Installing Debian using network booting

Inhaltsverzeichnis

Preface

Installation using network booting must not be confused with DebianNetworkInstall. In network install, you start with a CD (or USB flash memory or similar) to install a minimal Linux system before you proceed to download further packages over the network.

Installation using network booting is also different from using the network to boot an **already**-installed system: in the which case the client machine remains dependent on the server being up and running at every future boot. The Linux Server Terminal Project (LTSP) works like that, for example.

This page describes installing a new Debian system with CD, DVD, USB at all. A By the end of the installation process

the new machine is able to run without the support of the server. During the installation you will need a network boot server.

As there are no fiducial boot servers out in the wild, you need to set up your own. This is considerably more complicated than installing Debian from CD (shorthand for CD or USB or ...). Normally, network booting is only used if there is really no way to boot from CD. If booting from CD fails this may be due to BIOS problems that will equally prevent network booting (typical troubleshooting attempts include switching off secure booting, switching from UEFI to legacy boot mode, and similar).

In the Web, several articles can be found that describe in more or less detail how to setup a network boot server. They all have the same weakness: You are required to execute a long list of instructions without getting any feedback before the very end of the procedure when you try to boot. If it works, fine. If not, debugging will become very very difficult. Therefore in the following we break down the procedure into steps that can be debugged separately.

 http://www.debian.org/releases/stable/i386/ch04s05 - Debian Installation Guide - 4.5. Preparing Files for TFTP Net Booting

Preconditions

The computer you want to install to will be called the Client.

The computer you install from will be called the Server. We assume that the Server is running Debian.

To be specific, we assume that the Client and the Server are part of a LAN with the following IP addresses:

- 192.168.0.1 router (i.e. LAN default gateway) and DNS recursive server
- 192.168.0.2 the Server (will host a DHCP and TFTP server)
- 192.168.0.x the Client

You will find out the value of x later.

Note that many routers also provide a DHCP server: you will have to turn it off, since only one DHCP server can run in a given LAN. Unless you may configure your router's DHCP sever to comply with the ISC DHCP server configuration below, but this is outside of the scope of this document.

It is also possible that the router and the server are the same machine, i.e. that your Debian server is the default gateway for this LAN. This will work fine.

The following instructions have been tested with Debian 8.2 (jessie) in September 2015.

Activate PXE boot

Setup the BIOS boot menu of the Client to boot from the network.

Reboot. On most sytems this produces an output that contains the Client's MAC address. Then, it will fail with

```
PXE-E53: no boot filename received.
```

Note the MAC address, it will be helpful for interpreting log messages.

On many servers, it is also possible to temporary switch to PXE boot without permanently changing the BIOS settings. There will be some kind of key stroke to hit during BIOS POST. On Dell servers, F12 will do the trick (or Esc then @ from a serial or IPMI console).

Set up DHCP server

On the Server, we need to set up a DHCP server.

Current best practice seems to be to use the package DebianPkg: isc-dhcp-server, which provides a daemon *dhcpd*.

It's configuration file is */etc/dhcp/dhcpd.conf*. Modify this file so that it contains about the following; adapt IP and MAC addresses to your local needs:

```
default-lease-time 600;
max-lease-time 7200;

allow booting;

# in this example, we serve DHCP requests from 192.168.0.(3 to 253)
# and we have a router at 192.168.0.1
subnet 192.168.0.0 netmask 255.255.255.0 {
  range 192.168.0.3 192.168.0.253;
  option broadcast-address 192.168.0.255;
  option routers 192.168.0.1;             # our router
  option domain-name-servers 192.168.0.1; # our router, again
  filename "pxelinux.0"; # (this we will provide later)
}

group {
  next-server 192.168.0.2;                # our Server
  host tftpclient {
```

```
    filename "pxelinux.0"; # (this we will provide later)
  }
}
```

After each modification of the above, restart the DHCP server with

```
# /etc/init.d/isc-dhcp-server restart
```

or with the systemd equivalent

```
# systemctl restart isc-dhcp-server
```

Check that it is actually running:

```
# pgrep -lf dhcpd
32277 /usr/sbin/dhcpd -q
```

or

```
# systemctl status isc-dhcp-server
```

which gives slightly more information.

Before rebooting the client, you may like to run

```
# journalctl -fu isc-dhcp-server
```

which shows you the last few lines of the DHCP server log, then updates the screen with each new log entry. (If you do not want to "follow" the log, just leave out the "f")

Reboot the Client. On success, it will output the IP addresses of the Server ("DHCP"), of the router ("Gateway") and of itself (192.168.0.x). Then it will hang with a TFTP request, and finally write the error message:

```
PXE-E32: TFTP open timeout
```

and at the same time you will see log messages on the server screen showing the DHCP requests and offers similar to the output below the alternative command below

If you prefer not to use systemd, or wish to compare the traditional log output for diagnostic purposes, you can look up /var/log/syslog, for example with this command

```
# grep DHCP /var/log/syslog
```

where you should see something like:

```
Jun  3 09:53:46 server dhcpd: DHCPDISCOVER from 40:01:1c:47:44:1e via eth0
```

```
Jun  3 09:53:47 server dhcpd: DHCPOFFER on 192.168.0.3 to 40:01:1c:47:44:1e via eth0
Jun  3 09:53:51 server dhcpd: DHCPREQUEST for 192.168.0.3 (192.168.0.2) from 40:01:1c:47:44:1e via e
Jun  3 09:53:51 server dhcpd: DHCPACK on 192.168.0.3 to 40:01:1c:47:44:1e via eth0
```

(Note that earlier Debian releases used /var/log/daemon.log instead of syslog)

If nothing appears in the log with either command, check the network links between the Server and the Client. Note that some switches may impose severe limitations on DHCP traffic; for Cisco ones, use 'portfast' if possible (see http://www.cisco.com/en/US/products/hw/switches/ps708/products_tech_note09186a00800b1500.shtml).

Set up TFTP server

Next, we need to set up a TFTP server on the Server.

Again, there are several packages that provide TFTP (trivial FTP, unsafe, to be used in LAN's only). It seems best practice use the package DebianPkg: tftpd-hpa. On installation, a few question are asked. The response to these questions goes into a configuration file, */etc/default/tftpd-hpa*. There should be no need to modify the following default contents:

```
TFTP_USERNAME="tftp"
TFTP_DIRECTORY="/srv/tftp"
TFTP_ADDRESS="0.0.0.0:69"
TFTP_OPTIONS="--secure"
```

Ignore older Web sites that instruct you to insert something like 'RUN_DAEMON="yes"'.

After each modification of the above configuration file, restart the TFTP server with

```
# /etc/init.d/tftpd-hpa restart
```

or

```
# systemctl restart tftpd-hpa
```

On jessie the directory /srv/tftp will be automatically created. This means the next two steps are not necessary if you use jessie.

Initially, on pre-jessie versions, this might fail with a message like

```
Restarting HPA's tftpd: in.tftpd/srv/tftp missing, aborting.
```

Therefore, as root, create the directory */srv/tftp*. Restart the TFTP daemon. Check that it is actually running:

```
# pgrep -lf tftpd
12555 /usr/sbin/in.tftpd
```

or

```
# systemctl status tftp-hpa
```

which again gives a few lines of the log rather than just the fact that the task is (isn't) running.

It is useful to test your TFTP server with a TFTP client, you may simply use the DebianPkg: tftp-hpa package for this purpose:

```
# cd /tmp
# uname -a >/srv/tftp/test
# tftp 192.168.0.2
tftp> get test
tftp> quit
# diff test /srv/tftp/test
(nothing, they are identical)
```

It is also useful to see what log entries you get when you download a file that exists, and when you try to download one that doesn't. While using tftp to test your tftpd server, try tracking your experiemnts with old and new forms of the log command while you are using your tftp client to download files that do, and files that do not, exist.

The traditional command on jessie

```
# tail -f /var/log/syslog
```

(Note again that earlier Debian releases used /var/log/daemon.log instead of syslog)

On systemd

```
# journalctl -fu tftpd-hpa
```

Sadly these seem to give different results, as of January 2017. The systemd command does not display file requests for files that do not exist.

It is quite useful to know what the client is asking for, as it helps you move files to the expected place if you make mistakes later on. At present then it is probably worth using the traditional way of log tracking for TFTPD.

Reboot the Client. You should see error messages on the client screen starting with

```
PXE-T01: File not found
```

which is quite correct since we did not yet provide any files. On the server screen you will see exactly what the client did ask for.

Provide the boot image

Download *netboot/netboot.tar.gz* from a Debian mirror (see http://www.debian.org/distrib/netinst#netboot).

Optional: To verify the digitial signature, type these commands:

```
# wget http://"$YOURMIRROR"/debian/dists/wheezy/main/installer-"$ARCH"/current/images/netboot/netboo
```

```
# wget http://"$YOURMIRROR"/debian/dists/wheezy/main/installer-"$ARCH"/current/images/SHA256SUMS
# wget http://"$YOURMIRROR"/debian/dists/wheezy/Release
# wget http://"$YOURMIRROR"/debian/dists/wheezy/Release.gpg

# cat SHA256SUMS | grep -F netboot/netboot.tar.gz
ac278b204f768784824a108e7cf3ae8807f9969adcb4598effeff2b92055bb52  ./netboot/netboot.tar.gz
# sha256sum netboot.tar.gz
ac278b204f768784824a108e7cf3ae8807f9969adcb4598effeff2b92055bb52  netboot.tar.gz
(match!)

# sha256sum SHA256SUMS
4856ecb5015b93d7dd02249c91d03bd88890d44bd25d8a2d2a400bab63f9d7de  SHA256SUMS
# cat Release | grep -A 100000 '^SHA256' | grep -F installer-"$ARCH"/current/images/SHA256SUMS
4856ecb5015b93d7dd02249c91d03bd88890d44bd25d8a2d2a400bab63f9d7de    14289 main/installer-"$ARCH"/cur
(match!)

# gpg --verify Release.gpg Release
gpg: WARNING: multiple signatures detected.  Only the first will be checked.
gpg: Signature made Sat 15 Jun 2013 05:55:56 AM CDT using RSA key ID 473041FA
gpg: Good signature from "Debian Archive Automatic Signing Key (6.0/squeeze) <ftpmaster@debian.org>"
```

Unpack netboot.tar.gz to */srv/tftp*, which should now contain

```
  debian-installer/
```

```
pxelinux.0@
pxelinux.cfg@
version.info
```

It may be necessary to *chmod -R a+r ** to make all files in this directory readable for the TFTP daemon.

Restart the TFTP daemon, and again you may like to follow the log entries as they appear

```
# tail -f /var/log/syslog
```

HOSTORICAL NOTE /var/log/syslog is right for jessie -- in earlier Debian versions, if this does not seem to work also try /var/log/daemon.log

then reboot the Client. You should get to a Debian install screen.

If you lookup into /var/log/syslog, you will see what has been downloaded from the TFTP server by the PXE bootloader, and then by SYSLINUX. You might also see some "NAK" replies when SYSLINUX asked for files that do not exist (it tries several locations for some important files).

```
Jun  3 09:53:51 server tftpd.in[32698]: Serving pxelinux.0 to 192.168.0.3:2070
Jun  3 09:53:51 server tftpd.in[32698]: Serving pxelinux.0 to 192.168.0.3:2071
Jun  3 09:53:51 server tftpd.in[32698]: Serving pxelinux.cfg/44454c4c-5600-1048-8051-c7c04f575831 to
Jun  3 09:53:51 server tftpd.in[32698]: Serving pxelinux.cfg/40-01-b1-1c-47-44-1e to 192.168.0.3:570
Jun  3 09:53:51 server tftpd.in[32698]: Serving pxelinux.cfg/default to 192.168.0.3:57090
```

```
Jun  3 09:53:51 server tftpd.in[32698]: Serving bootmenu.txt to 192.168.0.3:57095
```

The PXE loader (the firmware in the BIOS or the network controller) try to load into that order:

- pxelinux.0 (or more exactly, what you told it to download in the 'filename' field of the DHCP response)

Then SYSLINUX/PXELINUX will try to search its configuration at different paths, from the most specific to the least:

- pxelinux.cfg/GUID
- pxelinux.cfg/MAC
- pxelinux.cfg/default

And if the configuration menu depends on other configuration items, they are also downloaded. Debian will at least need the 'bootmenu.txt' file which is the main menu.

By default you arrive at the graphical Debian install start menu screen. Press 'enter' to start intallation. Be patient: it may last over a minute before the next screen ('Select a language') appears.

Alternative way to obtain the boot image

If you have a Debian system of the same release as you wish to install, you can install the boot image using apt.

```
VERSION=8 # jessie, 7.0 for wheezy
ARCH=amd64 # or any other release architecture
apt-get install debian-installer-$VERSION-netboot-$ARCH
```

Now point the tftp server to `/usr/lib/debian-installer/images/$VERSION/$ARCH/$INTERFACE` where `INTERFACE=text` for the text mode installer or `INTERFACE=gtk` for the graphical installer. A simple way to achieve this is to turn `/srv/tftp` into a symbolic link.

Another Way - use Dnsmasq

DebianPkg: dnsmasq is a lightweight, easy to configure DNS forwarder and DHCP server with BOOTP/TFTP/PXE functionality. That is, you can replace DebianPkg: isc-dhcp-server and DebianPkg: tftpd-hpa with Dnsmasq.

Following is the /etc/dnsmasq.conf providing the same functionality as the way of isc-dhcpd-server and tftpd-hpa described above.

```
interface=eth1
domain=yourdomain.com
dhcp-range=192.168.0.3,192.168.0.253,255.255.255.0,1h
dhcp-boot=pxelinux.0,pxeserver,192.168.0.2
pxe-service=x86PC, "Install Linux", pxelinux
enable-tftp
tftp-root=/srv/tftp
```

Download the netboot.tar.gz and extract it in the /srv/tftp as previous description.

Potential Issues

If the kernel in the netboot image gets out of sync with the kernel module packages then the modules won't load and the install will fail, the usual symptoms are that messages about "missing symbols" appear in the ctrl-alt-f4 console.

To fix update the kernel and initrd on the netboot server.

There is probably a debian BTS issue open for this, but i can't find it now.

See Also

- http://www.debian.org/releases/stable/i386/ch04s05 - Debian Installation Guide - 4.5. Preparing Files for TFTP Net Booting
- The Lenny installer in `netboot.tar.gz` does NOT support installing using serial console by default now. More information on this issue, and a patch to enable serial console access, can be found in ~~Closed: #309223: [i386] serial console boot not possible ...: 309223~~.