The Linux Diskless Workstation HOWTO

Sumeet Madhukar Moghe *sumeet@ascentsolutionsonline.com* mailto:sumeet@ascentsolutionsonline.com
v1.0.2, May 2003

This HOWTO is a simplified guide to the creation of secure diskless terminals, using Linux as the underlying operating system. This article assumes no knowledge whatsoever of electronic technology, such as burning ROM chips, and hence concentrates on software aspects alone. This therefore aims to be a quick reference for an inexperienced system administrator who wants to get a cheap yet high-performance network running in no time.

Contents

1 Before we begin

1.1 Why this document?

The content of this document may well beg the above question, in light of the fact that several other HOWTOs and manuals exist on the concerned subject. I, however, believe that those documents either assume an in-depth working knowledge of electronic technology or UNIX system administration or are very cursory in nature. This document attempts to overcome those limitations and cater to less experienced system administrators who have just begun playing with their Linux boxes. In later versions of this document, I also intend to include detailed cost metrics and first-hand statistics on the performance of diskless workstations.

It is an honest attempt to provide in a simplistic form "everything of something" than "something of everything" !!

1.2 For whom is this document

This document is intended for the following audience:-

- Students with a basic knowledge of networking and exploring the virtues of Linux,
- System Administrators,
- Entrepreneurs wishing to reduce network/hardware/maintainence costs, and
- Colleges and educational institutions that wish to provide secure terminals for students in their computer labs

While this document can be perused and its contents executed with little preliminary knowledge of Linux system administration, it would be of help if the readers had a working knowlege of the following:

- TCP/IP and networking with Linux
- Dynamic Host Configuration protocol (DHCP)
- Network File System (NFS) and Trivial File Transfer Protocol (TFTP)
- The working of the X Windows System

Since the following HOWTO is based on a Red Hat 9 system, I guess `http://www.RedHat.com/docs/manuals/linux/RHL-9-Manual/ref-guide/` would be a great starting point for the above information.

1.3 Current versions of this document

The latest tarball of this document can be found *here* `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.tar.gz` .

The LinuxDoc SGML source for the document can be found *here* `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.sgml.gz`

You can read this document online *here* `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.html`

1.3.1 Other formats

Plain text `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.txt.gz`

Info pages `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.info.gz`

Rich text Format (RTF) `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.rtf.gz`

LaTex `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.tex.gz`

Device Independent (DVI) `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.dvi.gz`

Portable Document Format (PDF) `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.pdf.gz`

Postscript `http://www.geocities.com/smm_cipher/Diskless-Workstation/Diskless-Terminals-HOWTO.ps.gz`

1.4 About the author

I am Sumeet Madhukar Moghe; a die-hard FLOSS evangelist. I live by the day as a student of the arts (sic!) and by night as a hacker. My association with Linux dates back to 1998 and it has been a love story ever since. My home on the web can be found at `http://smith-morgan.mirrorz.com` .

1.5 Whom to contact for help

Before sending out an SOS signal to me (*sumeet@ascentsolutionsonline.com* `mailto:sumeet@ascentsolutionsonline.com`) be sure that your problem, with respect to this document, has not been already solved elsewhere. Search the web; *Google* `http://www.google.com` is a good starting point.

Join a Linux User Group somewhere in the vicinity and post your problems there for a solution. Read *this page* `http://www.catb.org/~esr/faqs/smart-questions.html` to get a good idea on how to ask smart questions on Usenet.

The best way to learn however, is to **RTFM** = **Read The Fine Manual**.

I hope this document will be useful to you. Happy hacking!!

2 Introduction

If you or your organisation has been working with computers for some time now, then it is pretty much assumed that over the years, you have retired many a computer, and now have a warehouse full of those supposedly useless machines. Did you know that those discarded boxes could be put to better use? Or that each of those boxes could be made to run a modern operating systems at nearly no cost, whatsoever?

Surprise, surprise!! Linux can breathe life into that pathetic 486 of yours! And what's more: it could run at a decent clip compared to a Pentium III! The idea is as simple as it gets. Take a simple bare-bones PC with some memory, a cheap display card and a decent network card, and turn it into a workstation for office work or surfing the net, or for scientific computing. In educational organisations you could use it as a terminal for students to perform their practicals. Interested?? Read on...

2.1 A historical perspective

Way back in the late 1970s and 1980s, memory and hard disk space were insanely expensive, and there arose a need to find ways and means to make workstations more affordable. However, as displays became more and more sophisticated and preliminary GUI applications replacing their legacy character-based counterparts, this task seemed more and more daunting.

In those days of UNIX, universities and large corporations found a method to do this. They used the X Window System and gave birth to a new breed to displays - X Terminals. These systems had no local storage and very little memory. All they were meant to do was to run an X display for their own purposes in order to support applications running on the server itself.

As workstations became more powerful, the need arose to run applications using the processing power of the workstation alone. Storage, however, continued to be expensive and hence servers continued to handle the storage for the networks.

Soon these terminals, were the "in thing", finding favor with corporates and small organizations, alike. Management of such networks was simple with a centralized server being all that needed to be configured and maintained. A single point failure in the network was easy to rectify, and terminals that failed could easily be swapped with working systems.

But as processing power increased to deal with computing demand, networks could not cope with the immense traffic generated by simulataneous read-write operations from various hosts. With secondary storage becoming cheaper, diskless systems were soon things of the past and the benefits of centralized computing were lost.

However with the recent increase in network bandwidth, diskless systems have seen a sort of a revival and have found usage in many a computing applications, such as public reservation systems, information kiosks, set-top boxes, etc. Diskless nodes are once again being considered as a tremendous cost-saving exercise and centralized computing is seeing a renaissance.

2.2 Look Ma...No disk!!

A truly diskless system can eliminate the following storage devices:

- Hard Disk
- CD-ROM Drive
- Floppy Drive

With a diskless setup, even the need for a monitor can be elimnated. The above setup, however, would need the expertise of burning the boot images on to a ROM chip, which (as indicated earlier), would be beyond the scope of this article. We shall therefore look at other options involving a single, removable storage device.

2.3 The pros and cons of going diskless

2.3.1 The Advantages

Diskless systems have many advantages to offer:

1. **Diskless sytems offer a very low-cost solution.**
    - The RAM on the central server is shared by many clients.

A single copy of a program is kept in memory and used by multiple clients. For example, in an internet cafe, despite the fact that various users on the network use a web browser, only one copy of the web browser is kept in memory, thereby eliminating wasted memory. Thus, by pooling memory on the server, memory costs can be reduced tremendously.

- There is no need for a UPS or airconditioned environment for the diskless clients. Providing the server alone, with a UPS, serves the purpose.
- The cost of the workstations, with typically low requirements, is minimal.

2. **With no local storage, diskless systems can be made very secure** These are ideal machines for students to experiment with. There is no local filesystem to corrupt if the machine crashes. Files on the server can be easily controlled or replaced. Diskless workstations are also safe from viruses, which function by copying themselves to the hard disk. The risk is limited to the central server.

3. **Administration of a group of diskless systems is simpler.**

   As discussed earlier, because administration is centralised, workstations can be added and removed with great ease. Backups need to be maintained only at the server end, and all configuration can be performed centrally. This is particularly useful when the network is spread over a campus or over several offices in a company.

4. **Diskless computers offer high-speed performance.**

   With the elimination of program loading time, diskless computers tend to work very fast. Once someone has loaded a memory hog like Star Office/Open Office into memory, the program loads in a flash for subsequent users.

2.3.2 The Disadvantages

With the advent of modern 100 MBit and 1 Gigabit ethernet cards, network bandwidth is no longer a consideration for the performance of diskless systems. However, with distributed computing having played a significant role in the proliferation of desktop PCs, the totally server-based solution represents a single point of failure. However, when compared to the central dependency that many modern systems have on various specialised, central servers, the diskless scenario doesn't look much worse. The bottom line is: *Take care of your server!!*

3 The Technology under the hood

To build a diskless network, it is important to understand how such a system works. This section deals with the basics of how such a system functions.

3.1 How does it work ?

Each time it boots up, the diskless system follows these steps:

1. The BIOS, configured to boot from either a floppy or a network card's BOOT ROM, reads the boot information from the specified device.
2. The media then does the following

- Broadcasts its MAC address on the network, searching for a DHCP server.
- Receives an IP address from the DHCP server and configures the NIC accordingly.
- Fetches a predefined kernel image from the server using TFTP (Trivial File Transfer Protocol).
- Loads this kernel image into the RAM.

3. Once the kernel image is loaded into the RAM, the root file system is fetched via NFS.
4. Depending on the methodology adopted, system initialization scripts are run and the system is made ready for use.

3.2 Methods of achieving diskless systems

Diskless systems can be achieved in 3 ways:-

1. 4 (LTSP) The Linux Terminal Server Project.
2. Network Booting with 5 (Etherboot)
3. 6 (CD Based distributions)

4 LTSP - The Linux Terminal Server Project

The LTSP provides a simple way to utilize low-cost workstations as either graphical or character based terminals on a GNU/Linux server. Using the LTSP, you can take very low-end PCs, remove the hard drives and CD-ROMs, and add floppies or bootable network cards. Many network cards have bootrom sockets just waiting for a bootrom to be inserted. During the boot phase, the diskless workstation obtains its IP info and a kernel from the server, then mounts the root filesystem from the server via NFS.

A workstation using the LTSP can be configured in the following ways:

1. **Graphical Workstation**The workstation uses the X-Window System to access applications on the Linux Terminal Server
2. **Character-Based Telnet Session**Multiple telnet sessions are invoked by the client to the telnet server. Users can switch between telnet sessions by pressing Alt+F1 through Alt+F9. However, telnet being an extremely insecure service, the use of this mode is strongly discouraged.
3. **Shell Prompt**This basically drops the user in a minimalistic bash prompt. This mode is basically a diagonistic or troubleshooting mode that helps debug problems with X or NFS.

Depending on the size of the server and the applications being run, a number of clients can be served by a single Linux server. The LTSP official documentation tells us: "*It's not unusual to have 40 workstations, all running Netscape and StarOffice from a Dual PIII 650 with 1GB of ram. We know this works. In fact, the load average is rarely above 1.0!*"

4.1 Configuring the Linux Terminal Server

Let us now look at how to configure LTSP on a Red Hat 9 server. Perform all the underlined steps as **root**

4.1.1 Packages required

The packages required for LTSP are available at *ltsp.org* `http://www.ltsp.org` . To start, you will need the following packages:

1. ltspcore-x.x.x.i386.rpm
2. ltsp_kernel-x.x.x.i386.rpm
3. ltsp_x_core-x.x.x.i386.rpm
4. ltsp_x_fonts-x.x.x.i386.rpm

Where x.x.x is the version of the package set downloaded.

4.1.2 Installing the packages

Install the packages as follows:

```
[root@quaker install_scripts]#rpm -ivh ltspcore-x.x.x.i386.rpm
[root@quaker install_scripts]#rpm -ivh ltsp_kernel-x.x.x.i386.rpm
[root@quaker install_scripts]#rpm -ivh ltsp_x_core-x.x.x.i386.rpm
[root@quaker install_scripts]#rpm -ivh ltsp_x_fonts-x.x.x.i386.rpm
```

4.1.3 Running the distro-specific install script

Once the packages are installed, change to the directory `/opt/ltsp/install_scripts/` and run the script `redhat.sh`

```
[root@quaker install_scripts]# cd /opt/ltsp/install_scripts/
[root@quaker install_scripts]# sh redhat.sh

Take a look in /tmp/ltsp.install.log for a complete log of the installation.

You now need to change to the /opt/ltsp/templates directory and
run the ltsp_initialize script to complete the installation
```

4.1.4 Initializing the Services

Now change to the directory `/opt/ltsp/templates` and run the script `ltsp_initialize`. Follow the steps as indicated below.

```
[root@quaker install_scripts]# cd /opt/ltsp/templates/
[root@quaker templates]# ./ltsp_initialize

The Linux Terminal Server Project (http://www.LTSP.org)
```

```
    About to update important system files.  If you would like
    to stop and review the changes that are about to be made,
    you can cancel now and look at the replacement files that
    are about to be installed.

Press <ENTER> to go on, or 'C' to cancel
```

<ENTER>

```
The Linux Terminal Server Project (http://www.LTSP.org)

The following files will be created/modified:

 /etc/X11/xdm/Xaccess              The config file to allow remote xdm log [Y]
 /etc/X11/xdm/Xservers             Config file for xdm to launch local Xse [Y]
 /etc/X11/xdm/Xsetup_workstation   Sets the logo of your login window      [Y]
 /etc/dhcpd.conf.example           Example config file for dhcp            [Y]
 /etc/exports                      The config file for nfs                 [Y]
 /etc/X11/gdm/gdm.conf             The config file for gdm                 [Y]
 /etc/X11/gdm/Init/Default         The gdm startup script                  [Y]
 /etc/hosts.allow                  Configuration file for tcp wrappers     [Y]
 /etc/inetd.conf                   Config file for inetd                   [Y]
 /etc/inittab                      Config file for init                    [Y]
 /etc/kde/kdm/kdmrc                The config file for kdm                 [Y]
 /etc/X11/xdm/ltsp.gif             The background logo for your login      [Y]
 /etc/rc.d/rc5.d/S60nfs            Startup links for nfs                   [Y]
 /etc/rc.d/rc5.d/S13portmap        Startup links for portmapper            [Y]
 /etc/rc.d/init.d/syslog           Startup script for syslogd              [Y]
 /etc/xinetd.d/tftp                Enable the tftp daemon                  [Y]
 /etc/X11/xdm/xdm-config           The main config file for xdm/kdm        [Y]

Ready to apply the changes? ( R-Review, A-Apply, C-Cancel )
```

A<ENTER>

```
The Linux Terminal Server Project (http://www.LTSP.org)

Doing the update
Xaccess.tmpl
Saving old /etc/X11/xdm/Xaccess as /etc/X11/xdm/Xaccess.2
Xservers.tmpl
Saving old /etc/X11/xdm/Xservers as /etc/X11/xdm/Xservers.2
Xsetup_workstation.tmpl
Saving old /etc/X11/xdm/Xsetup_workstation as /etc/X11/xdm/Xsetup_workstation.1
dhcpd.tmpl
Saving old /etc/dhcpd.conf.example as /etc/dhcpd.conf.example.1
```

```
Creating new dhcpd.leases file
Saving old /etc/exports as /etc/exports.2
gdm.conf.tmpl
Saving old /etc/X11/gdm/gdm.conf as /etc/X11/gdm/gdm.conf.2
gdm_Init_Default.tmpl
Saving old /etc/X11/gdm/Init/Default as /etc/X11/gdm/Init/:0
hosts.allow.tmpl
Saving old /etc/hosts.allow as /etc/hosts.allow.2
inetd.tmpl
Saving old /etc/inetd.conf as /etc/inetd.conf.1
inittab.tmpl
Saving old /etc/inittab as /etc/inittab.2
kdmrc.tmpl
Saving old /etc/kde/kdm/kdmrc as /etc/kde/kdm/kdmrc.2
ltsplogo.tmpl
Saving old /etc/X11/xdm/ltsp.gif as /etc/X11/xdm/ltsp.gif.1
nfs.tmpl
Saving old /etc/rc.d/init.d/syslog as /etc/rc.d/init.d/syslog.2
tftpd.tmpl
xdm-config.tmpl
Saving old /etc/X11/xdm/xdm-config as /etc/X11/xdm/xdm-config.2
```

4.1.5 Configuring the DHCP daemon

Bingo!! That sure does take care of a lot of stuff! Now all we need to do is perform some minor changes to a few configuration files. The first file I usually modify is `/etc/dhcpd.conf`.

In this example, I assume a server with IP address `192.168.100.13` and a diskless workstation which will be assigned the IP address `192.168.100.14` and hostname `ws001`. This workstation's ethernet card has a MAC address `00:0B:2B:0B:6B:2E`. The netmask is `255.255.255.0` and the domain name is `gamingheaven.com`.

In order to do this, open the file `/etc/dhcpd.conf` and make it read as follows (rather, change it to suit your setup ;)). This is a recommended minimum configuration:

```
ddns-update-style             ad-hoc;
default-lease-time            21600;
max-lease-time                21600;

option subnet-mask            255.255.255.0;
option broadcast-address      192.168.100.255;
option root-path              "192.168.100.13:/opt/ltsp/i386";
#The above line specifies where to fetch the root file system from

shared-network WORKSTATIONS {
    subnet 192.168.100.0 netmask 255.255.255.0 {
    }
}
```

```
group   {
    use-host-decl-names       on;
    option log-servers        192.168.100.13;
#configuring the first host
    host ws001 {
        hardware ethernet     00:0B:2B:0B:6B:2E;
        fixed-address         192.168.100.14;
        filename              "/lts/vmlinuz-2.4.9-ltsp-5";
#We need to copy the above file to the tftpboot directory.
    }
}
```

4.1.6 Configuring hostname mapping

Computers usually love communicating with numbers. Human beings on the other hand prefer names because we find them easier to remember. This is where the file `/etc/hosts` or DNS comes into play. Theoritically, IP address to hostname mapping should not be required. In an LTSP environment, however, we can't do without it because NFS throws permission errors when the workstation tries to mount the root file system over the network. We just need to add a single line to `/etc/hosts` describing the workstation, so that surely won't be too painful! In my case the line would look like this:

```
ws001.gamingheaven.com  ws001 192.168.100.14
```

Workstation-specific configurations We now need to edit the global LTSP configuration file `/opt/ltsp/i386/lts.conf` and perform the following steps. Again, this is a recommended minimum configuration.

1. In the default section edit the line containing SERVER and change it to read SERVER = <your server ip>

   In my case, it reads:

   ```
   SERVER = 192.168.100.13
   ```

2. In the section [ws001] (which you should change to the hostname you set in `/etc/dhcpd.conf`), you can play with the following options:

   - RUNLEVEL: Set this to 3 for a **Shell Prompt**, 4 for **Character based Telnet Session** and 5 for a **Graphical Workstation**
   - XSERVER: Set this if you want to force a specific X-Server module to be loaded for your workstation For example:

     ```
     XSERVER = vesa #This forcibly loads the generic vesa driver module for you X-Server
     ```

- X_MODE_0 through X_MODE_2: Up to 3 Modelines or resolutions can be configured for the workstation. This entry can take two different types of values: a resolution or a complete modeline.For example, if you have a really old monitor, you could use the following modelines. (Use Ctrl+Alt+KP+ to cycle between the resolutions in the same order as you list them here.)

```
X_MODE_0 = 640x480  31.5 640  664  704  832  480 489 492 520 +hsync +vsync
X_MODE_1 = 800x600  40   800  840  968  1056 600 601 605 628 +hsync +vsync
X_MODE_2 = 1024x768 44.9 1024 1048 1208 1264 768 776 784 817 interlace
```

- USE_XFS: Set this to Y if you wish to use the X Font server.
- MODULE_xx: (where xx is a number starting from O1) Set this to the name of a module you wish to load for your workstation. For example, I use the following line to initialize a Yamaha soundcard on one of my workstations:

```
MODULE_01       = opl3.o
```

- To setup printers attached to the workstation, use the parameters PRINTER_0_DEVICE and PRINTER_0_TYPE. Set the former to the device identity of the printer and the the latter to "S" for serial printers and "P" for parallel port printers. To add other printers use PRINTER_1_DEVICE and PRINTER_1_TYPE and so on. You can add up to 3 printers. For example:

```
PRINTER_0_DEVICE = /dev/lp0
PRINTER_0_TYPE = P
```

There are a number of other such parameters to play with, but discussing them all would be beyond the scope of this document.

4.1.7 Setting the Default Display Manager

The `ltsp_initialize` script does a great job handling all the scripts, but makes a the mistake of setting XDM as the display manager for the clients. I personally enjoy the insane eye-candy of GDM. To use GDM, open the file `/etc/inittab`, scroll down to the line that reads `x:5:respawn:/usr/X11R6/bin/xdm -nodaemon` and change it to:

```
x:5:respawn:/usr/bin/gdm -nodaemon
```

Well, that's it for the server configuration. Switch to runlevel 5.

```
[root@quaker etc] init 5
```

Whoa!! Your LTS is up and running. We still need to work a little to get our workstation up.

4.2 Building the bootdisk for the workstations.

Making the floppy is usually as simple as downloading the boot image from Marty Connor's `http://www.rom-o-matic.net` . Marty has done an execellent job with the site: it generates boot images on the fly when you enter your network card chip model. The default configuration usually doesnt need changing. For my Realtek 8139 cards (rt8139), I got the following boot image:

```
eb-5.0.8-rtl8139.lzdsk
```

To create the bootdisk, change to the directory containing the downloaded boot image and run the following command:

```
[sumeet@quaker sumeet]$ dd if=eb-5.0.8-rtl8139.lzdsk of=/dev/fd0
```

You could also use Etherboot to locally build boot images for various NICs. Etherboot is a package for creating ROM images that can download code over an Ethernet network, to be executed on an x86 computer. It can be downloaded freely from `http://etherboot.sourceforge.net/distribution.html` .

Creating boot images with Etherboot involves the following steps:

1. Untar and unzip the sources from the etherboot package
2. Dive into the "src" directory at the location where you untarred the package.
3. Insert a floppy into the floppy drive and run the following command:`make bin32/<network-card-name.fd0>`

   For example, with my RTL 8139 cards I need to execute the following steps

```
[sumeet@quaker sumeet]$ tar -zxvf etherboot-5.0.8.tar.gz
[sumeet@quaker sumeet]$ cd etherboot-5.0.8/src/
[sumeet@quaker src]$ make bin32/rtl8139.fd0
```

Now that your disk is ready, configure the BIOS of your diskless workstations to boot from the floppy and boot them up using this disk. You should get the GDM login window, with an LTSP.org background. Login and enjoy!!

4.3 Shutting down the diskless system

Shutting down a diskless node couldn't be easier. Just press the power switch and walk away. That's it!! Now isn't that easy?

5 Network Booting with Etherboot

This is a slightly more complicated method of getting diskless systems to work. However, once configured, it allows greater flexibility than LTSP. It allows users access to six terminals in addition to the standard X session.

bf	Boot File
dn	Domain Name
ds	Domain Name Server address List
ha	Host Hardware Address
ip	Host's IP Address
hn	Send client's hostname to client
ht	Hardware Type
sa	TFTP server address
rp	Root Path to mount as root
tc	Table Continuation

Etherboot works on the same principle as LTSP, but it uses BOOTP (Boot Protocol) instead of DHCP. A server running BOOTP will receive the request for an IP address and return the IP address the client system is to use. Since the BOOTP request is *broadcast* – that is, addressed to all computers on the network – the client doesnt need to know the address of the server.

Note: BOOTP is not a standard part of the Red Hat 9 distribution; you must download it from `http://www.rpmfind.net` .

5.1 Configuring BOOTP

Configuring BOOTP is extremely simple compared to DHCP. You just need to remember the following commonly used fields and flags:

Based on this, my `/etc/bootptab` file looks like this:

```
.def:ht=ethernet:hn=y
ws001:ha=000B2B0B6B2E:ip=192.168.100.14:tc=linux

linux:bf=/tftpboot/linux.diskless
```

Setup BOOTP to start up automatically in runlevel 5

```
[root@quaker root]#chkconfig --level 5 bootp on
```

5.2 Configuring TFTP

Edit the file `/etc/xinetd.d/tftp`. Change the line that reads:

`disable = yes`

to read:

`disable = no`

and the line that reads

`server_args = -s /tftpboot`

to read:

```
server_args = /tftpboot
```

Finally, setup TFTP to start automatically in runlevel 5

```
[root@quaker root]#chkconfig --level 5 tftp on
```

5.3 Building and configuring the kernel image for the Diskless Systems

We now need to build the kernel image that will be loaded into the RAM of the diskless system.

5.3.1 Configuring the kernel

Dive into the kernel sources directory and run `make menuconfig` Then do the following:

1. Enable the option [General Setup–>Networking Support].
2. Enable the option [File systems–>Network File Systems–>NFS file system support] and [File systems–>Network File Systems–>NFS file system support–>Root over NFS].
3. Enable the options [Networking Options–>IP: kernel level autoconfiguration] and [Networking Options–>BOOTP support].
4. Build your network card drivers into the kernel instead of selecting them as modules: [Network Device Support–>Select your card driver]

5.3.2 Building the Kernel

It takes just one command to build the kernel:

```
[root@quaker linux2.4]# make dep clean bzImage
```

This leaves a your new kernel image at `/usr/src/linux2.4/arch/i386/boot/bzImage`. Move this to the `/tftpboot` directory.

```
[root@quaker linux2.4]# mv i386/boot/bzImage /tftpboot/kernel.diskless
```

5.3.3 Making the kernel network bootable

We now need to explicitly make sure that the kernel is meant to boot from the network rather than a hard disk. To ensure this, we run the following set of commands:

```
[root@quaker linux2.4]# cd /tftpboot
[root@quaker tftpboot]# mknod /dev/nfs c 0 255
[root@quaker tftpboot]# rdev kernel.diskless /dev/nfs
```

5.3.4 Creating a Tagged Image

The last step that remains is to create the tagged image. A kernel image intended for a diskless client needs a special header which tells the network bootloader where the bytes go in the memory and at what address to start the program. This image is called the tagged image. Creating a tagged image is simple; all we need is the `mknbi-linux` tool from `the etherboot` package:

```
[root@quaker tftpboot]# mknbi-linux kernel.diskless > linux.diskless
```

5.4 Configuring NFS

A diskless system configured with IP address 192.168.100.14 attempts to mount the root filesystem from the location /tftpboot/192.168.100.14. We therefore add the following workstation-specific lines to `/etc/exports`

```
/tftpboot/192.168.100.14        ws001(rw,no_root_squash)
/usr    (ro)
/opt    (ro)
```

As you can see above, we export the directory `/tftpboot/192.168.100.14` with read+write options and allowing root access to the workstation ws001 (4.1.6 (Hostname Mapping) needs to be configured here). The /usr and /opt directories are made available in order to allow workstations to access installed programs. Finally, make the exported file systems available and setup NFS to start up automatically in runlevel 5:

```
[root@quaker root]# exportfs -a
[root@quaker root]# chkconfig --level 5 nfs on
```

5.5 Initializing the root filesystem

A minimal root filesystem for the client should contain the following directories in their entirety: /etc - Configuration files /var - Log files /lib - Libraries used at boot time /dev - device files

A simple script, `buildroot` automates the process of creating the root filesystem with the minimum of fuss. The script is as follows

```
#!/bin/bash
if [ $# != 1 ]
then
        echo Usage: $0 client-IP-address-or-name
        exit 1
fi

cd /
umask 022
mkdir -p /tftpboot/$1
```

```
for d in home mnt proc tmp usr opt
do
        mkdir /tftpboot/$1/$d
done

chmod 1777 /tftpboot/$1/tmp

touch /tftpboot/$1/fastboot
chattr +i /tftpboot/$1/fastboot
cp -a bin lib sbin dev etc root var /tftpboot/$1
cat << EOF
Now, in /tftpboot/$1/etc, edit

sysconfig/network
sysconfig/network-scripts/ifcfg-eth0
fstab
hosts
modules.conf

and configure rc.d/rc3.d
EOF
```

Move the script to `/usr/bin`. The script can be invoked from the command line, like this:

```
[root@quaker tftpboot] buildroot 192.168.100.14
```

5.5.1 Setting up NFS mounts

Now it's time to set up the filesystems exported from NFS. We do this in order that the client should mount them. The file `/etc/exports` should be set up as shown below:

```
192.168.100.13:/tftpboot/192.168.100.14 /       nfs     rw      0 0
192.168.100.13:/usr     /usr    nfs     ro      0 0
192.168.100.13:/opt     /opt    nfs     ro      0 0
proc    /proc   proc    defaults        0 0
devpts  /dev/pts        devpts  defaults        0 0
```

5.5.2 Setting up the network scripts

Set the values of HOSTNAME and IPADDR in the files `/tftpboot/192.168.100.14/etc/sysconfig/network` and `/tftpboot/192.168.100.14/etc/sysconfig/network-scripts/ifcfg-eth0` respectively. Finally, delete unwanted scripts in `/tftpboot/192.168.100.14/etc/rc.d/rc3.d/`.

5.6 Setting up the display manager

Setting up GDM as the display manager usually involves 2 simple steps:

1. Open the `/etc/X11/gdm/gdm.conf` file and scroll down to the section [xdmcp]. Add this line:

```
Enable = true
```

2. Open the file `/etc/inittab` and replace this line:

```
x:5:respawn:/etc/X11/prefdm -nodaemon
```

with this one:

```
x:5:respawn:/usr/bin/gdm -nodaemon
```

to force GDM as the display manager.

All that needs to be done now is to switch to runlevel 5:

```
[root@quaker root]# init 5
```

5.7 Building the bootable disk for the client

4.2 (Building the bootdisk for the diskless client) is the same as described for the 4 (LTSP configuration). Just pop in the disk and boot up your clients.

5.8 Running X on the client

To run X on the diskless node, execute the following command:

```
$ X -query 192.168.100.13
```

This should give you a GDM login window. Login and enjoy your remote X session. You can also switch between X and the 6 consoles from Ctrl+Alt+F1 through F6.

6 CD Based Distributions - Rapid Upgrade diskless systems

CD-based distributions and Live! Linux CD-ROMs are fast becoming popular ways to implement diskless nodes. These distributions use a compressed file system and boot off a CD-ROM. Note that they require over 128 MB of RAM for satisfactory performance, and they eliminate the possibility of workstation users having any storage facilities. They also depend on a pre-configured DHCP server if they are to have any kind of network capability. Also, for internet access, the additional task of configuring a proxy server is introduced. Despite these inherent limitations, CD-based distros are popular for the singular reason that they offer a *painless software upgrade scheme*: to get a completely new operating system, the user just trashes the old CD and starts booting from the new one.

A few extremely popular CD based distros are listed below:

- *Knoppix* `http://www.knopper.net`
- *Slackware Live!* `http://www.slackware.org`
- *Linux Bootable Business Card* `http://www.lnx-bbc.org`
- *DyneBolic Linux* `http://lab.dyne.org/DyneBolic`
- *Demo Linux* `http://www.demolinux.org`

7 Online References

- Official LTSP Documentation `http://www.ltsp.org/documentation/ltsp-3.0-4-en.html`
- Documentation index of the Etherboot project `http://etherboot.sourceforge.net/doc/html/documentation.html`
- Building a self-contained auto-configuring Linux system on an iso9660 filesystem `http://www.knopper.net/knoppix/knoppix-als2000-paper-html/`
- The LDP Diskless Nodes HOWTO `http://www.tldp.org/HOWTO/Diskless-HOWTO.html`

8 Copyright Notice