

[SLE] Remote Booting using SuSE AMD64 Enterprise Server 8

Source: <http://linux.derkeiler.com/Mailing-Lists/SuSE/2003-09/1433.html>

From: Manu Bhardwaj (manu_at_symonds.net)

Date: 09/13/03

To: suse-linux-e@suse.com

Date: Sat, 13 Sep 2003 11:26:18 -0700 (PDT)

Hi

This is a PXELinux problem that I'm facing. The problem, in brief, is that I have a set of diskless nodes (AMD Athlon XP 1800+ with 256MB RAM) connected via an internal 172.20.0.0/24 network to a dual AMD Opteron running SuSE Linux Enterprise Server 8 for AMD64. (This question is more relevant to this mailing list than suse-amd64.) I want the remote machines to diskless boot off the main server and run independently. The diskless machines have a PXE-compliant BIOS for booting off the network.

Details of absolutely everything I did has been written down at the end of this email. I will briefly summarize what I did here, and ask the questions here. For further details, please read the rest of this email and/or ask me specific questions.

Summary: I got the right linux kernel, I set up DHCP and the right TFTP (without `blksize` enabled) and NFS on the server. I switched on the remote diskless machine, and everytime I switch it on, I get the same RPC error which has something to do with being unable to mount NFS as root. I have even tried using an intermediate ramdisk, but it hasn't worked (possibly because my ramdisk wasn't ideal). My questions thus are:

- What specifically does one do to remote-mount a root partition over NFS while using such a setup?
- Would I even require a ramdisk in such a case, or is there a way to get the remote machines working after bypassing the ramdisk?
- I do not have much experience with creating ramdisks. If one is required, what does it need to contain to work properly?
- Will an option of using a floppy to mount a very basic root partition, followed by an NFS mount of root and a 'chroot' command, be an effective solution? Or will it be too slow/a stopgap approach?

ANY suggestions will help.

Thanks in advance,

SuSE: [SLE] Remote Booting using SuSE AMD64 Enterprise Server 8

Manu

--

Manu Bhardwaj <<http://manubhardwaj.net>>
<PGP: 0x7EF46A88>

--

Details:

I have searched the web and spent some days trying to get the system up and running. (Server address = 172.20.0.1, netmask 255.255.255.0). The server has been different at different points of time (sometimes TurboLinux for the AMD64, sometimes Debian Woody 32bit, and now SuSE for the AMD64. The errors are ALWAYS THE SAME.)

These are the steps I followed:

- Get ISC's DHCPD up and running off the server. It broadcasts addresses in the range 172.20.0.11 to 172.20.0.19. The relevant entries in /etc/dhcpd.conf for the subnet are:

```
range 172.20.0.11 172.20.0.13;
filename "/tftpboot/pxelinux.0";
#root-path "/";
```

- The diskless nodes, on startup, immediately obtain an address from the range. Typically, the first machine ALWAYS chooses the address 172.20.0.13. It also shows the correct netmask 255.255.255.0.

- TFTP then starts up on the client machine. (I use the inetd tftpd standalone server with the command line

```
# in.tftpd -l -s /tftpboot -v -r blksize
)
```

- TFTP downloads the 11kb pxelinux.0 kernel, and then reads the file /tftpboot/pxelinux.cfg/default (which is the only file in that directory). The contents of this file are:

```
DEFAULT vmlinuz_32bit_remote root=/dev/nfs nfsroot=172.20.0.1:/ ip=dhcp
#ipappend 1
```

The kernel vmlinuz_32bit_remote is a linux 2.4.18 kernel compiled on an Athlon XP machine. It has IP: kernel level autoconfiguration (DHCP, BOOTP, RARP) enabled; it also has NFS server and NFS as root options enabled. It also has ramdisk option enabled (4096kb). None of them are modules.

Sometimes, I have tried passing another static IP address to

- I have enabled NFS on the server. I tried using Knoppix and Gentoo Live on the diskless node, and tried remote mounting. It works perfectly and flawlessly. The contents of /etc/exports have been, at different points in time, one of these:

```
/tftpboot/remote 172.20.0.11(rw,no_root_squash)
/tftpboot/remote 172.20.0.12(rw,no_root_squash)
/tftpboot/remote 172.20.0.13(rw,no_root_squash)
or
/tftpboot/remote 172.20.0.0/24(rw,no_root_squash)
or
/tftpboot/remote 172.20.0.0/255.255.255.0(rw,no_root_squash)
or
/tftpboot/remote (rw,no_root_squash)
```

but have made no difference at any point in time. /tftpboot/remote has a complete Debian Woody 3.0r0 32-bit tree on it (got by using a hard drive on the remote machine, compiling a kernel for it and scp'ing it to the server before removing the hard drive from the remote machine).

- The contents of hosts.allow have been:

```
ALL : ALL@ALL : ALLOW
```

or

```
ALL : ALL@172.20.0.0/24 : ALLOW
```

etc. etc. Basically, everybody gets every permission. At other times, I have tried manual selection, such as

```
portmap : ALL@172.20.0.13 : ALLOW
```

```
rpc.mountd : ALL@172.20.0.13 : ALLOW
```

```
rpc.statd : ALL@172.20.0.13 : ALLOW
```

and all combinations of variations of these lines for the network, and for other

SuSE: [SLE] Remote Booting using SuSE AMD64 Enterprise Server 8

IP addresses.

(When this is the case, the remote machine searches every ASCII file from 01-00-00-0a-0b-0c and from A0ABCDEF (eg.) to A, before finally settling on the file 'default'. This takes 30 seconds when hosts.allow is set to "all all all all", but takes about 1 second when hosts.allow only allows portmap, mountd, etc. etc. selectively. Why? But that's beside the point.)

-The remote machine then easily downloads the linux kernel image via the network using TFTP, and then seems to work for about 10 seconds. I cannot actually type out all the contents of what is displayed (I grepped all files on the server but couldn't find where the remote machine's kernel logs go), but here is the jist of the output -

1. The kernel seems to identify hda and hdb from the SERVER rather than from itself. Obviously, it should not identify more than one device (the CDROM on its own machine - hdc - Secondary Master), but it identifies three drives - clearly the server's drives.

2. Most importantly, it stops at this point:

```
Looking up port of RPC 100003/2 on 172.20.0.1
```

```
RPC: sendmsg returned error 101
```

```
Root-NFS: Unable to get nfsd port number from server, using default
```

```
Lookingup port of RPC 100005/1 on 172.20.0.1
```

```
RPC: sendmsg returned error 101
```

```
Root-NFS: Unable to get mountd port number from server, using default
```

```
RPC sendmsg returned error 101
```

```
Mount: RPC call returned error 101
```

```
Root-NFS: Server returned error -101 while mounting /
```

```
VFS: Unable to mount root fs via NFS, trying floppy.
```

```
VFS: Insert root floppy and press ENTER
```

So I assumed it was an NFS error (though, as I mentioned before, a clean NFS mount using Knoppix works perfectly). So when I try

```
# rpcinfo -p
```

on the server, I get references to

```
100005/1 and 100005/2 (tcp and udp) mountd
```

as also references to 100003/something (tcp and udp) portmapper, along with some others. Obviously, they are working on the server.

I then created a ramdisk image, and modified pxelinux.cfg/default and appended the lines

```
root=/dev/ram0 initrd=image.gz
```

to the kernel. But no difference - RPC errors still thrown up.

End.

--

Check the headers for your unsubscription address

For additional commands send e-mail to suse-linux-e-help@suse.com

Also check the archives at <http://lists.suse.com>

Please read the FAQs: suse-linux-e-faq@suse.com