ttylinux User Guide

pc_i486 14.0

Maintained by Douglas Jerome
Based on Previous Work by Pascal Schmidt

April 8, 2012

Contents

1 Introduction

ttylinux is a small, minimal Linux distribution. It is freely available as a bootable CD-ROM image. The build system of shell scripts and configuration files that build the bootable CD-ROM image is also freely available.

This document provides information about using and installing ttylinux. The audience of this document should be comfortable with using the Bash command line.

The word ttylinux has no capital letters, ever. "TTYlinux", "TTY-Linux", "TtyLinux", "Ttylinux" and all other usages of a capital letter or extra symbol are wrong. When spoken, ttylinux sounds like "t - t - y - linux".

Using the ttylinux build distribution for building ttylinux is beyond the scope of this document. The build distribution has a short text file describing how to build ttylinux. When available, the Developer Guide describes building ttylinux.

This User Guide is primarily for the *PC* platform ttylinux availble for the **i486**, and is somewhat applicable to the **i686** and **x86_64** PC variants. This User Guide is also usefull for understanding the basics of the other ttylinux variants.

1.1 ttylinux Overview

ttylinux tries to use as little space as possible and be a familiar and complete command-line Linux system, with fairly up-to-date Linux kernel and program utilities. It provides multi-tasking, multi-user Linux with networking capabilities in no more than 8 MB of disk space. It is prepared for Internet access by Ethernet. A text-based web browser, command-line remote login secure client and server, NFS client, FTP server, and TFTP server are a part of ttylinux.

ttylinux can be installed onto a disk drive, both spinning hard drive or flash drive such as a USB memory stick; it can be manually installed or installed by using an installer script. Installation by installer script or manual installation can be done with ttylinux itself or by using a different Linux system.

The ttylinux file system, excluding the Linux kernel, is 8 MB in size. Using the ttylinux installer script, a Linux kernel between 2.5 and 3.5 MB will also be installed. This makes a minimum workable size of about 12 MB for a hard drive partition on which to install ttylinux, although 32 MB of RAM are needed to use the automated ttylinux installer script.

ttylinux includes a package management script, named **pacman**, capable of *installing*, *making*, *removing* and *querying* ttylinux packages and their files. The package manager can *list* and *install* packages from an external repository via http. Pacman is useful for adapting ttylinux to specific needs.

What ttylinux Is Not

ttylinux is not a typical Linux distribution; it does not have a graphical user interface, software development tools, music player, document preparation nor printing tools, databases nor network services such as BIND, News Server nor Mail Transfer Agents.

What ttylinux Can Do

ttylinux is intended to be useful as the basis of an embedded system or a directed-purpose system: with its small size ttylinux boots quickly from flash drives and CR-ROM; it has been used as a system fix/repair tool, as a simulation host, and is a good basis for a rescue or installation CD-ROM.

ttylinux provides a working Linux environment with its boot image, and custom task-specific scripts can mount other parts of the file system to provide a larger system.

ttylinux is useful on computers which are considered obsolete, such as 486SX PC. It is for people who want to have a minimal Linux distribution to run when little space is available or needed. Some users may want to use the ttylinux file system but configure and build their own Linux kernel.

ttylinux can serve as a rough prototype of a larger system, since it uses the same C library, glibc, as full Linux systems, compiling programs on a different Linux computer and copying them to the ttylinux file system can result in working programs. **However this is not a supported feature.** Programs compiled outside the normal ttylinux build process may require libraries not present in ttylinux. Worse, they may be compiled on a computer with different Linux kernel capabilities and make system calls not present in ttylinux.

ttylinux is for people who have Linux experience; it is not for beginners, unless you want to learn how a Linux system works underneath the Graphical User Interfaces. You must be able to use the interactive shell command-line, and it helps to know your way around Linux system. Most of the programs are **busybox** programs; these are smaller versions of the common Unix utilities.

1.2 Licenses

The software packages that are part of ttylinux are licensed under a number of different open source licenses, as listed below. The initialization and system service scripts developed by the ttylinux project are licensed under the GNU General Public License; a copy of this license is included in the file `COPYING`.

Package	Version	License
bash	4.2	GPL
busybox	1.19.3	GPL
dropbear	0.53.1	MIT
e2fsprogs	1.42.1	GPL
glibc	2.9	LGPL
iptables	1.4.12.2	GPL
lilo	23.2	BSD
ncurses	5.7	MIT
ppp	2.4.5	GPL
retawq	0.2.6c	GPL

For more information on the licenses, please visit the `opensource.org` website.

2 Starting with ttylinux

This section has a general overview of the ttylinux download CD-ROM image and also describes the system hardware requirements for using ttylinux, from where to download ttylinux, what to download and how to use the downloaded images.

ttylinux has three basic parts: a boot loader, a Linux kernel, and a root file system. All three of these are in the CD-ROM image; the CD-ROM image can be burned onto a blank CD-ROM disc and then booted. When booted, the root file system from the CD-ROM is decompressed and becomes a read/write root file system in a RAM disk in memory. Note that changes to any of the files while running ttylinux are lost, as they are in a RAM disk. Booting the ttylinux CD-ROM is further described in section 2.3.

Installing ttylinux from the bootable CD-ROM onto a hard drive is described in section 3.4. Installation onto a hard drive makes a system different from the bootable CD-ROM; the installed ttylinux has a read/write root file system directly on a spinning hard disk or solid state disk, not in a RAM disk. The advantage of a hard drive ttylinux system over the RAM disk system is that file changes are not lost.

ttylinux can be put onto a flash drive, such as a USB drive, which can be made bootable. This copies the RAM disk boot method to the flash drive; when the flash drive is booted, the root file system from the flash drive is decompressed and becomes a read/write root file system in a RAM disk in memory. As with the system booted from CD-ROM, the changes to files are lost when the system shuts down. The process of putting ttylinux onto a flash drive is described in more detail in section 3.3.

The ttylinux root file system is a compressed image file on the CD-ROM; it can be copied and used with a different custom kernel, one that you make, and put onto other media with your boot loader of choice. This process is beyond the scope of this document, but the requirements for a ttylinux custom kernel are described in more detail in section 2.1.1.

2.1 System Requirements

ttylinux is available for several different CPUs and hardware architectures; currently, ttylinux operation on a PC compatible i486 architecture is described in this User Guide, with much applicability to the i686 and x86_64 PC platforms. These systems are:
`ttylinux 14.0` - **i486**, specifically the i486 instruction set
`ttylinux 14.0` - **i686**, Pentium Pro instruction set
`ttylinux 14.0` - **x86_64**, AMD64/Intel 64 instruction set

CPUs and Computers

pc_i486 ttylinux requires an i486SX or newer processor in a PC compatible computer. It will not work with the i386 CPU; the glibc version in ttylinux uses CPU instructions the i386 CPU does not have. Any x86 compatible CPU supporting i486, and upward compatible, that is in a PC compatible computer should work.

pc_i686 ttylinux requires Pentium Pro or newer processor in a PC compatible computer. Any x86 compatible CPU supporting Pentium Pro, and upward compatible, that is in a PC compatible computer should work.

pc_x86_64 ttylinux supports generic 64-bit AMD64/Intel 64 processor in a PC compatible computer.

Memory

ttylinux uses an 8 MB RAM disk when booted from CD-ROM. The kernel on the CD-ROM is fairly large; it supports a broad range of hardware, so at least 28 MB of memory are required for full operation with pc_i486

ttylinux and at least 128 MB of memory are required for full operation with pc_i686 or pc_x86_64 ttylinux.

Using a custom kernel supporting only hardware for a particular computer, an i486 ttylinux system may require as little as 16 MB of memory to run. If the file system is installed onto a read/write disk drive, spinning or flash, and a custom kernel is used, an i486 ttylinux will run within 8 MB of RAM.

2.1.1 Custom Kernel Requirements

The ttylinux root file system is an 8 MB ext2 file system; the file system image is compressed and resides in the CD-ROM image. After burning the CD-ROM image to a blank CD-ROM disc, or mounting the CD-ROM image via loop device, you can find the compressed root file system; it is `boot/filesys.gz`. This root file system can be used with a custom kernel that you make.

pc_i486, pc_i686, and pc_x86_64 ttylinux are built with **Linux 2.6.34.6** header files. Linux kernels are not backwards compatible; software using the capabilities of a given kernel version cannot be *expected* to work with any previous kernel version. Using a kernel older than the kernel with which ttylinux was built cannot be supported in any way. With that described, with the small number of packages in the ttylinux system, ttylinux works to some extent with any Linux kernel from 2.6.0 upwards.

Your custom kernel needs to support all the hardware you want to use, plus some additional requirements for ttylinux itself.

A kernel used for running ttylinux needs to have **ramdisk** support, **initial ramdisk** support, and a default ramdisk size of at least **8192**. Note the kernel configuration has a default ramdisk size of 4096, which is not big enough.

If you want to use the basic firewall script of ttylinux, your kernel also needs iptables support with the **netlink** interface.

A ttylinux kernel needs to support **ext2 file systems**.

If you want to add a telnet server to ttylinux, your kernel will need to have **Unix98 pseudo terminal** support and support for the **devpts** file system.

2.2 File Downloads

The main ttylinux web site is accessed at `http://ttylinux.org/`, and `http://www.ttylinux.net/` should have the same content.

The ttylinux web site has a Download page that has several files available for downloading.

Bootable CD-ROM Images

There should be several ttylinux bootable CD-ROM ISO images available, at least one each for i486 PC, i686 PC, and x86_64 PC. The CD-ROM ISO images are each an El Torito bootable CD-ROM ISO 9660 file system with the Joliet and Rock Ridge extensions. El Torito enables CD-ROM to be bootable on PC. The Joliet and Rock Ridge extensions add longer file names to the ISO 9660 file system capabilities.

Source Code

The ttylinux web site should have links to the sources of the source code packages used to build ttylinux. ttylinux source ISO distributions, which have the source code packages comprising the ttylinux variants, are probably

available at the ttylinux web site. If you cannot find the source code package for a ttylinux component, then email douglas@ttylinux.org, and any needed arrangements will be made to supply the source code package.

Build System Distribution

The complete ttylinux build system distribution is available; it has a file How_To_Build_ttylinux.txt that describes the build process. When available at the ttylinux web site, the Developer Guide more fully describes building ttylinux.

Binary Run-time Packages

The binary packages that make up the entire ttylinux run-time system are available within the distribution ISO, or distribution TAR-file. These packages are available in the case any were removed from a ttylinux system and there is a desire or need to reinstall the removed packages. Packages are installed with pacman, the ttylinux package manager. Pacman is described in more detail in section 4.10 of this document.

2.3 Booting a CD-ROM Image

The PC variants of ttylinux are intended to boot on an appropriate 32-bit x86 or 64-bit x86_64 PC that can boot from a CD-ROM drive.

Download the CD-ROM ISO image file and burn it onto a blank CD-ROM disc as an ISO image. Then put the disc into the CD-ROM drive of an appropriate PC and boot the PC; ttylinux should start up automatically.

A computer's BIOS setup may not be set up to allow booting from CD-ROM; in that case you need to go into the BIOS setup screen(s) and make changes that allow the computer to boot from CD-ROM. If the computer has an old BIOS that is not able to boot from a CD-ROM device, there is software called *Smart Boot Manager* that may help. It can currently be found at: http://btmgr.sourceforge.net/about.html

Once ttylinux has booted, and you see the login prompt, login as user name "root", the administrator account, with password "password". There also is a non-administrator account "user", with password "password".

The CD-ROM can be used as a rescue system or simply for trying ttylinux. For installing or transferring ttylinux from the CD-ROM, or from the downloaded CD-ROM image file, to another disk device see section 3. See section 4 of this user guide for pointers about what you can do with a ttylinux system.

2.4 Setting Up a USB or Flash Drive

ttylinux can be put onto a USB drive, also known as flash drive, USB memory stick, pen drive, travel drive, etc. This also applies to flash drives that are not on USB. For these you probably want to boot a RAM disk system from your USB or flash drive. See section 3.3, but you should also read the preceeding parts of section 3.

3 Installing from CD-ROM

This section of the user guide describes the two types of ttylinux bootable installations and the methods for creating them from either the downloaded CD-ROM image file or a CD-ROM with the image burned onto it.

3.1 CD-ROM Image Overview

The ttylinux CD-ROM image is the source used for installation; it is a CD-ROM ISO 9660 file system. The pc_i486 CD-ROM has the following directory structure:

```
|-- AUTHORS
|-- COPYING
|-- LABEL
|-- boot
|   |-- System.map
|   |-- filesys.gz
|   |-- grub
|   |   `-- loopback.cfg
|   |-- isolinux
|   |   |-- boot.msg
|   |   |-- help_f2.msg
|   |   |-- help_f3.msg
|   |   |-- help_f4.msg
|   |   |-- isolinux.bin
|   |   `-- isolinux.cfg
|   |-- vmlinux
|   `-- vmlinuz
|-- config
|   |-- kernel-2.6.34.6.cfg
|   |-- syslinux
|   `-- ttylinux-setup
|-- doc
|   |-- ChangeLog-pc_i486
|   |-- Flash_Disk_Howto.txt
|   |-- User_Guide.html
|   |-- User_Guide.pdf
|   `-- User_Guide.tex
|-- packages
|   |-- bash-4.2-i486.tbz
|   |-- busybox-1.19.3-i486.tbz
|   |-- dropbear-0.53.1-i486.tbz
|   |-- e2fsprogs-1.42.1-i486.tbz
|   |-- glibc-2.9-i486.tbz
|   |-- gpm-1.20.6-i486.tbz
|   |-- iptables-1.4.12.2-i486.tbz
|   |-- lilo-23.2-i486.tbz
|   |-- ncurses-5.7-i486.tbz
|   |-- ppp-2.4.5-i486.tbz
|   |-- retawq-0.2.6c-i486.tbz
|   |-- ttylinux-basefs-1.0-i486.tbz
|   |-- ttylinux-devfs-1.0-i486.tbz
|   `-- ttylinux-utils-1.3-i486.tbz
`-- qemu-i486.sh
```

Several files are critical for installation, note their location in the CD-ROM image:
`CD-ROM/`**boot/filesys.gz** – ttylinux gzipped ext2 file system image
`CD-ROM/`**boot/vmlinuz** – gzipped ttylinux Linux kernel
`CD-ROM/`**config/ttylinux-setup** – user-maintained RAM disk startup

The ttylinux installation script automates the process of installing ttylinux; the scripts also copy the other documentation and information files to the new installed system.

For manual installation you can mount the CD-ROM or mount the CD-ROM image file via loop device for access to the critical files. The manual processes of installing ttylinux describes this in more detail.

3.2 RAM Disk or Persistent Storage Boot

There are two basic types of ttylinux installation, resulting in two type of booted systems: a RAM disk or persistent storage.

A **Ram Disk** installation results in a system that puts the root file system into RAM when it boots, which is what the bootable CD-ROM does. If you want to put ttylinux onto a flash drive, pen drive, USB memory stick, travel drive, etc. then you very probably want this sort of installation. This is typical for flash drives, but not for spinning hard disks, for at least two reasons: 1) flash drives have been much slower than hard drives, so maintaining a live file system on a flash drive has been intolerably slow, and 2) these are removable drive which have been difficult to consistently mount as they move between interfaces and computers. With this booting scheme, the boot loader takes the root file system from the drive and gives it to the kernel which decompresses it and mounts it as a read/write root file system in a RAM disk in memory. Changes to files in the root file system are lost when the system shuts down; persistent changes must be stored elsewhere. However, ttylinux has specific support for persistent changes to its boot-time startup with a RAM disk system, this is described section 3.3.5. The program `/sbin/ttylinux-flash` can be used to copy ttylinux from the CD-ROM to another drive and configure it to boot in this manner. The processes of putting ttylinux onto a drive for booting a RAM disk system and configuring its boot-time startup support is described in more detail in the next section 3.3.

A **Persistent Storage** installation results in a system that boots with the read/write root file system maintained directly on the spinning hard disk or solid state drive. If you have a spinning hard disk or one of the new fast solid state drives, and it is not removable, then you probably want this sort of installation. If you want to install onto removable media and will move the media to different slots or computers, then you do not want this kind of installation. The program `/sbin/ttylinux-installer` can be used to install ttylinux from the CD-ROM onto a spinning hard disk or solid state drive and configure it to boot in this manner. The processes of installing ttylinux onto a hard drive and booting a persistent storage root file system is described in more detail in section 3.4.

3.3 Transfer from CD-ROM – Make a RAM Disk Boot System

The section describes transferring a few files from the CD-ROM image to a drive, probably a flash drive, and configuring it to boot ttylinux into a RAM disk, as does the CD-ROM.

A Linux system, either ttylinux or some other Linux system, can be used to make a ttylinux bootable flash drive.

The ttylinux script `/sbin/ttylinux-flash` makes a bootable ttylinux on a flash drive, and it does this in such a way that the new flash drive ttylinux copy can then be used in place of a CD-ROM as the *source* for another ttylinux transfer, but only if you again use the ttylinux script.

3.3.1 Source Directory

The ttylinux CD-ROM is used as the source, or the ttylinux CD-ROM image file mounted with a loop device can be used. Even the kernel and file system image files removed from the ttylinux CD-ROM image can be used as the source if they are in a directory structure as found in the CD-ROM image.

Using the ttylinux CD-ROM Disc

For the following examples of mounting the CD-ROM disc, /mnt/cdrom references the mount point in your file system to which the CD-ROM disc mounts. If you are not using ttylinux then your actual mount point may be different; substitute accordingly. Have the ttylinux boot CD-ROM disc in the CD-ROM drive and mount it. You need to know which device in /dev to use; if you do not know which device to use then section 3.4.1 might help. The CD-ROM should be mounted as type iso9660 e.g., mounted by the following command.

```
mount -t iso9660 /dev/<partition> /mnt/cdrom
```

Using a ttylinux CD-ROM ISO Image File

If you have a downloaded ttylinux CD-ROM image file, ttylinux-i486-14.0.iso.gz, ttylinux-i686-14.0.iso.gz or ttylinux-x86_64-14.0.iso.gz, then you can mount it via loopback device with the following commands; substitue i686 or x86_64 for i486 where appropriate.

```
mkdir -p mnt/cdrom
gunzip ttylinux-i486-14.0.iso.gz
mount -t iso9660 -o loop ttylinux-i486-14.0.iso mnt/cdrom
```

Critical ttylinux CD-ROM Files

Of the following three files, you must have access to the first two; the third one is very usefull, but not critical. In the following, "CD-ROM/" is meant to be wherever you mounted the CD-ROM disc or CD-ROM image file.

CD-ROM/**boot/filesys.gz** – ttylinux gzipped ext2 file system image
CD-ROM/**boot/vmlinuz** – gzipped ttylinux Linux kernel
CD-ROM/**config/ttylinux-setup** – user-maintained RAM disk startup

3.3.2 Target Directory

The target directory is the directory where ttylinux will be put; it must be the top-level, root directory on the disk partition being used. You need to know, or find out, the device name for the disk partition onto which you want to transfer ttylinux. If you are not sure what a disk partition is you can read a little more description of ttylinux target partitions in section 3.4.2, but do not continue until you understand enough about disk devices and partitions to understand the rest of this section.

Due to the combined space requirements of the 8 MB ttylinux file system and the 2.5 to 3.5 MB ttylinux Linux kernel, and considering some margin, the minimum partition size onto which you can install ttylinux and have it work is at least 12 MB. These sizes are much larger for the i686 and x86_64 ttylinux variants.

This rest of this section describes manually mounting a disk partition that has the directory to transfer ttylinux onto. If your target directory is already mounted, or automatically mounts, and you will use the ttylinux script to transfer ttylinux, then **delete everything in the target directory** or the script will not transfer ttylinux onto it.

In order to manually mount a disk you need to know the disk device node e.g. /dev/sdc and its mountable partition you want to use e.g. /dev/sdc1. Read the previous sentence again, note the distinction between the

disk and partition devices.

A USB drive partition probably should be mounted with the following command. For the following example of mounting the drive, /mnt/flash references the mount point in your file system to which the drive mounts. If you are not using ttylinux then your actual mount point may be different; substitute accordingly. If you are mounting a Linux file system then change to the appropriate file system type in the following command.

```
mount -t vfat /dev/<partition> /mnt/flash
```

The device partition in the above example is the device node of the mountable partition on the disk that you want to use e.g. sdc1, in which case it represents /dev/sdc1.

If you will use the ttylinux script to transfer ttylinux, then **delete everything in the target directory** or the script will not transfer ttylinux onto it.

3.3.3 Running the Transfer Script

The ttylinux shell script, `/sbin/ttylinux-flash` automates the process of copying the ttylinux system from the source directory into the target directory and making the target drive bootable. This transfer typically is from CD-ROM disc to flash drive; the CD-ROM disc should be mounted with option `-t iso9660` to specify the correct file system type, and USB drives are usually FAT32 and those should be mounted with option `-t vfat` to specify the correct file system type.

The script is invoked with a command line option telling it which boot loader to use, lilo or syslinux. The following is the help output from the `ttylinux-flash` script, it describes how to invoke the script.

```
ttylinux-flash
(C) 2008-2010 Douglas Jerome <douglas@ttylinux.org>

Usage: ttylinux-flash --lilo     <source path> <flash path> <flash dev>
       ttylinux-flash --syslinux <source path> <flash path> <flash dev>

Parameters:

     -l | --lilo ....... Use lilo method to make bootable flash disk.
     -s | --syslinux ... Use syslinux method to make bootable flash disk.

     <source path> is the mounted ttylinux CD-ROM directory, or any equivalent
                   USB or hard drive directory of the ttylinux CD-ROM layout
                   and contents.

     <flash path>  is a rooted path to the flash disk root file system to be
                   loaded from the source path.  For the syslinux method this
                   must be a Windows FAT file system, but for the lilo method
                   this can be either an EXT2 or FAT file system.

     <flash dev>   is the /dev/* that is the whole disk block device node,
                   such as /dev/sdc, NOT a partition block device node
                   like /dev/sdc1.
```

The transfer script checks if the source CR-ROM device contains a ttylinux CD-ROM; if the CD-ROM is found then a summary of what is to be transfered onto which device is printed and you are given a choice of continuing or aborting. Enter "yes" to continue the transfer.

The transfer script copies the ttylinux files onto drive and then installs the bootloader.

After the installer is finished you can remove the CD-ROM disc from your computer and reboot.

3.3.4 Manual Transfer

The manual process is described in appendix B of this document. It also is a text file, `Flash_Disk_Howto.txt`, in the doc/ directory on the ttylinux CD-ROM disc.

3.3.5 Configuring the System

This transfered ttylinux system is a RAM disk system. Its file system is a gzipped image on the boot drive; its directories and files are very inaccessible. When the system is running, changes made to files are not retained for the next boot. Customizing or configuring this system to your needs takes some specific support.

A ttylinux system properly transfered, such as by using the `/sbin/ttylinux-flash` script, has a special boot-time startup feature: when ttylinux boots it attempts to mount the USB/flash drive that is boots from and run a script on that drive. Since the script is in a boot drive directory, and not burried away in a compressed file system, it can be changed and maintained by you. You can change this script to perform ttylinux configuration for ttylinux to do when it boots.

The feature of mounting the booted drive and running a startup script on that drive provides the user with a persistent boot-time startup configuration that is retained from one boot to the next. This user-maintainable startup script on the boot drive is in the `config/ttylinux` directory; a default version of this script is put onto the boot drive when the `/sbin/ttylinux-flash` script is run. This startup script is a very convenient place to configure your particular network interface upon ttylinux startup.

3.4 Install from CD-ROM – Make a Persistent Storage Boot System

NOTE Your computer needs at least 32 MB of RAM to run the `ttylinux-installer` script.

The section describes installing ttylinux from the CD-ROM image to a drive partition, probably on a spinning hard disk, and configuring it to boot ttylinux with the root file system residing on the boot drive, not in RAM.

A Linux system, either ttylinux or some other Linux system, can be used to install ttylinux.

WARNING: Running the installer can easily destroy all operating systems, and anything else, currently present on the target machine. Proceed with caution and backup all *important* data before installing ttylinux. Really.

To install ttylinux onto disk from the bootable CD-ROM, you first need to burn the ttylinux CD-ROM ISO image onto a blank CD-ROM disc and, if using ttylinux to perform the installation, boot into it as described in the previous section 2.3.

Once logged into ttylinux as root, you can start the installation. You need to know three things to run the installer: 1) what your CD-ROM device is, 2) which drive partition you want to install ttylinux, and 3) where you want to put the boot loader.

If you don't know the answers to those three questions after reading the following instructions, the safe bet would be **not** to proceed with installation; the ttylinux installer is not yet automated or user-friendly enough for you.

3.4.1 Source CD-ROM Device

The correct CD-ROM device name depends on whether the drive is an IDE or SATA device. If your system uses IDE, the following device names are possible:

Device Name	Description
/dev/hda	Master Device on First IDE Controller
/dev/hdb	Slave Device on First IDE Controller
/dev/hdc	Master Device on Second IDE Controller
/dev/hdd	Slave Device on Second IDE Controller

Among the above, /dev/hda is not likely to be your CD-ROM device unless you are using a modern laptop. A more likely possibility is /dev/hdc. /dev/hda normally is the device name of your hard disk, but a modern computer will use SATA for the hard drive and many of those have IDE CD-ROM drive.

If your system uses SATA (Serial ATA), use this table:

Device Name	Description
/dev/scd0	First SATA CD-ROM Device
/dev/scd1	Second SATA CD-ROM Device
/dev/scd2	Third SATA CD-ROM Device
/dev/scd3	Fourth SATA CD-ROM Device

Usually the SATA CD-ROM device will be /dev/scd0.

3.4.2 Target Partition Device

You need to know, or find out, the device name for the disk partition on which you want to install ttylinux. The device names for disk partitions are formed by appending a number to the device name of the corresponding disk. For example, if your disk device is /dev/hda, the device /dev/hda3 is the third partition on that disk. Numbers 1-4 are the primary partitions, extended partitions start at 5.

ATTENTION: If you plan on installing onto a USB drive, or some other frequently moved disk device, then do not install ttylinux with the instructions here; use the instructions in section 3.3. The disk and partition devices used by this installation process would likely be different between different computers, so this installation may not correctly boot when booted on a computer other than the computer on which the installation is performed.

Due to the combined space requirements of the 8 MB ttylinux file system and the 3 MB ttylinux kernel, and considering some margin, the minimum partition size onto which you can install ttylinux and have it work is about 12 MB.

IDE disks use the same device names as given for IDE CD-ROM devices above. For SATA, the names are as follows:

Device Name	Description
/dev/sda	First SATA Disk Device
/dev/sdb	Second SATA Disk Device
/dev/sdc	Third SATA Disk Device
/dev/sdd	Fourth SATA Disk Device

Note that if you want to create a dual-boot setup with Windows and ttylinux on the same disk, a topic not covered here, you can't use the first partition /dev/hda1 or /dev/sda1 as your ttylinux target partition, because that is where Windows needs to be installed to work.

Here are some examples of possible device names for your target partition:

Device Name	Description
/dev/hda1	First Primary Partition on Primary IDE Master
/dev/hdb5	First Extended Partition on Primary IDE Slave
/dev/sda2	Second Primary Partition on First SATA Disk
/dev/sdc6	Second Extended Partition on Third SATA Disk

Note that depending on the BIOS, booting might be possible only from the first two disks installed in the system.

Also, you can look at the directory listing of `/sys/block` to see which block devices the kernel has detected, as disk drives are block devices.

Drive Partitioning, if Needed

What to do if your target disk is not partitioned yet? Linux systems, including ttylinux, have the `fdisk` program that can be used to partition disks. For example, to partition a disk connected as master to the first IDE controller, use:

```
fdisk /dev/hda
```

The user interface of `fdisk` is somewhat primitive, *so be careful*. If you haven't used it before, a good idea would be to search the Internet for instructions. The basic commands you may need are "d" to delete a partition, "n" to create a new partition, "p" to print the current partition table, and "w" to write the edited partition table to disk. You can also use "q" to exit `fdisk` without saving your changes.

3.4.3 Boot Loader Location

The LILO boot loader is installed in one of two places: either the Master Boot Record (MBR) of the *disk device* or the boot sector of the *partition device* in which ttylinux is being installed.

With LILO installed in the MBR of the first disk, it will completely take over the entire boot process of the computer. If there are other operating systems installed on the computer they need to be specified in the LILO configuration file, `/etc/lilo.conf`, in order to boot them.

With LILO installed in the boot sector of the target partition or in the MBR of a disk other than the first one in your computer, the bootloader installed in the MBR of the first disk needs to be configured to boot the ttylinux target partition.

3.4.4 Running the Installer

Once you have decided on target device and boot loader location, you can run the installer. The script is called `ttylinux-installer`. The following is the help output from the `ttylinux-installer` script, it describes how to invoke the script. The square brackets indicate an optional parameter, the *partition* device is used for the installation target device.

```
ttylinux-installer
(C) 2008-2011 Douglas Jerome <douglas@ttylinux.org>

Usage: $(basename $0) [ <options> ] <source device> <target device>
Usage: $(basename $0) --config=<file> <source device>

Options:
     --help ......... Show this help.
     -m | --mbr ..... Put lilo boot loader onto the Master Boot Record of the
                      disk device containing the <target device> disk partition.
     --nolilo ....... Do not put the lilo boot loader onto the disk drive.
     --vcs=<name> ... Use <name> for the Virtual Context Script.

<source device> is the CD-ROM device that has the ttylinux CD-ROM.

<target device> is the disk partition device onto which ttylinux is installed;
                it needs to be a disk partition device, not the whole disk
                device.  An ext2 file system is created on this device.

Create an ext2 file system on the disk partition device <target device>,
install ttylinux from the CD-ROM <source device> into the new file system on
<target device>, and then maybe put a lilo boot loader onto the target disk.
The lilo boot loader is put onto the disk partition <target device> unless the
-m or --mbr option is present, in which case the it is put onto the Master Boot
Record of the disk device.  No lilo boot loader is put onto the disk if the
--nolilo option is preset or if the running architecture does not support lilo
e.g., ttylinux PowerPC.
```

For example, to install from the CD-ROM device /dev/hdc into partition device /dev/hda2 and placing LILO on the MBR, /dev/hda disk device, you would use:

```
ttylinux-installer -m /dev/hdc /dev/hda2
```

Another example, installing from the second SATA CD-ROM device /dev/scd1 into the third partition device of the second SATA disk and placing LILO on the boot sector *of the target partition*:

```
ttylinux-installer /dev/scd1 /dev/sdb3
```

The installer checks if the source CR-ROM device contains a ttylinux CD-ROM disc; if the CD-ROM disc is found then a summary of what is to be installed on which device is printed and you are given a choice of continuing or aborting. Enter "yes" to continue the installation.

The installer creates an ext2 file system on the target partition then copies the ttylinux distribution files onto the new file system, and then installs the LILO bootloader.

After the installer is finished you can remove the CD-ROM disc from your computer and reboot.

Using the Installation Configuration File

When you boot ttylinux from the CD-ROM image and log in as root, you will find a file named "install.conf" in the /root directory; this file is an example of an installation configuration file that can be used to direct the automated installtion activites of the ttylinux installer script. The example "install.conf" file is well documented; read it to learn how to use it.

You must first manually partition the hard drive, and then make sure the installation configuration file's fstab section matches the actual disk partitions.

Example, to install from the CD-ROM device /dev/hdc into partition device /dev/hda2, using the installation configuration file:

```
ttylinux-installer --config=install.conf /dev/hda2
```

Using a Virtual Context Script

The ttylinux installer script supports a Virtual Context Script option in which the installed system is locked-down and boots with a startup sequence that mounts an iso9660 (CD-ROM) file system and runs a contextualization script from that file system. This is for running the installed ttylinux system in a virtual environment that may be prone to unauthorized log-in attempts.

The `--vcs=<name>` option names the Virtual Context Script to be invoked at boot startup each time the installed system boots. The named Virtual Context Script is inkvoked from an iso9660 file system that is mounted at boot startup; the block device for this file system must be available as /dev/hda, /dev/hdc, /dev/sr0, or /dev/cdrom.

When the `--vcs=<name>` option is used, the installed system is locked down by:
1. the user account is removed
2. the getty logins are all disabled
3. dropbear (sshd) is configured to run with passwords disabled
4. the firewall is enabled allowing only the SSH port
5. the Virtual Context Script is configured to be invoked at startup
6. the startup is configured to create a random root password

It is expected that the Virtual Context Script, which is invoked each time the installed system boots, puts an ssh-compliant public key into the /root/.ssh/authorized_keys file. This allows root login via SSH by someone with the corresponding SSH-compliant private key. See section 4.5 for some more information on dropbear and public/private key usage.

3.4.5 Manual Installation – Setup and Installation

This description uses `LILO` for boot loading; other boot loaders such as `grub` and maybe `loadlin` and `syslinux` will also work.

There are two files to take from the ttylinux CD-ROM image, either by burning the image to a blank CD-ROM disc and mounting it, or mounting the CR-ROM image via loop device. In the following commands, the ttylinux version 14.0 CD-ROM image file is named `ttylinux-i486-14.0.iso.gz`; decompress it and mount it via loop device with the following commands, substituting your actual CPU `x86_64` or `i686`, if needed.

```
mkdir -p mnt/ttylinux
gunzip ttylinux-i486-14.0.iso.gz
mount -t iso9660 -o loop ttylinux-i486-14.0.iso mnt/ttylinux
```

The two files needed from the CD-ROM are the ttylinux root file system, `boot/filesys.gz`, and the Linux kernel, `boot/vmlinuz`. You can, of course, use a different Linux kernel, following the ttylinux custom kernel requirements described in section 2.1.1.

There are two ways to install ttylinux for booting, one is to have ttylinux boot with the root file system in RAM disk, the other is to have ttylinux boot with the root file system directly on the hard drive.

Install a ttylinux to Boot Using RAM Disk

Copy the ttylinux file system `filesys.gz` image and the desired Linux kernel into your boot files directory;

probably, this directory is /boot. After copying the two files, unmount the loop device with the following command.

```
umount -d mnt/ttylinux
```

These two files, the kernel and the file system image, can have names other than the file names from the ttylinux CD-ROM. For this example the file names are changed from the names on the CD-ROM: the compressed ttylinux file system image file is called ttylinux-filesys.gz, the Linux kernel is called ttylinux-vmlinuz and the boot dirctory is /boot. Add the following section to /etc/lilo.conf:

```
image = /boot/ttylinux-vmlinuz
        label  = ttylinux
        initrd = /boot/ttylinux-filesys.gz
        root   = /dev/ram0
        read-only
```

Run the LILO boot loader installer by typing /sbin/lilo. The next boot will have the option of selecting ttylinux at the LILO boot prompt.

Install a ttylinux to Boot with File System on a Hard Drive

A hard drive partition, or a flash drive partition, of at least 8 MB is needed to install ttylinux. For this example ttylinux is being installed on drive partition device /dev/hda8 and the kernel and file system files are available via the loop device instructions above. A loop device also is used to mount the ttylinux file system image file.

```
cp mnt/ttylinux/boot/filesys.gz filesys.gz
gunzip filesys.gz
mkdir -p mnt/filesys
mkdir -p mnt/newroot
mount -t ext2 -o loop ./filesys mnt/filesys
mount -t ext2 /dev/hda8 mnt/newroot
cp -a mnt/filesys/* mnt/newroot
cp mnt/ttylinux/boot/vmlinuz mnt/newroot/boot/ttylinux-vmlinuz
umount -d mnt/ttylinux
umount -d mnt/filesys
```

The new ttylinux root file system is still mounted; it needs to be customized before booting. Configuration is described in the following section 3.4.6; it includes a description of a LILO configuration for booting the new ttylinux installation. After configuration unmount mnt/newroot.

3.4.6 Manual Installation – System Configuration

This section covers the minimum configuration needed to run ttylinux. More system configuration can be done; see the system guide, section 4, below for information.

The configuration files and options described in this section are present in a ttylinux system installed from the bootable CD-ROM. Following the installation instructions above, the file system is in mnt/newroot, that is the example starting point for the following configuration descriptions.

/etc/fstab

/etc/fstab needs to have the correct device for the root directory, the manually installed ttylinux /etc/fstab still specifies a RAM disk device for the root directory. Change the RAM disk device, /dev/ram0, to be the disk partition device in which the ttylinux root file system was installed. In the above example /dev/hda8 was used,

so for that example the root directory in /etc/fstab would be specified as:

```
/dev/hda8      /      ext2      defaults      0 0
```

Boot Loader

The boot loader needs to specify the ttylinux kernel and root file system device. Following the installation instructions above, the LILO configuration file /etc/lilo.conf would include the following. Note the *initrd* specifier is removed and the *root* specifier is changed to /dev/hda8.

```
image = /boot/ttylinux-vmlinuz
        label = ttylinux
        root  = /dev/hda8
        read-only
```

Keyboard Map

To use the current keyboard map from the Linux computer being used to install ttylinux, use the following commands.

```
        rm mnt/newroot/etc/i18n/kmap
        mnt/newroot/bin/dumpkmap >mnt/newroot/etc/i18n/kmap
```

Timezone

The best way to set the ttylinux timezone is to use the boot parameters as described in section 4.1. This needs to be done only one time for an installed ttylinux system as this boot option becomes persistent.

Dial-up Network Information

ttylinux does not *directly* support dial-up networking with PPP and has no support at all for ISDN. Previous versions of ttylinux did have PPP and ISDN support; their package structure is being re-organized and they may return in some later version of ttylinux.

ttylinux does have the PPP binaries: /usr/sbin/pppd and /usr/sbin/chat, but currently it is up to you to configure and use them.

Unmount and Reboot

Now unmount the newly installed partition.

```
        umount mnt/newroot
```

And reboot to run the new ttylinux system.

4 System Guide

This section gives a short overview of the ttylinux system, its configuration and some of the installed programs.

4.1 Boot Parameters

Boot parameters are typed as a command line in the boot loader, before the Linux kernel is loaded. You will see these options when you boot the ttylinux CD-ROM. The vga=<mode> and modules=<module>[,<module>] boot parameters are not used by **i486** ttylinux.

```
console=<ttyS*> ........... Use serial dev <ttyS*> for console login. The
                            kernel will use it for console output and ttylinux
                            will also put a getty login on it.
                            For <ttyS*> use one of ttyS0, ttyS1, ttyS2 or ttyS3.

vga=<mode> ................ Use VESA frame buffer with <mode>.  Try vga=ask
                            to get a list of modes support by the kernel.
                          vga=0x301 640x480   8-bit | vga=0x161 1152x864   8-bit
                          vga=0x311 640x480  16-bit | vga=0x163 1152x864  16-bit
                          vga=0x303 800x600   8-bit | vga=0x307 1280x1024  8-bit
                          vga=0x313 800x600  16-bit | vga=0x31A 1280x1024 16-bit
                          vga=0x305 1024x768  8-bit | vga=0x31C 1600x1200  8-bit
                          vga=0x317 1024x768 16-bit | vga=0x31E 1600x1200 16-bit

quiet ..................... Don't print a bunch of stuff while booting, but do
                            show anomalies and errors.

enet ...................... Startup networking for Ethernet interfaces found by
                            the kernel. DHCP is used. Any started interface will
                            be eth0, eth1, eth2 or eth3.

nofsck .................... Do not run fsck on any file systems.

nosshd .................... Do not start sshd or make ssh keys; recommended for
                            any CPU slower than 1 Ghz. A script that makes the
                            ssh keys will be left in the /root directory.

modules=<module>[,<module>] ... Load specific kernel module(s) named <module>.
                                Use this to load your sound card module, or
                                secure digital module (sdhci_pci if on
                                the PCI bus), etc.

hwclock=(local|utc) ....... The hardware (CMOS) clock keeps local or UCT time.

tz=<timezone> ............. Set timezone to <timezone> by setting the TZ
                            environment variable. Example: tz=GMT-8  See the
                            following URL for a complete description of TZ.
         http://www.gnu.org/s/libc/manual/html_node/TZ-Variable.html#TZ-Variable

host=<name.domain.tld> .... Set the hostname to <name.domain.tld>, which by
                            this example is a Fully Qualified Domain Name. You
                            can use a simple <name>.

login=<tty*,tty*,...> ..... Allow login on devices e.g., ttyS1, etc. Embedded
```

```
                              system might use this to put a getty login(s) on a
                              serial port(s).

nologin=<tty*,tty*,...> ...   Disallow login on devices e.g., tty1, tty2, etc.
                              Embedded system might use this to prevent a getty
                              login(s) on a virtual console(s).
```

If you've installed ttylinux and are using lilo, or some other boot loader, ttylinux will still use these boot options even if the boot loader does not show them.

4.2 Basic Features

Upon boot-up, ttylinux provides 6 text consoles for login. There are two initial accounts: `root`, the administrator account, with password `password`; `user`, a user account, with password `password`

The `syslogd` and `klog` daemons are running and logging kernel and system messages to the file `/var/log/messages`.

The available text editor is `vi`; invoke it by typing `vi /path/to/filename`. This version of `vi` is a minimal version provided by `busybox`. Documentation and help for using `vi` is available in many places on the web.

For manipulation of users, groups and passwords, the tools `adduser`, `addgroup`, `deluser`, `delgroup` and `passwd` are present.

If you have not changed the keyboard settings as outlined in the configuration section, section 3.4.6 above, ttylinux will use its default keyboard settings. The default keyboard mapping is for a US keyboard.

The `inetd` super-server and the `dropbear` SSH server are running by default. An FTP or TFTP server will be forked by `inetd` when receiving either a connect from an FTP or TFTP client, respectively.

ttylinux has **no** `telnet` server or client; the `dropbear` SSH client is used to remotely log in to other hosts.

ttylinux includes a basic packet filtering firewall which is enabled by default. Section 4.7 below describes the configuring the ttylinux firewall.

4.3 Bootup, Shutdown and System Configuration

On system bootup, the init process runs the `/etc/rc.d/rc.sysinit` script to setup the system, such as setting the clock, system font, keyboard map and checking the file systems. `rc.sysinit` also runs the `/etc/rc.d/rc.local` script *and then* runs all the programs in the `/etc/rc.d/rc.startup` directory, all with the command line parameter `start`.

For ttylinux systems installed by the `/sbin/ttylinux-flash` script, the system startup script `/etc/rc.d/rc.sysinit` attempts to find and mount the drive that is booted from and then run a script on that drive. Remember: this ttylinux system is a RAM disk system, changes made to files are not retained for the next boot because the file system is in RAM. This feature of mounting the booted drive and running a startup script provides the user with a persistent boot-time startup configuration that is retained from one boot to the next. This user-maintainable startup script on the boot drive is in the `config/ttylinux` directory; a default version of this script is put onto the drive when the `/sbin/ttylinux-flash` script is run. This script is a very convenient place to configure your particular network interface for ttylinux startup.

On system shutdown, the script /etc/rc.d/rc.sysdone runs. This script runs all the programs in the /etc/rc.d/rc.shutdown directory *and then* runs the /etc/rc.d/rc.local script, all with the command line parameter stop.

All the programs in /etc/rc.d/rc.startup and /etc/rc.d/rc.shutdown are symbolic links that reference actual shell scripts or binary programs; they are run in the ASCII order of their symbolic link file names. These symbolic links are named with leading numbers to help control their ordering e.g., 10.network is the symbolic link the the network startup program. The actual programs are in /etc/rc.d/init.d. Removing a symbolic link disables the program from starting up. These programs typically are shell scripts; they are commonly called *initscripts*.

Initscripts can be interactively invoked. The following command runs the network script /etc/rc.d/init.d/syslog with the command line option stop.

```
service syslog stop
```

All scripts use the command line options start, stop, reload, restart and status. They print a list of supported options if they are called with no option present.

The initscripts define the basic ttylinux bootup system configuration. The initscripts are configurable to an extent; thus the bootup configuration is configurable to an extent. The bootup system configuration is specified in ASCII text files in the /etc/sysconfig directory; this directory is intended to have only files that are read by the various initscripts. All files read by initscripts for configuration options should reside in /etc/sysconfig.

There are two files in /etc that describe your ttylinux build-time configurations. /etc/ttylinux-host is a text file that describes something about the host architecture that built your ttylinux distribution. /etc/ttylinux-target is a text file that describes some things about the architecture of your running ttylinux distribution.

4.4 Shell Environment

The default shell used by ttylinux is GNU bash. The shell environment of aliases and variables is in /etc/profile; view this file after login to become familiar with the default shell environment.

Upon login, the PATH environmental variable has the following paths in the order listed.

```
/bin
/usr/bin
/sbin
/usr/sbin
```

Put additional, or overriding, shell environment in scripts in the /etc/profile.d directory; do not change /etc/profile in order to avoid losing changes when updating ttylinux.

4.5 Using Dropbear for SSH

SSH, or secure shell, is a protocol for remote login with an advantage over telnet being that it can use public key authentication instead of passwords. Another advantage over the telnet protocol is that plain text is not transfered; the data sent between the host connections is encrypted.

dropbear is a small SSH v2 server and client package. Keys are generated and the server is started on system bootup by default, unless either the ttylinux dropbear starup script detects the CPU is slower than 1 GHz or the nosshd boot options was specified.

dropbear allows password and public key authentication. Public key authentication can use DSS and RSA keys and works with keys generated by the popular OpenSSH package. Having a public key from OpenSSH in the file .ssh/authorized_keys should allow secure login from the machine that has the corresponding private key. The permissions on the .ssh directory must not include group or other write permission, otherwise dropbear will refuse public key authentication.

The SSH client program is called dbclient. It is different from the server in that it cannot use keys in OpenSSH format. You can use the dropbearconvert program to convert an OpenSSH format key for use by dbclient or you can use dropbearkey to create a new key.

To convert an OpenSSH key stored in ~/.ssh/id_rsa, do:

```
dropbearconvert openssh dropbear \
        ~/.ssh/id_rsa ~/.ssh/id_rsa.db
```

The new key will be stored in ~/.ssh/id_rsa.db. You can use the -i switch to dbclient to make it use your new key for authentication. The public key part of the old OpenSSH key can be used as-is for pasting into your ~/.ssh/authorized_keys file. Conversion is only needed for the private key.

To create a new RSA key to store in ~/.ssh/id_rsa.db, you can use the following command:

```
dropbearkey -t rsa -f ~/.ssh/id_rsa.db
```

The public key part of the new key will be printed to the screen. You can put it into the ~/.ssh/authorized_keys file on all machines where you want to be able to login using your new private key stored in ~/.ssh/id_rsa.db. You can create a DSS key instead of an RSA key by using -t dss instead of -t rsa. Should you lose the public key, you can always get it back by using the private key and the -y switch to dropbearkey:

```
dropbearkey -y -f ~/.ssh/id_rsa.db
```

If you want to use scp to copy files from another machine, the standard scp program from OpenSSH is included with dropbear and ttylinux.

4.6 Using an Ethernet Network

ttylinux is ready to use Ethernet networking. DHCP will be used when starting up the Ethernet network, unless configured otherwise.

The Ethernet network interface configuration is specified in the text file:

```
/etc/sysconfig/network-scripts/ifcfg-eth0
```

This file has specification in the form of "ITEM=value". Edit this file to set the proper Ethernet interface IP addresses, change the Ethernet DHCP usage and to enable Ethernet networking. To enable Ethernet networking, the line ENABLE=no must be changed to ENABLE=yes. To disable DHCP, the line DHCP=yes must be changed to DHCP=no.

After configuring the Ethernet network interface, restart the networking subsystem with the following command.

```
service network restart
```

See the description of the /sbin/sysconfig script in section 4.11 for scripted help in setting up the Ethernet network interface configuration.

The Ethernet network interface, commonly referred to as eth0, can be started and stopped independently from the entire network subsystem with the following commands.

Startup eth0 with:

```
ifup eth0
```

Shutdown eth0 with:

```
ifdown eth0
```

Help! I can use only IP addresses and not domain names.
/etc/resolv.conf

If you use only static IP addresses and no DHCP, as specified in your /etc/sysconfig/network-scripts/ifcfg-eth* files, then you probably have no /etc/resolv.conf file and no domain name resolution. In this case only IP addresses will work, such as "ping 212.123.44.77", and domain names will **NOT** work, such as "ping sun.com".

In this case, make your own /etc/resolv.conf file with these contents:

```
# OpenDNS Servers
nameserver 208.67.222.222
nameserver 208.67.220.220
```

The following bash commands will do this for you:

```
rm -f /etc/resolv.conf
>/etc/resolv.conf
echo "# OpenDNS Servers"          >>/etc/resolv.conf
echo "nameserver 208.67.222.222" >>/etc/resolv.conf
echo "nameserver 208.67.220.220" >>/etc/resolv.conf
```

4.7 Using the Firewall

The ttylinux firewall script sets the firewall to drop all new network input except for the ports explicitly specified in the firewall configuration file /etc/firewall.conf. The default firewall configuration specified in /etc/firewall.conf allows connections for FTP, TFTP, SSH, HTTP and the unprivileged UDP ports 1024 through 65535. The /etc/firewall.conf firewall configuration file has a very simple syntax that includes comments; the default file contains comments explaining its syntax and should be easy to understand and update.

Outgoing traffic is not firewalled at all and there is no configuration file for controlling outgoing traffic.

4.8 Using NFS

ttylinux can be an NFS client; NFS versions 2 and 3 are supported.

tylinux is prepared to automatically mount NFS entries you add to the /etc/fstab file.

Example manual commands to mount a NFS directory:

```
mount -t nfs -o nolock,rw,vers=2 <nfs server>:<exported dir> /mnt/nfs
mount -t nfs -o nolock,rw,vers=3 <nfs server>:<exported dir> /mnt/nfs
```

4.9 Using Dialup

ttylinux does not *directly* support dial-up networking with PPP and has no support at all for ISDN. Previous versions of ttylinux did have PPP and ISDN support; their package structure is being re-organized and they may return in a later version of ttylinux.

ttylinux does have the PPP binaries: `/usr/sbin/pppd` and `/usr/sbin/chat`, but currently it is up to you to configure and use them.

4.10 Package Management

Package management is handled by `pacman`; it is invoked from the shell by typing its name, `pacman`. Use `pacman` to install and remove packages, and to query the local database of installed packages and files. When the network is up `pacman` can query, download and install packages from appropriate repositories. `http://ttylinux.net/` currently is the only known ttylinux package repository. `pacman` also can make ttylinux packages, which may be handy as `pacman` is a bash script that can run on any Linux system.

ttylinux packages are `tar` archives compressed with `bzip2`. All the packages that come with the ttylinux distribution are available in the CD-ROM ISO image; this is for reinstalling any packages that may have been removed from a ttylinux system.

A list of hostnames to be used as ttylinux repositories may be kept in file `/etc/ttylinux-repo`; however this file does not need to exist. A package repository hostname can be given to any pacman command that accesses a ttylinux package repository, and `ttylinux.net` is allways the default if no package repository is given. If `/etc/ttylinux-repo` exists and has hostnames in it, `ttylinux.net` will be search first if no package repository is given on the pacman command line. You are not likely to use the `/etc/ttylinux-repo` file, nor the repo option in general, as there currently is no known ttylinux package repository.

`pacman` uses directory `/usr/share/ttylinux` as a database location. In this directory, ttylinux has one file per installed package; each file lists of all the file pathnames that belong that package. `pacman` makes and removes these package database files as needed. Also in the `/usr/share/ttylinux` directory will be similar repository cache files, one each for each ttylinux repository that `pacman` uses.

Pacman Usage
Some of the information from "pacman –help":

```
Usage: pacman [option ...] operation name [name ...]

Options:
  --repo=<name>  refer to a particular external repository
  --vers=<num>   download for ttylinux version V<num>
  -v|--verbose   verbose operation

Operations:
  -h|--help          display this option summary
  -d|--download      download package files
  -e|--erase         remove packages
  -i|--install       install package files
```

```
-m|--make          make a package
-qa|--query-all    list all installed packages
-qf|--query-file   show package that has file
-ql|--query-list   list files from named package
-qr|--query-repo   list packages in external repositiory
```

The pacman command line above shows a specific order for *options*, *operations* and *names*; however, they actually can be arraged in any order.

Options
Options can be repeated. If multiple conflicting options are given, the last one is used and the others are silently ignored. Each option applies only to some operations.

Operations
One operation must be supplied, and only one operation is allowed; all other uses of pacman display a help summary.

Package Names
There are two kinds of package names used with pacman.

Package *download*, *installation*, *make* and *query repo* operations refer to actual binary package file names. These files are tar archives compressed with bzip2.

Package *erase*, *query all*, *query file* and *query list* operations refer to package names without the CPU architecture and .tbz suffix. This shorter name conceptually refers to the package as its files are installed in various directories; it is not a name of a literal package file. With the *erase* and *query list* operations the name is given on the pacman command line. With the *query all* and *query file* operations the name is listed as output from pacman.

4.10.1 Using pacman with ttylinux

Package Download

Examples

```
pacman --download bash-3.2.48-x86_64.tbz lilo-22.8.src-x86_86.tbz
pacman -d bash-3.2.48-x86_64.tbz --repo=ttylinux.org
pacman -d bash-3.2.48-x86_64.tbz lilo-22.8.src-x86_86.tbz --vers=9.1
pacman --repo=palooka.net --download --vers=9.1 bash-3.2.48-x86_64.tbz
```

Use the -d or --download option to download a package from a repository via the network. The package name given to the command is the actual name of the package file. Multiple package names can be used to download multiple packages.

If no repository is given with the --repo= option, then all known repositories are searched. If the --repo= option is used, then only the given repository is search. The first package found matching the given package name is downloaded if there is a ttylinux version match, and then the download command stops; there is no further package search after a download attempt.

The version of the running ttylinux from which the download command is given must match the ttylinux version for a matching package name, otherwise the package is skipped and the download command continues searching the repository. Matching package names are displayed with their ttylinux version.

The --vers= option is used to override the running ttylinux version. For example, with this option you can download a package generated for ttylinux-9.1 while running ttylinux-9.3, which in many cases is a valid option;

furthermore, this option must be used with pacman when running pacman from a non-ttylinux host. See section 4.10.2 for using pacman on a non-ttylinux host.

Package Erase

Examples

```
pacman --erase e2fsprogs-1.41.3
pacman -e e2fsprogs-1.41.3 bash-3.2.48
```

Use the `-e` or `--erase` option to remove an installed package's files and remove the package database file. The *name* is the name of the package as shown by the pacman query operations; this is not the name of the actual binary package file. Multiple package names can be used to remove multiple packages.

`pacman` will show the package name ask to continue to remove the given package, and it will always list all the removed files.

Use the `-v` or `--verbose` option to get verbose output during package removal.

Package Install

Examples

```
pacman -i packages/bash-3.2.48-i486.tbz
pacman -i bash-3.2.48-i486.tbz e2fsprogs-1.41.3-i486.tbz
pacman --install --repo=ttylinux.net bash-3.2.48-i486.tbz
pacman --repo=ttylinux.net --install --vers=9.1 bash-3.2.48-i486.tbz
```

Use the `-i` or `--install` option to install a package by giving the package file name. The package name can be a pathname; the actual package file must be as named in the pacman command unless the `--repo=` option is used. Multiple package names can be used to install multiple packages.

To install from a repository use the `--repo=` option to give the hostname of the ttylinux package repository. Matching package names are displayed with their ttylinux version, but if the package's ttylinux version does not match the running ttylinux then the package is not installed. For the package installation command, the `--vers=` option is used only with the `--repo=` option to override the running ttylinux version. For example, with this option you can install a package generated for ttylinux-9.1 while running ttylinux-9.3, which in many cases is a valid option.

Package Make

Examples

```
pacman -m dropbear-0.52-i486.tbz
pacman -make bash-3.2.48-i486.tbz e2fsprogs-1.41.3-i486.tbz
```

Use the `-m` or `--make` option to make ttylinux packages. This is a very difficult command to use. This command can be used on ttylinux but it is intended to be used on a non-ttylinux host to assemble ttylinux packages.

This command looks for a package database file. The database file name is the package name without the CPU architecture and `.tbz` suffix. For the dropbear example above, the database file name will be `pkg-dropbear-0.52-FILES` and that database file must be in the `/usr/share/ttylinux` directory.

Use the `-v` or `--verbose` option to get verbose output during package making.

See section 4.10.2 for using pacman on a non-ttylinux host to make ttylinux packages.

Query All *List the Installed Packages*

Examples

```
pacman -qa
pacman --query-all
```

Use the `-qa` or `--query-all` option to see the list of all installed packages. This command shows the general package names, not the actual binary package file names. The package names shown by this command are the package names to use with the `pacman` erase command.

Query File *Show the Package to which a File Belongs*

Examples

```
pacman -qf /bin/login
pacman --query-file /bin/ls /usr/bin/pacman
```

Use the `-qf` or `--query-file` option to find out which package a file belongs to. If the given file name does not actually exists on the system there will be no output from `pacman`. If the given file name is not in an installed package, then there will be no output from `pacman`.

Query List *List the Files of an Installed Package*

Examples

```
pacman -ql e2fsprogs-1.41.3
pacman --query-list e2fsprogs-1.41.3 bash-3.2.48
```

Use the `-ql` or `--query-list` option to list all files in the given package.

Query Repository

Examples

```
pacman -qr
pacman --query-repo calc e2fsprogs-1.41.3-i486.tbz
pacman -qr --repo=waldo.net
```

Use the `-qr` or `--query-repo` option to list packages in an external ttylinux package repository.

When the `--repo=` option is used, then only that repository is used; otherwise the `ttylinux.net` repository *and* all the repositories named in the `/etc/ttylinux-repo` file are used.

The package name is the actual name of the binary package file; this is the same package name to use when installing a package.

When no package name is given all packages in the appropriate repositories are listed. This can be bothersome when looking for a particular package, as the name of the package may scroll way off the screen as the repository is listed. Use the name of a package, or a partial name, with this command to limit the output to the package of interest. The package names given with this command can be shortened; the partial name must be from the beginning of the package name, with no gaps or wildcards. For example, a repository query for package name `calc` will find and list only the binary package files beginning with `calc`, listing the actual package names such as `calc-2.12.4.0-i486.tbz` and `calc-2.12.4.0-x86_64.tbz`. All appropriate repositories will be search in this manner.

4.10.2 Using pacman on a non-ttylinux Host

Package Making Issues

The intended use of pacman on non-ttylinux hosts is for assembling ttylinux packages.

Package making seems an easy task as ttylinux packages are tar archives compressed with bzip2. A complicating factor in using pacman to make ttylinux packages is that the package making operation is an exact inverse of package installation, and there is a controlling list of files that comprises the files in the package. This controlling list of files is the database file associated with the package.

A package database file is created by the pacman package installation operation; it is read by the pacman package making operation.

All the package database files are in the /etc/share/ttylinux directory, which your non-ttylinux host probably does not have, nor should you want that directory on your non-ttylinux host.

Also, package installation puts files in directories all over your system, and for the inverse operation of package making you do not want to get files from directories all over your system. Putting files all around your system's directories in order to make a ttylinux package is a bothersome at best, and risky at worst.

Ideally, you would have a staging area, with the ttylinux files for which you want to create a package, under a single convenient private directory. And the package database file in a convenient private directory.

Alternate Directories

pacman can use a user-specified root directory from which it gets the files to make a package, and it can use a user-specified directory to look for the package database file. These user-specified directories, file root and package database directories, are used by **all** the pacman operations except package *download* and *query repo* operations. With this capabilities, pacman can install, query and make packages all with an alternate private root directory, and using an alternate private directory for managing the package database files.

The appropriate way to use pacman on a non-ttylinux host is to set the pacman file root and package database directories to your own alternate private directories. This is done by using environment variables. It may be convenient to use shell scripts in which the environment variables that control the user-specified directories are set, and sequences of pacman commands work within this environment.

There are three environment variables that customize pacman behaviour.

```
PACMAN_FILES_ROOT_DIR  This sets the root directory that pacman uses to install
                       files and also to look for files when making packages or
                       removing packages. The default that pacman uses when
                       this is not set is the system's root directory, /.

PACMAN_PACKAGE_DB_DIR  This sets the directory in which to refer to, make or
                       remove the package database files. The default that
                       pacman uses when this is not set is /usr/share/ttylinux.

PACMAN_REPO_CACHE_DIR  This sets the directory that pacman uses to make
                       repository cache files for all repository operations.
                       The default that pacman uses when this is not set is
                       /usr/share/ttylinux.
```

An example usage of these environment variables for using pacman on a non-ttylinux host is:

```
#!/bin/bash

# Script to merge the bash and busybox packages into one single package.
# The pacman script must be in the $PATH.

export PACMAN_FILES_ROOT_DIR=$(pwd)/p_root
export PACMAN_PACKAGE_DB_DIR=$(pwd)/p_root/usr/share/ttylinux
export PACMAN_REPO_CACHE_DIR=$(pwd)/p_root/usr/share/ttylinux

# Remove everything created by this script so that this script is repeatable.
#
rm -rf p_root
rm -rf new-stuff-i486.tbz

# Make an alternate root directory.
#
mkdir -p p_root/usr/share/ttylinux

pacman --repo=ttylinux.net --install --vers=9.1 bash-3.2.48-i486.tbz
#
# Now these two files are installed:
#       $(pwd)/p_root/bash
#       $(pwd)/p_root/sh -> bash
# And there is a package database file:
#       $(pwd)/p_root/usr/share/ttylinux/pkg-bash-3.2.48-FILES
# And there is a repository cache file:
#       $(pwd)/p_root/usr/share/ttylinux/repo-ttylinux.net

pacman --repo=ttylinux.net --install --vers=9.1 busybox-1.15.3-i486.tbz
#
# Now there are a bunch of files in:
#       $(pwd)/p_root/bin/
#       $(pwd)/p_root/sbin/
#       $(pwd)/p_root/usr/bin/
#       $(pwd)/p_root/usr/sbin/
#       <etc>
# And there is a package database file:
#       $(pwd)/p_root/usr/share/ttylinux/pkg-busybox-1.15.3-FILES
# And there is a repository cache file:
#       $(pwd)/p_root/usr/share/ttylinux/repo-ttylinux.net

# Merge the two packages and make a new single package.
#
cat $(pwd)/p_root/usr/share/ttylinux/pkg-bash-3.2.48-FILES \
    $(pwd)/p_root/usr/share/ttylinux/pkg-busybox-1.15.3-FILES \
    >$(pwd)/p_root/usr/share/ttylinux/pkg-new-stuff-FILES
pacman --make new-stuff-i486.tbz

# Show the new package binary file.
#
ls -hl new-stuff-i486.tbz

exit 0
```

4.11 Using the sysconfig Script

The /sbin/sysconfig shell script can be used to set, and to show, the fields in various ttylinux system configuration files; it can set or show any value for any "ITEM=value" line in any configuration file in the /etc/sysconfig and /etc/sysconfig/network-scripts directories.

The following commands sets "ENABLE=yes" and "DHCP=yes" in the /etc/sysconfig/network-scripts/ifcfg-eth0 file.

```
sysconfig -nc ifcfg-eth0.enable=yes
sysconfig -nc ifcfg-eth0.dhcp=yes
```

The "-nc" option in the above examples tells the sysconfig script to work on files in the /etc/sysconfig/network-scripts directory. The second option is in the form file.item=value.

To change the IP address of the Ethernet network interface, with 192.168.1.100 as the example IP address, with a netmask of 255.255.255.0 and standard subnet gateway and broadcast addresses, use the following sequence of sysconfig script commands.

```
sysconfig -nc ifcfg-eth0.ipaddress=192.168.1.100
sysconfig -nc ifcfg-eth0.network=192.168.1.0
sysconfig -nc ifcfg-eth0.netmaks=255.255.255.0
sysconfig -nc ifcfg-eth0.gateway=192.168.1.1
sysconfig -nc ifcfg-eth0.broadcast=192.168.1.255
```

Use "-sc" for the first option to the sysconfig script in order to work with system configuration files in the /etc/sysconfig directory.

Use the following command to get complete, up-to-date help description directly from /sbin/sysconfig

```
sysconfig --help
```

4.12 Depricated and Legacy Items

4.12.1 Dial-up Networking

ttylinux does not *directly* support dial-up networking with PPP and has no support at all for ISDN. Previous versions of ttylinux did have PPP and ISDN support; their package structure is being re-organized and they may return in a later version of ttylinux.

ttylinux does have the PPP binaries: /usr/sbin/pppd and /usr/sbin/chat, but currently it is up to you to configure and use them.

5 Add-ons

Add-ons packages, such as `thttpd` a tiny web server, **MIGHT** be available at the ttylinux web site. There also are links to any known off-site ttylinux add-on resources. New add-ons submitted to ttylinux will be considered for inclusion at the web site.

The ttylinux package manager, `pacman`, has the ability to install add-ons directly from the ttylinux web site. Section 4.10 describes the `pacman` ttylinux package manager.

6 Contact and Help

Reporting bugs in ttylinux and its documents is appreciated. For bug reports, suggestions, or anything else about ttylinux that you think is important, feel free to contact me. You can reach me by email at:

Douglas Jerome <douglas@ttylinux.org>

There is a web-based forum that is active from time to time; it is active when this was written, April 2010, and is intended to be active as long as minimalinux is supporting ttylinux, barring spammer abuse.

http://www.minimalinux.org/forum/

Help may be available on irc, although it is very low bandwith and usually more appropriate for inane banter.

irc.freenode.net #ttylinux

A ttylinux-specific Commands Overview

Separate from the initscripts in `/etc/rc.d/initd` directory, the following table lists the ttylinux-specific scripts intended to be available for ttylinux root users.

Script	Directory	Usage
ifdown	/sbin	Shutdown Ethernet Network Interface
ifup	/sbin	Startup Ethernet Network Interface
service	/sbin	Execute a script in `/etc/rc.d/init.d`
shutdown	/sbin	Reboot or Shutdown the System
sysconfig	/sbin	Modify a System Configuration File
ttylinux-flash	/sbin	Copy ttylinux to Flash Disk
ttylinux-installer	/sbin	Install ttylinux onto A Disk
pacman	/usr/bin	ttylinux Package Manager

B Flash_Disk_Howto.txt

```
How to Put ttylinux on a Flash Disk and Make it Bootable
Copyright (C) 2008-2010 Douglas Jerome <douglas@ttylinux.org>


FILE NAME

        $RCSfile: Flash_Disk_Howto.txt,v $
        $Revision: 1.11 $
        $Date: 2010/03/01 02:33:11 $

PROGRAM INFORMATION

        Developed by:   ttylinux project
        Developer:      Douglas Jerome, drj, <douglas@ttylinux.org>

FILE DESCRIPTION

        This document is a guide to putting ttylinux on a flash disk and making
        the it bootable.

CHANGE LOG

        28feb10 drj     Corrected for the latest CD-ROM layout and added timeout
                        to the boot loaders to allow for boot options.

        19dec09 drj     Corrected the description of the two required flash
                        drive directories. credit <legendre@nerp.net>

        01sep09 drj     Updated to be consistent with revised ttylinux-flash
                        script and the CD-ROM directory and file structure.

        07dec08 drj     Changed some descriptions for using the syslinux
                        executable program on the ttylinux CD-ROM.

        04dec08 drj     Added suggestions on mounting the CD-ROM and USB disk.

        22nov08 drj     Added failure path descriptions.  Finished testing the
                        installation processes.  Added section numbers and the
                        outline.

        22nov08 drj     Changed ram0 location from flash disk to /tmp.  Fixed
                        the device referenced by the syslinux command.  Added
                        description of lilo's anomalous behavior.  Fixed the
                        fdisk usage in the description of boot problems.

        21nov08 drj     Finished and baselined first version for ttylinux.


                ------------------------------------------------


How to Put ttylinux on a Flash Disk and Make it Bootable
```

```
-- Document Outline --
1. Preface
2. Introduction
3. Lilo Method
4. Syslinux Method
5. Automated Help
6. Boot Problems


==========
1. Preface
==========

Caveat:  The syslinux method is known to work with syslinux-3.72.
Caveat:  Instead of booting ttylinux, your flash disk may become unusable, but
         that is not known to have happened.

Advice:  Read before doing; reading does not take very long.  Look at the end
         of this short document for problems and possible resolutions.


===============
2. Introduction
===============

Flash disks include USB disks which are often called flash drives, pen drives,
USB memory sticks, travel drives, etc.

This file describes two methods of copying ttylinux from its bootable CD-ROM
and putting it onto a flash disk that is also made bootable.  The syslinux
and lilo methods both can be done by ttylinux, but notice the syslinux program
that makes the flash disk become bootable is not in the ttylinux file system,
it is in the root directory on the ttylinux CD-ROM.  These methods probably
only make sense on a Linux system, particularly the lilo method.

You should save all your data on the flash disk to somewhere else and then
remove all files and directories from the flash disk.  Making a mistake in this
process can endanger any data on the flash disk.  Also, if the Linux kernel is
too far from the beginning of the flash disk memory it may not be bootable;
this has nothing to do with where the file name is in a directory listing or in
Windows explorer.

You can format the flash disk to be a Linux file system, but leaving a USB disk
in Windows format, probably vfat aka W95 FAT32, is very convenient.

Prerequisites:  Depending upon the method you use, you need to have privilege
                to write to the flash disk device e.g. /dev/sdc or to write to
                its mountable partition you want to use e.g. /dev/sdc1, and
                with the lilo method you need to create a device node.  It is
                therefore very likely you need to be root.

                You need to *know* the flash disk device node e.g. /dev/sdc and
                its mountable partition you want to use e.g. /dev/sdc1.  Read
                the previous sentence again, note the distinction between the
                disk and partition devices.

In the following descriptions <disk> and <partition> are used to represent
```

```
device nodes in the /dev directory.

<disk> is the device node of the entire flash disk e.g. sdc, in which case
       /dev/<disk> represents /dev/sdc.

<partition> is the device node of the mountable partition on the flash disk
            that you want to use to store the Linux kernel and ttylinux file
            system e.g. sdc1, in which case /dev/<partition> represents
            /dev/sdc1.

In the following descriptions, /mnt/flash references the mount point in your
file system to which the flash disk mounts.  Your actual mount point may be
different, substitute accordingly.

A USB disk partition probably should be mounted with the following mount
command.  The second command gives you the UUID of the mounted partition, it
may not work, but if it does then write down or otherwise save the UUID.

     $ mount -t vfat /dev/<partition> /mnt/flash
     $ blkid /dev/<partition>

/mnt/cdrom represents the location of the mounted CD-ROM in the following
descriptions.

Have the ttylinux boot CD-ROM in the CD-ROM drive and mount it.  The CD-ROM
should be mounted as type iso9660 e.g., mounted by the following command.

     $ mount -t iso9660 /dev/<disk> /mnt/cdrom

If you have an image of the ttylinux CD-ROM mounted via loopback device, or
have the files from the ttylinux CD-ROM in another directory, you can use that.

In the following descriptions there are example commands; they are prefixed by
a shell prompt of "$␣", and comments to shell commands begin with the shell
comment character '#".


==============
3. Lilo Method
==============

Warning:  After performing this method subsequent uses of the syslinux method
          may have no affect, or misboot with odd errors, or the lilo boot
          loader may remain on the flash disk and continue to boot the kernel.
          I've never seen the syslinux method work after using this lilo
          method.  There is a way to fix this; it is described at the end of
          the syslinux method.

Mount the flash disk.  The following description uses /mnt/flash to reference
the mount point of the flash disk.  Did you remember to first save everything
you want to keep off the flash disk and remove everything from it?  After
mounting the flash disk, create two new directories named "boot" and "config"
on the flash disk.

     $ mkdir /mnt/flash/boot
     $ mkdir /mnt/flash/config
```

```
The flash disk should now have nothing on it except the two empty directories
just made, /boot and /config.

Copy the ttylinux Linux kernel and ttylinux file system image from the CD-ROM
onto the flash disk; put them into the boot directory.

     $ cp /mnt/cdrom/boot/vmlinuz          /mnt/flash/boot/
     $ cp /mnt/cdrom/boot/filesys.gz       /mnt/flash/boot/
     $ cp /mnt/cdrom/config/ttylinux-setup /mnt/flash/config/ttylinux

You need a ram0 device node for lilo to reference during the boot installation.
If you don't have one in /dev then you need to make one somewhere; it is better
to NOT make one in /dev in the case your system uses udev.  You can make one in
/tmp with the following command.

     $ mknod -m 660 /tmp/ram0 b 1 0

A lilo configuration file is needed.  It is convenient to put it on the flash
disk in the boot directory; the file is /mnt/flash/boot/lilo.conf.  Use the
following example lilo.conf file, changing <disk> and </mnt/flash> and
</dev/ram0> to be the actual values.  Use ttylinux-flash=<UUID> ONLY if you got
the UUID when previously mounting the USB disk partition, replacing <UUID> with
the actual UUID value.

NOTE The location of the ram0 device is the actual one you want to use; if you
     didn't create one then it probably is /dev/ram0.

NOTE Everything between the dashed lines is the /mnt/flash/boot/lilo.conf file.

-------------------------------------------------------------------------------
boot = /dev/<disk>
disk = /dev/<disk> bios=0x80
map  = </mnt/flash>/boot/map

install     = menu
menu-scheme = Yb:Yk:kb:Yb
menu-title  = "LILO␣(LInux␣LOader)␣boot␣ttylinux"

compact
default = ttylinux
lba32
prompt
timeout = 150

image=</mnt/flash>/boot/vmlinuz
     append = "ro␣ttylinux-flash=<UUID>"
     label  = ttylinux
     root   = </dev/ram0>
     initrd = </mnt/flash>/boot/filesys.gz
     read-only
-------------------------------------------------------------------------------

After the lilo.conf file is correct, execute lilo to make the flash disk
bootable with these two commands.

     $ lilo -M /dev/<disk> mbr
     $ lilo -C /mnt/flash/boot/lilo.conf
```

```
There probably are many possible problems.  If there were no FATAL problems
reported from lilo, unmount and reboot the flash disk.

----------------
Possible Problem
----------------

Lilo may detect a partition problem and give you message like the following:

     Warning: boot record relocation beyond BPB is necessary: /dev/sdc
     Added ttylinux *
     Fatal: LILO internal error:  Would overwrite Partition Table

--------------------
Possible Resolutions
--------------------

If you have this problem you may want to do one of the following:

=> If you are using a USB disk then you can use a Windows-based USB boot disk
   tool; several are freely available.

=> Use a commercial partition tool to fix the flash disk partition table.

=> Use a different flash disk.


==================
4. Syslinux Method
==================

You need to have the syslinux executable program.  The root directory of the
ttylinux CD-ROM should have the syslinux executable program from syslinux-3.72.

Other syslinux sources: You may have it in your current linux distribution.  Or
you can get the latest version from
http://www.kernel.org/pub/linux/utils/boot/syslinux/ and after untarring it,
find the syslinux executable in the linux directory.

Caveat:  The syslinux method is only known by the author to work with
         syslinux-3.72; it probably works with newer versions and a few of the
         older versions.

Mount the flash disk.  The following description uses /mnt/flash to reference
the mount point of the flash disk.  Did you remember to first save everything
you want to keep off the flash disk and remove everything from it?  The flash
disk should now have nothing on it.

NOTE  The following lilo fixup also fixes many USB disks that do not properly
      boot.

NOTE  If you are doing this with a flash disk that previously was booting from
      a lilo boot loader e.g., you previously used the above lilo method, then
      perform this lilo operation before continuing:

      $ lilo -M /dev/<disk> mbr
```

```
Mount the flash disk.  The following description uses /mnt/flash to reference
the mount point of the flash disk.  Did you remember to first save everything
you want to keep off the flash disk and remove everything from it?  After
mounting the flash disk, create some new directories on the flash disk:

     $ mkdir /mnt/flash/boot
     $ mkdir /mnt/flash/boot/syslinux
     $ mkdir /mnt/flash/config

Copy the syslinux help message files from the CD-ROM onto the flash disk.  Copy
the ttylinux Linux kernel and ttylinux file system image files from the CD-ROM
onto the flash disk:

     $ cp /mnt/cdrom/boot/vmlinuz          /mnt/flash/boot/
     $ cp /mnt/cdrom/boot/filesys.gz       /mnt/flash/boot/
     $ cp /mnt/cdrom/boot/boot.msg         /mnt/flash/boot/syslinux/
     $ cp /mnt/cdrom/boot/help.msg         /mnt/flash/boot/syslinux/
     $ cp /mnt/cdrom/config/syslinux       /mnt/flash/config/syslinux
     $ cp /mnt/cdrom/config/ttylinux-setup /mnt/flash/config/ttylinux

A syslinux configuration file is needed.  It must be put on the flash disk in
the boot/syslinux directory; the file is /mnt/flash/boot/syslinux/syslinux.cfg.
Use the following example syslinux.cfg file.  Use ttylinux-flash=<UUID> ONLY if
you got the UUID when previously mounting the USB disk partition, replacing
<UUID> with the actual UUID value.  Everything between the dashed lines is the
/mnt/flash/boot/syslinux/syslinux.cfg file.
long.

--------------------------------------------------------------------------------
default ttylinux
display boot.msg
prompt  1
timeout 150

F1 boot.msg
F2 help.msg

label ttylinux
        kernel /boot/vmlinuz
        append initrd=/boot/filesys.gz root=/dev/ram0 ro ttylinux-flash=<UUID>
--------------------------------------------------------------------------------

Now make the flash disk bootable with syslinux; notice the partition device is
used, not the disk device.

     $ syslinux -d boot/syslinux /dev/<partition>

There probably are many possible problems.  If there were no problems, unmount
and reboot the flash disk.

----------------
Possible Problem
----------------

When executing the syslinux command you see an error message something like
"Cluster␣sizes␣larger␣than␣16K␣not␣supported".
```

```
--------------------
Possible Resolutions
--------------------

Install a more recent version of syslinux.


=================
5. Automated Help
=================

It really is best to use the script described herein.

For the automated help described below, both the CD-ROM and the flash disk must
be mounted before executing the ttylinux-flash script.

There is a shell script in the ttylinux file system that does a variation of
the lilo and syslinux methods.  Backup anything you want to keep from your
flash disk before using the script.  The script is invoked with a command line
option telling it which method to use; guess which option does which.

ttylinux-flash --lilo     <CD-ROM path> <flash disk path> <flash disk device>
ttylinux-flash --syslinux <CD-ROM path> <flash disk path> <flash disk device>

The following command examples use the same conventions as above for the paths
and device nodes.

ttylinux-flash --lilo     /mnt/cdrom /mnt/flash /dev/<disk>
ttylinux-flash --syslinux /mnt/cdrom /mnt/flash /dev/<disk>

If you want to run this script from a Linux system other than ttylinux, then
run it from the ttylinux mounted at /mnt/cdrom, it will be
/mnt/cdrom/sbin/ttylinux-flash.


================
6. Boot Problems
================

General
-------

Some flash disks seem to have a boot problem, something wrong with their zero
block Master Boot Record (MBR).  Run fdisk on the disk device /dev/<disk> to
see if the Boot flag is set on the partition that has the Linux kernel,
/dev/<partition>.

     # Check for the Boot flag
     #
     fdisk -l /dev/<disk>

If the Boot flag is not set, use fdisk to toggle the bootable flag; the fdisk
command is 'a'.  The fdisk usage will look something like the following, if the
partition with the Linux kernel is 1.

     $ fdisk /dev/<disk>
```

```
     Command (m for help): a
     Partition number (1-8): 1
     Command (m for help): w

It also is best to use this lilo command, after having used fdisk to set the
partition bootable flag:

     $ lilo -M /dev/<disk> mbr

Strange Lilo Boot Errors
------------------------

If you get part of the word LILO and then nothing or a repeating sequence of
numbers or words, or if you get "Can't␣load␣operating␣system" or even nothing
at all: put the flash disk back into the computer from wich you where loading
it with ttylinux and try this lilo command:

     $ lilo -M /dev/<disk> mbr

Try bootable again after executing the above command; if the flash disk still
doesn't correctly boot, you may need to repeat either the lilo or syslinux
method of installing ttylinux.


[eof]
```