы! Вы должны будете написать скрипт для монтирования флешки, который делает chroot в корневой каталог. В противном случае, вы вынуждены будете работать с ReadOnly разделом, величиной в 5 Мб. Но это уже совсем другая история.

Unnx Davis T, B.

Для непросвещенных:

Томас Матеджисек — создатель Slax и Linux Live Scripts.
Роберто де Лео — создатель Movix.

Ловкость рук и никакого мошенничества: практические советы по ускоренному обучению дизайну для разработчиков 12

ПЕРЕВОД

Cloud4Y • вчера в 15:52

Падение Stack Overflow: что случилось 13

it_man • вчера в 16:35

Qt: Пишем обобщенную модель для QML ListView 3

ИЗ ПЕСОЧНИЦЫ

koldoon • вчера в 13:41

Битва за сетевой нейтралитет: судебные войны и общественные протесты 5

VASExperts • вчера в 18:33

Альтернатива платному отключению рекламы в бесплатном приложении Android 16

ИЗ ПЕСОЧНИЦЫ

web_alex • вчера в 15:28

Лучшее на Geektimes

Пишите на Гиктаймс, это классно 23

Cerium • вчера в 21:06

Почему обтекатели для ракет такие дорогие? 19

Nubus • сегодня в 01:56

Астроном-любитель проверял новую камеру и случайно сфотографировал вспышку сверхновой 23

alizar • вчера в 17:47

SpaceX продолжает работу: два первых спутника глобальной беспроводной интернет-сети и охота за обтекателями 26

marks • вчера в 18:55

Что сейчас известно о суперсимметрии в физике 7

ПЕРЕВОД

SLY_G • вчера в 12:00