NFS−Root Mini−Howto

Table of Contents

NFS−Root Mini−Howto

Andreas Kostyrka, `andreas@ag.or.at`

V8, 8 August 1997

This Mini−HOWTO tries explains how to setup a ``disc−less'' Linux workstation, which mounts it's root filesystems via NFS. The newest version of this Mini−Howto can always be found in `ftp://sunsite.unc.edu/pub/Linux/docs/HOWTO/mini/NFS-Root` *or on any sunsite mirror NEAR YOU.*

---

1. Copyright

If you have questions, please contact Andreas Kostyrka `<mailto:andreas@ag.or.at>`, the author of this mini−HOWTO, or Tim Bynum, the Linux HOWTO coordinator, at `<mailto:linux−howto@sunsite.unc.edu>` via email.

1.1 Contributors

- Avery Pennarun `<apenwarr@foxnet.net>` (how to boot without **LILO**)
- Ofer Maor `<ofer@hadar.co.il>` (providing a better mini howto about setting up discless workstations.)
- Christian Leutloff `<leutloff@sundancer.tng.oche.de>` (providing infos about netboot.)

---

2. General Overview

Generally speaking there are the following problems for the workstation:

- It must find out it's own IP−address, and if needed also the rest of the Ethernet configuration.
- It must know the *NFS*−server and the mount path to it's root filesystem.

The current implementation of *NFSROOT* in the Linux kernel (as of 1.3.7x) allows for the following ``solutions'':

- The IP−address may be discovered by *RARP*, or the full ethernet configuration may be passed to the kernel via kernel parameters by **LILO** or **LOADLIN**.
- The *NFS*−path to mount can be passed via kernel parameters. If this is not done, the kernel assumes the *RARP*−server also as *NFS*−server, and uses compiled in default for the path part. (current default value in the kernel: `/tftpboot/<`*`IP−address of the machine`*`>`.)
- The client configuration may be discovered by *BOOTP*.

Before starting to setup a discless enviroment, you should decide if you will be booting via **LILO** or **LOADLIN**. The advantage of doing so is flexibility, the disadvantage is speed. Booting a Linux kernel without **LILO** is faster. This may or may not be a consideration.

---

3. Setup on the server

3.1 Compiling the kernels

RARP support in the kernel of the server will probably be a good idea. You must have it if you will boot without kernel parameters. On the other hand it doesn't help you, if the client isn't on the same subnet than the server.

The kernel for the workstation needs the following as a minimum set compiled in:

- *NFS*−filesystem compiled in. (It doesn't need to have *ext2*−support compiled in, a module suffices.)
- ``Root on NFS'' must be enabled.
- The Ethernet driver for the network card of the workstation must be compiled in.
- Depending upon your needs you may have to include *RARP* or *BOOTBP* support for NFS−Root. (By this I mean the questions that are asked after the NFS question in make config.)

If the workstation will be booted without kernel parameters, you need also to set the root device to 0:255. Do this by creating a dummy device file with mknod /dev/nfsroot b 0 255. After having created such a device file, you can set root device of the kernel image with rdev <*kernel−image*> /dev/nfsroot.

3.2 Creation of the root filesystem

Copying the filesystem

Warning: while these instruction might work for you, they are by no means sensefull in a production enviroment. For a better way to setup a root filesystem for the clients, see the NFS−Root−Client mini howto by Ofer Maor <ofer@hadar.co.il>.

After having decided where to place the root tree, create it with (e.g.) mkdir −p <*directory*> and tar cClf / − | tar xpCf <*directory*> −.

If you boot your kernel without LILO, then the rootdir has to be /tftpboot/<*IP−address*>. If you don't like it, you can change it in the top Makefile in the kernel sources, look for a line like: NFS_ROOT = −DNFS_ROOT="\"/tftpboot/%s\"" If you change this, you have to recompile the kernel.

Changes to the root filesystem

Now trim the unneeded files, and check the /etc/rc.d scripts. Some important points:

- One important thing is eth0 setup. The workstation comes up with a, at least partially, setup eth0. Setting up the IP−address of the workstation to the the IP−Address of the server is not considered a clever thing to do. (As it happened to the author on one of his early attempts.)
- Another point is the /etc/fstab of the workstation. It should be setup for nfs filesystems.
- **WARNING**: Don't confuse the server root filesystem and the workstation root filesystem. (I've already patched up a rc.inet1 on the server, and wondered why the workstation still didn't work.)

Exporting the filesystem

Export the root dir to the work station. See `exports(5)`. You most likely will have to restart the nfsd/mountd after this change. Under RedHat this can easily be done by typing `/etc/rc.d/init.d/nfs stop ; /etc/rc.d/init.d/nfs start .`

RARP setup

Setup the *RARP* somewhere on the net. If you boot without a nfsroot parameter, the *RARP* server has to be the *NFS* server. Usually this will be the *NFS* server. To do this, you will need to run a kernel with *RARP* support.

To do this, execute (and install it somewhere in `/etc/rc.d` of the server!):

`/sbin/rarp −s <ip-addr> <hardware-addr>`

where

ip−addr

is the IP address of the workstation, and

hardware−addr

is the Ethernet address of the network card of the workstation.

example: `/sbin/rarp −s 131.131.90.200 00:00:c0:47:10:12`

You can also use a symbolic name instead of the IP−address, as long the server is able to find out the IP−address. (/etc/hosts or *DNS* lookups)

BOOTP setup

For *BOOTP* setup you need to edit `/etc/bootptab`. Please consult the *bootpd(8)* and *bootptab(5)* man pages.

Finding out hardware addresses

I don't know the hardware address! How can I find it out?

- Boot the kernel disk you made, and watch for the line where the network card is recognized. It usually contains 6 hex bytes, that should be the address of the card.
- Boot the workstation with some OS with TCP/IP networking enabled. Then ping the workstation from the server. Look in the ARP−cache by executing: `/sbin/arp −a`

---

4. Booting the workstation

4.1 Using a boot rom

As I have not used such a beast myself yet, I can give you only the following tips (courtesy of Christian Leutloff `<leutloff@sundancer.tng.oche.de>`):

- You can't use ``normal'' bootroms.
- There is a `netboot` packet by Gero Kuhlmann, that provides for bootroms for Linux, and further information. `netboot` is available from the local Linux mirror, or as a Debian package (`netboot−0.4`).
- Read the documentation coming with your boot rom carefully.
- You probably will have to enable the tftpd on the server, but this depends upon your boot rom's way of loading the kernel.
- *Any informations on bootrom vendors of these Linux variety, mentioned above, as not everybody has access to prom burner :((especially in europe, as I'm located there.) welcome, I'll include them then here.*

4.2 Using a raw kernel disc

If you have exported the root filesystem with the correct name for the default naming and your *NFS* server is also the *RARP* server (which implies that the boxes are on the same subnet.), than you can just boot the kernel by `cat`ing it to a disc. (You have to set the root device in the kernel to 0:255.) This assumes, that the root directory on the server is `/tftpboot/`*IP−Address* (this value can be changed when compiling the kernel.)

4.3 Using a bootloader & *RARP*

Give the kernel all needed parameters when booting, and add `nfsroot=<server−ip−addr>:</path/to/mount>` where *server−ip−addr* is the IP−address of your NFS−server, and */path/to/mount* is the path to the root directory.

Tips:

- When using **LILO** consider using the ``lock'' feature: Simply type in once all the correct parameters and add ``lock''. Next time when booting let LILO timeout.
- When generating a workstation specific boot disk, you can also use the `append=` feature in `lilo.conf`.

4.4 Using a bootloader without *RARP*

In addition to `nfsroot` give a `nfsaddrs=<wst−IP>:<srv−IP>:<gw−IP>:<netm−IP>:<hostname>` commandline argument for the kernel. The kernel will setup `eth0` with the given parameters:

wst−IP

machine's IP−Address

srv−IP

NFS−server IP−Address

gw−IP

gateway

netm−IP

netmask

hostname

machine name

---

5. Known problems

5.1 /sbin/init doesn't start.

A popular problem with /sbin/init is, that some (at least) current distributions come with /sbin/init dynamically linked. So you have to provide a correct /lib setup to the client. One easy thing one could try is replacing /sbin/init (for the client) with a statically linked ``Hello World'' program. This way you know if it is something more basic, or ``just'' a problem with dynamic linking.

5.2 /dev troubles.

If you get some garbled messages about ttys when booting, then you should run a MAKEDEV from the client in the /dev directory. There are rumors that this doesn't work with certain server oses which use 64−bit dev numbers, should you run into this, please mail me with which os you have the troubles. A potential solution would be to create a small /dev ram disc early in the boot process, and reinstall the device nodes each time.

---

6. Other topics

- There is BOOTP client:
  `ftp://sunsite.unc.edu/system/Network/admin/bootpc.v045.tgz`

  With initrd (which is included in Linux 2.0), it could be made to work for diskless stations quite nicely. initrd is actually always an advanced option for more customized setups.

- For plain bootpd based boots this is actually probably not needed as Linux 2.0 contains also the option to use BOOTP instead of RARP. (To be more precise, you can compile both in the kernel, and the faster response wins.)
- In the Documentation directory of kernel source there is a file documenting NFS−Root systems.
- There is a patch floating around, that allows for swapping over NFS. It was send to me (during a private high workload phase), but I somehow managed to loose the mail. :(

You can get it probably from http://www.linuxhq.com/ in the unofficial patches section.

- My PGP public key can be fetched by fingering andreas@ag.or.at. The fingerprint is: F1 F7 43 D5 07 C4 6C 87 BF 6B 33 A2 2C EE 5A F9.