Setting up a Fedora Network Install Server

Quick Start Guide

Version 0.01

Author Mark Dickinson, December 2010

Document source: http://www.mdickinson.dyndns.org/public/doc/books/Fedora_TFTPBOOT_Server.pdf

This is how I do it, I'm not saying it's best practise, but it works

AFTER READING THIS DOCUMENT YOU WILL BE ABLE TO DO
hands off fully automated network install
manual network install
booting into recovery mode from a network source

Table of Contents

Introduction and environment assumed

This document is a really quick faststart to setting up a **Fedora** server to provide Network install functionality via a full hands off tftp network boot and install server.

As a result of this exercise you will also have created an environment you can perform manual network installs after booting off the install DVD, how to do that using the environment you have setup is covered in the troubleshooting section at the end of this document; as this document assumes the desired goal is a hands off install and any manual work will be just trouble-shooting.

It is assumed you are running Fedora Core 14, and that your web server is running under the default install location for Fedora of /var/www/html. If you have your apache documentroot anywhere else remember to use the directory paths for your server in the commands shown.

Check you have the required software installed

As well as the standard apache packages to provide the web server you must have xinetd, dhcpd, dhcpv6, tftp, tftp-server and syslinux installed in order to provide a tftp boot server.

Just run the yum update command, if any packages are already installed it will just tell you so, and install the missing ones. The yum command is

```
yum install dhcp dhcpv6 tftp tftp-server xinetd syslinux
```

Create your tftpboot directory structure, and required files

Create required directory structure

As noted, I assume you are using the default /var/www/html directory structure for apache.

```
mkdir -p /var/www/html/tftpboot
mkdir /var/www/html/tftpboot/pxelinux.cfg
mkdir /var/www/html/tftpboot/configs
mkdir /var/www/html/tftpboot/FC14
```

Copy the entire contents of the Fedora install DVD

Insert your Fedora Core 14 install DVD

```
cd /var/www/html/tftpboot/FC14
cp -rp DVDMOUNTPOINT/* .
```

Replace DVDMOUNTPOINT of course with whatever your DVD automounted on.

Copy the required pxe files to the tftpboot root directory

Note that I name my vmlinuz and initrd.img copies to include the version number of fedora as I have multiple versions available on my tftpboot server and the files are not interchangeable between between versions. The pxelinux is generally useable across versions so I normally just use the latest version from the fedora syslinux package.

```
cd /var/www/html/tftpboot
cp -p FC14/isolinux/vmlinuz vmlinux_FC14
cp -p FC14/isolinux/initrd.img initrd_FC14.img
cp -p /usr/share/syslinux/pxelinux.0 pxelinux.0
```

note: it you do not have a /usr/share/syslinux/pxelinux.0 file you have not installed syslinux, go back to the start of the document and follow the instructions. And yes, at a minimum the vmlinuz and initrd.img files, whatever you name them, must be in the root of the tftpboot directory.

Configure System files to allow tftp

Update /etc/inetd.d/tftp to enable tftp

vi the file **/etc/xinetd.d/tftp and** change disabled to no and change the -s path to the tftpboot directory you setup above. The file should look similar to the below one.

```
# default: off
# description: The tftp server serves files using the trivial file transfer \
#       protocol.  The tftp protocol is often used to boot diskless \
#       workstations, download configuration files to network-aware printers, \
#       and to start the installation process for some operating systems.
# MID - changed disable yes to disable no, I want it on
#    change below to my setup also
#       server_args             = -s /var/lib/tftpboot
service tftp
{
        disable    = no
        socket_type             = dgram
        protocol        = udp
        wait                    = yes
        user                    = root
        server                  = /usr/sbin/in.tftpd
        server_args             = -s /var/www/html/tftpboot
        per_source              = 11
        cps                     = 100 2
        flags                   = IPv4
}
```

You may also want to change the user to the userid that runs your web server (ie: apache), however I have never attemped that in my home network so can't promise that will work.

Update /etc/dhcp/dhcpd.conf for your servers

During a network install the dhcp server is queried to get the ip-address and hostname associated with the MAC address of the server you are booting from the network. This also identifies the servers that can be PXE network booted.

Key issues to bear in mind when configuring this file are

- **next-server should be your dhcp(tftp) server or it just won't work, I assume without that stopper it just forever searches**
- servername must be the hostname you have given your boot server, it should be in the hosts file
- limit the avaailable lease range to the servers you will be PXEboot installing to
- hardware ethernet is the MAC of the server requesting the lease, use individual server entries to force assignment of an ipaddr (should be the final ipaddr, anyway for firewalls use a know ipaddr)
- domainname is irrelevant as long as its ok for your internal network
- I always turn off my internet connection before enabling dhcp/tftp.
- Fyi: The pxelinux.0 file is the one you copied into the directory structure you created earlier, the xinetd changes made in the tftp file earlier resolve the directory path.

You will see there is also a dhcpd6.conf file in this directory. Unfortunately I didn't document how I got tftpboot working across ipv6 and cannot remember now, so this tutorial will be ipv4 only.

If a MAC address is not explicitly defined associated for a server then that MAC address will not be permitted to network boot from this server.

```
#
# See dhcp6c.conf(5) man page for details.
#

#interface eth0 {
#    #information-only;
#    send rapid-commit;
#    request prefix-delegation;
#    #request temp-address;
#};
```

```
ddns-update-style interim;
server-name "falcon";
default-lease-time 3600;
max-lease-time 4800;
allow booting;
allow bootp;

subnet 192.168.1.0 netmask 255.255.255.0 {
  option domain-name "anything.co.nz";
  range 192.168.1.191 192.168.1.200;
  option broadcast-address 192.168.1.255;
  option routers 192.168.1.183;
  option domain-name-servers 192.168.1.183;
  option subnet-mask 255.255.255.0;
  option time-offset -12;
  next-server 192.168.1.183;
}

host osprey {
  filename "pxelinux.0";
  hardware ethernet 00:04:23:13:CE:85;
  fixed-address 192.168.1.182;
  option host-name "osprey";
}

host nagiosvm {
  filename "pxelinux.0";
  hardware ethernet 08:00:27:71:00:D2;
  fixed-address 192.168.1.180;
  option host-name "nagiosvm";
}
```

Create the tftp server specific configuration files now

Obtain the MAC address of the server(s) you are network booting

You need to know the MAC address of the server you are network booting from, it is the MAC address that locates the correct tftp configuration file, which is actually named the MAC address. This is why I sugested above you should put known hostnames/MAC combinations into the dhcp configuration file, as a ls on your configuration files won't tell you anything.

The configuration files also must reside in directory pxelinux.cfg under the tftpboot directory identified by the -s option when you edited the tftp config file. That directory you created earlier of you have been following the instructions.

There are many ways to get the MAC address of the server you are intending to install on

- if the server is already running you can of course use ifconfig
- if you are installing to a VM you can provide the MAC address in the VM configuration so you know what it will be
- if you are installing onto a new server with no OS you can simply do a network boot and tail /var/log/messages on your tftpboot server which will show in the dhcpd messages what MAC address is searching for a configuration file. Or you could boot off a recovery CD/DVD on that server and ifconfig out the Mac address

But the bottom line is, you need to know the MAC address of the server you are providing a network boot for.

TRAP: can you spot it on the next page, my 'real hardware' server I do regular bare metal installs onto is osprey, this is the Mac address details and the name of the configuration file I use for that

This is the result of an ifconfig on one of my servers, I have highlighted the hardware MAC address and the PXE boot configuration filename that will be run when a network install is done from that MAC address.

```
[mark@osprey ~]$ ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 00:04:23:13:CE:85
          inet addr:192.168.1.182  Bcast:192.168.255.255  Mask:255.255.0.0
          inet6 addr: fe80::204:23ff:fe13:ce85/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:556330 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1795044 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:48418639 (46.1 MiB)  TX bytes:2475231540 (2.3 GiB)

[root@falcon pxelinux.cfg]# ls
00-0c-6e-63-c7-a2     default
01-00-04-23-13-ce-85  fd24274f-11e0-4c50-889e-4e23d6767ca3
01-08-00-27-c6-67-a3
```

The TRAP is that for physical hardware you need a 01 in front (so far thats all I've found), VM's will sometimes want a 00 in front rather than a 01; it's confusing. I'm sure there is a way of determining what it is likely to be but I find using the method of tailing /var/log/messages to see what dhcpd reports the MAC address as is the easiest way to find out what is actually required for any given server.
Also note, the filename has – instead of : to seperate the numbers of the MAC address, and you must use lowercase.

The long winded filename ignore, I'll cover that if I ever update this document to cover ipv6 network installs.
The default file is covered at the end of this document.

Create a PXE configuration file for the server MAC address

As mentioned earlier, the tftp configuration file must reside in directory pxelinux.cfg under your tftpboot directory; so lets go there and create the file.

```
cd /var/www/html/tftpboot/pxelinux.cfg
vi 01-macaddryougotearlier
```

Note: the filename uses – instead of : from the MAC address, other than that the filename needs to be the exact MAC address.

And heres a working sample of what the contents should look like, this is taken from a working configuration on my tftpboot server, the file highlighted above actually.

```
prompt 1
default linux
timeout 100

label linux
kernel vmlinuz_FC14
append initrd=initrd_FC14.img ramdisk_size=9216 noapic acpi=off ks=http://192.168.1.183/tftpboot/configs/osprey_FC14.cfg

# PXE boot for osprey mac addr
```

So what are the key fields you need to think about

- all the append parameters are passed to the defined vmlinuz and these are the parameters you would normally append using [tab] at the install/boot menu if you were doing the install from a local DVD for unatended re-install/refresh you obviously need to use a kickstart file to provide the commands to be used during the install, which is defined ks= option. **The ks option provides the full URL to the kickstart configuration file that contains the installation script/commands**. I choose to keep them under the tftpboot directory structure, they must be accessable by the web server so that is a logical place.
- the kernel option must be the correct vmlinuz version for the OS version you are installing
- the append option must specify the correct initrd.img for the OS version you are installing

Simple so far isn't it.

Finally you get to create a Kickstart configuration script

The kickstart configuration file used by a tftp install is specified by the ks= entry in the PXE configuration file you just set up above.

In the kickstart file you should specify at a minimum

- the disk layout to be used
- the packages to install, or explicitly omit
- the timezone
- a password for the root user

But don't panic (unless you are installing to a headless server). If you are network installing to a server with a keyboard and screen (or a VM) any missing or invalid parameters in the kickstart file will result in a prompt for corrective action.

Simple example

An very cutdown example is shown and explained here. The key things to note in this sample are

- the first line is what is going to be done, we are doing a fresh 'install'
- the second line instructs that we are installing from a network URL, not from local media
- the third line ensures the install assigns the expected static ipaddr
- the tricky one, the root password is set The string below rootpw is actually part of the same line (just wouldn't fit on the page). Get this value from cut/paste from the shadow file from one of your other servers (actually I changed the root password on one of my servers to 'install', copied the sring to this config file, and of course changed the real password back to something else; but I know the root password will be install after a network install on all servers, you need to do something simililar so you know what the password is). If the rootpw is not provided the server/desktop the install is being done onto will prompt which you don't want on a hands off install, so provide a password here. You could of course provide an unencrypted password, but remember this config file is readable by any web browser than can connect to your web server
- then we enable ssh through the firewall, we want to be able to login to fix problems
- note: the bootloader line is all on one line also, darn line breaks
- from clearpart (blow away the entire disk) to prior to the %packages line is repartioning the disk and setting up the volume groups
- in the %packages section you select the @packagename or individual rpmname to install, or use -rpmname to specifically omit a package
- the interesting section is the %post section. At this point you are in a chrooted environment on your newly installed OS on the server and can do

all the customisation you need. It is at this stage I 'harden' configuration files, use wget to pull custom packages and database backups from my local repo server, install custom rpms, restore the database, add additional users etc.., anything you want to customise really.

- The last line, the %end, is enforced in FC14, it wasn't needed in prior Fedora releases but is needed now. It's just, the end of the kickstart script

```
# Example kickstart script
install
url --url http://192.168.1.183/tftpboot/FC14
lang en_US.UTF-8
keyboard us
network --device eth0 --bootproto static --ip 192.168.1.182 --netmask 255.255.0. 0 --hostname osprey --gateway 192.168.1.1
# pw is install
rootpw  --iscrypted  $6$Xz/2TJUZ$PV5lVP1Bc5CTqp3kc9D9Igz4LoEXmCUd9lBh6jT1GKw9r4fWg7T4cMP4KW5xKs1j.85K4RYPbBE3udRkTsDTN.
firewall --service=ssh
authconfig --enableshadow --passalgo=sha512 --enablefingerprint
selinux --enforcing
timezone --utc Pacific/Auckland

bootloader --location=mbr --driveorder=sda --append=" LANG=en_US.UTF-8 SYSFONT=l
atarcyrheb-sun16 KEYBOARDTYPE=pc KEYTABLE=us rhgb quiet"

clearpart --all --drives=sda
part /boot --fstype ext4 --size=200 --ondisk=sda
part pv.6 --size=1 --grow --ondisk=sda
volgroup VolGroup00 --pesize=32768 pv.6
logvol / --fstype ext4 --name=LogVol00 --vgname=VolGroup00 --size=1024 --grow
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=640 --grow
--maxsize=1280

%packages
@mysql
@admin-tools
...
radeontool
wireshark-gnome
...
-openoffice.org-impress
-openoffice.org-xsltfilter
```

```
%post
(
# ------------------------------------------------------------------------
# Post install commands to be run, these are run on the new system
# while it is in a chroot / to the new filesystem.
# ------------------------------------------------------------------------
# Create repos.d entry for my local repository so when this brand new FC14 install
# does a yum update it gets most of the rpms from my network.
# Disable automatic updates as well.
/sbin/chkconfig yum-updatesd off
cat << EOF > /etc/yum.repos.d/local-updates.repo
[local-updates]
version=14
name=LOCAL \$releasever updated RPM Packages  -
failovermethod=priority
baseurl=http://192.168.1.186/tftpboot/FC\$releasever-updates/
enabled=1
gpgcheck=0
retries=1
priority=50
EOF

# and any more stuff you might want to do
# - add extra users
# - wget and install custom rpms
# - configure databases etc.
) 2>&1 >> /root/custom_install.log

%end
```

And an important tip, the installer creates one

It may look complex to create a kickstart file, but remember that whenever you install Fedora on a local machine using standard install media is creates a file /root/anaconda.log... updated, in FC14 it seems to create a file /root/anaconda-ks.cfg ?. But check in the /root directory of one of your servers and from any 'manual' install you have done there will be a file created which is a kickstart template that can be used in a network install to exactly replicate the manual install that was just done; after you have copied the file to your tftp server, and changed the file to set the install to a URL instead of a local device.

All the disk formatting options will be commented out in the file created by that initial install, which is wise, you may wish to tweak them later. Also from past experience you need to review them as at least up until FC13 the generated script tries to create LVMs before physical partitions were created. So tweak and uncomment before using that log file as a the basis for a kickstart script.

But it is a good starting point to use.

Also if you want to test a network install by doing a manual install from the tftpboot install system, that is covered in the troubleshooting section.

Prepare your tftpboot server for network installs

You could do two things, I use the later, its up to you

- use chkconfig to ensure dhcpd starts at every reboot, and implement all the dangerous firewall rules required
- or when doing a network install, on the network install server - flush your firewall rules and 'service start dhcpd', then when done reload the firewall iptables and stop the dhcpd service (just less window of oportunity to be hacked if yoy are using a internet facing server; which you should not be by the way).

All, done; you are ready to go

Now all you have to do is go to the server you wish to install onto, work out what series of keys you need to use on it to get it to initiate a network boot, and boot the server. If will poll out into the network for a dhcp/tftp server and begin an OS install for you.

Trouble Shooting

Test the kickstart file is being run via tftp

If you want to test your kickstart script is actually being run, you can change the configuration file in /var/www/http/configs/whatevernameyouaretesting to contain only the lines

```
install
url --url http://yourtftpbootserveripaddr/tftpboot/FC14
interactive
```

which will test the dhcp and tftp setup is correct and start an interactive install from the server.

If an interactive install starts, then all your end to end bits are in place; the problem was in your origional kickstart file. I would suggest at that point completing the manual install and use the /boot/anaconda-ks.cfg log file created from that install as the basis of your next attempt at customising a kickstart file (remember to add the %end at the end of it though).

But also tail the /var/log/messages file on the network install server, you might spot the issue there.

Important: you can omit the install option and get the standard install menu instead, from which you can select a normal url install to test http (described below) or enter recover mode.

Test the http setup, use a manual URL install

This will test only that your web server setup is correct.
Insert the physical FC14 install DVD and boot off it. At the installation menu prompt use [tab] and enter 'askmethod' to the prompt. The install will then go through the usual setup prompts but eventually ask where you want to install from. At which point select the URL option and enter the http://yourtftpbootserveripaddr/tftpboot/FC14 URL (the url path to where you copied the Fc14 install DVD contents), and the manual install will continue using the network media.
If that works your http setup is correct, review the dhcp and tftp setup.

Oh no, It won't boot; yes you can network recover

If you have a recoverable situation, if you could boot; but don't have the install DVD handy to boot from... you can still boot into recovery mode if you have setup a network install server.

This isn't part of the tftpboot setup document, but usefull to know.
You can change configuration file in the pxelinux.cfg directory for the MAC address of this server; by simply removing the 'ks=urltokickstartfile' from the append line and replacing it with 'linux rescue' when you network boot it will simply use the DVD image you so carefully copied onto you install server many chapters ago.

When network booting into 'linux rescue' mode you will have to reply to the selection of keyboard/language/network etc but will then be prompted for the location of the rescue media; at this point you select the Url option and provide the URL of the directory on the network server you placed the copy of the FC14 DVD.

Assuming it's not a network problem that killed the server of course.

Note: I actually use the default file in the pxelinux.cfg directory to provide that functionality, and trigger it by just temporarily renaming out the file with the MAC address name, and renaming it back when done.

The pxelininux.cfg/default file

If you have read the entire document you will have noticed a file named 'default' in the PXE configuration directory, along with all those files with Mac address names.

From memory if tftpd cannot find a matching configuraion file for the MAC address being used it will try to run the default file, I havent really followed this option up as I am using specific files for each MAC address as they are more self documenting, and should continue working if support for 'default' is ever dropped. For all I know it already has been dropped.

This is my default file.

```
prompt 1
default rescue
timeout 100

label rescue
kernel vmlinuz_FC14
append initrd=initrd_FC14.img ramdisk_size=9216 noapic acpi=off  linux rescue

label linux
kernel vmlinuz_FC14
append initrd=initrd_FC14.img ramdisk_size=9216 noapic acpi=off

label fc13osprey
kernel vmlinuz_FC13
append initrd=initrd_FC13.img ramdisk_size=9216 noapic acpi=off ks=http://169.254.218.183/kickstart/configs/osprey_FC13.cfg

label FC8
kernel vmlinuz_FC8
append initrd=initrd_FC8.img ramdisk_size=9216 noapic acpi=off
```

I can't remember how to use the labels anymore, note to self, remember so I can document that. My default setting is to boot a server into recovery mode to fix problems.

Appendix A – An example kickstart scripts

This example will not work as a cut/paste job for you, it is a copy of a working kickstart script I use but I have ripped huge chunks out of it in order to get it as small as possible for this document while still showing what is possible in a post install script.

If has been stripped down to show

- you can add additional users
- you can customise system files to your hearts content
- you can create scripts
- you can build cron tables
- you can wget RPMs or full applications from an install source and install them
- you can do anything you want really

but as I have ripped out huge blocks you may spot post install steps refering to files that are not in the wget list etc., ignore all that, this is an example of methods that you can use in a kickstart script, not a working kickstart script.

```
#
# Heavily customised install script
install
url --url http://192.168.1.183/tftpboot/FC14
lang en_US.UTF-8
keyboard us
network --device eth0 --bootproto static --ip 192.168.1.182 --netmask 255.255.0.0 --hostname osprey --gateway 192.168.1.1
rootpw  --iscrypted
$6$Xz/2TJUZ$PV5lVP1Bc5CTqp3kc9D9Igz4LoEXmCUd9lBh6jT1GKw9r4fWg7T4cMP4KW5xKs1j.85K4RYPbBE3udRkTsDTN.
firewall --service=ssh
authconfig --enableshadow --passalgo=sha512 --enablefingerprint
selinux --enforcing
timezone --utc Pacific/Auckland
```

```
bootloader --location=mbr --driveorder=sda --append=" LANG=en_US.UTF-8 SYSFONT=latarcyrheb-sun16 KEYBOARDTYPE=pc
KEYTABLE=us rhgb quiet"
clearpart --all --drives=sda
part /boot --fstype ext4 --size=200 --ondisk=sda
part pv.6 --size=1 --grow --ondisk=sda
volgroup VolGroup00 --pesize=32768 pv.6
logvol / --fstype ext4 --name=LogVol00 --vgname=VolGroup00 --size=1024 --grow
logvol swap --fstype swap --name=LogVol01 --vgname=VolGroup00 --size=640 --grow --maxsize=1280

%packages
@mysql
@admin-tools
@editors
@system-tools
@fonts
@text-internet
@core
@base
@hardware-support
@java
@web-development
@web-server
@server-cfg
@legacy-fonts
php-mysql
emacs
radeontool
fuse
audit
w3m
```

```
lynx
mtools
digikam
kipi-plugins
libsane-hpaio
im-chooser
urw-fonts
amarok
spamassassin
procmail
policycoreutils
policycoreutils-gui
policycoreutils-python
-openoffice.org-impress
-openoffice.org-writer
-openoffice.org-math
-openoffice.org-calc
-openoffice.org-graphicfilter
-openoffice.org-draw
-openoffice.org-xsltfilter

%post
(
# ------------------------------------------------------------------------
# Post install commands to be run, these are run on the new system
# while it is in a chroot / to the new filesystem.
# ------------------------------------------------------------------------
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# change default runlevel from 5 to 3, this is a headless server
```

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
echo "******* Changing server default runlevel to 3 ********"
/bin/cp -p /etc/inittab /etc/inittab.`/bin/date +"%Y%m%d"`
/bin/cat /etc/inittab.`/bin/date +"%Y%m%d"` | /bin/sed -e 's/id:5:initdefault:/id:3:initdefault:/' > /etc/inittab
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 2 - create a new motd and a ssh login banner
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
echo "******* Creating modt and sshd banners ********"
/bin/cat > /etc/motd << EOF
Use of this server is restricted to authorised users only.
Unauthorised access will be prosecuted to the full extent of the law.
All user activity on this server is being logged, and reviewed by automation tools.
EOF
/bin/cat > /etc/ssh/prelogin_banner << EOF
**************************************************************************
*                                                                        *
*               Access to this system is restricted                      *
*                                                                        *
* Your ip address has already been logged, if you are not authorised to use  *
* this system then you should disconnect immediately.                      *
*                                                                        *
**************************************************************************
EOF
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# lock down ssh
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

```
#
echo "******* Locking down ssh access ********"
# Note, need to replace the authorized_keys line with one without tabs
cat /etc/ssh/sshd_config | grep -v "ssh\/authorized_keys" > /etc/ssh/sshd_config.`/bin/date +"%Y%m%d"`
echo '#AuthorizedKeysFile .ssh/authorized_keys' >> /etc/ssh/sshd_config.`/bin/date +"%Y%m%d"`
/bin/cat > /tmp/sedcmds << EOF
s/#ListenAddress 0.0.0.0/ListenAddress 192.168.1.182/
s/#Protocol 2/Protocol 2/
s/#PermitRootLogin yes/PermitRootLogin no/
s/#PermitEmptyPasswords no/PermitEmptyPasswords no/
s/#PrintMotd yes/PrintMotd yes/
s/#PrintLastLog yes/PrintLastLog yes/
s'#Banner none'Banner /etc/ssh/prelogin_banner'
s'Subsystem'#Subsystem'
s'#LogLevel INFO'LogLevel INFO'
s'#MaxAuthTries 6'MaxAuthTries 6'
#NO LONGER REQD#s'#AuthorizedKeysFile .ssh/authorized_keys'AuthorizedKeysFile .ssh/authorized_keys2'
#NO LONGER REQD#s'#PubkeyAuthentication yes'PubkeyAuthentication yes'
s'#UseDNS yes'UseDNS no'
EOF
/bin/cat /etc/ssh/sshd_config.`/bin/date +"%Y%m%d"` | /bin/sed -f /tmp/sedcmds > /etc/ssh/sshd_config
/bin/rm /tmp/sedcmds
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# lock down services
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
echo "******* Disabling unused services ********"
/sbin/chkconfig nfs off
/sbin/chkconfig nfslock off
```

```
/sbin/chkconfig nmb off
/sbin/chkconfig smb off
/sbin/chkconfig cups off
/sbin/chkconfig cups-config-daemon off
/sbin/chkconfig squid off
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# add my userid, and give it an initial password
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
echo "******* Adding a non-root userid ********"
useradd -n -m -c "Mark Dickinson" -d /home/mark -g users -u 500  mark
echo "install" | passwd --stdin mark
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# install additional software
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# wget runs in the background, so delay after all these to ensure we got it all.
echo "******* Retrieving additional software ********"
/bin/mkdir /installs
cd /installs
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/osprey_rc.firewall
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/elm-2.5.5-2mdk.i586.rpm
# we want to recover back the last database dump automagically exported to the boot server as well
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/website_live_mysql_backup.tar.gz
# website databases are used to restore service, need those
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/latest_website_backup.tar.gz
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/latest_osprey_website_config.tar.gz
# ddclient updates our dynamic ip address at dyndns
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/ddclient_custom.tar.gz
```

```
# selinux customisations needed
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/webserver_FC13_selinux_rules.tar.gz
# authorized_keys to allow falcon to pull mysql backups off the server
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/osprey_authorized_keys
# Some scripts needed
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/make_website_map.sh
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/osprey_scan_weblogs.sh
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/my_extras/backuputil.tar.gz
# needed for php-gd
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/FC14-extras/php-gd-5.3.3-1.fc14.i686.rpm
wget -a wget.log -nv --tries 5 --no-clobber http://192.168.1.183/tftpboot/FC14-extras/t1lib-5.1.2-6.fc14.i686.rpm
/bin/sleep 30
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# NETWORK CUSTOMISATION
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
cat << EOF > /etc/resolv.conf
domain mdickinson.dnsalias.org
nameserver 192.168.1.1
EOF
# Custom rules fror my webserver
mv /installs/osprey_rc.firewall /etc/rc.firewall
chmod 744 /etc/rc.firewall
#
# cron jobs
/bin/cat >> /var/spool/cron/root << EOF
# System performance indicator checks, and filesystem and filesize checks
1,6,11,16,21,26,31,36,41,46,51,56 * * * * /root/custom/cron_jobs/r005_process_busy_check.sh
0,5,10,15,20,25,30,35,40,45,50,55 * * * * /root/custom/cron_jobs/r005_load_average_check.sh
0,10,20,30,40,50 * * * * /root/custom/cron_jobs/r010_check_disk_busy.sh
```

```
8,18,28,38,48,58 * * * * /root/custom/cron_jobs/r010_check_free_mem.sh
6,21,36,51 * * * * /root/custom/cron_jobs/r015_system_check.sh all
#7,22,37,52 * * * * /root/custom/cron_jobs/r015_server_ping_check.sh
2,32 * * * * /root/custom/cron_jobs/r030_filesize_check.sh
#15 * * * * /root/custom/cron_jobs/r060_systemstats.sh
01 23 * * * /root/custom/cron_jobs/backup_mysql_databases.sh
01 23 * * * sh -c "/root/custom/scan_weblogs.sh 2>&1 > /home/mark/backups/weblog_extract"
01 23 * * * /root/custom/cron_jobs/backup_mail.sh
50 22 * * * sh -c "cp /home/httpd/newsite/html/logs/* /home/mark/backups;chmod 644 /home/mark/backups/*log"
30 23 * * * /root/custom/cron_jobs/check_changes.sh
EOF
#
# Now the extra cron job to backup the mysql databases is installed
#
cat << EOF > /root/custom/cron_jobs/backup_mysql_databases.sh
#!/bin/bash
if [ ! -d /home/mark/backups ];
then
   mkdir -p /home/mark/backups
   chown mark /home/mark/backups
fi
if [ -f /home/mark/backups/website_live_mysql_backup.tar.gz ];
then
   /bin/rm -f /home/mark/backups/website_live_mysql_backup.tar.gz
fi
cd /var/lib
/sbin/service mysqld stop
tar -zcf /home/mark/backups/website_live_mysql_backup.tar.gz mysql
/sbin/service mysqld start
chown mark /home/mark/backups/website_live_mysql_backup.tar.gz
```

```
# done
exit 0
EOF
chmod 744 /root/custom/cron_jobs/backup_mysql_databases.sh
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# need to add some host entries
# replace the whole file as we must get our server name on line 1 anyway
# no unsolicited traffic is permitted out of this server but hostname entries
# are needed for some of my inbound traffic access checks.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
echo "******* Customising /etc/hosts ********"
/bin/cat > /etc/hosts << EOF
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.1.182 osprey osprey.localdomain osprey.mdickinson.dnsalias.org mdickinson.dnsalias.org www.mdickinson.dnsalias.org
192.168.1.183 falcon falcon.localdomain
192.168.1.184 firebird
192.168.1.186 eagle
EOF
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# 10. harden up the server
#   - create and secure home directories that do not exist
#   - delete users and groups we do not want
#   - initialse the ftp users file to deny ftp usage to all users
#   - change passsword expiry and minimum length in login.defs
#   - resecure some files with bad security as installed by default
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
#
```

```
echo "******* Performing server hardening ********"
/bin/mkdir /var/adm
/bin/chown adm:adm /var/adm
/bin/chmod 700 /var/adm
/bin/mkdir /var/gopher
/usr/sbin/userdel ftp
/usr/sbin/userdel news
/usr/sbin/userdel uucp
/usr/sbin/userdel gopher
/bin/cat /etc/passwd | awk -F: {'print $1'} | while read uname
do
   echo "$uname" >> /etc/ftpusers
done
/bin/chmod 644 /etc/ftpusers
# Can't really use sed here as there are tabs in the file, so just drop
# what we are changing and then add the new values to the end of the file.
/bin/cp -p /etc/login.defs /etc/login.defs.`/bin/date +"%Y%m%d"`
/bin/cat /etc/login.defs.`/bin/date +"%Y%m%d"` | grep -v PASS_MAX_DAYS | grep -v PASS_MIN_LEN > /etc/login.defs
/bin/cat >> /etc/login.defs << EOF
# Updated by kickstart installation script
PASS_MAX_DAYS   31
PASS_MIN_LEN    6
EOF
# these fail security checks as world writable, change that at install time
/bin/chmod 644 /etc/dumpdates
/bin/chmod 644 /usr/share/doc/docbook-dtds-1.0/4.1-sgml/*.txt
/bin/chmod 644 /usr/share/doc/docbook-dtds-1.0/4.1-sgml/ChangeLog
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Install the http directories and configurations
```

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "******* Installing the web site ********"
service httpd stop
cd /
tar -xf /installs/latest_website_backup.tar.gz
rm -f /installs/latest_website_backup.tar.gz
tar -xf /installs/latest_osprey_website_config.tar.gz
rm -f /installs/latest_osprey_website_config.tar.gz
# Yup, I don't use /var/www
cd /home
chcon -R system_u:object_r:httpd_sys_content_t:s0 httpd
chcon -R system_u:object_r:httpd_sys_script_exec_t:s0 httpd/newsite/cgi-bin
/sbin/chkconfig --add httpd
/sbin/chkconfig httpd on
cd /etc/httpd/conf
/bin/cp -p httpd.conf httpd.conf.`/bin/date +"%Y%m%d"`
/bin/cat > /tmp/sedcmds << EOF
s'dickinson.co.nz'mdickinson.dnsalias.org'g
EOF
/bin/cat httpd.conf.`/bin/date +"%Y%m%d"` | /bin/sed -f /tmp/sedcmds > httpd.conf
/bin/rm /tmp/sedcmds
cd /etc/httpd/conf.d
# The untarred symbolic links get blocked by SeLinux, recreate them
cd /home/httpd/newsite/html
/bin/rm login_ok.php
/bin/ln -s site/security/login_ok.php login_ok.php
# Still probs with some access, selinux confusing perms with homedir perms
# as I have the website in apaches homedir. Allow homedirs until I sort this out.
setsebool -P httpd_enable_homedirs=1
cd /home/httpd/newsite/html/personal_space/pics
```

```
/bin/rm -rf archives
#
# Add a the script I use to keep an eye on whos on the website
mv /installs/osprey_scan_weblogs.sh /root/custom/scan_weblogs.sh
chmod 744 /root/custom/scan_weblogs.sh
#
# make the site map and link into the robots.txt file
mv /installs/make_website_map.sh /root/custom/make_website_map.sh
mv /installs/osprey_scan_weblogs.sh /root/custom/scan_weblogs.sh
chmod 744 /root/custom/make_website_map.sh
/root/custom/make_website_map.sh
#
# Update the server latest news file
datenow=`date +"%Y/%m/%d"`
/bin/cat > /home/httpd/newsite/html/news/site_news.txt << EOF
<div class="infobar">
<b>${datenow}</b><br />
Server re-installed to test the kickstart image.<br />
No news updates since then.
</div>
EOF
chown apache:apache /home/httpd/newsite/html/news/site_news.txt
chmod 644 /home/httpd/newsite/html/news/site_news.txt
chcon system_u:object_r:httpd_sys_content_t:s0 /home/httpd/newsite/html/news/site_news.txt
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# initialise mysql
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "******* Installing and customising MySQL database backups ********"
/bin/cp -p /etc/my.cnf /etc/my.cnf.`/bin/date +"%Y%m%d"`
```

```
/bin/cat > /etc/my.cnf << EOF
[client]
#password = your_password
port = 3306
socket = /var/lib/mysql/mysql.sock

[mysqld]
datadir=/var/lib/mysql
port = 3306
bind-address = 192.168.1.182
socket=/var/lib/mysql/mysql.sock
# skip-innodb
skip-locking
key_buffer = 16k
max_allowed_packet = 1M
table_cache = 4
sort_buffer_size = 64k
net_buffer_length = 2k
thread_stack = 64k
# Default to using old password format for compatibility with mysql 3.x
# clients (those using the mysqlclient10 compatibility package).
old_passwords=1

# to disable network access all together use the below, will break phpBB
#skip-networking

# uncomment the below to log updates
#log-bin

# uncoment the below if you are not using BDB tables
```

```
#skip-bdb

# leave the below commented unless you want to use InnoDB tables tuning
# the InnoDB tables allow transaction begin/end/abort processing and
# will be created with the defaults if this section is left commented.
#innodb_data_home_dir = @localstatedir@/
#innodb_data_file_path = ibdata1:10M:autoextend
#innoddb_log_group_home_dir = @localstatedir@/
#innoddb_log_arch_home_dir = @localstatedir@/
# buffer pool can be up to 50-80% of RAM, but setting it
# too high will cause you problems
#innodb_buffer_pool_size = 16M
#innodb_additional_mem_pool_size=2M
# set ..log_file_size to 25% of buffer pool size
#innodb_log_file_size = 5M
#innodb_log_buffer_size = 8M
#innodb_flush_log_at_trx_commit = 1
#innodb_lock_wait_timeout = 50

[mysql.server]
user=mysql
basedir=/var/lib/mysql

[mysqld_safe]
log-error=/var/log/mysqld.log
err-log=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid

[mysqldump]
quick
```

```
max_allowed_packet = 16M

[mysql]
no-auto-rehash
safe-updates

[isamchk]
key_buffer = 8M
sort_buffer_size = 8M
[myisamchk]
key_buffer = 8M
sort_buffer_size = 8M

[mysqlhotcopy]
interactive-timeout
EOF

# install the archived databases now
cd /var/lib
##tar -zxf /installs/falcon_mysql_backup.tar.gz
tar -zxf /installs/website_live_mysql_backup.tar.gz
rm -f /var/lib/mysql/mysql.sock
##rm -f /installs/falcon_mysql_backup.tar.gz
rm -f /installs/website_live_mysql_backup.tar.gz
#
# Load all the archived databases
#
# Databases and Password will have already been set in the archived databases
# /bin/su - mysql -c "/usr/bin/mysql_install_db"
# /bin/su - mysql -c "cd /usr ; /usr/bin/mysqld_safe &"
```

```
# /bin/sleep 5
# /usr/bin/mysqladmin -u root password 'guardian'
# set password for this server
/usr/bin/mysqladmin -u root -h osprey password 'guardian' -pguardian &
# service mysqld stop
#
# set to autostart
/sbin/chkconfig --add mysqld
/sbin/chkconfig mysqld on
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Need some customised selinux rules, without these the website cannot
# connect to the search engine; sendmail cannot write to users directories,
# and the apache certwatch will not work.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "******* Creating customised selinux rules *******"
cd /root
tar -zxf /installs/webserver_FC13_selinux_rules.tar.gz
/bin/rm -f /installs/webserver_FC13_selinux_rules.tar.gz
cd selinux/sources
if [ -f localmisc.pp ];
then
   /bin/rm -f localmisc.pp
fi
make
if [ -f localmisc.pp ];
then
   semodule -i localmisc.pp
fi
#
```

```
echo "# The FC11 bug where the floopy drive spins forever is" >> /etc/rc.local
echo "# back in FC12. And FC13, and FC14, so keep this in." >> /etc/rc.local
echo "modprobe floppy" >> /etc/rc.local
echo "rmmod floppy" >> /etc/rc.local
#
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# And private ssh keys needed for the mysql backup to be pulled back
# up to the jumpstart server every six hours.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "*** Loading ssh keys needed by falcon backups ***"
cd /home/mark
mkdir .ssh
chown mark:users .ssh
chmod 700 .ssh
cd .ssh
mv /installs/osprey_authorized_keys authorized_keys2
chmod 600 authorized_keys2
chown mark:users authorized_keys2

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# makewhatis kills the server, remove it
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "*** removing makewhatis ***"
/bin/rm -f /etc/cron.weekly/makewhatis.cron
/bin/rm -f /etc/cron.daily/makewhatis.cron

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# php needs more memory for openid functions
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "*** Increasing php memory limit and setting timezone ***"
```

```
cd /etc
/bin/cp -p php.ini php.ini.`/bin/date +"%Y%m%d"`
/bin/cat > /tmp/sedcmds << EOF
s'memory_limit = 32M'memory_limit = 64M'g
s';date.timezone ='date.timezone = "Pacific/Auckland"'g
EOF
/bin/cat php.ini.`/bin/date +"%Y%m%d"` | /bin/sed -f /tmp/sedcmds > php.ini
/bin/rm /tmp/sedcmds

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# We boot from the kickstart server, we do NOT want web updates !, as they
# will be backed out every time we kickstart anyway; a waste of my bandwidth.
# B U T
# Add my local network repositories with a higher precedence in the
# yum search path so I can manually get the updates from my own network
# servers as needed.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
/sbin/chkconfig yum-updatesd off
cat << EOF > /etc/yum.repos.d/local-updates.repo
[local-base]
name=LOCAL \$releasever RPM Packages -
version=14
failovermethod=priority
baseurl=http://192.168.1.186/tftpboot/FC\$releasever/
enabled=1
gpgcheck=0
retries=1
priority=50

[local-source]
```

```
version=14
name=LOCAL \$releasever Source RPM Packages -
failovermethod=priority
baseurl=http://192.168.1.186/tftpboot/FC\$releasever-source/
enabled=0
gpgcheck=0
retries=1
priority=50

[local-updates]
version=14
name=LOCAL \$releasever updated RPM Packages  -
failovermethod=priority
baseurl=http://192.168.1.186/tftpboot/FC\$releasever-updates/
enabled=1
gpgcheck=0
retries=1
priority=50
EOF

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Osprey will be the master ntp server for my network as well.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
cat << EOF > /etc/ntp.conf
# For more information about this file, see the man pages
# ntp.conf(5), ntp_acc(5), ntp_auth(5), ntp_clock(5), ntp_misc(5), ntp_mon(5).
# make osprey the master, my other servers will query this server
# first if at all possible.
driftfile /var/lib/ntp/drift
# Permit time synchronization with our time source, but do not
```

```
# permit the source to query or modify the service on this system.
restrict default kod nomodify notrap nopeer noquery
restrict -6 default kod nomodify notrap nopeer noquery
restrict 127.0.0.1
restrict -6 ::1
restrict 192.168.1.0 mask 255.255.255.0 nomodify notrap
server 2.nz.pool.ntp.org
server 1.oceania.pool.ntp.org
server 0.oceania.pool.ntp.org
broadcast 192.168.1.255 autokey      # broadcast server
includefile /etc/ntp/crypto/pw
# Key file containing the keys and key identifiers used when operating
# with symmetric key cryptography.
keys /etc/ntp/keys
EOF
chkconfig ntpd on

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Turn off network manager, we want to use our own static ip addresses.
# And NetworkManager buggered the DNS lookups so force those to be correct.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
chkconfig NetworkManager off
chkconfig network on
cat << EOF >> /etc/resolv.conf
nameserver 192.168.1.1
EOF
echo "DNS1=192.168.1.1" >> /etc/sysconfig/network-scripts/ifcfg-eth0

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Other miscellaneous stuff
```

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# don't need smart card support
chkconfig pcscd off
# ..... quick cleanups ......
# Preserve the custom install log
mv /installs/wget.log /root/custom_install_wget.log
# Add alias I use to maintain website easily
echo "alias website='cd /home/httpd/newsite/html'" >> /root/.bashrc
echo "alias mvs='cd /home/mark/hercules/turnkey3'" >> /home/mark/.bashrc
# ..... end quick cleanups ......

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# The two additional RPMs needed for the website to work
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
cd /installs
rpm -i t1lib-5.1.2-6.fc14.i686.rpm php-gd-5.3.3-1.fc14.i686.rpm
/bin/rm -f t1lib-5.1.2-6.fc14.i686.rpm php-gd-5.3.3-1.fc14.i686.rpm

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# I have done my configs, disable additional prompts done by firstboot or
# the reboot will hang.
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "RUN_FIRSTBOOT=NO" > /etc/sysconfig/firstboot
chown root:root /etc/sysconfig/firstboot
chmod 644 /etc/sysconfig/firstboot

# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
# Reboot now, full filesystem check on the way up
# ACTUALLY the reboot doesn't get run, smarts somewhere ?.
# the server will reboot itself at the end of the post step.
```

```
# - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
echo "******* Post-Install completed. Rebooting in 60 seconds to complete install ********"
sleep 60
echo "******* Rebooting ********"
sync
sync
shutdown -Fr now
) 2>&1 >> /root/custom_install.log

%end
```