SCST Configuration How-To Using Gentoo

1 REQUIREMENTS

1.1 Hardware Requirements:

1.1.1 High-end Desktop
1.1.2 3 Gigabit or better Network Cards
1.1.3 6 x 3.0 TB 7200 RPM HDD's
1.1.4 1 x 500 GB HDD
1.1.5 4 GB Memory Stick

1.2 Software Requirements:

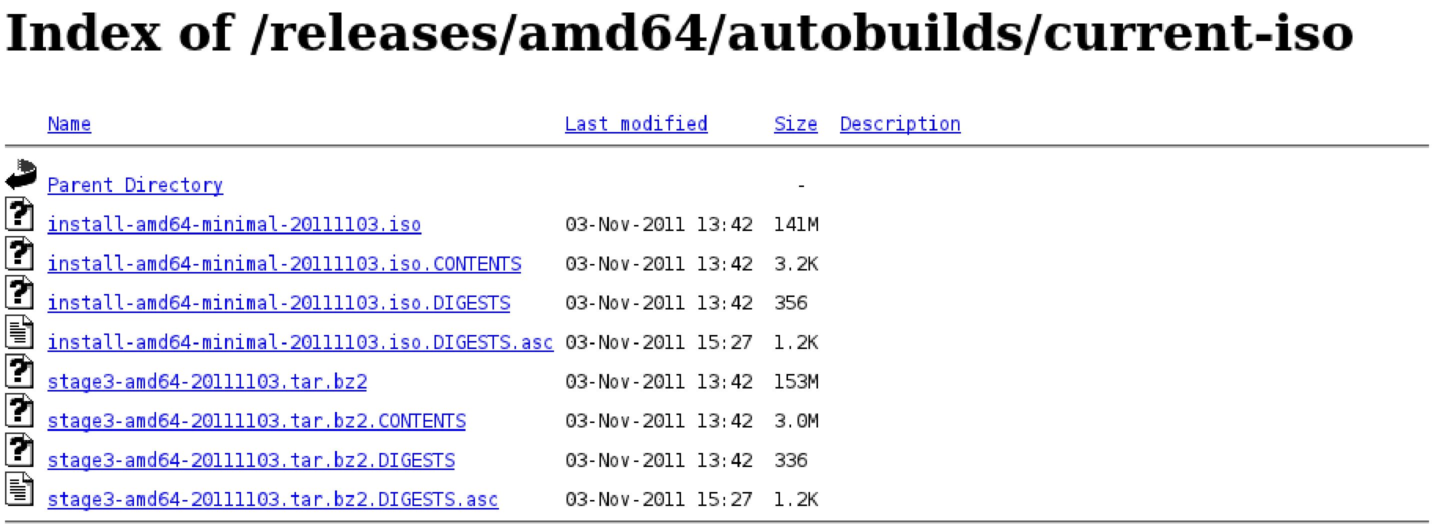
1.2.1 Gentoo Linux x64 Base Install Image
1.2.2 Latest Gentoo Stage3 Tarball
1.2.3 Latest Portage snapshot.
1.2.4 SCST 2.2.0 Sources

2 GENTOO INSTALLATION

2.1 Downloads

2.1.1 Download Gentoo x64 Base installation Disk ISO from

http://distfiles.gentoo.org/releases/amd64/autobuilds/current-iso/ - Download the one with the latest date. **Once downloaded, Burn it to disc.**

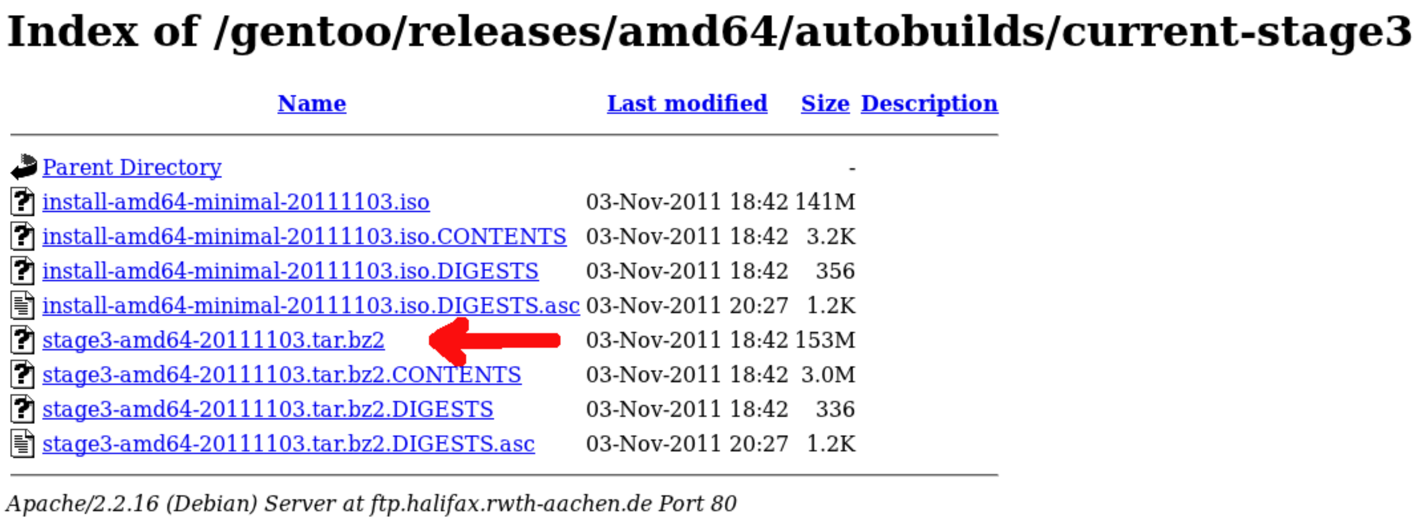


Index of /releases/amd64/autobuilds/current-iso

Name	Last modified	Size	Description
Parent Directory		-	
install-amd64-minimal-20111103.iso	03-Nov-2011 13:42	141M	
install-amd64-minimal-20111103.iso.CONTENTS	03-Nov-2011 13:42	3.2K	
install-amd64-minimal-20111103.iso.DIGESTS	03-Nov-2011 13:42	356	
install-amd64-minimal-20111103.iso.DIGESTS.asc	03-Nov-2011 15:27	1.2K	
stage3-amd64-20111103.tar.bz2	03-Nov-2011 13:42	153M	
stage3-amd64-20111103.tar.bz2.CONTENTS	03-Nov-2011 13:42	3.0M	
stage3-amd64-20111103.tar.bz2.DIGESTS	03-Nov-2011 13:42	336	
stage3-amd64-20111103.tar.bz2.DIGESTS.asc	03-Nov-2011 15:27	1.2K	

2.1.2 Next we need to download the latest Stage 3 Tarball for Gentoo x64, to do this – go to:

http://distfiles.gentoo.org/releases/amd64/autobuilds/current-stage3/ and download the latest one.



Index of /gentoo/releases/amd64/autobuilds/current-stage3

Name	Last modified	Size	Description
Parent Directory		-	
install-amd64-minimal-20111103.iso	03-Nov-2011 18:42	141M	
install-amd64-minimal-20111103.iso.CONTENTS	03-Nov-2011 18:42	3.2K	
install-amd64-minimal-20111103.iso.DIGESTS	03-Nov-2011 18:42	356	
install-amd64-minimal-20111103.iso.DIGESTS.asc	03-Nov-2011 20:27	1.2K	
stage3-amd64-20111103.tar.bz2	03-Nov-2011 18:42	153M	
stage3-amd64-20111103.tar.bz2.CONTENTS	03-Nov-2011 18:42	3.0M	
stage3-amd64-20111103.tar.bz2.DIGESTS	03-Nov-2011 18:42	336	
stage3-amd64-20111103.tar.bz2.DIGESTS.asc	03-Nov-2011 20:27	1.2K	

Apache/2.2.16 (Debian) Server at ftp.halifax.rwth-aachen.de Port 80

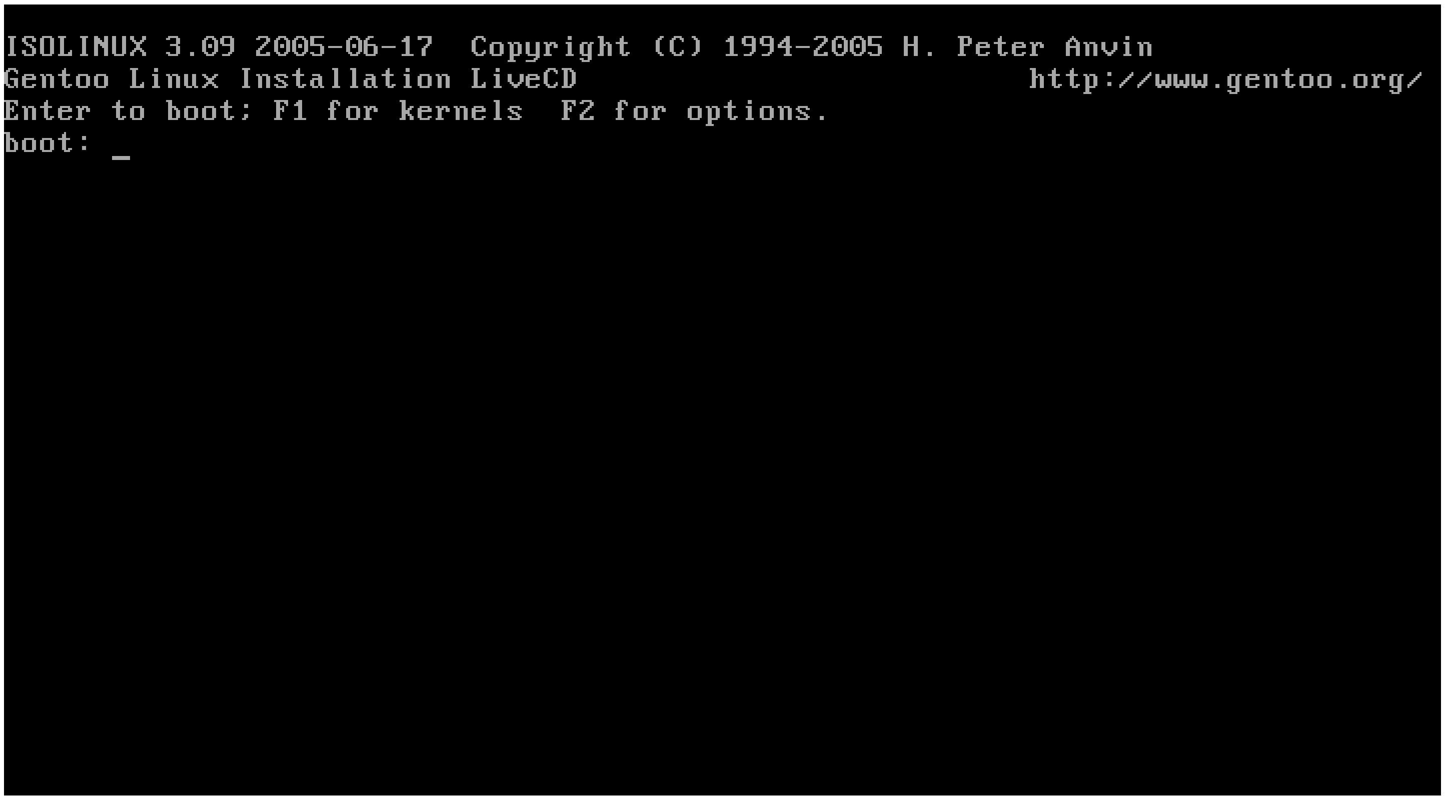
Once Downloaded, copy the file to the memory stick.

2.1.3 Next we need to download the latest Portage snapshot. Go download the latest snapshot from this link: http://de-mirror.org/gentoo/snapshots/portage-latest.tar.bz2 .

Once Downloaded, copy the file to the memory stick.

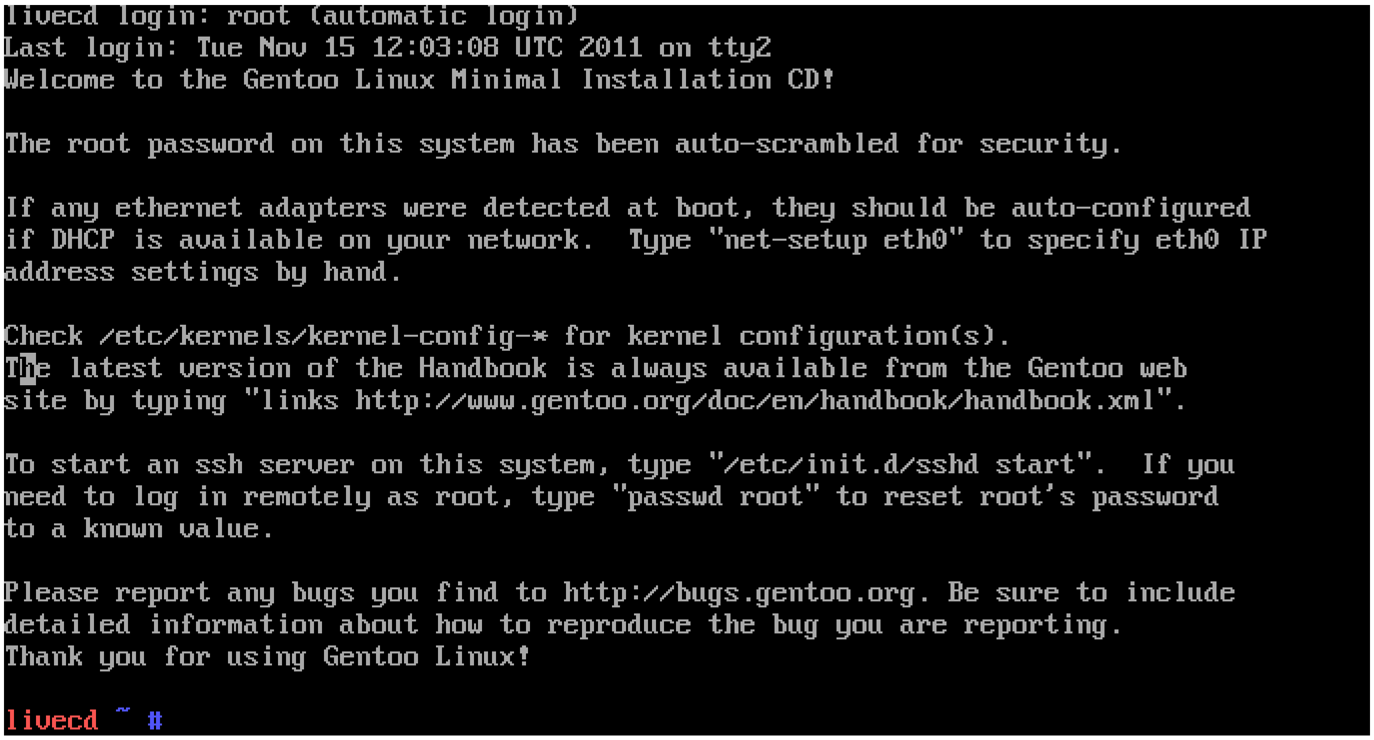
2.2 Installation

2.2.1 Insert the Gentoo CD you burned into the CD Drive on the PC, and press enter on the boot prompt that pops up.



```
ISOLINUX 3.09 2005-06-17  Copyright (C) 1994-2005 H. Peter Anvin
Gentoo Linux Installation LiveCD                        http://www.gentoo.org/
Enter to boot; F1 for kernels  F2 for options.
boot: _
```

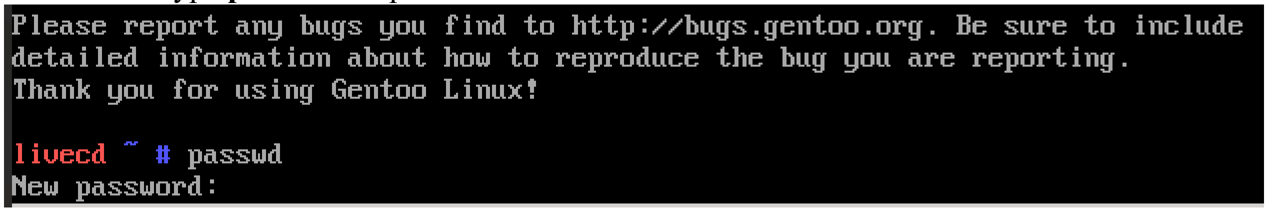
2.2.2 Once booted, you will see this screen:



```
livecd login: root (automatic login)
Last login: Tue Nov 15 12:03:08 UTC 2011 on tty2
Welcome to the Gentoo Linux Minimal Installation CD!

The root password on this system has been auto-scrambled for security.

If any ethernet adapters were detected at boot, they should be auto-configured
if DHCP is available on your network.  Type "net-setup eth0" to specify eth0 IP
address settings by hand.

Check /etc/kernels/kernel-config-* for kernel configuration(s).
The latest version of the Handbook is always available from the Gentoo web
site by typing "links http://www.gentoo.org/doc/en/handbook/handbook.xml".

To start an ssh server on this system, type "/etc/init.d/sshd start".  If you
need to log in remotely as root, type "passwd root" to reset root's password
to a known value.

Please report any bugs you find to http://bugs.gentoo.org. Be sure to include
detailed information about how to reproduce the bug you are reporting.
Thank you for using Gentoo Linux!

livecd ~ #
```

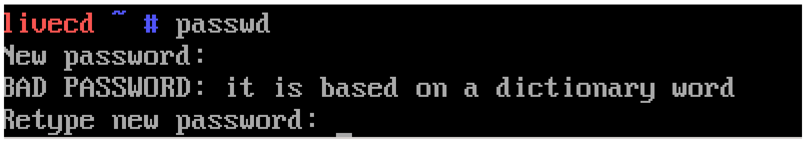
2.2.3 We first need to change the root password

⟶ Type **passwd** and press Enter.



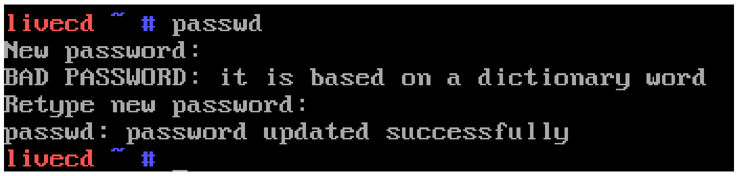
```
Please report any bugs you find to http://bugs.gentoo.org. Be sure to include
detailed information about how to reproduce the bug you are reporting.
Thank you for using Gentoo Linux!

livecd ~ # passwd
New password:
```

⟶ At the prompt, type your new password, and press ENTER.



```
livecd ~ # passwd
New password:
BAD PASSWORD: it is based on a dictionary word
Retype new password: _
```

⟶ Confirm your pasword, and press ENTER again:

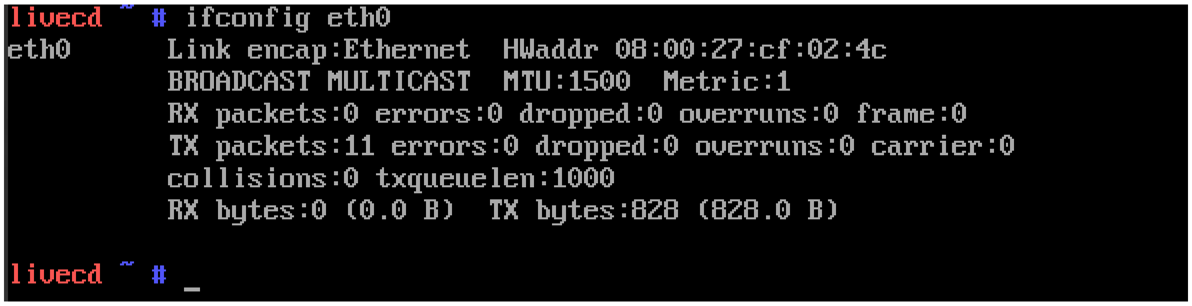


```
livecd ~ # passwd
New password:
BAD PASSWORD: it is based on a dictionary word
Retype new password:
passwd: password updated successfully
livecd ~ # _
```

2.2.4 Next we need to set up the network, we will simply use dhcp to set up the network card, and set a fixed IP later. **Make sure that you have Internet access**!

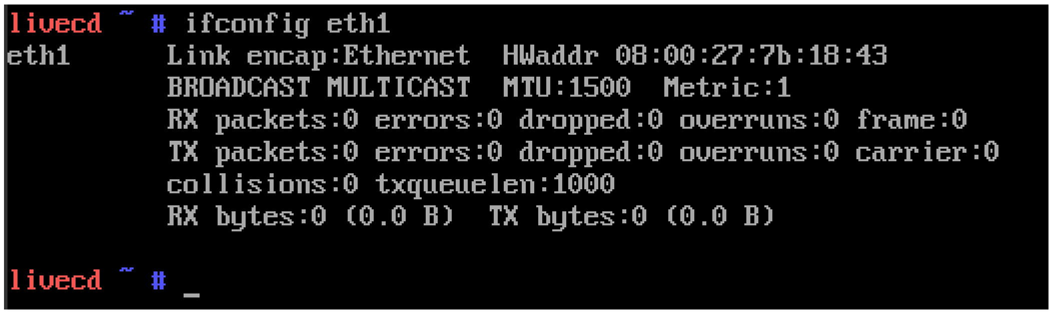
2.2.4.1 First we need to check if the live cd picked up our network cards properly. We have 3 of them so in linux – the names of the cards will be **eth0**, **eth1** and **eth2** respectively. To test if the live cd picked them all up.

⟶ Type **ifconfig eth0** and press Enter.



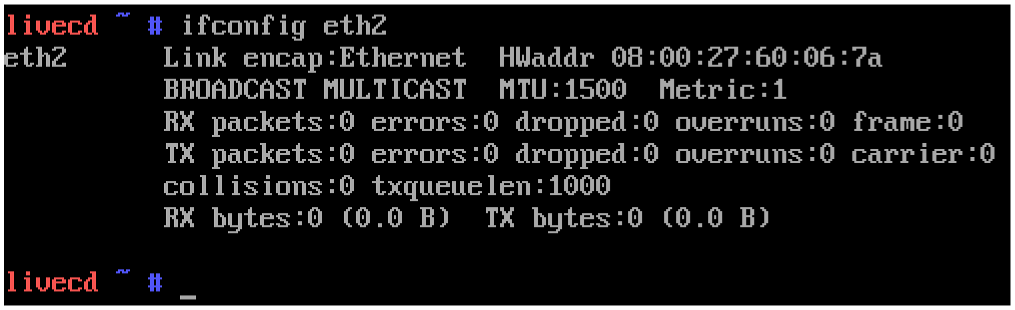
```
livecd ~ # ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:cf:02:4c
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:11 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:828 (828.0 B)

livecd ~ # _
```

⟶ Type **ifconfig eth1** and press Enter.



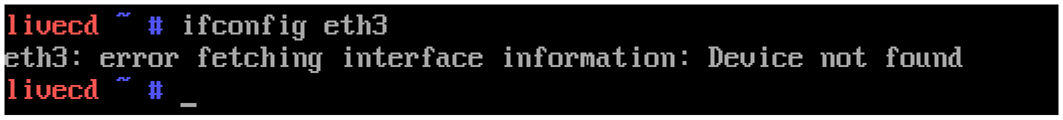
```
livecd ~ # ifconfig eth1
eth1      Link encap:Ethernet  HWaddr 08:00:27:7b:18:43
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

livecd ~ # _
```

⟶ Type **ifconfig eth2** and press Enter.



```
livecd ~ # ifconfig eth2
eth2      Link encap:Ethernet  HWaddr 08:00:27:60:06:7a
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

livecd ~ # _
```

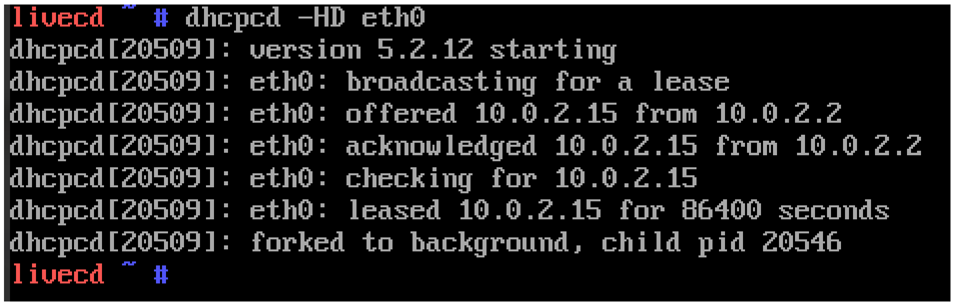
If you see the image below for any of the network cards, swop it out with another one. It does not neccecarily mean the card is faulty, it could be that it has no drivers for that model of card. Loading extra drivers is beyond the scope of this document.



```
livecd ~ # ifconfig eth3
eth3: error fetching interface information: Device not found
livecd ~ # _
```

2.2.4.2 Now to set up the primary NIC – the one we going to use to manage the box – eth0.

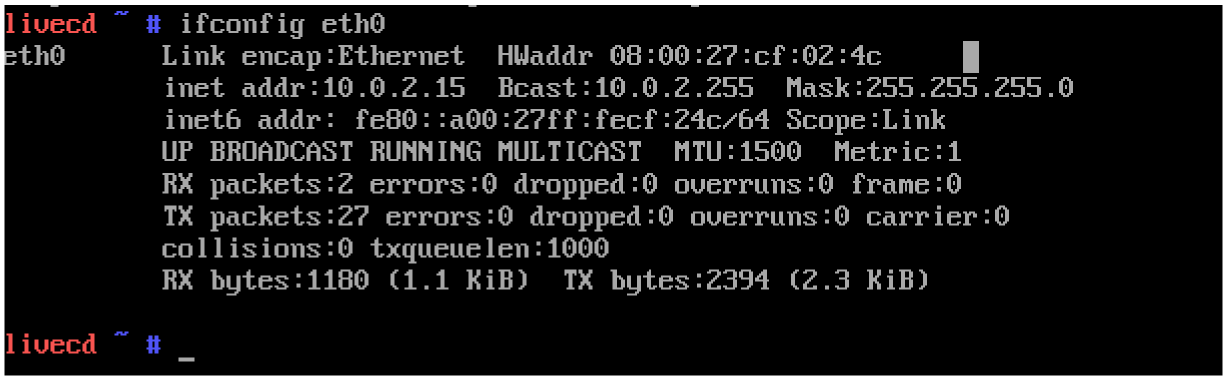
⟶ Type **dhcpcd -HD eth0** and press Enter.



```
livecd ~ # dhcpcd -HD eth0
dhcpcd[20509]: version 5.2.12 starting
dhcpcd[20509]: eth0: broadcasting for a lease
dhcpcd[20509]: eth0: offered 10.0.2.15 from 10.0.2.2
dhcpcd[20509]: eth0: acknowledged 10.0.2.15 from 10.0.2.2
dhcpcd[20509]: eth0: checking for 10.0.2.15
dhcpcd[20509]: eth0: leased 10.0.2.15 for 86400 seconds
dhcpcd[20509]: forked to background, child pid 20546
livecd ~ #
```

Next we need to check if we got an ip address. You will see the following, if it is missing an ip address – check your network cable / DHCP Server:

⟶ Type **ifconfig eth0** and press Enter.

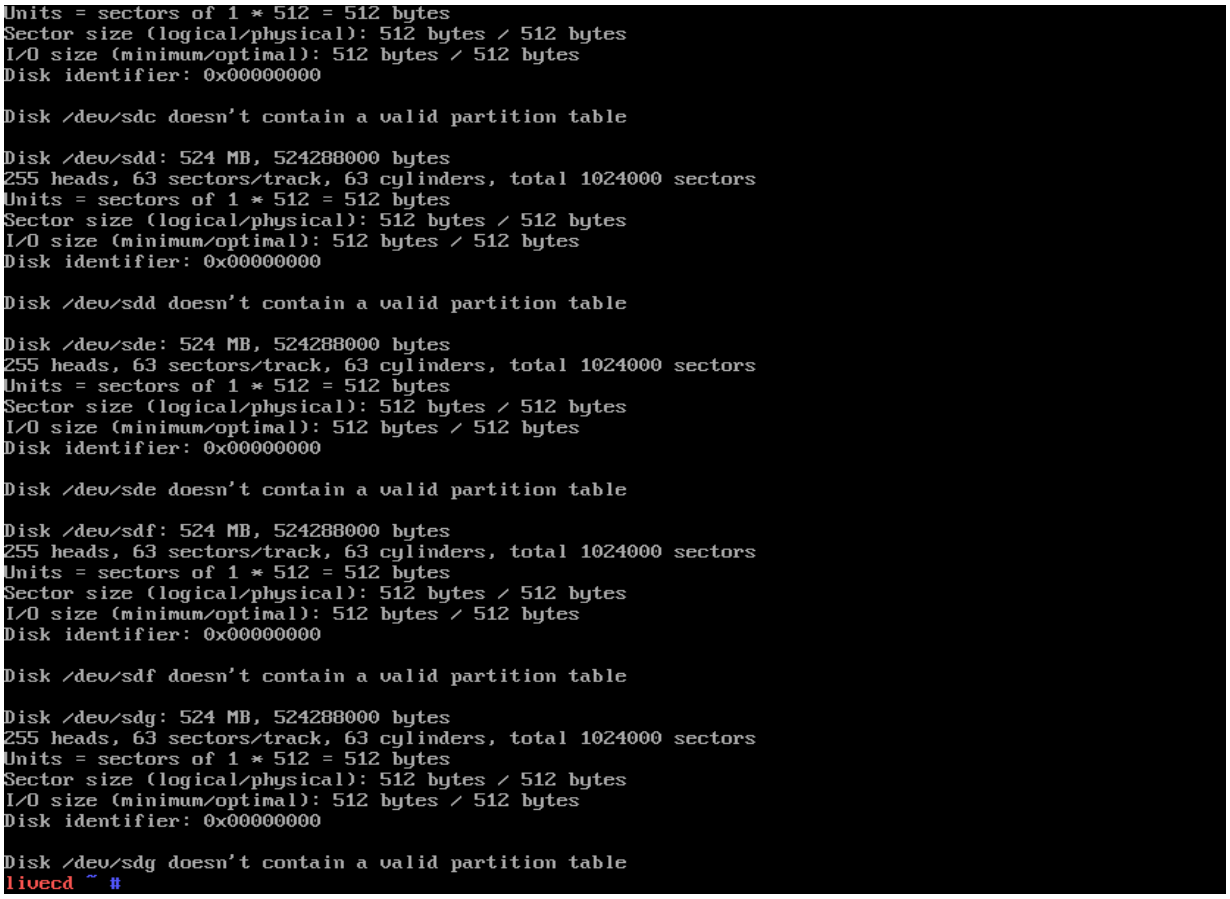


```
livecd ~ # ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:cf:02:4c
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fecf:24c/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2 errors:0 dropped:0 overruns:0 frame:0
          TX packets:27 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:1180 (1.1 KiB)  TX bytes:2394 (2.3 KiB)

livecd ~ # _
```

2.2.5 Time to ready our main hard drive for the installation.

2.2.5.1 Lets first list what drives we have available to us. The output will look as follows:

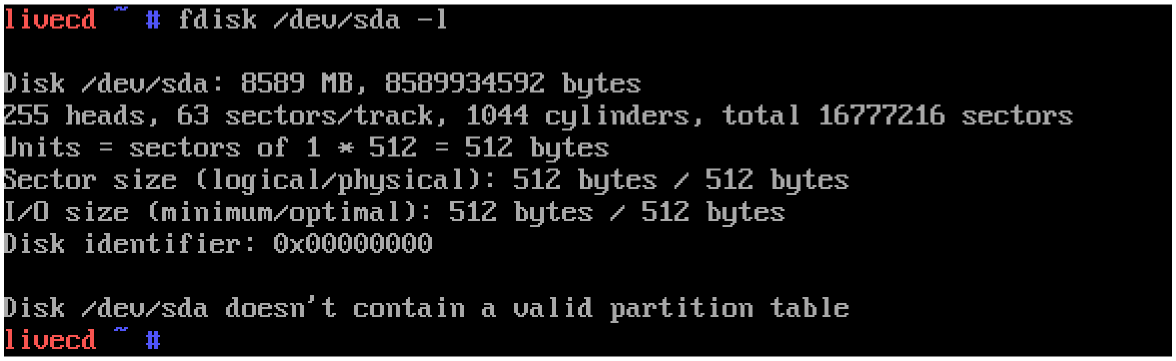
→ Type **fdisk -l** and press Enter.



```
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdc doesn't contain a valid partition table

Disk /dev/sdd: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdd doesn't contain a valid partition table

Disk /dev/sde: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sde doesn't contain a valid partition table

Disk /dev/sdf: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdf doesn't contain a valid partition table

Disk /dev/sdg: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sdg doesn't contain a valid partition table
livecd ~ #
```

As you can see, it is displaying all the disks installed in the system, and our installation disk is out of the display. As it is connected to port 1 on the SATA / SAS controller, it should be /dev/sda, Let's confirm that – on this test setup I used to get the screenshots – the system disk is 8 GB in size, so lets list only /dev/sda on the fdisk output. Output should show similar to this(your disk size obviously)**:**

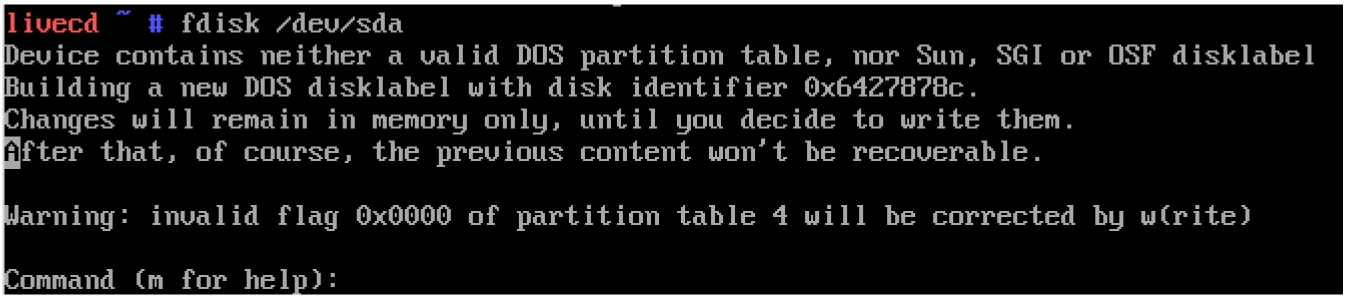
→ Type **fdisk /dev/sda -l** and press Enter.



```
livecd ~ # fdisk /dev/sda -l

Disk /dev/sda: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 cylinders, total 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/sda doesn't contain a valid partition table
livecd ~ #
```

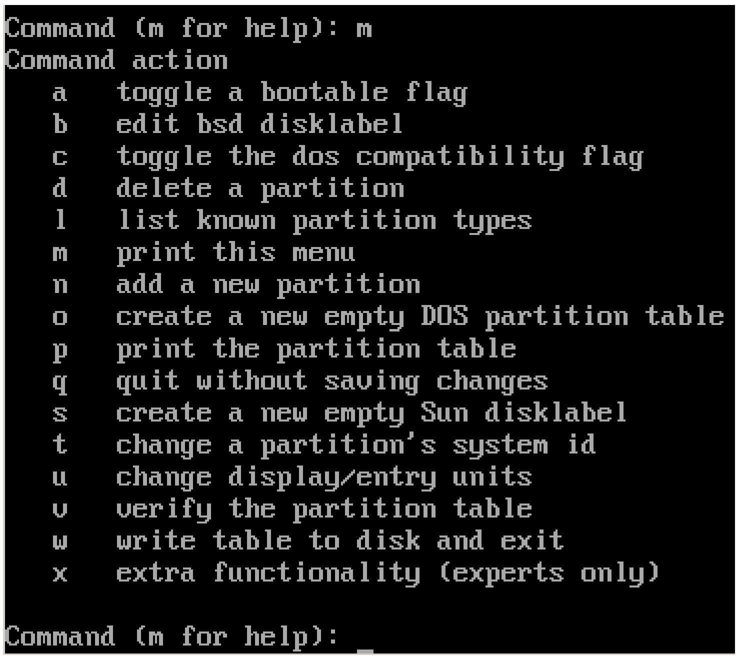
2.2.5.2 Ok, so now that we know that it is picking up our drive correctly, we can start partitioning it to hold our operating system. To do this, we have to run fdisk again, but this time we will not be passing ti the **-l** parameter.

⟶ Type **fdisk /dev/sda** and press Enter.



```
livecd ~ # fdisk /dev/sda
Device contains neither a valid DOS partition table, nor Sun, SGI or OSF disklabel
Building a new DOS disklabel with disk identifier 0x6427878c.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help):
```

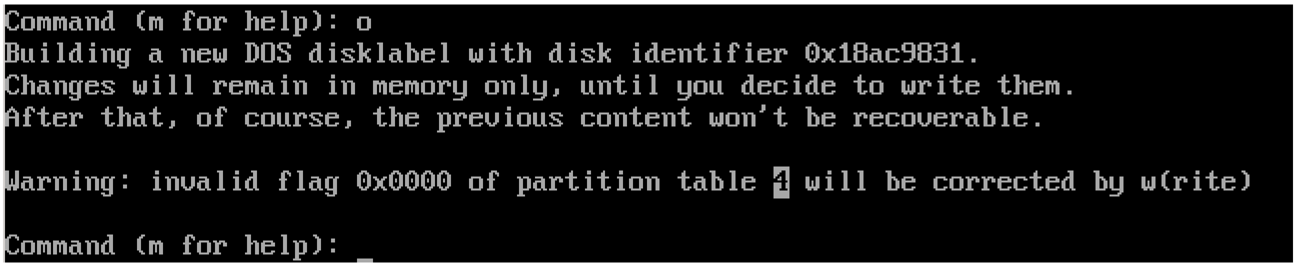
⟶ **m** will show a list of the available options:



```
Command (m for help): m
Command action
   a   toggle a bootable flag
   b   edit bsd disklabel
   c   toggle the dos compatibility flag
   d   delete a partition
   l   list known partition types
   m   print this menu
   n   add a new partition
   o   create a new empty DOS partition table
   p   print the partition table
   q   quit without saving changes
   s   create a new empty Sun disklabel
   t   change a partition's system id
   u   change display/entry units
   v   verify the partition table
   w   write table to disk and exit
   x   extra functionality (experts only)

Command (m for help): _
```

First we need to create a DOS partition table. The output will be similar to this:

⟶ Type **o** and press Enter.



```
Command (m for help): o
Building a new DOS disklabel with disk identifier 0x18ac9831.
Changes will remain in memory only, until you decide to write them.
After that, of course, the previous content won't be recoverable.

Warning: invalid flag 0x0000 of partition table 4 will be corrected by w(rite)

Command (m for help): _
```

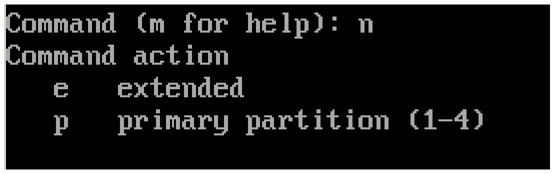
Now that we have our partition table, we can start partitioning the disk. First lets look at the partitions we want to create. We need 2 basic partitions at minimum, one for the root of the filesystem, and one for swapspace. We will be creating 3 - / , /boot and swap.

To Summarize:

Partition	**Size**	**Use**
/dev/sda1	200MB	/boot
/dev/sda2	7365MB	/
/dev/sda3	1024MB	swap

This will be the layout for my 8 GB Disk, You want to assign the majority of your disk space to the / partition, as this will be holding your operating system. Lets create our first partition.
It now asks us what kind of partition we want to create – as seen below:

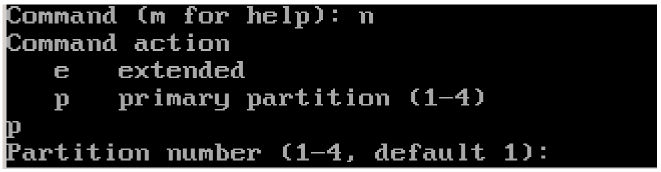
⟶ Type **n** and press enter.



```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
```

We want to create a primary partition. It will now ask us what partition number we want to create.

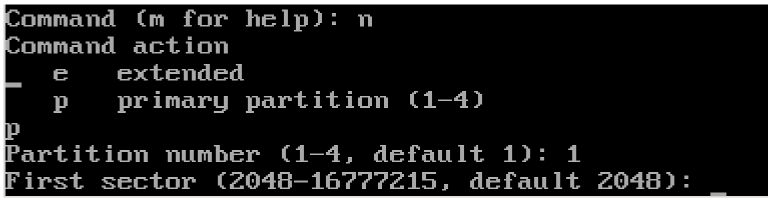
⟶ Type **p** and press Enter



```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4, default 1):
```

This is the first partition.

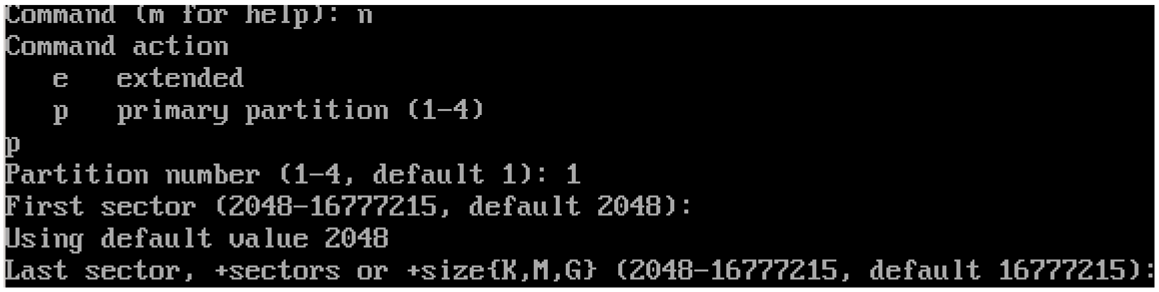
⟶ Type **1** and then Enter.



```
Command (m for help): n
Command action
_  e   extended
   p   primary partition (1-4)
p
Partition number (1-4, default 1): 1
First sector (2048-16777215, default 2048): _
```

At the First Sector Prompt, press Enter to accept the default value. The following will show afterwards:

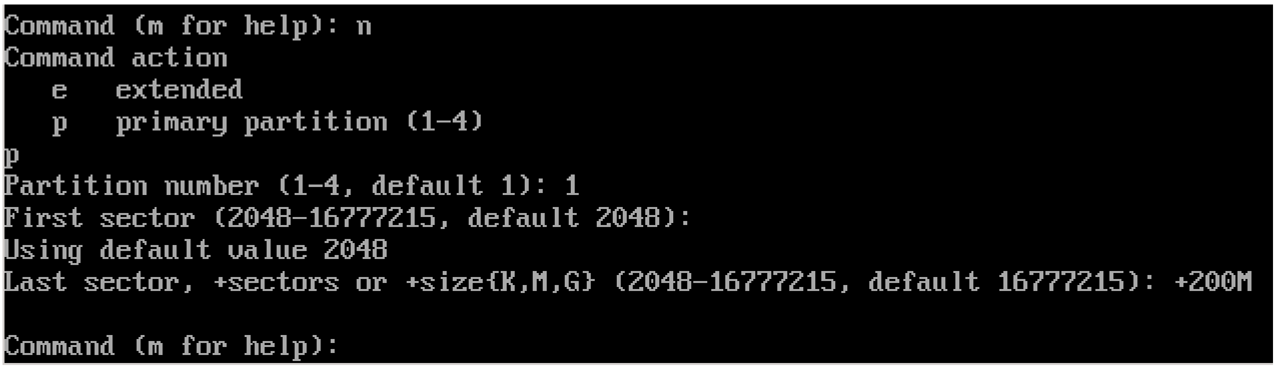
⟶ Press Enter to accept the default value.



```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4, default 1): 1
First sector (2048-16777215, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-16777215, default 16777215):
```

At this prompt, we need to tell fdisk how big we want our partition to be. We are creating the /boot partition, which we want to be 200MB. We are then returned to the “main menu”, our partition has been created successfully!

⟶ Type **+200M** and press enter

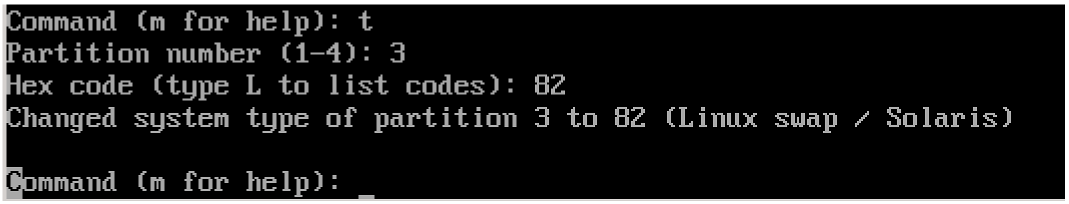


```
Command (m for help): n
Command action
   e   extended
   p   primary partition (1-4)
p
Partition number (1-4, default 1): 1
First sector (2048-16777215, default 2048):
Using default value 2048
Last sector, +sectors or +size{K,M,G} (2048-16777215, default 16777215): +200M

Command (m for help):
```

Repeat the above steps for each of the partitions, changing only the size of the partition and the partition number (2nd partition = 2 etc...) accordingly. (**Note: on the last partition you create, just press Enter on all the prompts to accept the defaults.**)

Now we just need to tell linux that /dev/sda3 is a swap partition.

⟶ To do this press **t** and Enter**,** then **3** and Enter, then **82** and Enter

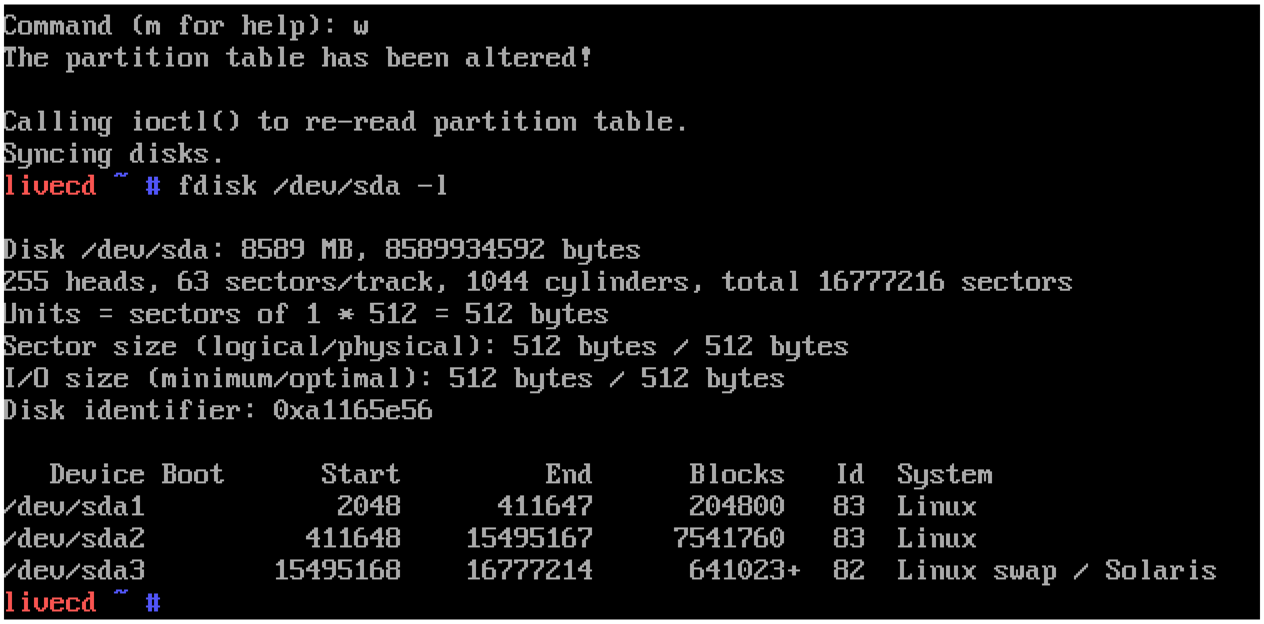


```
Command (m for help): t
Partition number (1-4): 3
Hex code (type L to list codes): 82
Changed system type of partition 3 to 82 (Linux swap / Solaris)

Command (m for help): _
```

When done, press **w** at the main menu to write the changes to disk.

Let's also look at the new partition structure on /dev/sda**.**

⟶ Type **fdisk /dev/sda -l** and press Enter



```
Command (m for help): w
The partition table has been altered!

Calling ioctl() to re-read partition table.
Syncing disks.
livecd ~ # fdisk /dev/sda -l

Disk /dev/sda: 8589 MB, 8589934592 bytes
255 heads, 63 sectors/track, 1044 cylinders, total 16777216 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xa1165e56

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1            2048      411647      204800   83  Linux
/dev/sda2          411648    15495167     7541760   83  Linux
/dev/sda3        15495168    16777214      641023+  82  Linux swap / Solaris
livecd ~ #
```

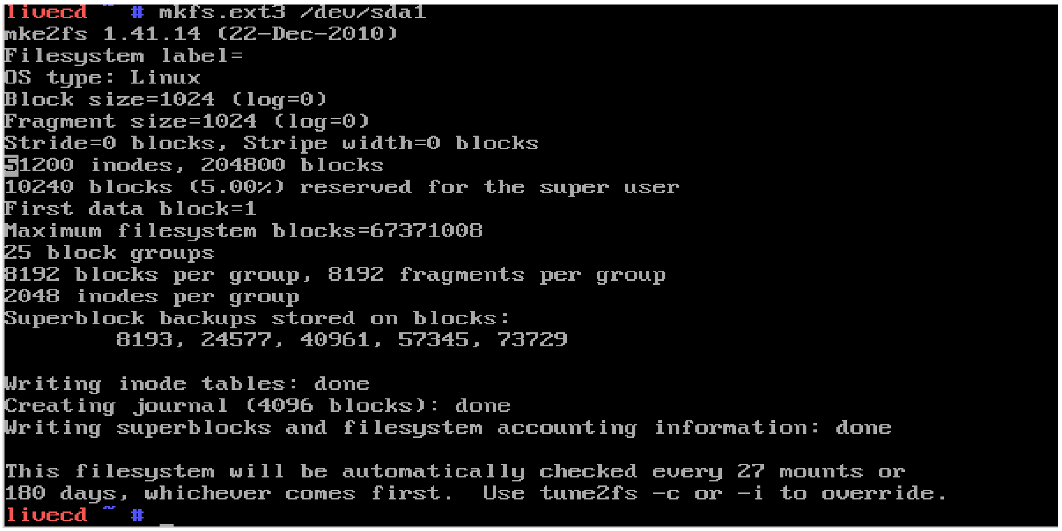
2.2.5.3 Now we need to format our partitions with the appropriate filesystems, and create & activate the swap space.

First off, lets decide what filesystems to use:

Partition	**Mount Point**	**Filesystem**
/dev/sda1	/boot	EXT3
/dev/sda2	/	EXT4
/dev/sda3	Swap	N/A

To create our first filesystem for /dev/sda1.

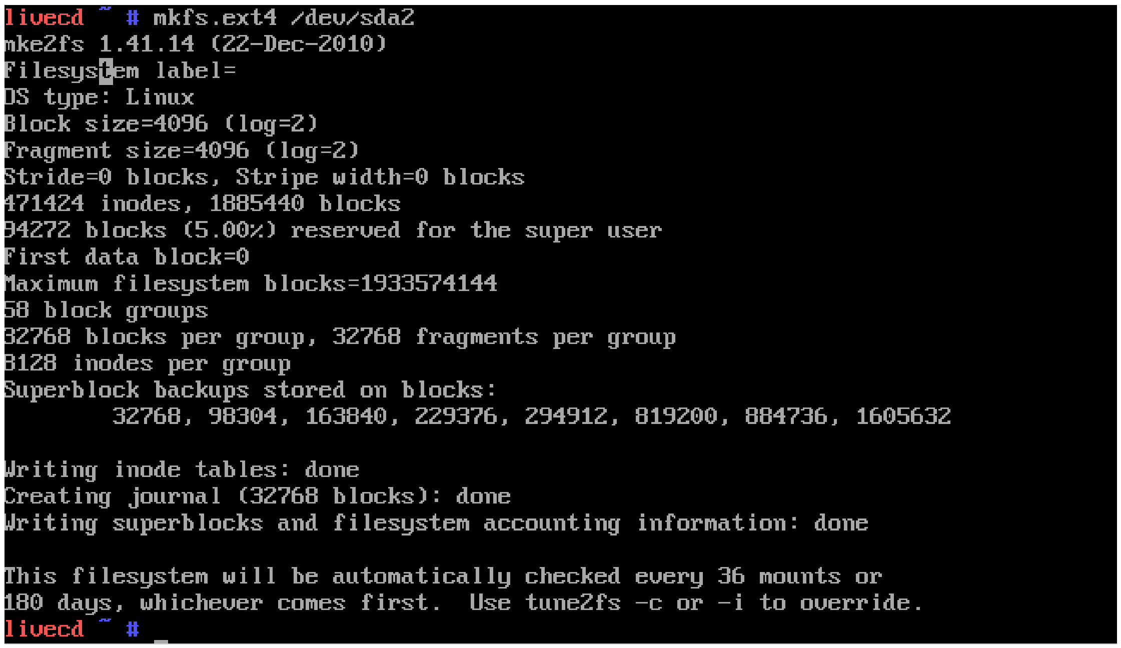
→ Type **mkfs.ext3 /dev/sda1** and press Enter.



```
livecd ~ # mkfs.ext3 /dev/sda1
mke2fs 1.41.14 (22-Dec-2010)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
51200 inodes, 204800 blocks
10240 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67371008
25 block groups
8192 blocks per group, 8192 fragments per group
2048 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729

Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 27 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
livecd ~ # _
```

To create our filesystem for /dev/sda2.

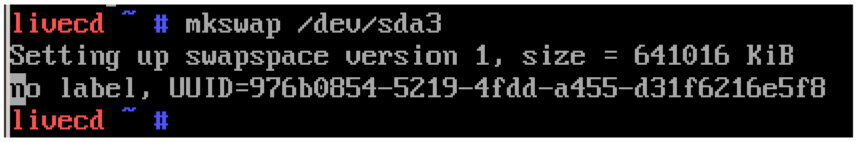
→ Type **mkfs.ext4 /dev/sda2** and press Enter.



```
livecd ~ # mkfs.ext4 /dev/sda2
mke2fs 1.41.14 (22-Dec-2010)
Filesystem label=
OS type: Linux
Block size=4096 (log=2)
Fragment size=4096 (log=2)
Stride=0 blocks, Stripe width=0 blocks
471424 inodes, 1885440 blocks
94272 blocks (5.00%) reserved for the super user
First data block=0
Maximum filesystem blocks=1933574144
58 block groups
32768 blocks per group, 32768 fragments per group
8128 inodes per group
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632

Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 36 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
livecd ~ # _
```

Now, to create our swapspace.

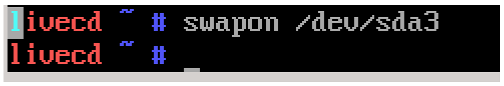
⟶ Type **mkswap /dev/sda3** and press Enter



```
livecd ~ # mkswap /dev/sda3
Setting up swapspace version 1, size = 641016 KiB
no label, UUID=976b0854-5219-4fdd-a455-d31f6216e5f8
livecd ~ #
```

And finally – we need to activate the Swapspace.

⟶ Type **swapon /dev/sda3** and press Enter.

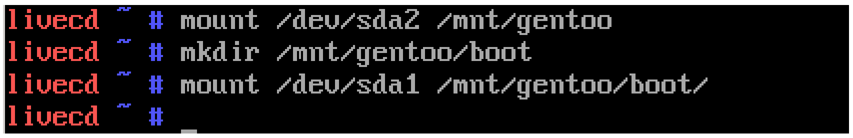


```
livecd ~ # swapon /dev/sda3
livecd ~ # _
```

You'll notice – there's no output telling us that it has been done – but it has.

2.2.5.4 Now that your partitions are initialized and are housing a filesystem, it is time to mount those partitions. Use the mount command. Don't forget to create the necessary mount directories for every partition you created. As an example we mount the root and boot partition:

⟶ Type **mount /dev/sda2 /mnt/gentoo** and press Enter.
⟶ Type **mkdir /mnt/gentoo/boot** and press Enter.
⟶ Type **mount /dev/sda1 /mnt/gentoo/boot** and press Enter,



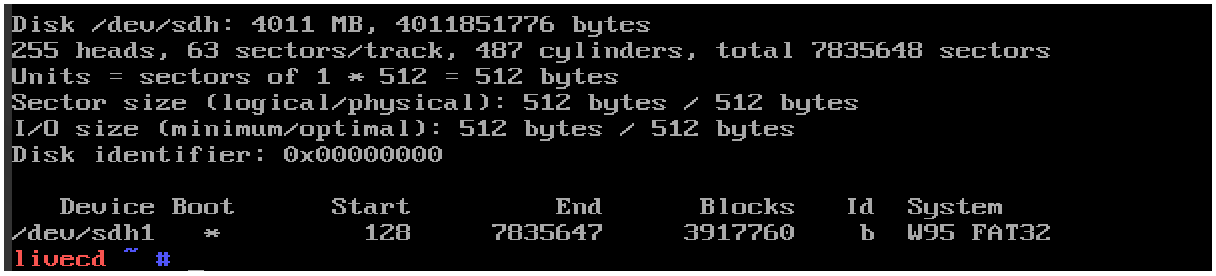
```
livecd ~ # mount /dev/sda2 /mnt/gentoo
livecd ~ # mkdir /mnt/gentoo/boot
livecd ~ # mount /dev/sda1 /mnt/gentoo/boot/
livecd ~ # _
```

2.2.6 Ok! Now we get to the fun part, installing Gentoo.

2.2.6.1 Firstly insert the usb memory stick containing the two .tar.gz files into the pc.

Let's check whether it has been detected, to do this we use the **fdisk -l** command again. It should be the last drive on the list – in my case **/dev/sdh**:

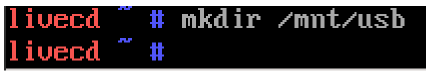
⟶ Type **fdisk -l** and press Enter.



```
Disk /dev/sdh: 4011 MB, 4011851776 bytes
255 heads, 63 sectors/track, 487 cylinders, total 7835648 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/sdh1   *         128     7835647     3917760    b  W95 FAT32
livecd ~ # _
```

Now we need to mount the memory stick so that we can access the data on it, to do this we first need to create a folder to mount it on.

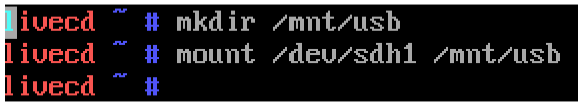
⟶ Type mkdir **/mnt/usb,** this will create a folder under **/mnt** called **usb.**



```
livecd ~ # mkdir /mnt/usb
livecd ~ #
```

Now to mount our memory stick. In my case the device name **/dev/sdh**, and the partition we want to mount is **/dev/sdh1**.

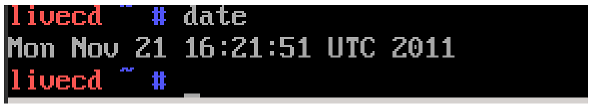
⟶ Type **mount /dev/sdh1 /mnt/usb** and press Enter.



```
livecd ~ # mkdir /mnt/usb
livecd ~ # mount /dev/sdh1 /mnt/usb
livecd ~ #
```

Before you continue you need to check your date/time and update it. A mis-configured clock may lead to strange results in the future!

⟶ To verify the current date/time, type **date** and press Enter**.**

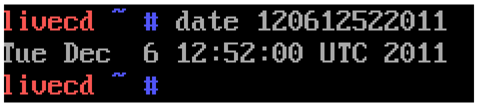


```
livecd ~ # date
Mon Nov 21 16:21:51 UTC 2011
livecd ~ # _
```

If the date/time displayed is wrong, update it using the **date MMDDhhmmYYYY** syntax (**M**onth, **D**ay, **h**our, **m**inute and **Y**ear). At this stage, you should use UTC time. You will be able to define your timezone later on.

For instance, to set the date to December 06th, 12:52 in the year 2011,

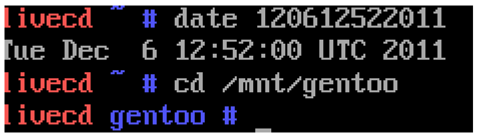
⟶ Type **date 120612522011** and press Enter**.**



```
livecd ~ # date 120612522011
Tue Dec  6 12:52:00 UTC 2011
livecd ~ #
```

Go to the Gentoo mountpoint at which you mounted your filesystems (/mnt/gentoo).

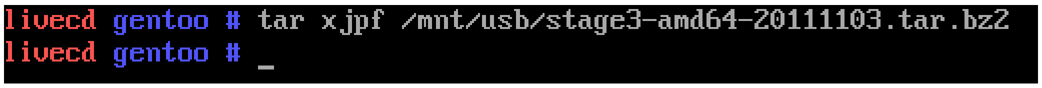
⟶ Type **cd /mnt/gentoo** and press Enter.



```
livecd ~ # date 120612522011
Tue Dec  6 12:52:00 UTC 2011
livecd ~ # cd /mnt/gentoo
livecd gentoo # _
```

Now unpack your downloaded stage 3 tarball onto your system. We use **tar** to proceed as it is the easiest method.

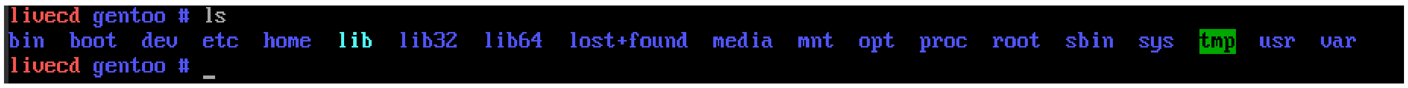
⟶ Type **tar xvjpf /mnt/usb/stage3-*.tar.bz** and press Enter**.** (the stage3-*.tar.bz should be the name of your stage3 file on the memory stick):



```
livecd gentoo # tar xjpf /mnt/usb/stage3-amd64-20111103.tar.bz2
livecd gentoo # _
```

Lets check that the extraction has been completed.

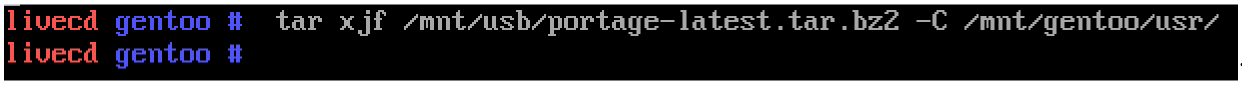
⟶ Type **ls** and press Enter.



```
livecd gentoo # ls
bin  boot  dev  etc  home  lib  lib32  lib64  lost+found  media  mnt  opt  proc  root  sbin  sys  tmp  usr  var
livecd gentoo # _
```

In the next step, we extract the Portage snapshot onto your filesystem. Make sure that you use the exact command; the last option is a capital C, not c

⟶ Type **tar xvjf /mnt/usb/portage-latest.tar.bz2 -C /mnt/gentoo/usr** and press Enter.

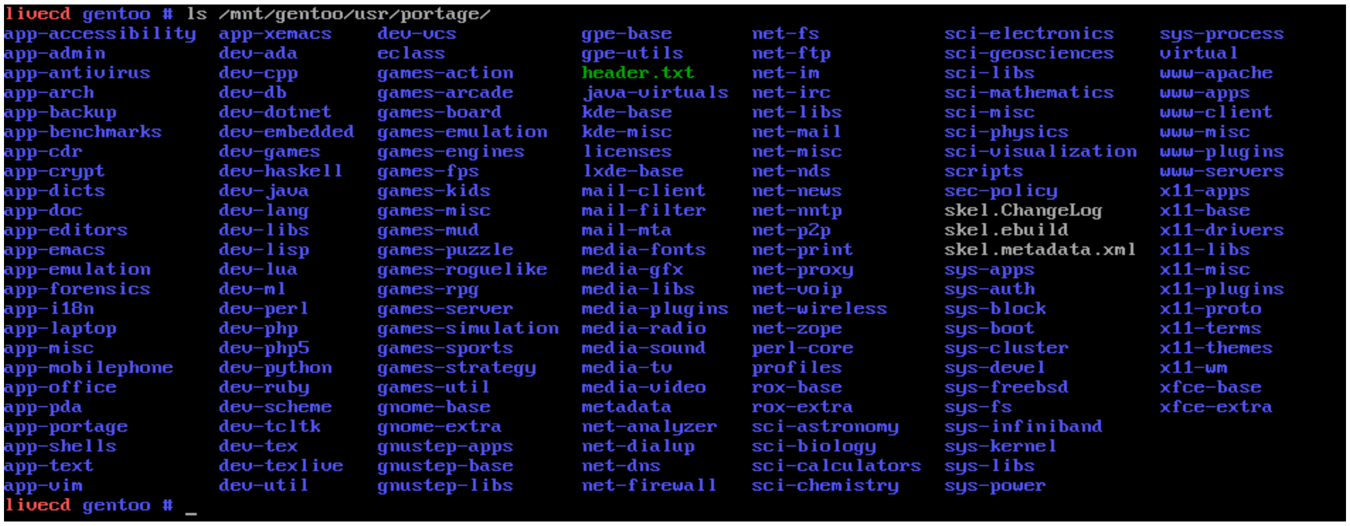


```
livecd gentoo #  tar xjf /mnt/usb/portage-latest.tar.bz2 -C /mnt/gentoo/usr/
livecd gentoo #
```

.

Lets check that the extraction has been completed.

⟶ Type **ls /mnt/gentoo/usr/portage** and press Enter.



```
livecd gentoo # ls /mnt/gentoo/usr/portage/
app-accessibility  app-xemacs    dev-vcs           gpe-base       net-fs           sci-electronics    sys-process
app-admin          dev-ada       eclass            gpe-utils      net-ftp          sci-geosciences    virtual
app-antivirus      dev-cpp       games-action      header.txt     net-im           sci-libs           www-apache
app-arch           dev-db        games-arcade      java-virtuals  net-irc          sci-mathematics    www-apps
app-backup         dev-dotnet    games-board       kde-base       net-libs         sci-misc           www-client
app-benchmarks     dev-embedded  games-emulation   kde-misc       net-mail         sci-physics        www-misc
app-cdr            dev-games     games-engines     licenses       net-misc         sci-visualization  www-plugins
app-crypt          dev-haskell   games-fps         lxde-base      net-nds          scripts            www-servers
app-dicts          dev-java      games-kids        mail-client    net-news         sec-policy         x11-apps
app-doc            dev-lang      games-misc        mail-filter    net-nntp         skel.ChangeLog     x11-base
app-editors        dev-libs      games-mud         mail-mta       net-p2p          skel.ebuild        x11-drivers
app-emacs          dev-lisp      games-puzzle      media-fonts    net-print        skel.metadata.xml  x11-libs
app-emulation      dev-lua       games-roguelike   media-gfx      net-proxy        sys-apps           x11-misc
app-forensics      dev-ml        games-rpg         media-libs     net-voip         sys-auth           x11-plugins
app-i18n           dev-perl      games-server      media-plugins  net-wireless     sys-block          x11-proto
app-laptop         dev-php       games-simulation  media-radio    net-zope         sys-boot           x11-terms
app-misc           dev-php5      games-sports      media-sound    perl-core        sys-cluster        x11-themes
app-mobilephone    dev-python    games-strategy    media-tv       profiles         sys-devel          x11-wm
app-office         dev-ruby      games-util        media-video    rox-base         sys-freebsd        xfce-base
app-pda            dev-scheme    gnome-base        metadata       rox-extra        sys-fs             xfce-extra
app-portage        dev-tcltk     gnome-extra       net-analyzer   sci-astronomy    sys-infiniband
app-shells         dev-tex       gnustep-apps      net-dialup     sci-biology      sys-kernel
app-text           dev-texlive   gnustep-base      net-dns        sci-calculators  sys-libs
app-vim            dev-util      gnustep-libs      net-firewall   sci-chemistry    sys-power
livecd gentoo # _
```

Now we need to configure our make.conf file, for the sake of making things easier, just make sure your file **/mnt/gentoo/etc/make.conf** looks the same as below.

⟶ Type **nano /mnt/gentoo/etc/make.conf** and press Enter.

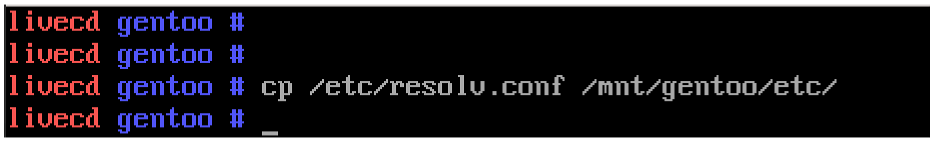
```
CFLAGS="-O2 -march=k8 -pipe -mno-tls-direct-seg-refs"
CXXFLAGS="${CFLAGS}"
CHOST="x86_64-pc-linux-gnu"
MAKEOPTS="-j2"                ## -j2 means 2 core processor – so for quad core type -j4
FEATURES="parallel-fetch ccache"

DISTDIR=/home/portage/distfiles
KERNEL_DIR=/usr/src/linux

ACCEPT_LICENSE="*"

USE="mmx sse sse2"

GENTOO_MIRRORS="http://distfiles.gentoo.org http://ftp.twaren.net/Linux/Gentoo/
http://mirror.switch.ch/ftp/mirror/gentoo http://ftp.snt.utwente.nl/pub/os/linux/gentoo"
```

```
SYNC="rsync://rsync1.de.gentoo.org/gentoo-portage"
```

2.2.6.2 Now we get to installing the Gentoo Base System.

One thing still remains to be done before we enter the new environment and that is copying over the DNS information in **/etc/resolv.conf**. You need to do this to ensure that networking still works even after entering the new environment. **/etc/resolv.conf** contains the nameservers for your network.

⟶ Type **cp /etc/resolv.conf /mnt/gentoo/etc/** and press Enter.



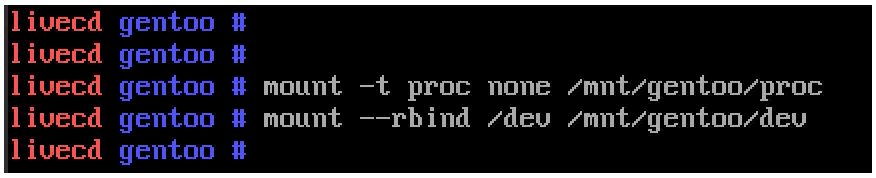
```
livecd gentoo #
livecd gentoo #
livecd gentoo # cp /etc/resolv.conf /mnt/gentoo/etc/
livecd gentoo # _
```

In a few moments, we will change the Linux root towards the new location. To make sure that the new environment works properly, we need to make certain file systems available there as well.

Mount the /proc filesystem on /mnt/gentoo/proc to allow the installation to use the kernel-provided information within the chrooted environment, and then mount-bind the /dev filesystem.

⟶ Type **mount -t proc none /mnt/gentoo/proc** and press Enter.

⟶ Type **mount --rbind /dev /mnt/gentoo/dev** and press Enter.



```
livecd gentoo #
livecd gentoo #
livecd gentoo # mount -t proc none /mnt/gentoo/proc
livecd gentoo # mount --rbind /dev /mnt/gentoo/dev
livecd gentoo #
```

2.2.6.3 Now that all partitions are initialized and the base environment installed, it is time to enter our new installation environment by chrooting into it. This means that we change from the current installation environment (Installation CD or other installation medium) to your installation system (namely the initialized partitions).

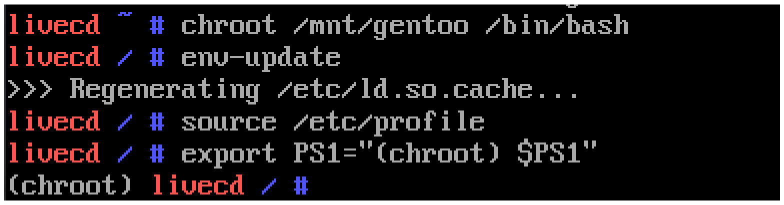
This chrooting is done in three steps. First we will change the root from / (on the installation medium) to /mnt/gentoo (on your partitions) using **chroot**. Then we will create a new environment using **env-update**, which essentially creates environment variables. Finally, we load those variables into memory using **source**.

⟶ Type **chroot /mnt/gentoo /bin/bash** and press Enter.

⟶ Type **env-update** and press Enter. (If you get errors, check your make.conf file)

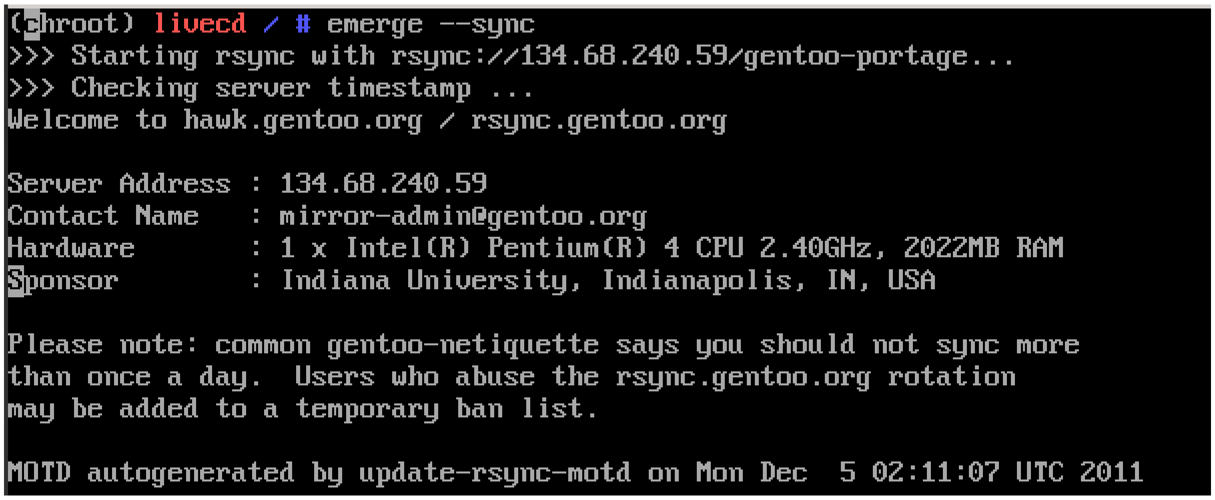
⟶ Type **source /etc/profile** and press Enter.

⟶ Type **export PS1="(chroot) $PS1"** and press Enter.



```
livecd ~ # chroot /mnt/gentoo /bin/bash
livecd / # env-update
>>> Regenerating /etc/ld.so.cache...
livecd / # source /etc/profile
livecd / # export PS1="(chroot) $PS1"
(chroot) livecd / #
```

You should now update your Portage tree to the latest version. **emerge --sync** does this for you.



```
(chroot) livecd / # emerge --sync
>>> Starting rsync with rsync://134.68.240.59/gentoo-portage...
>>> Checking server timestamp ...
Welcome to hawk.gentoo.org / rsync.gentoo.org

Server Address : 134.68.240.59
Contact Name   : mirror-admin@gentoo.org
Hardware       : 1 x Intel(R) Pentium(R) 4 CPU 2.40GHz, 2022MB RAM
Sponsor        : Indiana University, Indianapolis, IN, USA

Please note: common gentoo-netiquette says you should not sync more
than once a day.  Users who abuse the rsync.gentoo.org rotation
may be added to a temporary ban list.

MOTD autogenerated by update-rsync-motd on Mon Dec  5 02:11:07 UTC 2011
```

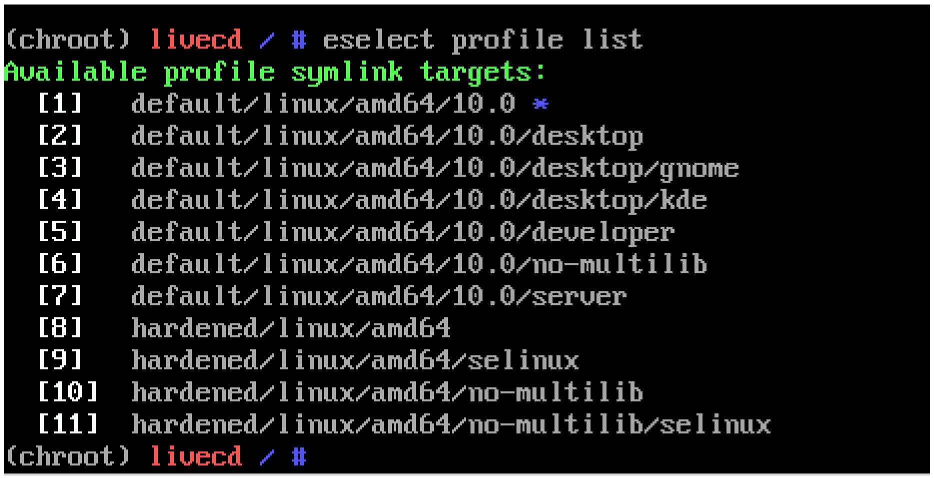
First, a small definition is in order.

A profile is a building block for any Gentoo system. Not only does it specify default values for USE, CFLAGS and other important variables, it also locks the system to a certain range of package versions. This is all maintained by the Gentoo developers.

Previously, such a profile was untouched by the users. However, there may be certain situations in which you may decide a profile change is necessary.

You can see what profile you are currently using with the following command:

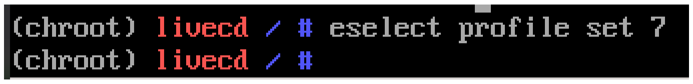
⟶ Type **eselect profile list** and press Enter.



```
(chroot) livecd / # eselect profile list
Available profile symlink targets:
  [1]   default/linux/amd64/10.0 *
  [2]   default/linux/amd64/10.0/desktop
  [3]   default/linux/amd64/10.0/desktop/gnome
  [4]   default/linux/amd64/10.0/desktop/kde
  [5]   default/linux/amd64/10.0/developer
  [6]   default/linux/amd64/10.0/no-multilib
  [7]   default/linux/amd64/10.0/server
  [8]   hardened/linux/amd64
  [9]   hardened/linux/amd64/selinux
  [10]  hardened/linux/amd64/no-multilib
  [11]  hardened/linux/amd64/no-multilib/selinux
(chroot) livecd / #
```

We want to run the Server profile, so type the following to change to it:

⟶ Type eselect profile set 7 and press Enter. (**If the Server profile number differs on your list, change the 7 to the corresponding number.**)

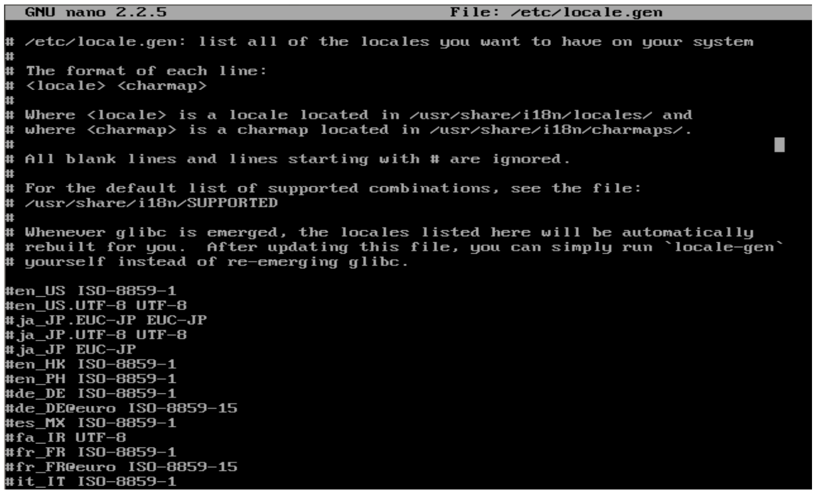


```
(chroot) livecd / # eselect profile set 7
(chroot) livecd / #
```

2.2.6.4 Next, we'll be configuring the kernel.

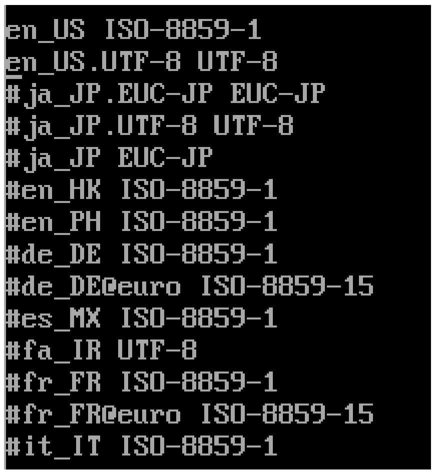
First we need to set the glibc locales, we need to edit the file **/etc/locale.gen.**

⟶ Type **nano /etc/locale.gen** and press Enter.



```
  GNU nano 2.2.5                   File: /etc/locale.gen

# /etc/locale.gen: list all of the locales you want to have on your system
#
# The format of each line:
# <locale> <charmap>
#
# Where <locale> is a locale located in /usr/share/i18n/locales/ and
# where <charmap> is a charmap located in /usr/share/i18n/charmaps/.
#
# All blank lines and lines starting with # are ignored.
#
# For the default list of supported combinations, see the file:
# /usr/share/i18n/SUPPORTED
#
# Whenever glibc is emerged, the locales listed here will be automatically
# rebuilt for you.  After updating this file, you can simply run `locale-gen`
# yourself instead of re-emerging glibc.

#en_US ISO-8859-1
#en_US.UTF-8 UTF-8
#ja_JP.EUC-JP EUC-JP
#ja_JP.UTF-8 UTF-8
#ja_JP EUC-JP
#en_HK ISO-8859-1
#en_PH ISO-8859-1
#de_DE ISO-8859-1
#de_DE@euro ISO-8859-15
#es_MX ISO-8859-1
#fa_IR UTF-8
#fr_FR ISO-8859-1
#fr_FR@euro ISO-8859-15
#it_IT ISO-8859-1
```

Uncomment the first two in the list by removing the # before them (the ones starting with ***en_US***):

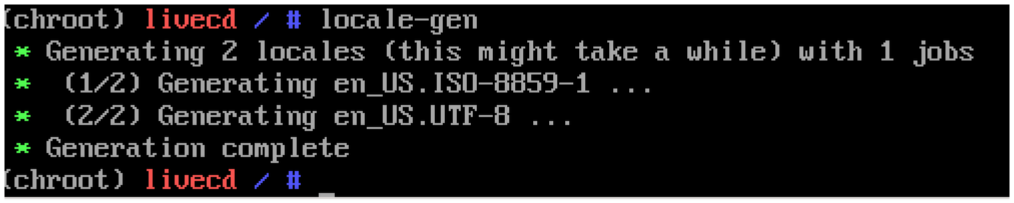


```
en_US ISO-8859-1
en_US.UTF-8 UTF-8
#ja_JP.EUC-JP EUC-JP
#ja_JP.UTF-8 UTF-8
#ja_JP EUC-JP
#en_HK ISO-8859-1
#en_PH ISO-8859-1
#de_DE ISO-8859-1
#de_DE@euro ISO-8859-15
#es_MX ISO-8859-1
#fa_IR UTF-8
#fr_FR ISO-8859-1
#fr_FR@euro ISO-8859-15
#it_IT ISO-8859-1
```

To save, press **CTRL + O** then **Enter**, and then exit by pressing **CTRL + X**. You'll be returned to the command prompt.

Now we need to generate the new locales.

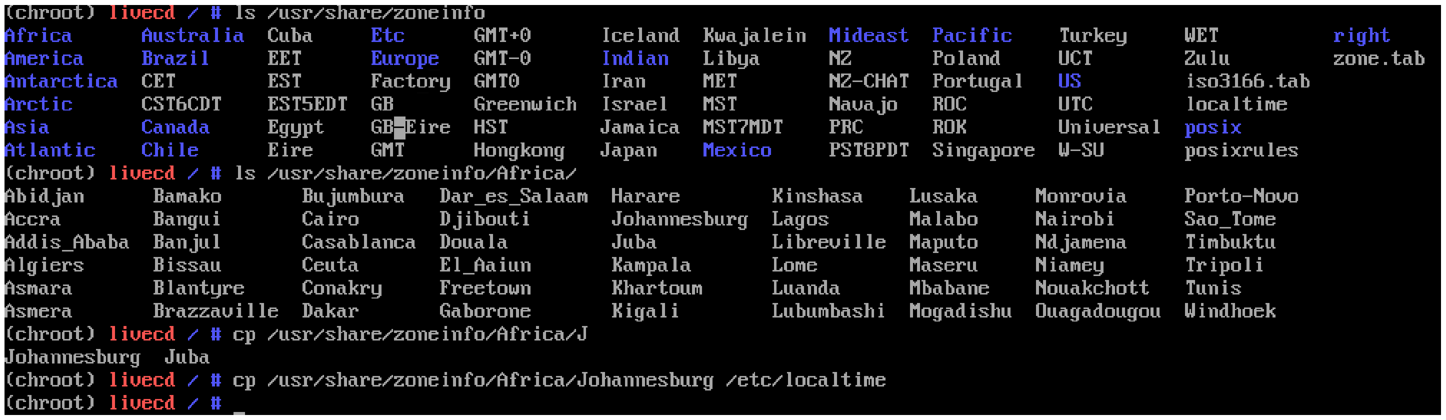
⟶ Type **locale-gen** and press **Enter**:



```
(chroot) livecd / # locale-gen
 * Generating 2 locales (this might take a while) with 1 jobs
 *  (1/2) Generating en_US.ISO-8859-1 ...
 *  (2/2) Generating en_US.UTF-8 ...
 * Generation complete
(chroot) livecd / # _
```

Now we need to select your timezone so that our system knows where it is located. Look for your timezone in **/usr/share/zoneinfo**, then copy it to **/etc/localtime**. Please avoid the **/usr/share/zoneinfo/Etc/GMT*** timezones as their names do not indicate the expected zones. For instance, GMT-8 is in fact GMT+8.

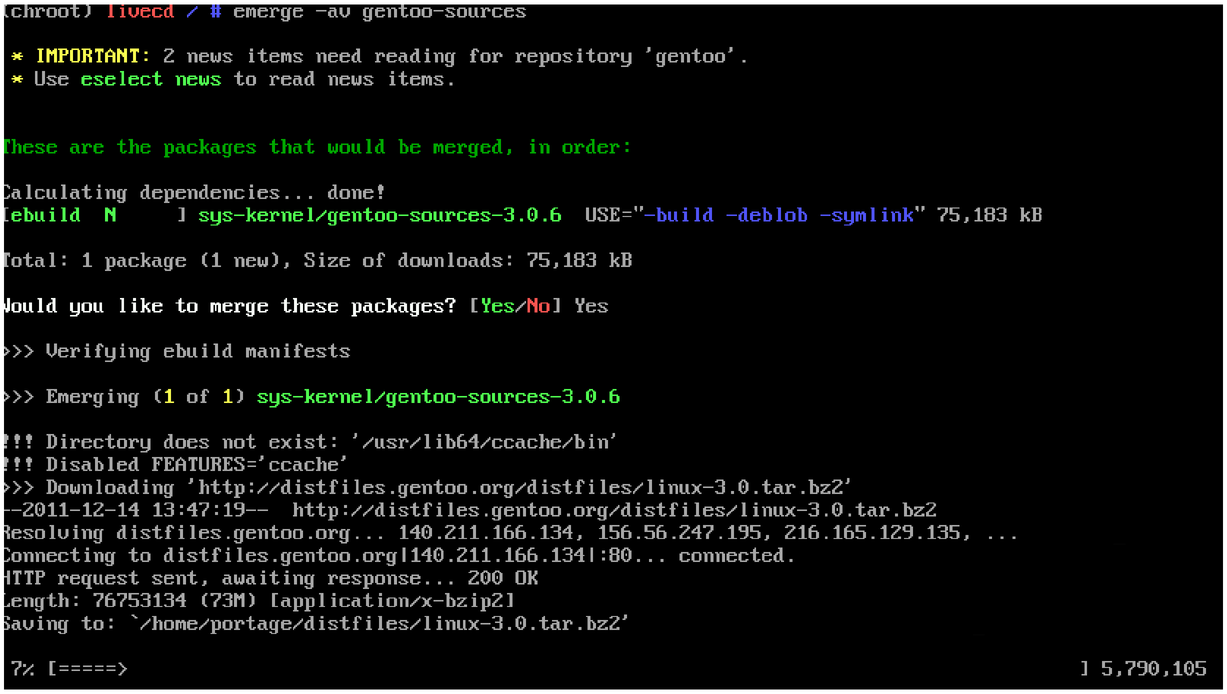
⟶ Type **cp /usr/share/zoneinfo/Africa/Johannesburg /etc/localtime** and press Enter.



```
(chroot) livecd / # ls /usr/share/zoneinfo
Africa      Australia  Cuba       Etc      GMT+0      Iceland  Kwajalein  Mideast  Pacific    Turkey     WET          right
America     Brazil     EET        Europe   GMT-0      Indian   Libya      NZ       Poland     UCT        Zulu         zone.tab
Antarctica  CET        EST        Factory  GMT0       Iran     MET        NZ-CHAT  Portugal   US         iso3166.tab
Arctic      CST6CDT    EST5EDT    GB       Greenwich  Israel   MST        Navajo   ROC        UTC        localtime
Asia        Canada     Egypt      GB-Eire  HST        Jamaica  MST7MDT    PRC      ROK        Universal  posix
Atlantic    Chile      Eire       GMT      Hongkong   Japan    Mexico     PST8PDT  Singapore  W-SU       posixrules
(chroot) livecd / # ls /usr/share/zoneinfo/Africa/
Abidjan      Bamako       Bujumbura    Dar_es_Salaam  Harare        Kinshasa     Lusaka     Monrovia     Porto-Novo
Accra        Bangui       Cairo        Djibouti       Johannesburg  Lagos        Malabo     Nairobi      Sao_Tome
Addis_Ababa  Banjul       Casablanca   Douala         Juba          Libreville   Maputo     Ndjamena     Timbuktu
Algiers      Bissau       Ceuta        El_Aaiun       Kampala       Lome         Maseru     Niamey       Tripoli
Asmara       Blantyre     Conakry      Freetown       Khartoum      Luanda       Mbabane    Nouakchott   Tunis
Asmera       Brazzaville  Dakar        Gaborone       Kigali        Lubumbashi   Mogadishu  Ouagadougou  Windhoek
(chroot) livecd / # cp /usr/share/zoneinfo/Africa/J
Johannesburg  Juba
(chroot) livecd / # cp /usr/share/zoneinfo/Africa/Johannesburg /etc/localtime
(chroot) livecd / # _
```

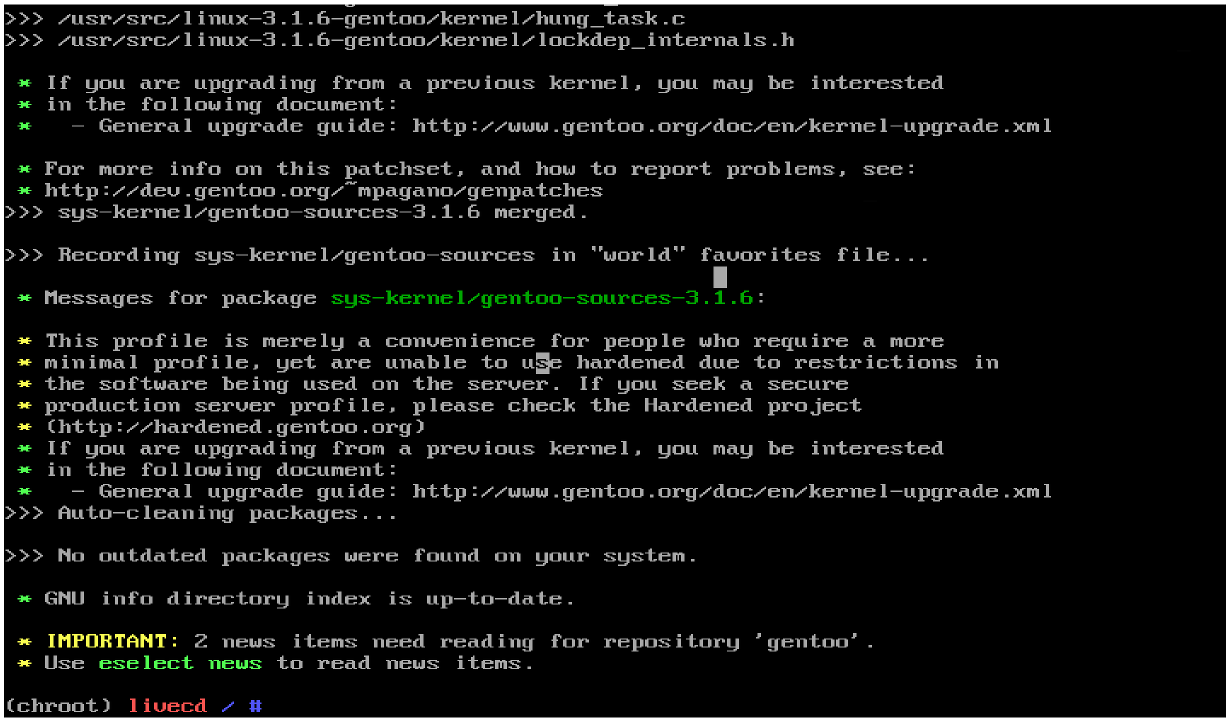
The core around which all distributions are built is the Linux kernel. It is the layer between the user programs and your system hardware. Gentoo provides its users several possible kernel sources. A full listing with description is available at the Kernel Guide. For AMD64-based systems we have **gentoo-sources** (kernel source patched for extra features). Install your kernel source.

⟶ Type **emerge -av gentoo-sources** and press Enter:



```
(chroot) livecd / # emerge -av gentoo-sources

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] sys-kernel/gentoo-sources-3.0.6  USE="-build -deblob -symlink" 75,183 kB

Total: 1 package (1 new), Size of downloads: 75,183 kB

Would you like to merge these packages? [Yes/No] Yes

>>> Verifying ebuild manifests

>>> Emerging (1 of 1) sys-kernel/gentoo-sources-3.0.6

!!! Directory does not exist: '/usr/lib64/ccache/bin'
!!! Disabled FEATURES='ccache'
>>> Downloading 'http://distfiles.gentoo.org/distfiles/linux-3.0.tar.bz2'
--2011-12-14 13:47:19--  http://distfiles.gentoo.org/distfiles/linux-3.0.tar.bz2
Resolving distfiles.gentoo.org... 140.211.166.134, 156.56.247.195, 216.165.129.135, ...
Connecting to distfiles.gentoo.org|140.211.166.134|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 76753134 (73M) [application/x-bzip2]
Saving to: `/home/portage/distfiles/linux-3.0.tar.bz2'

 7% [=====>                                                                   ] 5,790,105
```

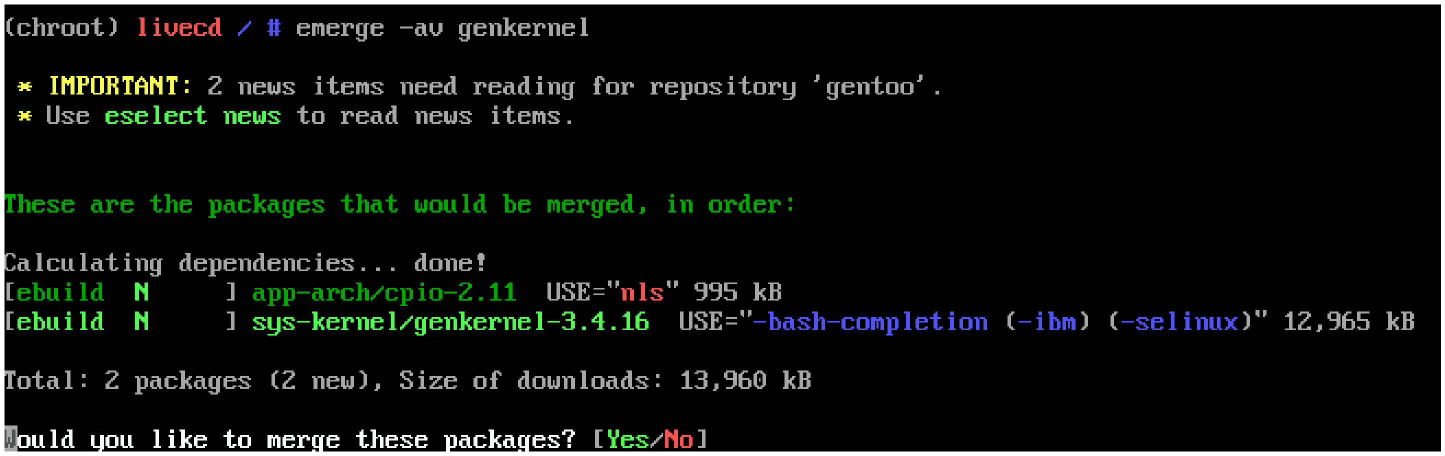
Confirm that there are no errors when done, you should see the following:



```
>>> /usr/src/linux-3.1.6-gentoo/kernel/hung_task.c
>>> /usr/src/linux-3.1.6-gentoo/kernel/lockdep_internals.h

 * If you are upgrading from a previous kernel, you may be interested
 * in the following document:
 *   - General upgrade guide: http://www.gentoo.org/doc/en/kernel-upgrade.xml

 * For more info on this patchset, and how to report problems, see:
 * http://dev.gentoo.org/~mpagano/genpatches
>>> sys-kernel/gentoo-sources-3.1.6 merged.

>>> Recording sys-kernel/gentoo-sources in "world" favorites file...

 * Messages for package sys-kernel/gentoo-sources-3.1.6:

 * This profile is merely a convenience for people who require a more
 * minimal profile, yet are unable to use hardened due to restrictions in
 * the software being used on the server. If you seek a secure
 * production server profile, please check the Hardened project
 * (http://hardened.gentoo.org)
 * If you are upgrading from a previous kernel, you may be interested
 * in the following document:
 *   - General upgrade guide: http://www.gentoo.org/doc/en/kernel-upgrade.xml
>>> Auto-cleaning packages...

>>> No outdated packages were found on your system.

 * GNU info directory index is up-to-date.

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.

(chroot) livecd / #
```

Now we need to configure & Install the kernel. For simplicity's sake – we will be using genkernel to do this. Install genkernel.

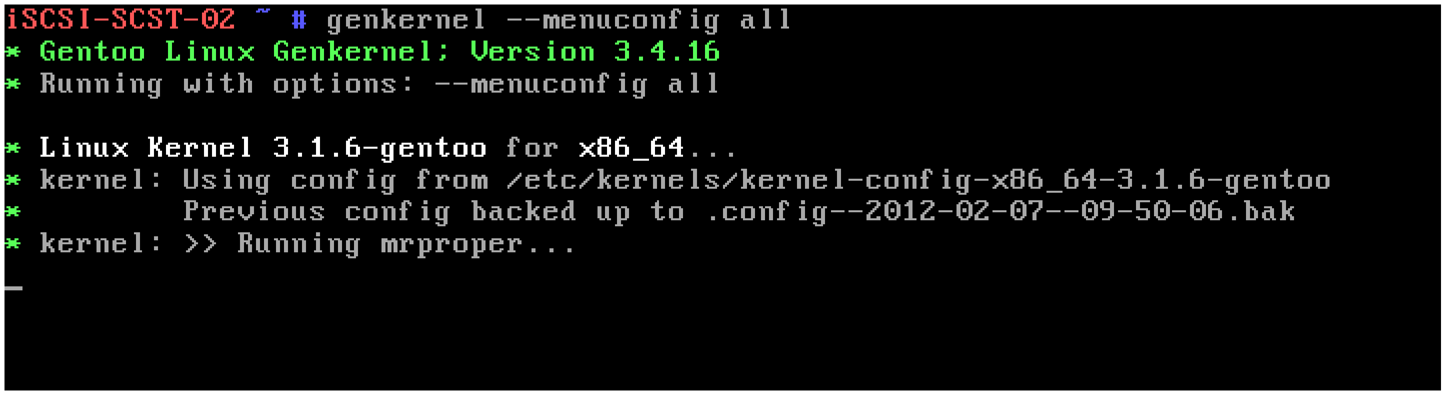
⟶ Type **emerge -av genkernel** and press **Enter**:



```
(chroot) livecd / # emerge -av genkernel

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] app-arch/cpio-2.11  USE="nls" 995 kB
[ebuild  N     ] sys-kernel/genkernel-3.4.16  USE="-bash-completion (-ibm) (-selinux)" 12,965 kB

Total: 2 packages (2 new), Size of downloads: 13,960 kB

Would you like to merge these packages? [Yes/No]
```

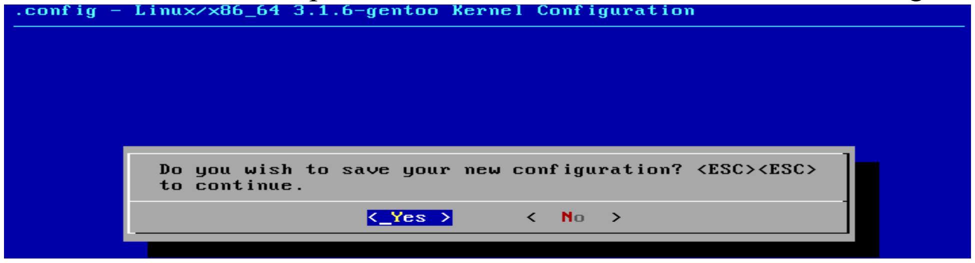
Now that Genkernel's Installed, we can compile the kernel. We also need to add the bonding module to the kernel, since we will need it later.

⟶ Type **genkernel --menuconfig all** and pressing **Enter**:



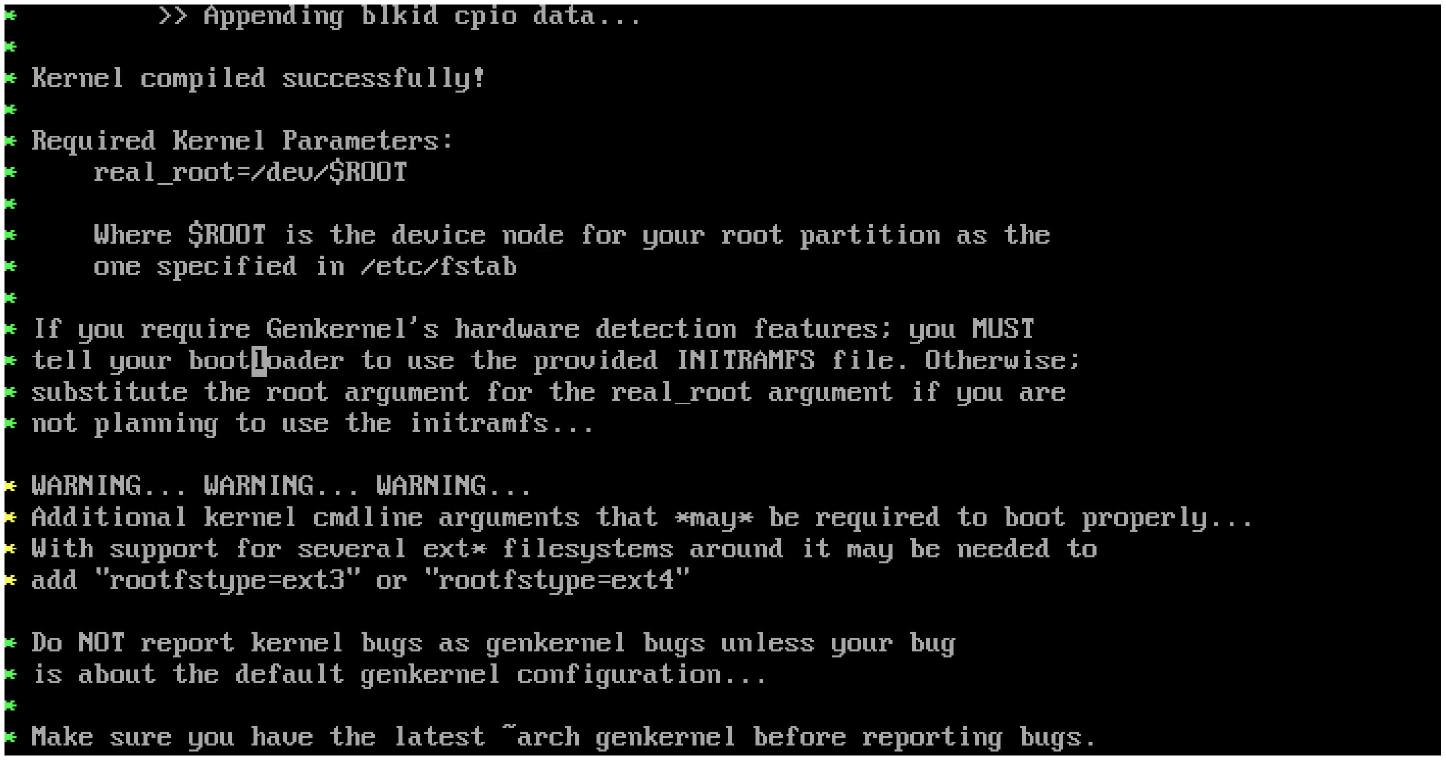
```
iSCSI-SCST-02 ~ # genkernel --menuconfig all
* Gentoo Linux Genkernel; Version 3.4.16
* Running with options: --menuconfig all

* Linux Kernel 3.1.6-gentoo for x86_64...
* kernel: Using config from /etc/kernels/kernel-config-x86_64-3.1.6-gentoo
*         Previous config backed up to .config--2012-02-07--09-50-06.bak
* kernel: >> Running mrproper...
_
```

On the menu that comes up, navigate to **Device Driver => Network device support**, and select **Bonding driver support** by highlighting it and pressing **M**. Now press ESC 4 times to return to main menu. Next we need to enable GPT Partition support, navigate to **File systems => Partition Types** and make sure that **EFI GUID Partition Support** has a * next to it. Afterwards, press ESC 6 times and select Yes to save changes.



```
.config - Linux/x86_64 3.1.6-gentoo Kernel Configuration

    Do you wish to save your new configuration? <ESC><ESC>
    to continue.

                 < Yes >          <  No  >
```

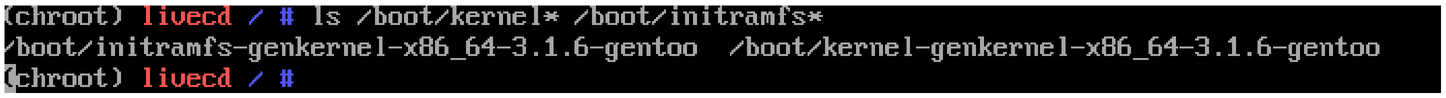
If finished successfully, you will see the following:



```
*         >> Appending blkid cpio data...
*
* Kernel compiled successfully!
*
* Required Kernel Parameters:
*     real_root=/dev/$ROOT
*
*     Where $ROOT is the device node for your root partition as the
*     one specified in /etc/fstab
*
* If you require Genkernel's hardware detection features; you MUST
* tell your bootloader to use the provided INITRAMFS file. Otherwise;
* substitute the root argument for the real_root argument if you are
* not planning to use the initramfs...

* WARNING... WARNING... WARNING...
* Additional kernel cmdline arguments that *may* be required to boot properly...
* With support for several ext* filesystems around it may be needed to
* add "rootfstype=ext3" or "rootfstype=ext4"

* Do NOT report kernel bugs as genkernel bugs unless your bug
* is about the default genkernel configuration...
*
* Make sure you have the latest ~arch genkernel before reporting bugs.
```

Once genkernel completes, a kernel, full set of modules and initial ram disk (initramfs) will be created. We will use the kernel and initrd when configuring a boot loader later in this document. Write down the names of the kernel and initrd as you will need it when writing the bootloader configuration file.

⟶ Type **ls /boot/kernel* /boot/initramfs*** and press Enter.



```
(chroot) livecd / # ls /boot/kernel* /boot/initramfs*
/boot/initramfs-genkernel-x86_64-3.1.6-gentoo  /boot/kernel-genkernel-x86_64-3.1.6-gentoo
(chroot) livecd / #
```

2.2.6.5 Configuring the Filesystem.

Under Linux, all partitions used by the system must be listed in /etc/fstab. This file contains the mount points of those partitions (where they are seen in the file system structure), how they should be mounted and with what special options (automatically or not, whether users can mount them or not, etc.)

/etc/fstab uses a special syntax. Every line consists of six fields, separated by whitespace (space(s), tabs or a mixture). Each field has its own meaning:

The **first field** shows the partition described (the path to the device file)

The **second field** shows the mount point at which the partition should be mounted

The **third field** shows the filesystem used by the partition

The **fourth field** shows the mount options used by mount when it wants to mount the partition. As every filesystem has its own mount options, you are encouraged to read the mount man page (man mount) for a full listing. Multiple mount options are comma-separated.

The **fifth field** is used by dump to determine if the partition needs to be dumped or not. You can generally leave this as 0 (zero).

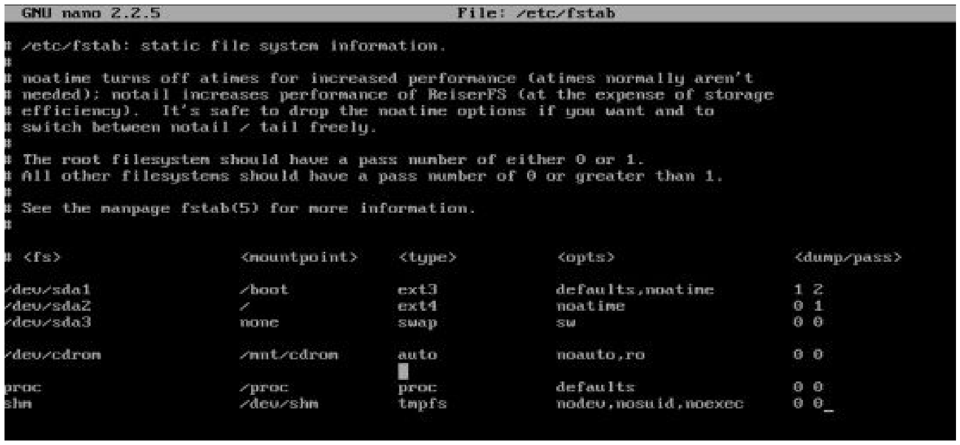
The **sixth field** is used by fsck to determine the order in which filesystems should be checked if the system wasn't shut down properly. The root filesystem should have 1 while the rest should have 2 (or 0 if a filesystem check isn't necessary).

Important: The default /etc/fstab file provided by Gentoo is not a valid fstab file. You have to create your own /etc/fstab.

Open /etc/fstab with nano, and replace the contents with the following:

→ Type **nano /etc/fstab** and press Enter.

```
/dev/sda1   /boot    ext3     defaults,noatime     1 2
/dev/sda2   /        ext4     noatime              0 1
/dev/sda3   none     swap     sw                   0 0

/dev/cdrom  /mnt/cdrom   auto    noauto,user          0 0
proc        /proc        proc    defaults             0 0
shm         /dev/shm     tmpfs   nodev,nosuid,noexec  0 0
```

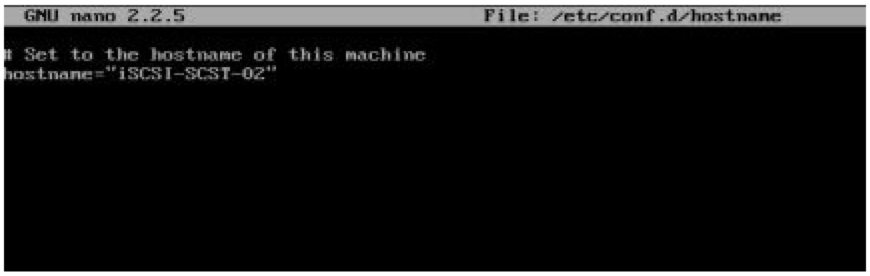


```
  GNU nano 2.2.5                        File: /etc/fstab

# /etc/fstab: static file system information.
#
# noatime turns off atimes for increased performance (atimes normally aren't
# needed); notail increases performance of ReiserFS (at the expense of storage
# efficiency).  It's safe to drop the noatime options if you want and to
# switch between notail / tail freely.
#
# The root filesystem should have a pass number of either 0 or 1.
# All other filesystems should have a pass number of 0 or greater than 1.
#
# See the manpage fstab(5) for more information.
#

# <fs>                  <mountpoint>    <type>          <opts>                  <dump/pass>

/dev/sda1               /boot           ext3            defaults,noatime        1 2
/dev/sda2               /               ext4            noatime                 0 1
/dev/sda3               none            swap            sw                      0 0

/dev/cdrom              /mnt/cdrom      auto            noauto,ro               0 0

proc                    /proc           proc            defaults                0 0
shm                     /dev/shm        tmpfs           nodev,nosuid,noexec     0 0
```

2.2.6.6 Networking & Hostname

Next, we need to set the hostname of our machine, to do this we need to edit the file ***/etc/conf.d/hostname.***

→ Type **nano /etc/conf.d/hostname** and press Enter.

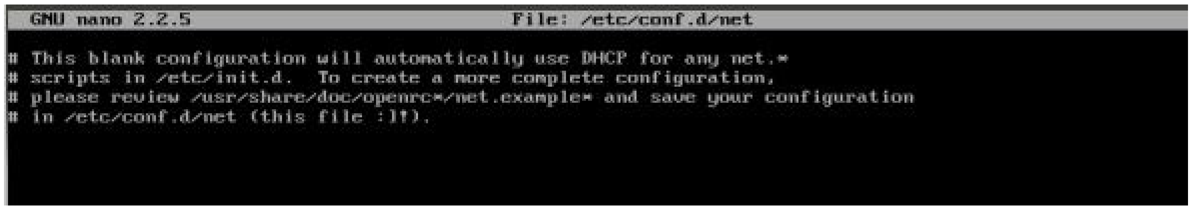


```
  GNU nano 2.2.5                    File: /etc/conf.d/hostname

# Set to the hostname of this machine
hostname="iSCSI-SCST-02"
```

Change the name inside the quotes to what you want to name the machine.

Next we need to configure our network settings.

→ Type **nano -w /etc/conf.d/net** and press Enter.



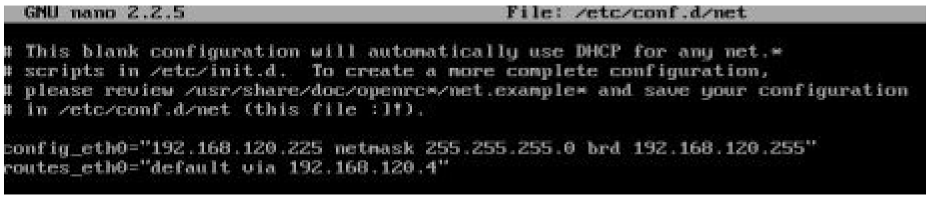
```
  GNU nano 2.2.5                    File: /etc/conf.d/net

# This blank configuration will automatically use DHCP for any net.*
# scripts in /etc/init.d.  To create a more complete configuration,
# please review /usr/share/doc/openrc*/net.example* and save your configuration
# in /etc/conf.d/net (this file :]!).
```

To enter your own IP address, netmask and gateway, you need to set both ***config_eth0*** and ***routes_eth0***:

→ Type **config_eth0="*IP_Address* netmask *Subnet* brd *broadcast"***

→ Type **routes_eth0="default via *router_ip*"**

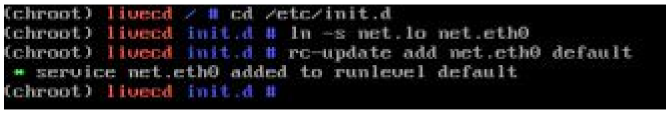
NB: *Replace the italic values with your network settings*



```
  GNU nano 2.2.5                    File: /etc/conf.d/net

# This blank configuration will automatically use DHCP for any net.*
# scripts in /etc/init.d.  To create a more complete configuration,
# please review /usr/share/doc/openrc*/net.example* and save your configuration
# in /etc/conf.d/net (this file :]!).

config_eth0="192.168.120.225 netmask 255.255.255.0 brd 192.168.120.255"
routes_eth0="default via 192.168.120.4"
```

To have your network interface ***eth0*** activated at boot, you need to add them to the default runlevel, execute the commands below:

→ Type **cd /etc/init.d/** and press Enter.
→ Type **ln -s net.lo net.eth0** and press Enter.
→ Type **rc-update add net.eth0 default** and press Enter.



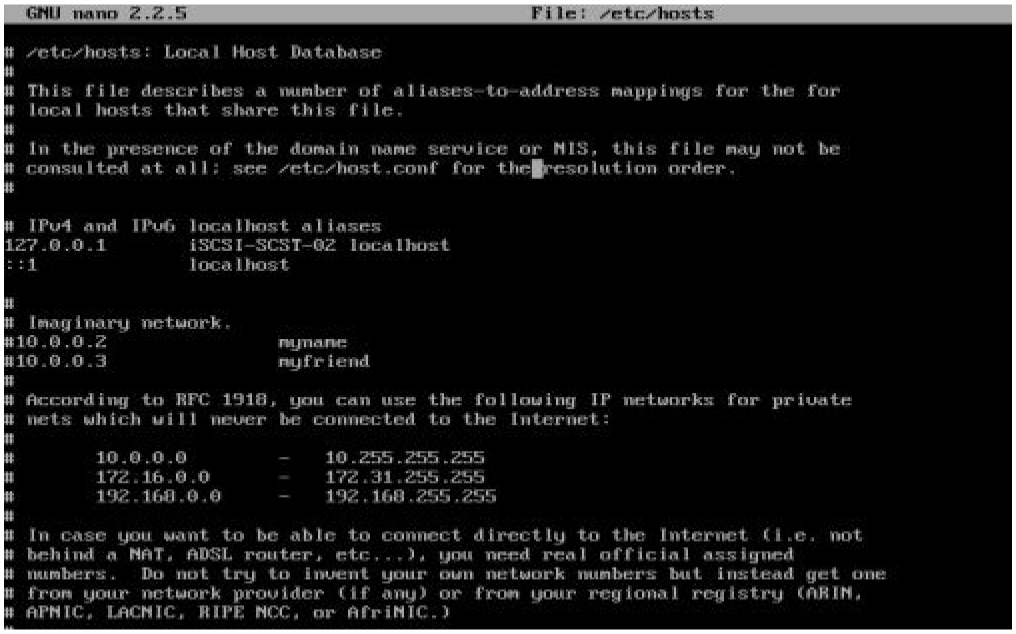
```
(chroot) livecd / # cd /etc/init.d
(chroot) livecd init.d # ln -s net.lo net.eth0
(chroot) livecd init.d # rc-update add net.eth0 default
 * service net.eth0 added to runlevel default
(chroot) livecd init.d #
```

We will configure the other interfaces later.

You now need to inform Linux about your network. This is defined in ***/etc/hosts*** and helps in resolving host names to IP addresses for hosts that aren't resolved by your nameserver. You need to define your system. You may also want to define other systems on your network if you don't want to set up your own internal DNS system.

Open the file using nano and fill in the appropriate values –

→ Type **nano /etc/hosts** and press Enter.

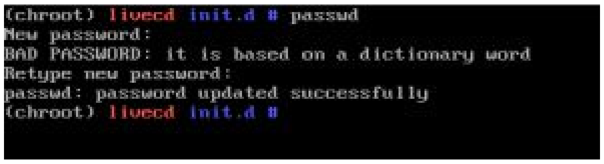


```
  GNU nano 2.2.5                          File: /etc/hosts

# /etc/hosts: Local Host Database
#
# This file describes a number of aliases-to-address mappings for the for
# local hosts that share this file.
#
# In the presence of the domain name service or NIS, this file may not be
# consulted at all; see /etc/host.conf for the resolution order.
#

# IPv4 and IPv6 localhost aliases
127.0.0.1       iSCSI-SCST-02 localhost
::1             localhost

#
# Imaginary network.
#10.0.0.2               myname
#10.0.0.3               myfriend
#
# According to RFC 1918, you can use the following IP networks for private
# nets which will never be connected to the Internet:
#
#       10.0.0.0        -   10.255.255.255
#       172.16.0.0      -   172.31.255.255
#       192.168.0.0     -   192.168.255.255
#
# In case you want to be able to connect directly to the Internet (i.e. not
# behind a NAT, ADSL router, etc...), you need real official assigned
# numbers.  Do not try to invent your own network numbers but instead get one
# from your network provider (if any) or from your regional registry (ARIN,
# APNIC, LACNIC, RIPE NCC, or AfriNIC.)
```

2.2.6.7 System Information

Now we need to set the root password for our new installation.

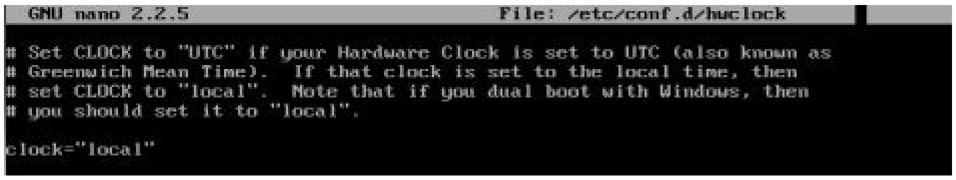
→ Type **passwd** and press Enter.
→ At the prompt, type your new password, and press ENTER.
→ Confirm your pasword, and press ENTER again.



```
(chroot) livecd init.d # passwd
New password:
BAD PASSWORD: it is based on a dictionary word
Retype new password:
passwd: password updated successfully
(chroot) livecd init.d #
```

Gentoo uses ***/etc/conf.d/hwclock*** to set clock options. Edit it according to your needs. If your hardware clock is not using UTC, you need to add ***clock="local"*** to the file. Otherwise you will notice some clock skew.

→ Type **nano /etc/conf.d/hwclock** and press Enter.(***Edit it according to your needs.***)



```
 GNU nano 2.2.5                    File: /etc/conf.d/hwclock

# Set CLOCK to "UTC" if your Hardware Clock is set to UTC (also known as
# Greenwich Mean Time).  If that clock is set to the local time, then
# set CLOCK to "local".  Note that if you dual boot with Windows, then
# you should set it to "local".

clock="local"
```

2.2.6.8 Installing Necessary System Tools.

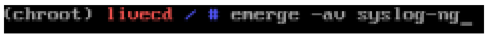
Some tools are missing from the stage3 archive because several packages provide the same functionality. It is now up to you to choose which ones you want to install.

The first tool you need to decide on has to provide logging facilities for your system. Unix and Linux have an excellent history of logging capabilities -- if you want you can log everything that happens on your system in logfiles. This happens through the system logger.

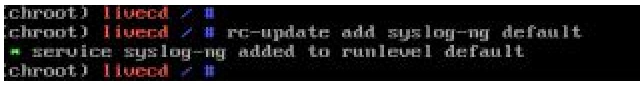
To install the system logger of your choice, emerge it and have it added to the default runlevel using rc-update. The following example installs syslog-ng. Of course substitute with your system logger:

⟶ Type **emerge -v syslog-ng** and press Enter.

⟶ Type **rc-update add syslog-ng default** and press Enter.



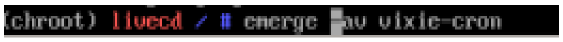
```
(chroot) livecd / # emerge -av syslog-ng_
```



```
(chroot) livecd / #
(chroot) livecd / # rc-update add syslog-ng default
 * service syslog-ng added to runlevel default
(chroot) livecd / #
```

Next is the cron daemon. Although it is optional and not required for your system, it is wise to install one. But what is a cron daemon? A cron daemon executes scheduled commands. It is very handy if you need to execute some command regularly (for instance daily, weekly or monthly).

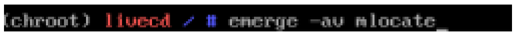
⟶ Type **emerge -v vixie-cron** and press Enter.



```
(chroot) livecd / # emerge -av vixie-cron
```

If you want to index your system's files so you are able to quickly locate them using the **locate** tool, you need to install ***sys-apps/mlocate.***

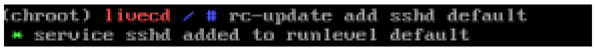
⟶ Type **emerge -v mlocate** and press Enter.



```
(chroot) livecd / # emerge -av mlocate_
```

If you need to access your system remotely after installation, don't forget to add ***sshd*** to the default runlevel:

⟶ Type **rc-update add sshd default** and press Enter.

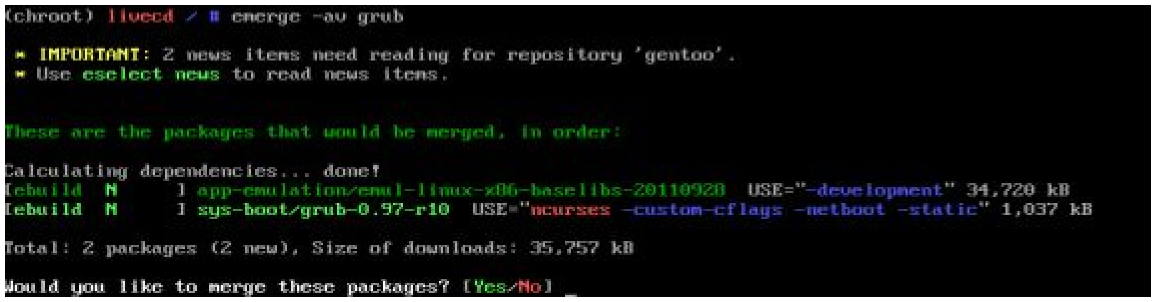


```
(chroot) livecd / # rc-update add sshd default
 * service sshd added to runlevel default
```

2.2.6.9 Configuring The Bootloader

Now that your kernel is configured and compiled and the necessary system configuration files are filled in correctly, it is time to install a program that will fire up your kernel when you start the system. Such a program is called a bootloader.

→ To install GRUB, type ***emerge -av grub***:



```
(chroot) livecd / # emerge -av grub

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.

These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] app-emulation/emul-linux-x86-baselibs-20110928  USE="-development" 34,720 kB
[ebuild  N     ] sys-boot/grub-0.97-r10  USE="ncurses -custom-cflags -netboot -static" 1,037 kB

Total: 2 packages (2 new), Size of downloads: 35,757 kB

Would you like to merge these packages? [Yes/No] _
```

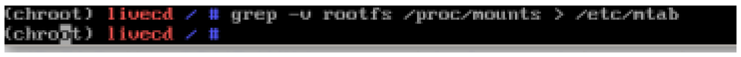
Next we need to configure Grub. Type ***nano -w /boot/grub/grub.conf***, and copy the following into it, substituting the bold names with the ones you wrote down when compiling your kernel.

→ Type **nano -w /boot/grub/grub.conf** and press Enter.

```
default 0
timeout 30
splashimage=(hd0,0)/boot/grub/splash.xpm.gz

title Gentoo Linux 2.6.34-r1
root (hd0,0)
kernel /boot/kernel-genkernel-amd64-2.6.34-gentoo-r1 real_root=/dev/sda2
initrd /boot/initramfs-genkernel-amd64-2.6.34-gentoo-r1
```

To install GRUB you will need to issue the grub-install command. However, grub-install won't work off-the-shelf since we are inside a chrooted environment. We need to create /etc/mtab which lists all mounted filesystems. Fortunately, there is an easy way to accomplish this - just copy over /proc/mounts to /etc/mtab, excluding the rootfs line if you haven't created a separate boot partition. The following command will work in both cases:

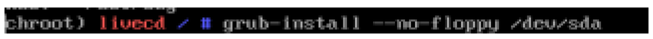
→ Type **grep -v rootfs /proc/mounts > /etc/mtab** and press Enter.



```
(chroot) livecd / # grep -v rootfs /proc/mounts > /etc/mtab
(chroot) livecd / #
```

Now we can "install" the bootloader:

→ Type **grub-install --no-floppy /dev/sda** and press Enter.



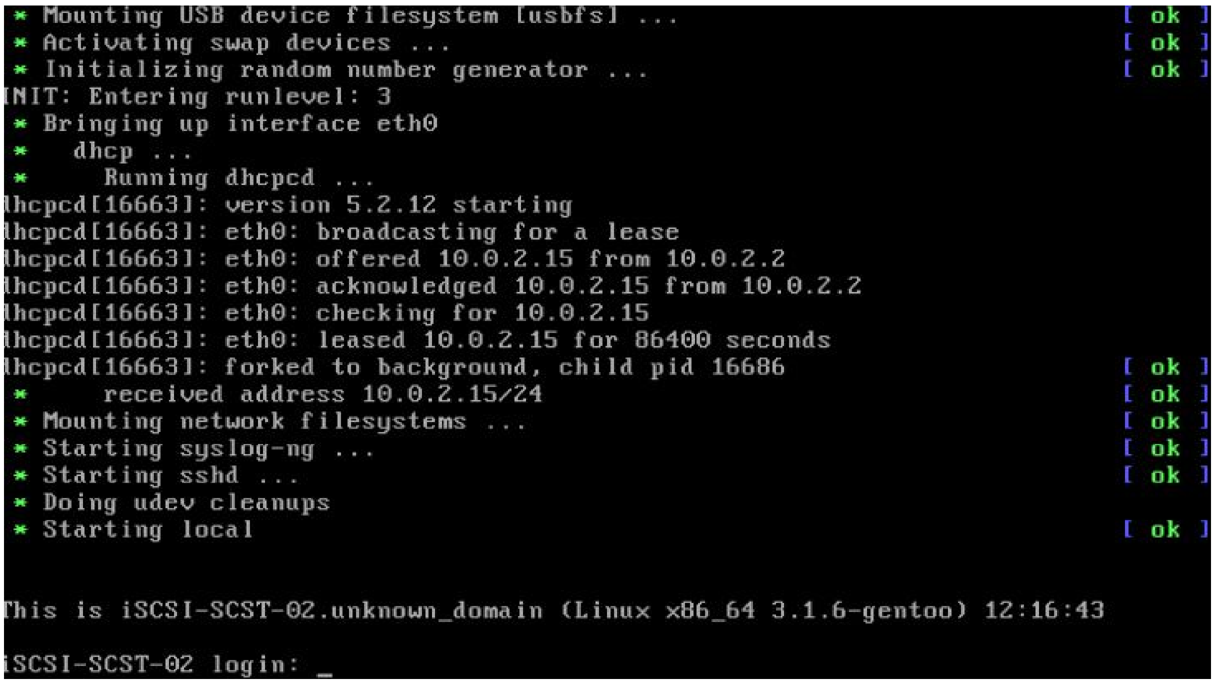
```
chroot) livecd / # grub-install --no-floppy /dev/sda
```

Exit the chrooted environment and unmount all mounted partitions. Then type in that one magical command you have been waiting for: reboot.

⟶ Type **exit** and press Enter.
⟶ Type **cd** and press Enter.
⟶ Type **umount -l /mnt/gentoo/dev{/shm,/pts,}** and press Enter.
⟶ Type **umount -l /mnt/gentoo{/boot,/proc,}** and press Enter.
⟶ Type **reboot** and press Enter.

```
# exit
# cd
# umount -l /mnt/gentoo/dev{/shm,/pts,}
# umount -l /mnt/gentoo{/boot,/proc,}
# reboot
```

And now you can pat yourself on the back, you have just installed Gentoo Linux!



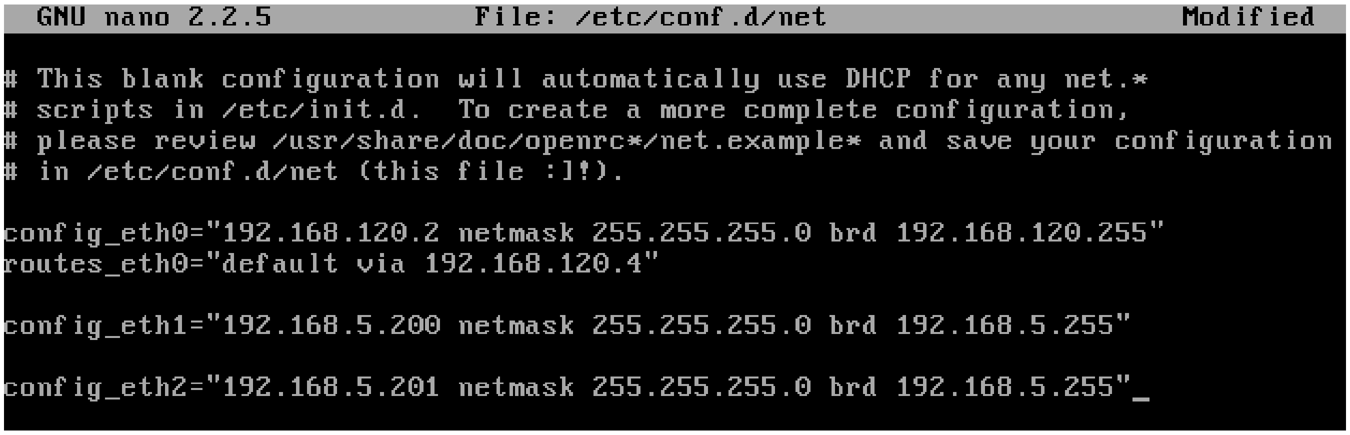
```
 * Mounting USB device filesystem [usbfs] ...                              [ ok ]
 * Activating swap devices ...                                             [ ok ]
 * Initializing random number generator ...                                [ ok ]
INIT: Entering runlevel: 3
 * Bringing up interface eth0
 *   dhcp ...
 *     Running dhcpcd ...
dhcpcd[16663]: version 5.2.12 starting
dhcpcd[16663]: eth0: broadcasting for a lease
dhcpcd[16663]: eth0: offered 10.0.2.15 from 10.0.2.2
dhcpcd[16663]: eth0: acknowledged 10.0.2.15 from 10.0.2.2
dhcpcd[16663]: eth0: checking for 10.0.2.15
dhcpcd[16663]: eth0: leased 10.0.2.15 for 86400 seconds
dhcpcd[16663]: forked to background, child pid 16686                       [ ok ]
 *     received address 10.0.2.15/24                                       [ ok ]
 * Mounting network filesystems ...                                        [ ok ]
 * Starting syslog-ng ...                                                  [ ok ]
 * Starting sshd ...                                                       [ ok ]
 * Doing udev cleanups
 * Starting local                                                          [ ok ]


This is iSCSI-SCST-02.unknown_domain (Linux x86_64 3.1.6-gentoo) 12:16:43

iSCSI-SCST-02 login: _
```

2.2.6.10 Finishing Touches

Now all that is left is to configure the RAID on the data disks & our other two network interfaces, namely: **eth1** & **eth2.**

We'll start off with the two network interfaces, we configure them in the file **/etc/conf.d/net**. We need to create the configuration settings for the interfaces, same as we did for eth0. The only difference is the name of the configuration parameter changes slightly.

config_eth0 becomes **config_eth1** for the eth1 interface, and **config_eth2** for the eth2 interface. Also the two interfaces ideally should be on a different subnet than your production network. See example below:

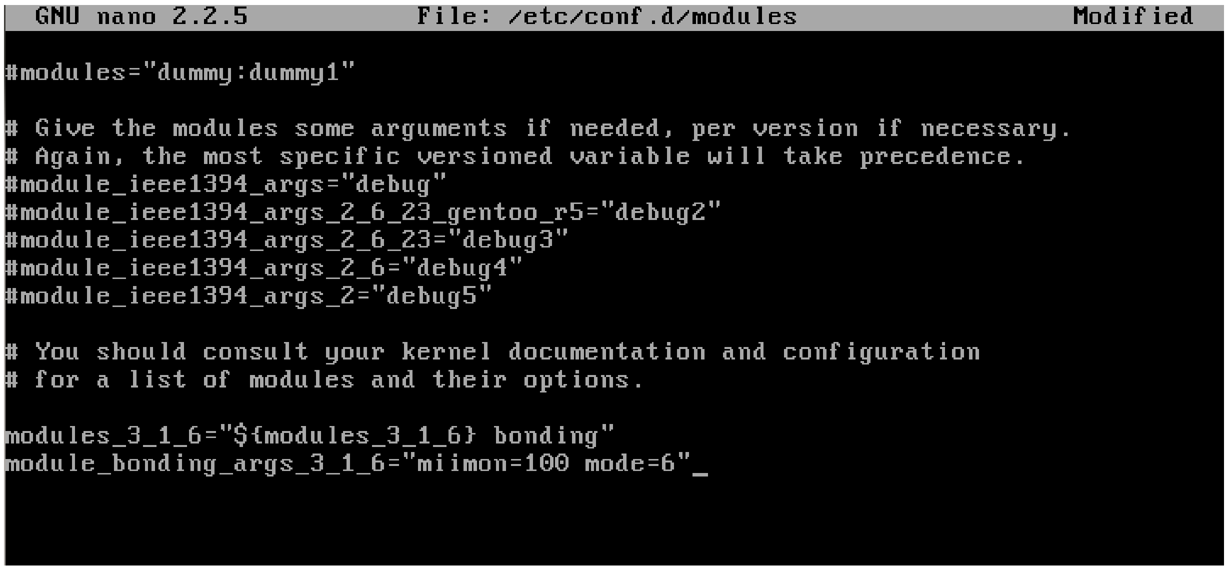


```
  GNU nano 2.2.5            File: /etc/conf.d/net                       Modified

# This blank configuration will automatically use DHCP for any net.*
# scripts in /etc/init.d.  To create a more complete configuration,
# please review /usr/share/doc/openrc*/net.example* and save your configuration
# in /etc/conf.d/net (this file :]!).

config_eth0="192.168.120.2 netmask 255.255.255.0 brd 192.168.120.255"
routes_eth0="default via 192.168.120.4"

config_eth1="192.168.5.200 netmask 255.255.255.0 brd 192.168.5.255"

config_eth2="192.168.5.201 netmask 255.255.255.0 brd 192.168.5.255"_
```

But we are going to set up the remaining interfaces using NIC bonding, linking the two network cards together so they are seen as 1 interface. This increases the bandwidth that will be available to our iSCSI clients.

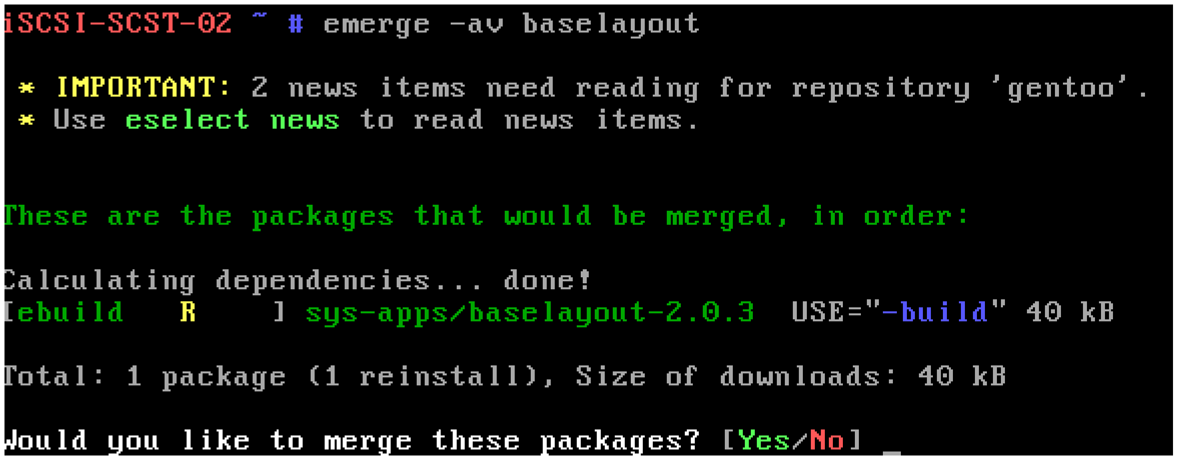
Firstly we need to tell Gentoo to load the bonding kernel modules on startup, to do this we need to edit **/etc/conf.d/modules**

⟶ Type **nano /etc/conf.d/modules** and press Enter. Edit the file as below, replacing the **_3_1_6** with your kernel version.


```
  GNU nano 2.2.5              File: /etc/conf.d/modules                    Modified

#modules="dummy:dummy1"

# Give the modules some arguments if needed, per version if necessary.
# Again, the most specific versioned variable will take precedence.
#module_ieee1394_args="debug"
#module_ieee1394_args_2_6_23_gentoo_r5="debug2"
#module_ieee1394_args_2_6_23="debug3"
#module_ieee1394_args_2_6="debug4"
#module_ieee1394_args_2="debug5"

# You should consult your kernel documentation and configuration
# for a list of modules and their options.

modules_3_1_6="${modules_3_1_6} bonding"
module_bonding_args_3_1_6="miimon=100 mode=6"_
```

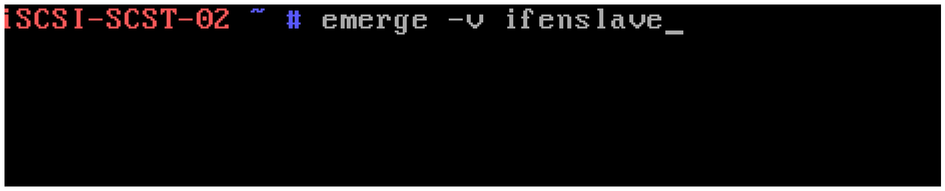
Next we need to install **baselayout.**

⟶ Type **emerge -v baselayout** and press Enter.


```
iSCSI-SCST-02 ~ # emerge -av baselayout

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild   R    ] sys-apps/baselayout-2.0.3  USE="-build" 40 kB

Total: 1 package (1 reinstall), Size of downloads: 40 kB

Would you like to merge these packages? [Yes/No] _
```

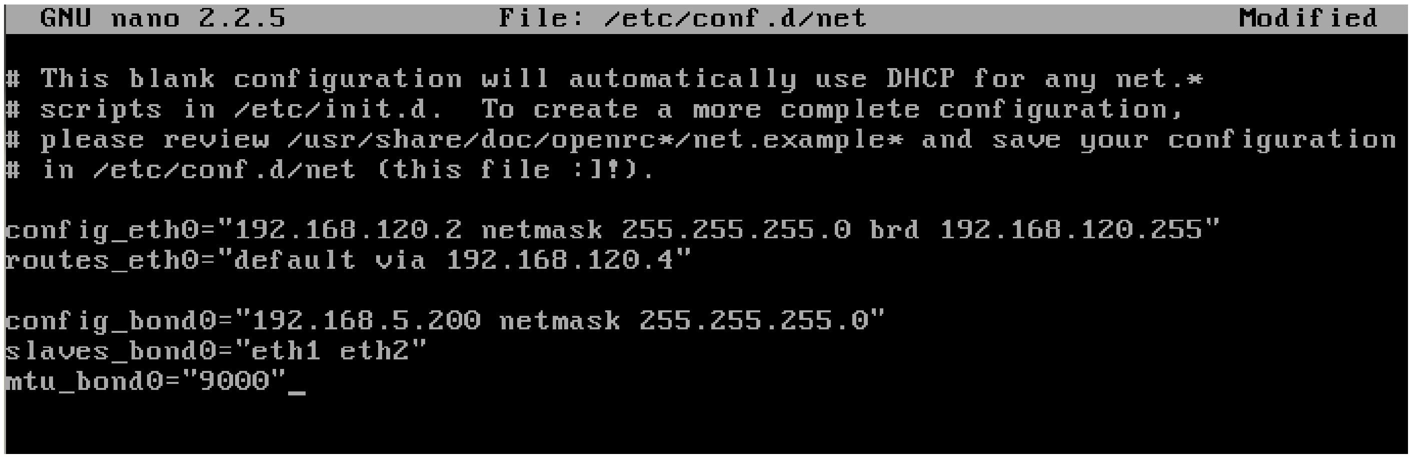
Next we need to install **ifenslave.**

⟶ Type **emerge -v ifenslave** and press Enter.


```
iSCSI-SCST-02 ~ # emerge -v ifenslave_
```

Next we need to set up the network configuration for the bond. We do this in **/net/etc/conf.d/net**.

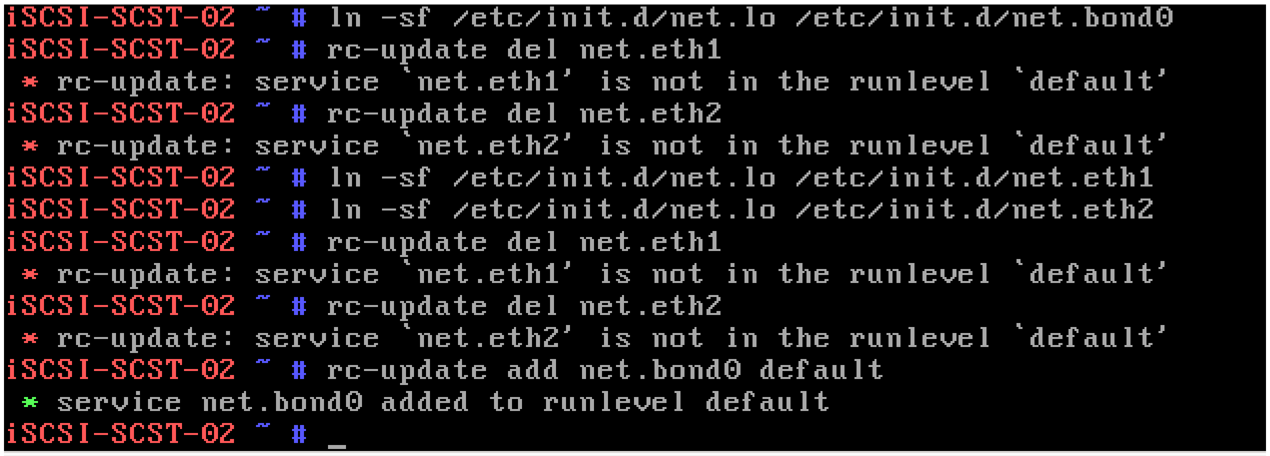
→ Type **nano /etc/conf.d/net** and press Enter. Edit the file accordingly, filling in the values exactly as they are in the screenshot (Your own IP/Network information obviously).



```
  GNU nano 2.2.5              File: /etc/conf.d/net                    Modified

# This blank configuration will automatically use DHCP for any net.*
# scripts in /etc/init.d.  To create a more complete configuration,
# please review /usr/share/doc/openrc*/net.example* and save your configuration
# in /etc/conf.d/net (this file :]!).

config_eth0="192.168.120.2 netmask 255.255.255.0 brd 192.168.120.255"
routes_eth0="default via 192.168.120.4"

config_bond0="192.168.5.200 netmask 255.255.255.0"
slaves_bond0="eth1 eth2"
mtu_bond0="9000"_
```

Next we need to add the bond to the startup procedure.

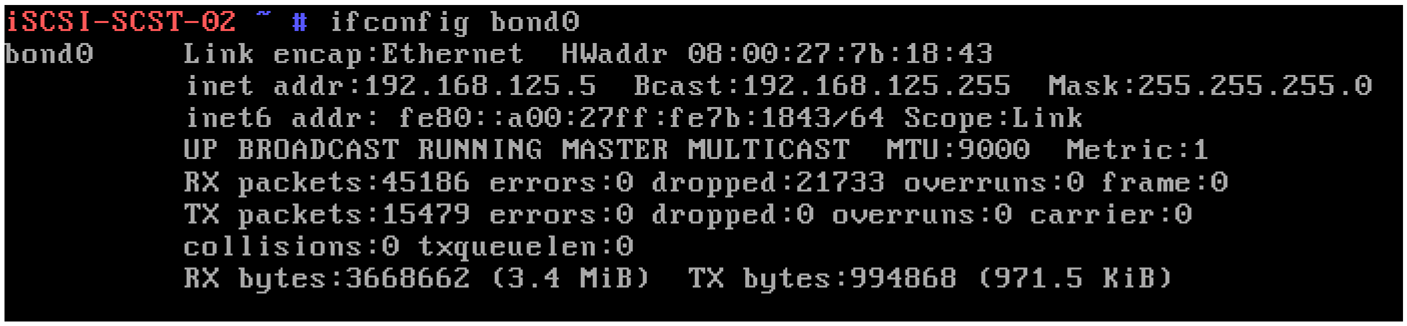
→ Type **ln -sf /etc/init.d/net.lo /etc/init.d/net.bond0** and press Enter.
→ Type **rc-update del net.eth1** and press Enter.
→ Type **rc-update del net.eth2** and press Enter.
→ Type **rc-update add net.bond0 default** and press Enter.



```
iSCSI-SCST-02 ~ # ln -sf /etc/init.d/net.lo /etc/init.d/net.bond0
iSCSI-SCST-02 ~ # rc-update del net.eth1
 * rc-update: service `net.eth1' is not in the runlevel `default'
iSCSI-SCST-02 ~ # rc-update del net.eth2
 * rc-update: service `net.eth2' is not in the runlevel `default'
iSCSI-SCST-02 ~ # ln -sf /etc/init.d/net.lo /etc/init.d/net.eth1
iSCSI-SCST-02 ~ # ln -sf /etc/init.d/net.lo /etc/init.d/net.eth2
iSCSI-SCST-02 ~ # rc-update del net.eth1
 * rc-update: service `net.eth1' is not in the runlevel `default'
iSCSI-SCST-02 ~ # rc-update del net.eth2
 * rc-update: service `net.eth2' is not in the runlevel `default'
iSCSI-SCST-02 ~ # rc-update add net.bond0 default
 * service net.bond0 added to runlevel default
iSCSI-SCST-02 ~ # _
```

Now reboot the machine. The bonded interface should come up and be assigned an IP address, lets check that.

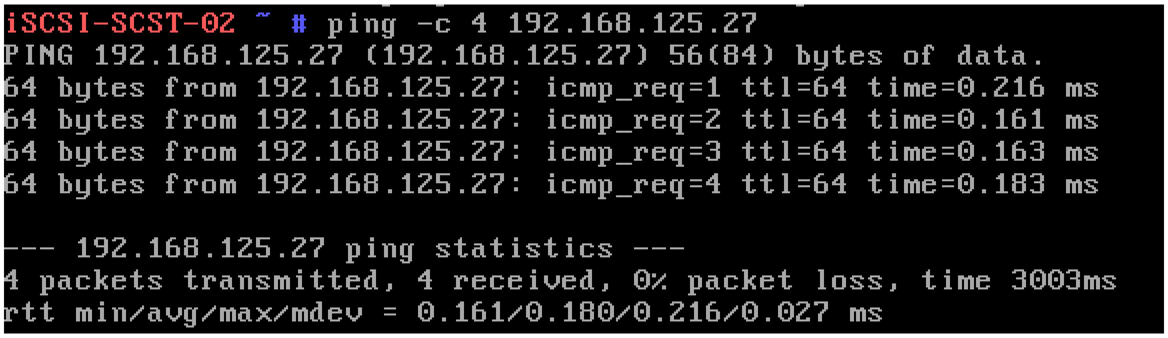
⟶ Type **ifconfig bond0** and press Enter.



```
iSCSI-SCST-02 ~ # ifconfig bond0
bond0     Link encap:Ethernet  HWaddr 08:00:27:7b:18:43
          inet addr:192.168.125.5  Bcast:192.168.125.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe7b:1843/64 Scope:Link
          UP BROADCAST RUNNING MASTER MULTICAST  MTU:9000  Metric:1
          RX packets:45186 errors:0 dropped:21733 overruns:0 frame:0
          TX packets:15479 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:3668662 (3.4 MiB)  TX bytes:994868 (971.5 KiB)
```

Next lets see if we can ping an ip address through the bond. If all is well you should see the output in the screenshot.

⟶ Type **ping -c 4 <ip on your network(same subnet as bond)>** and press Enter.

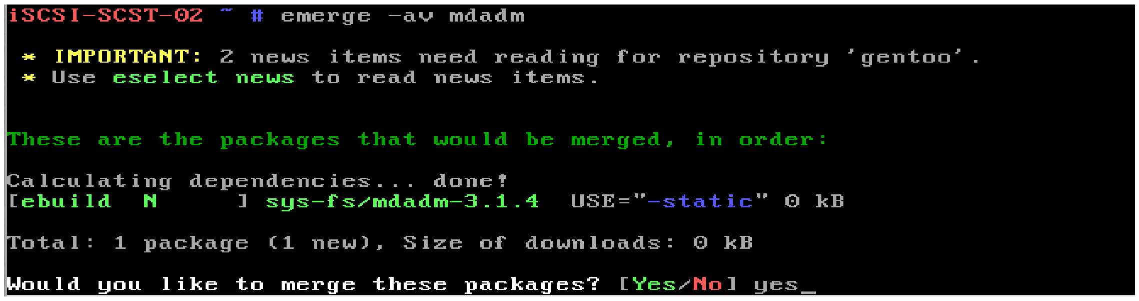


```
iSCSI-SCST-02 ~ # ping -c 4 192.168.125.27
PING 192.168.125.27 (192.168.125.27) 56(84) bytes of data.
64 bytes from 192.168.125.27: icmp_req=1 ttl=64 time=0.216 ms
64 bytes from 192.168.125.27: icmp_req=2 ttl=64 time=0.161 ms
64 bytes from 192.168.125.27: icmp_req=3 ttl=64 time=0.163 ms
64 bytes from 192.168.125.27: icmp_req=4 ttl=64 time=0.183 ms

--- 192.168.125.27 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3003ms
rtt min/avg/max/mdev = 0.161/0.180/0.216/0.027 ms
```

Now that our NIC bond has been set up successfully, we can start setting up our RAID Array. You could do this with hardware RAID, but I prefer using Linux SW RAID for the flexibility. SW RAID does add extra overhead, so it is not as fast as HW RAID, but seeing as we have a beast of a machine, this should not be an issue.

We will be using **mdadm** to set up the array, so let's install it.

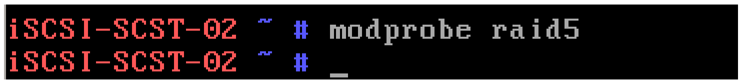
⟶ Type **emerge -v mdadm** and press Enter.



```
iSCSI-SCST-02 ~ # emerge -av mdadm

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] sys-fs/mdadm-3.1.4  USE="-static" 0 kB

Total: 1 package (1 new), Size of downloads: 0 kB

Would you like to merge these packages? [Yes/No] yes_
```

Then we need to load the appropriate kernel modules. There are a few, but we will be using RAID5.

⟶ Type **modprobe raid5** and press Enter.

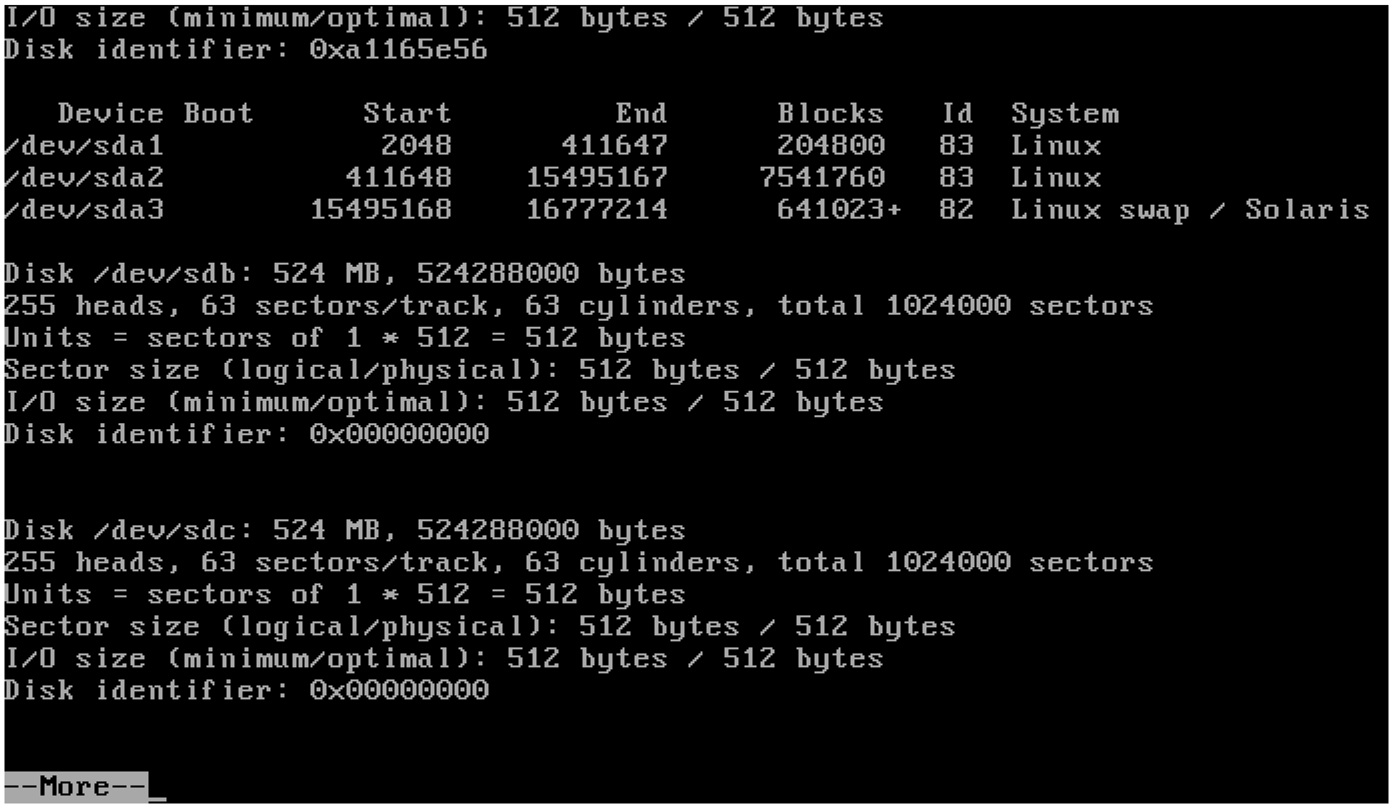


```
iSCSI-SCST-02 ~ # modprobe raid5
iSCSI-SCST-02 ~ # _
```

Next, we need to partition our disks, we will set up the first disk and then simply copy the partition table to the other disks.

First let's check what drives are available to us, we know that /dev/sda is our system disk, so we will be leaving that one out of this operation.

⟶ Type **fdisk -l | more** and press Enter. Scroll through the list by pressing Enter repeatedly, until you are returned to the command prompt.

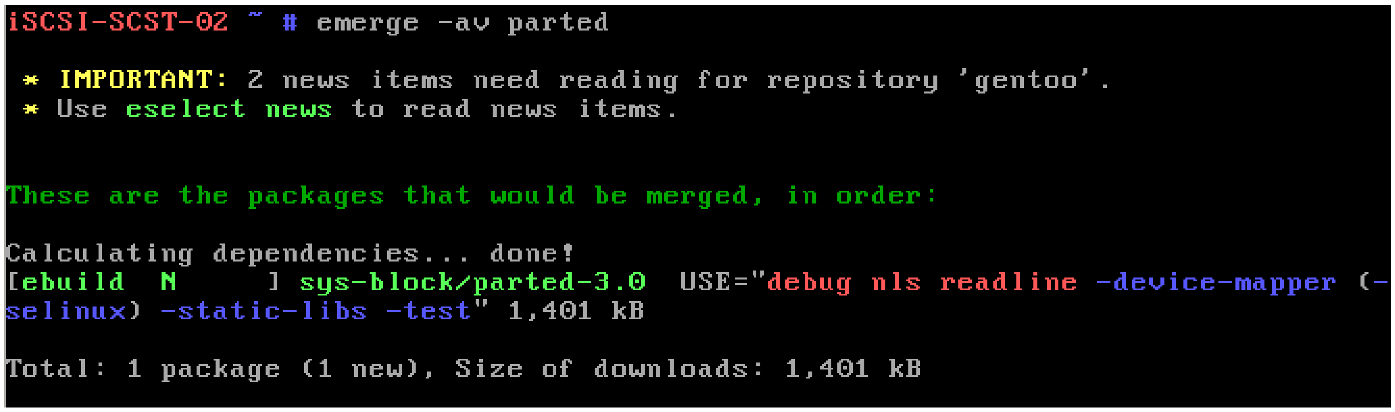


```
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xa1165e56

   Device Boot      Start         End      Blocks   Id  System
/dev/sda1            2048      411647      204800   83  Linux
/dev/sda2          411648    15495167     7541760   83  Linux
/dev/sda3        15495168    16777214      641023+  82  Linux swap / Solaris

Disk /dev/sdb: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000


Disk /dev/sdc: 524 MB, 524288000 bytes
255 heads, 63 sectors/track, 63 cylinders, total 1024000 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000


--More--_
```

I am going to assume that you will be using 2TB or larger disks for your data drives, therefore we will be partitioning the disks using a GPT partition table via **parted**.

Of course, we will have to install **parted** first.

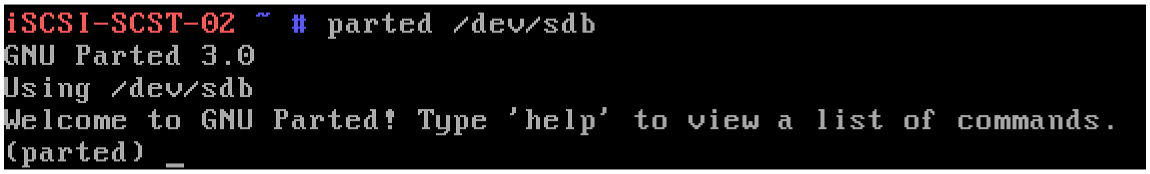
⟶ Type **emerge -v parted** and press Enter.



```
iSCSI-SCST-02 ~ # emerge -av parted

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] sys-block/parted-3.0  USE="debug nls readline -device-mapper (-
selinux) -static-libs -test" 1,401 kB

Total: 1 package (1 new), Size of downloads: 1,401 kB
```

Okay, now we need to partition our first disk, namely **/dev/sdb**.

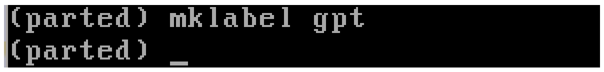
⟶ Type **parted /dev/sdb** and press Enter.



```
iSCSI-SCST-02 ~ # parted /dev/sdb
GNU Parted 3.0
Using /dev/sdb
Welcome to GNU Parted! Type 'help' to view a list of commands.
(parted) _
```

Next, lets create our GPT partition table.

⟶ Type **mklabel gpt** and press Enter.

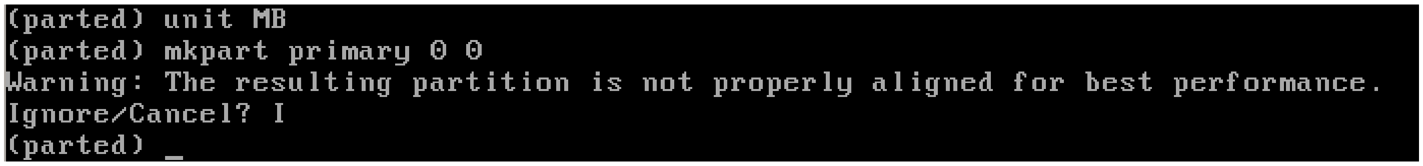


```
(parted) mklabel gpt
(parted) _
```

Next we need to create our partition, to simplify this I will change the default unit size parted uses to TB(Terrabyte), and create a 3 TB partition (*If your disk size differs, adjust accoringly, you'll notice that my commands in the screenshots dont reflect the unit type change or 3.0 TB disk size, I am using a VM for the purpose of creating this document and therefore opted to use 500MB disks.*)

⟶ Type **unit TB** and press Enter

⟶ Type **mkpart primary 0 0** and press Enter.

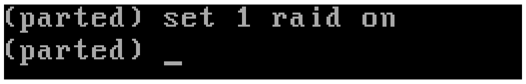


```
(parted) unit MB
(parted) mkpart primary 0 0
Warning: The resulting partition is not properly aligned for best performance.
Ignore/Cancel? I
(parted) _
```

Next we need to set the partition type to RAID.

⟶ Type **set 1 raid on** and press Enter.

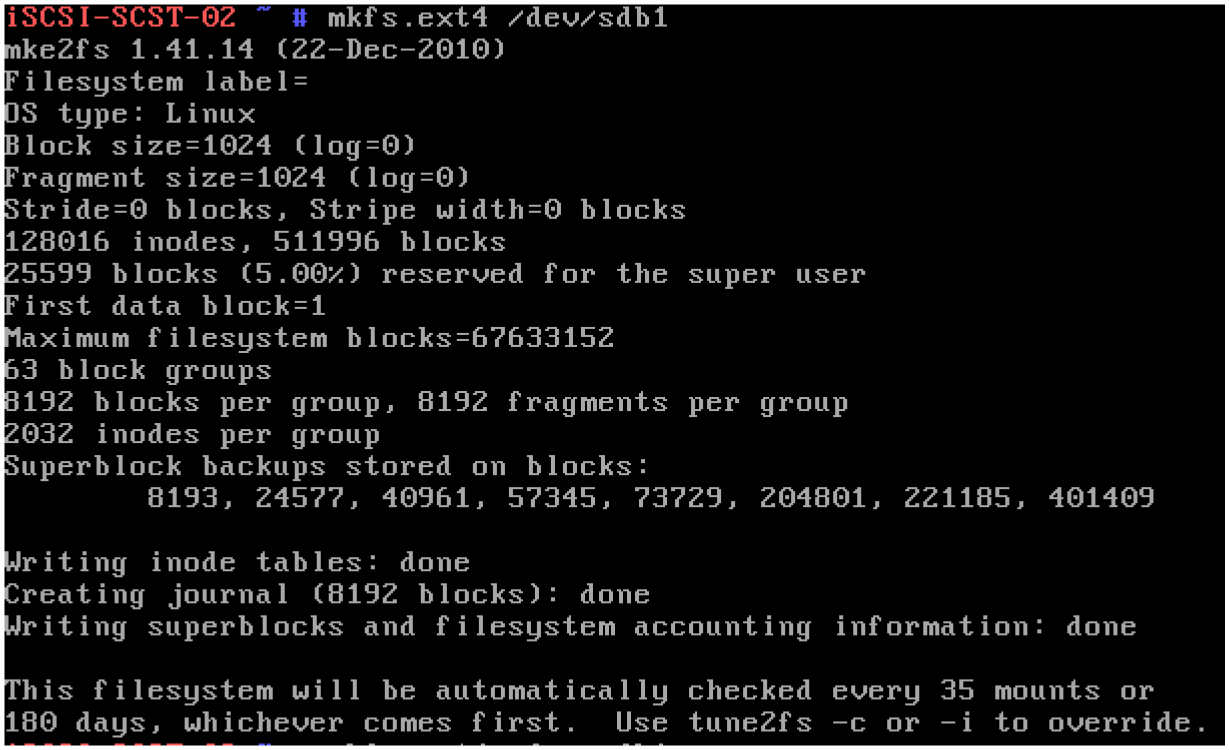
⟶ Type **quit** and press Enter.



```
(parted) set 1 raid on
(parted) _
```

Now we need to create a filesystem for our partition.

⟶ Type **mkfs.ext4 /dev/sdb1** and press Enter.

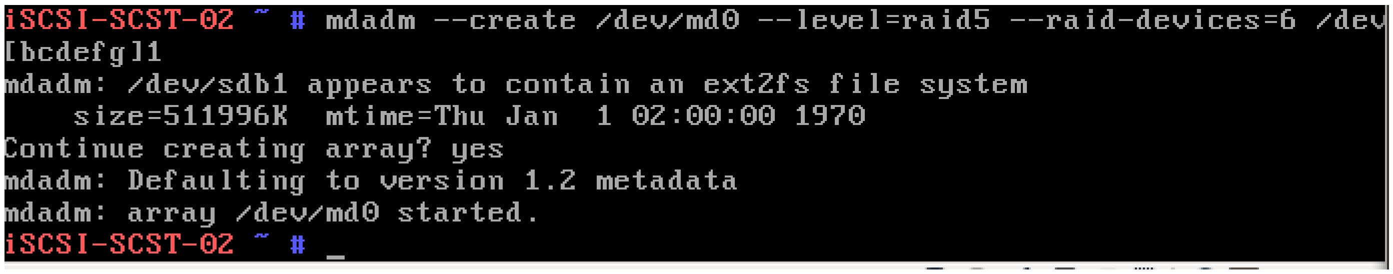


```
iSCSI-SCST-02 ~ # mkfs.ext4 /dev/sdb1
mke2fs 1.41.14 (22-Dec-2010)
Filesystem label=
OS type: Linux
Block size=1024 (log=0)
Fragment size=1024 (log=0)
Stride=0 blocks, Stripe width=0 blocks
128016 inodes, 511996 blocks
25599 blocks (5.00%) reserved for the super user
First data block=1
Maximum filesystem blocks=67633152
63 block groups
8192 blocks per group, 8192 fragments per group
2032 inodes per group
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729, 204801, 221185, 401409

Writing inode tables: done
Creating journal (8192 blocks): done
Writing superblocks and filesystem accounting information: done

This filesystem will be automatically checked every 35 mounts or
180 days, whichever comes first.  Use tune2fs -c or -i to override.
```

Repeat the above steps (starting with creating the partition table) for each of your disks, then continue with the instructions below.

And finally, we create our RAID Array!

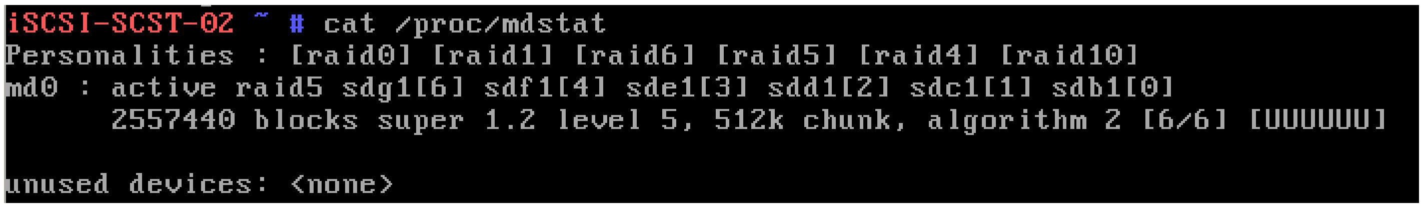
→ Type **mdadm --create /dev/md0 --level=raid5 --raid-devices=6 /dev/sd[bcdefg]1** and press Enter, type **yes** at prompt.



```
iSCSI-SCST-02 ~ # mdadm --create /dev/md0 --level=raid5 --raid-devices=6 /dev
[bcdefg]1
mdadm: /dev/sdb1 appears to contain an ext2fs file system
    size=511996K  mtime=Thu Jan  1 02:00:00 1970
Continue creating array? yes
mdadm: Defaulting to version 1.2 metadata
mdadm: array /dev/md0 started.
iSCSI-SCST-02 ~ # _
```

Now lets check the status of our RAID array.

→ Type **cat /proc/mdstat** and press Enter.

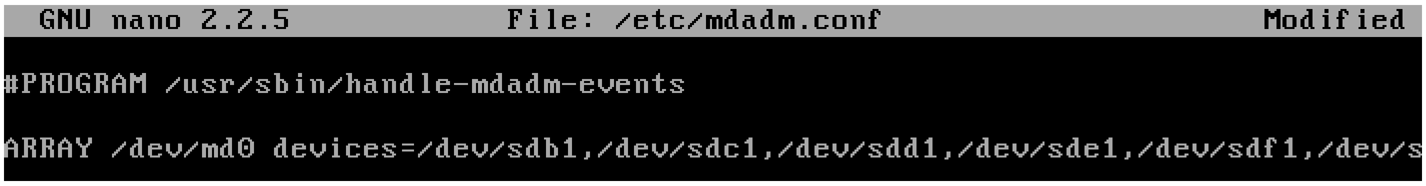


```
iSCSI-SCST-02 ~ # cat /proc/mdstat
Personalities : [raid0] [raid1] [raid6] [raid5] [raid4] [raid10]
md0 : active raid5 sdg1[6] sdf1[4] sde1[3] sdd1[2] sdc1[1] sdb1[0]
      2557440 blocks super 1.2 level 5, 512k chunk, algorithm 2 [6/6] [UUUUUU]

unused devices: <none>
```

Next we need to add a configuration entry to the file **/etc/mdadm.conf**

→ Type **nano /etc/mdadm.conf** and press Enter. Then add the following to the bottom of the file:

ARRAY /dev/md0 devices=/dev/sdb1,/dev/sdc1,/dev/sdd1,/dev/sde1,/dev/sdf1,/dev/sdg1

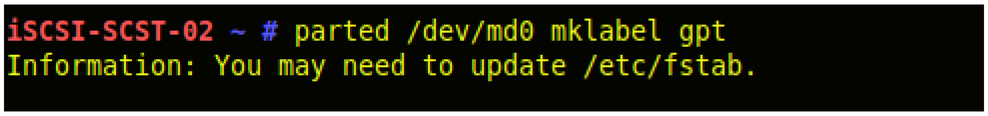


```
  GNU nano 2.2.5              File: /etc/mdadm.conf                    Modified

#PROGRAM /usr/sbin/handle-mdadm-events

ARRAY /dev/md0 devices=/dev/sdb1,/dev/sdc1,/dev/sdd1,/dev/sde1,/dev/sdf1,/dev/s
```

Next, reboot the machine and make sure that the RAID Array comes online and has the device name **md0**.

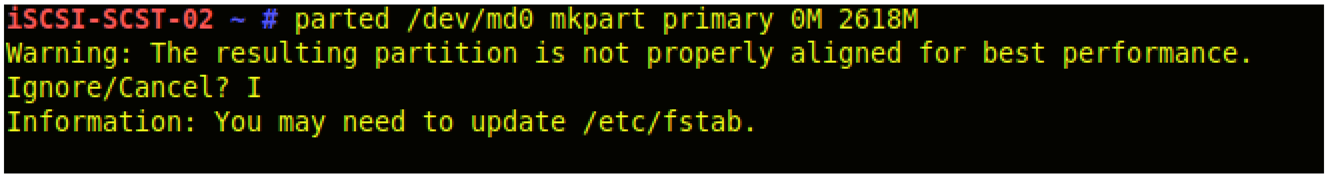
Next we need to create the partition for the RAID array.

⟶ Type **parted /dev/md0 mklabel gpt** and press Enter.



```
iSCSI-SCST-02 ~ # parted /dev/md0 mklabel gpt
Information: You may need to update /etc/fstab.
```

⟶ Type **parted /dev/md0 mkpart primary 0% 100%** and press Enter,

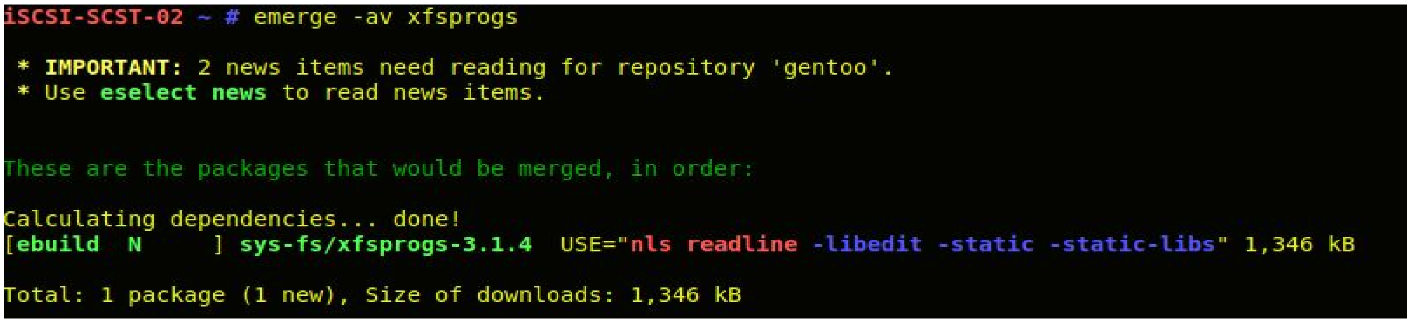


```
iSCSI-SCST-02 ~ # parted /dev/md0 mkpart primary 0M 2618M
Warning: The resulting partition is not properly aligned for best performance.
Ignore/Cancel? I
Information: You may need to update /etc/fstab.
```

Next we need to create a filesystem for the partition. For the purpose of this setup, which will be handling large files, we will be using the XFS filesystem, as it is well suited to handle large files.

First we need to install the necessary tools for the XFS filesystem.

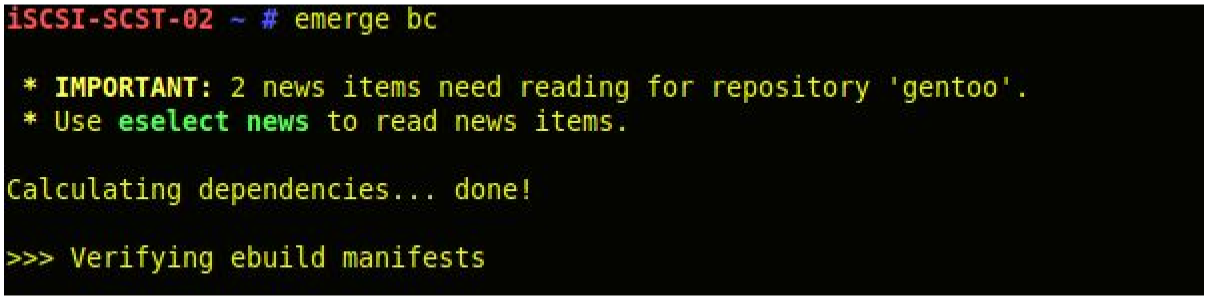
⟶ Type **emerge -v xfsprogs** and press Enter.



```
iSCSI-SCST-02 ~ # emerge -av xfsprogs

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] sys-fs/xfsprogs-3.1.4  USE="nls readline -libedit -static -static-libs" 1,346 kB

Total: 1 package (1 new), Size of downloads: 1,346 kB
```

We are going to use a script to do this for us, as it optimizes the filesystem for RAID. This script requires the bash calculator, so lets install that.

⟶ Type **emerge -v bc** and press Enter.



```
iSCSI-SCST-02 ~ # emerge bc

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.

Calculating dependencies... done!

>>> Verifying ebuild manifests
```

→ Type **nano xfscreate.sh**, and copy the following into the file.

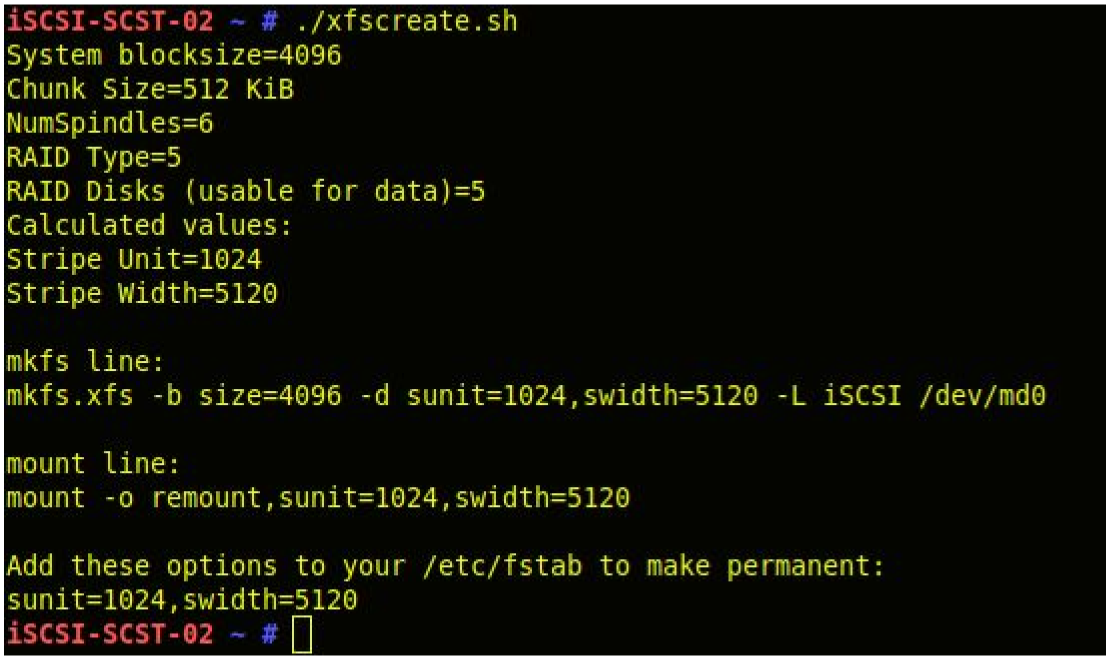
```
#!/bin/bash
BLOCKSIZE=4096                    # Make sure this is in bytes
CHUNKSIZE=512                     # This is your RAID Chunk size, type cat /proc/mdstat to check your raid chunk size.
NUMSPINDLES=6                     # Number of disks in array
RAID_TYPE=5                       # RAID Level (5 in our case)
RAID_DEVICE_NAME="/dev/md0" # Specify device name for your RAID device
FSLABEL="iSCSI"                   # Specify filesystem label for generating mkfs line here

case "$RAID_TYPE" in
0)
   RAID_DISKS=${NUMSPINDLES};
   ;;
1)
   RAID_DISKS=${NUMSPINDLES};
   ;;
10)
   RAID_DISKS=${NUMSPINDLES};
   ;;
5)
   RAID_DISKS=`echo "${NUMSPINDLES} - 1" | bc`;
   ;;
6)
   RAID_DISKS=`echo "${NUMSPINDLES} - 2" | bc`;
   ;;
*)
   echo "Please specify RAID_TYPE as one of: 0, 1, 10, 5, or 6."
   exit
   ;;
esac

SUNIT=`echo "${CHUNKSIZE} * 1024 / 512" | bc`
SWIDTH=`echo "$RAID_DISKS * ${SUNIT}" | bc`

echo "System blocksize=${BLOCKSIZE}"
echo "Chunk Size=${CHUNKSIZE} KiB"
echo "NumSpindles=${NUMSPINDLES}"
echo "RAID Type=${RAID_TYPE}"
echo "RAID Disks (usable for data)=${RAID_DISKS}"
echo "Calculated values:"
echo "Stripe Unit=${SUNIT}"
echo -e "Stripe Width=${SWIDTH}\n"
echo "mkfs line:"
echo -e "mkfs.xfs -b size=${BLOCKSIZE} -d sunit=${SUNIT},swidth=${SWIDTH} -L ${FSLABEL} $
{RAID_DEVICE_NAME}\n"
echo "mount line:"
echo -e "mount -o remount,sunit=${SUNIT},swidth=${SWIDTH}\n"
echo "Add these options to your /etc/fstab to make permanent:"
echo "sunit=${SUNIT},swidth=${SWIDTH}"
```

Next we need to make the file executable.

→ Type **chmod +x xfscreate.sh** and press Enter.

And now we execute it, so it shows us which parameters to use when creating the filesystem.

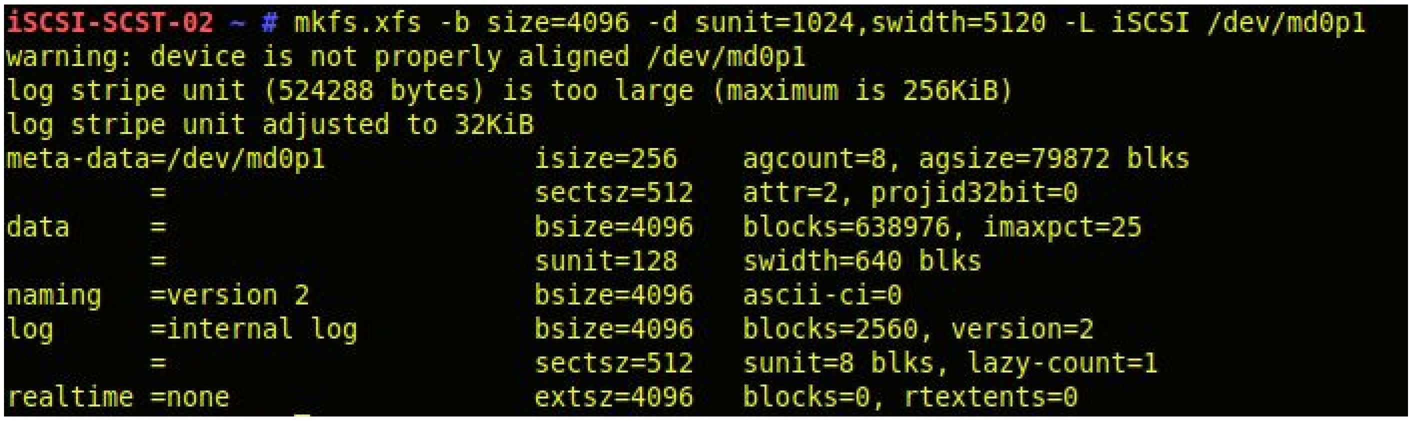
→ Type **./xfscreate.sh** and press Enter.



```
iSCSI-SCST-02 ~ # ./xfscreate.sh
System blocksize=4096
Chunk Size=512 KiB
NumSpindles=6
RAID Type=5
RAID Disks (usable for data)=5
Calculated values:
Stripe Unit=1024
Stripe Width=5120

mkfs line:
mkfs.xfs -b size=4096 -d sunit=1024,swidth=5120 -L iSCSI /dev/md0

mount line:
mount -o remount,sunit=1024,swidth=5120

Add these options to your /etc/fstab to make permanent:
sunit=1024,swidth=5120
iSCSI-SCST-02 ~ #
```

Now we create the filesystem. (Note: The values for **sunit** and **swidth** should be the ones you saw in the script output)

→ Type **mkfs.xfs -b size=4096 -d sunit=1024,swidth=5120 -L iSCSI /dev/md0p1 -f** and press Enter.

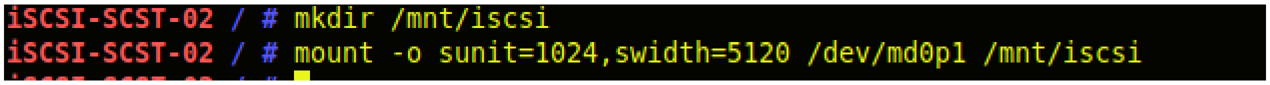


```
iSCSI-SCST-02 ~ # mkfs.xfs -b size=4096 -d sunit=1024,swidth=5120 -L iSCSI /dev/md0p1
warning: device is not properly aligned /dev/md0p1
log stripe unit (524288 bytes) is too large (maximum is 256KiB)
log stripe unit adjusted to 32KiB
meta-data=/dev/md0p1             isize=256    agcount=8, agsize=79872 blks
         =                       sectsz=512   attr=2, projid32bit=0
data     =                       bsize=4096   blocks=638976, imaxpct=25
         =                       sunit=128    swidth=640 blks
naming   =version 2              bsize=4096   ascii-ci=0
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=8 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

Next, lets create a mountpoint and mount our filesystem.

⟶ Type **mkdir /mnt/iscsi** and press Enter.

⟶ Type **mount -o sunit=1024,swidth=5120 /dev/md0p1 /mnt/iscsi** and press Enter. (***Note: The values for sunit and swidth should be the ones you saw in the script output***)

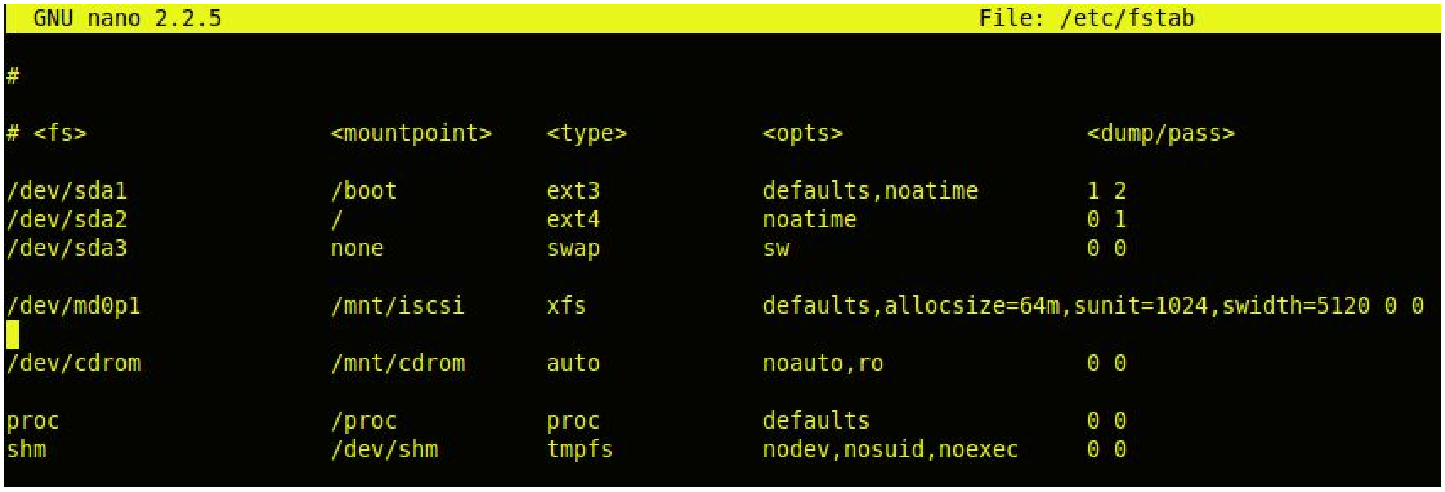


```
iSCSI-SCST-02 / # mkdir /mnt/iscsi
iSCSI-SCST-02 / # mount -o sunit=1024,swidth=5120 /dev/md0p1 /mnt/iscsi
```

Next, we need to add the partition to fstab, so that it is mounted at system boot.

⟶ Type **nano /etc/fstab** and press Enter. Add the following line to fstab, as seen in the screenshot below:

/dev/md0p1   /mnt/iscsi   xfs   defaults,allocsize=64m,sunit=1024,swidth=5120   0 0



```
  GNU nano 2.2.5                                                         File: /etc/fstab

#

# <fs>                    <mountpoint>    <type>          <opts>                  <dump/pass>

/dev/sda1                 /boot           ext3            defaults,noatime        1 2
/dev/sda2                 /               ext4            noatime                 0 1
/dev/sda3                 none            swap            sw                      0 0

/dev/md0p1                /mnt/iscsi      xfs             defaults,allocsize=64m,sunit=1024,swidth=5120 0 0

/dev/cdrom                /mnt/cdrom      auto            noauto,ro               0 0

proc                      /proc           proc            defaults                0 0
shm                       /dev/shm        tmpfs           nodev,nosuid,noexec     0 0
```

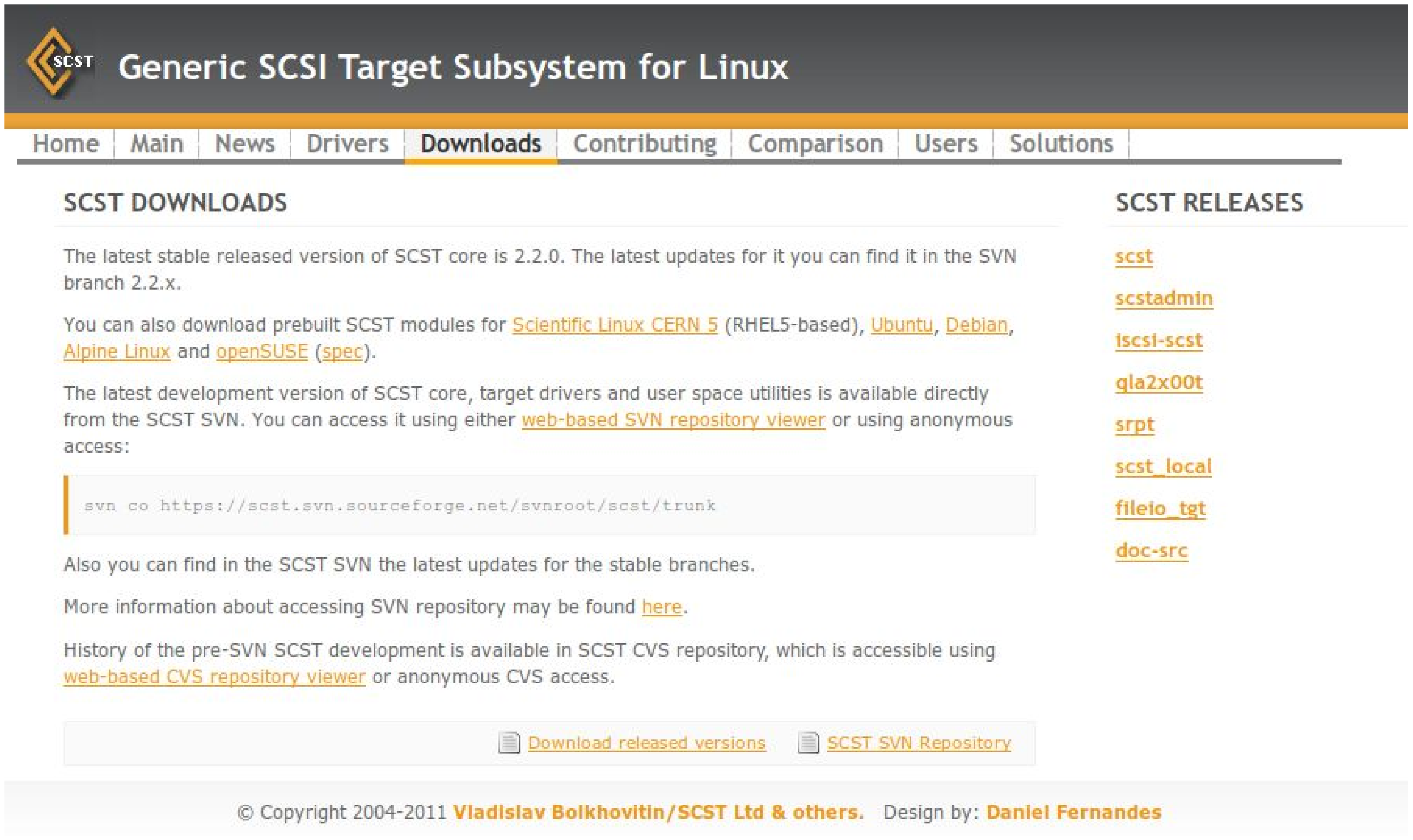
And congratulations, you have just set up a Linux SW RAID 5 Array with a XFS Filesystem Optimized for our RAID volume. we can now continue on the installlation of SCST and its dependencies.

3 Installing SCST

Now we can start with the final phase of this project – Installing SCST.

Firstly we need to do some preperation, we need to download the source from the SCST webpage. Open your browser, and navigate to → http://scst.sourceforge.net/downloads.html

Download the latest version of scst from here.



SCST
Generic SCSI Target Subsystem for Linux
Home | Main | News | Drivers | Downloads | Contributing | Comparison | Users | Solutions
SCST RELEASES
scst
scstadmin
iscsi-scst
qla2x00t
srpt
scst_local
fileio_tgt
doc-src

SCST DOWNLOADS

The latest stable released version of SCST core is 2.2.0. The latest updates for it you can find it in the SVN branch 2.2.x.

You can also download prebuilt SCST modules for Scientific Linux CERN 5 (RHEL5-based), Ubuntu, Debian, Alpine Linux and openSUSE (spec).

The latest development version of SCST core, target drivers and user space utilities is available directly from the SCST SVN. You can access it using either web-based SVN repository viewer or using anonymous access:

```
svn co https://scst.svn.sourceforge.net/svnroot/scst/trunk
```

Also you can find in the SCST SVN the latest updates for the stable branches.

More information about accessing SVN repository may be found here.

History of the pre-SVN SCST development is available in SCST CVS repository, which is accessible using web-based CVS repository viewer or anonymous CVS access.

Download released versions   SCST SVN Repository

© Copyright 2004-2011 Vladislav Bolkhovitin/SCST Ltd & others.   Design by: Daniel Fernandes

Click on the "Download Released Versions" link at the bottom of the page. On the next page, download the latest released versions of the following, make sure the version numbers correspond:

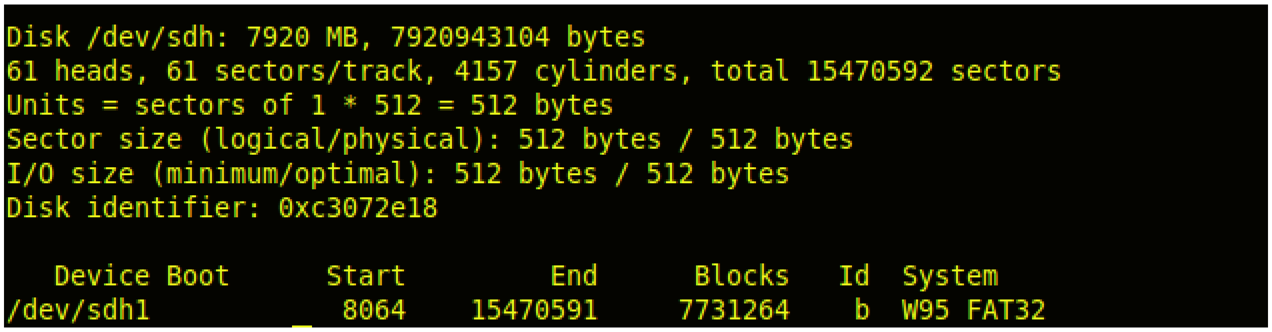
scst
iscsi-scst
scstadmin

Save the scst, iscsi-scst & scstadmin tarballs to a USB Flash disk, and put the flash disk in your SCST machine.

Next, we need to mount the memory stick.

Lets check which device it is.

⟶ Type **fdisk -l** and press Enter. It should be the last device on the list.

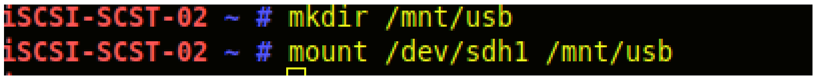


```
Disk /dev/sdh: 7920 MB, 7920943104 bytes
61 heads, 61 sectors/track, 4157 cylinders, total 15470592 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0xc3072e18

   Device Boot      Start         End      Blocks   Id  System
/dev/sdh1            8064    15470591     7731264    b  W95 FAT32
```

Now we mount it!

⟶ Type **mkdir /mnt/usb** and press Enter.

⟶ Type **mount /dev/sdh1 /mnt/usb** and press Enter.



```
iSCSI-SCST-02 ~ # mkdir /mnt/usb
iSCSI-SCST-02 ~ # mount /dev/sdh1 /mnt/usb
```

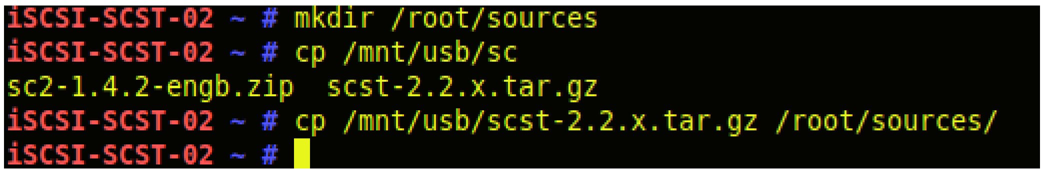
Next, lets create a folder to work from – where we can extract the SCST source and compile it. For convenience we will be using root's home directory. Also copy the source there.

⟶ Type **mkdir /root/sources** and press Enter.

⟶ Type **cp /mnt/usb/scst-*.tar.gz /root/sources/** and press Enter.

⟶ Type **cp /mnt/usb/scstadmin-*.tar.bz2 /root/sources/** and press Enter.

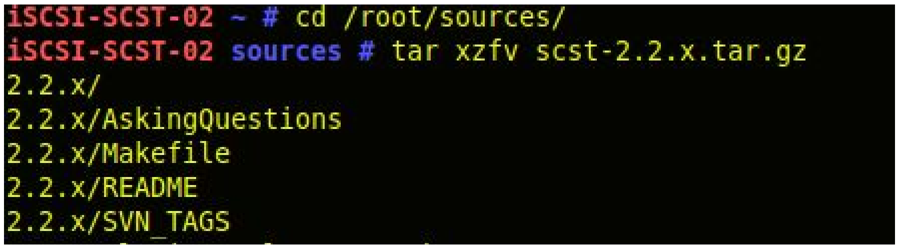
⟶ Type **cp /mnt/usb/iscsi-scst-*.tar.bz2 /root/sources/** and press Enter.



```
iSCSI-SCST-02 ~ # mkdir /root/sources
iSCSI-SCST-02 ~ # cp /mnt/usb/sc
sc2-1.4.2-engb.zip  scst-2.2.x.tar.gz
iSCSI-SCST-02 ~ # cp /mnt/usb/scst-2.2.x.tar.gz /root/sources/
iSCSI-SCST-02 ~ # 
```

Next, we need to extract the tarball.

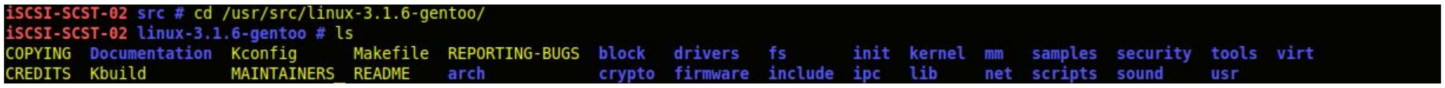
→ Type **cd /root/sources** and press Enter.
→ Type **tar xjpf scst-*.tar.bz2** and press Enter.
→ Type **tar xjpf scstadmin-*.tar.bz2** and press Enter.
→ Type **tar xjpf iscsi-scst-*.tar.bz2** and press Enter.



```
iSCSI-SCST-02 ~ # cd /root/sources/
iSCSI-SCST-02 sources # tar xzfv scst-2.2.x.tar.gz
2.2.x/
2.2.x/AskingQuestions
2.2.x/Makefile
2.2.x/README
2.2.x/SVN_TAGS
```

OK, Now we need to apply the kernel patches for SCST. First we navigate to the kernel source directory.

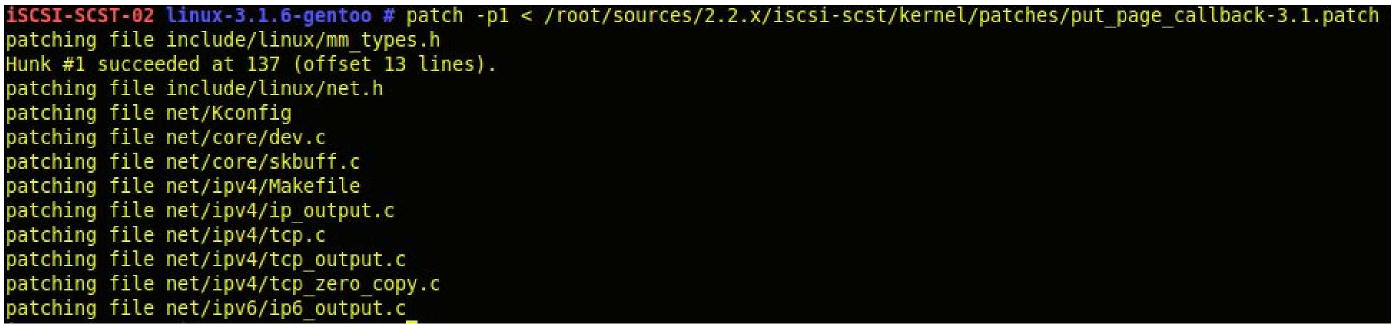
→ Type **cd /usr/src/linux-3.1.6-gentoo/** and press Enter. (replace the "-3.1.6" with your kernel version)



```
iSCSI-SCST-02 src # cd /usr/src/linux-3.1.6-gentoo/
iSCSI-SCST-02 linux-3.1.6-gentoo # ls
COPYING  Documentation  Kconfig      Makefile  REPORTING-BUGS  block   drivers   fs       init  kernel  mm   samples  security  tools  virt
CREDITS  Kbuild         MAINTAINERS  README    arch            crypto  firmware  include  ipc   lib     net  scripts  sound     usr
```

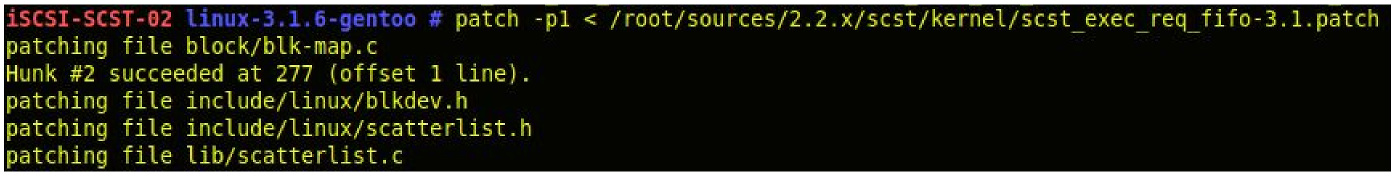
Now we apply the patches, make sure you apply the ones relevant to your kernel version, also the path to the patches might differ in your case, adjust accordingly.

→ Type **patch -p1 < /root/sources/iscsi-scst/kernel/patches/put_page_callback-3.1.patch** and press Enter.



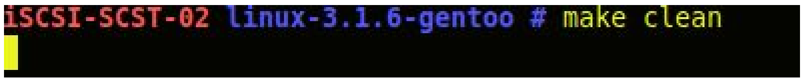
```
iSCSI-SCST-02 linux-3.1.6-gentoo # patch -p1 < /root/sources/2.2.x/iscsi-scst/kernel/patches/put_page_callback-3.1.patch
patching file include/linux/mm_types.h
Hunk #1 succeeded at 137 (offset 13 lines).
patching file include/linux/net.h
patching file net/Kconfig
patching file net/core/dev.c
patching file net/core/skbuff.c
patching file net/ipv4/Makefile
patching file net/ipv4/ip_output.c
patching file net/ipv4/tcp.c
patching file net/ipv4/tcp_output.c
patching file net/ipv4/tcp_zero_copy.c
patching file net/ipv6/ip6_output.c
```

→ Type **patch -p1 < /root/sources/scst/kernel/scst_exec_req_fifo-3.1.patch** and press Enter.



```
iSCSI-SCST-02 linux-3.1.6-gentoo # patch -p1 < /root/sources/2.2.x/scst/kernel/scst_exec_req_fifo-3.1.patch
patching file block/blk-map.c
Hunk #2 succeeded at 277 (offset 1 line).
patching file include/linux/blkdev.h
patching file include/linux/scatterlist.h
patching file lib/scatterlist.c
```

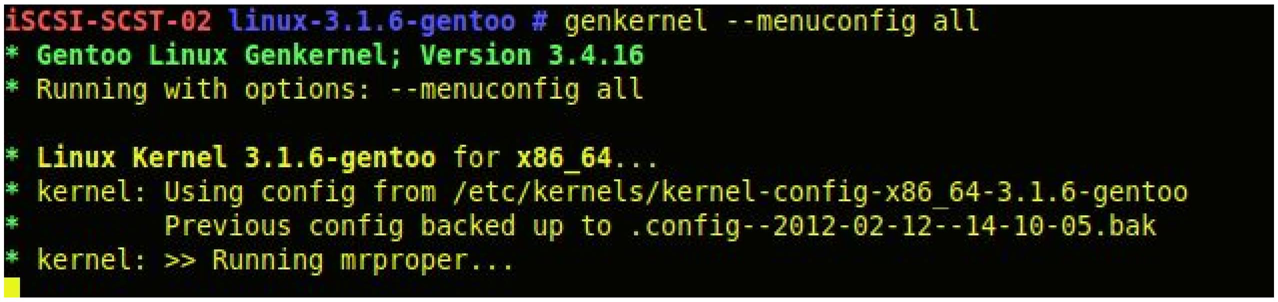
⟶ Type **make clean** and press Enter.



```
iSCSI-SCST-02 linux-3.1.6-gentoo # make clean
```

Now we need to select the kernel modules and recompile the kernel.

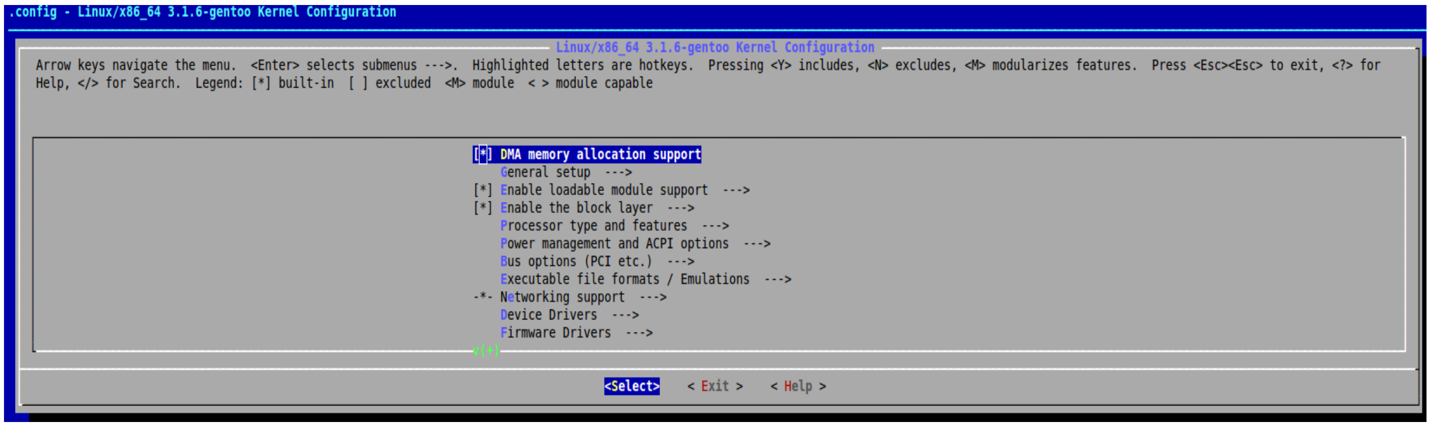
⟶ Type **genkernel –menuconfig all** and press Enter.



```
iSCSI-SCST-02 linux-3.1.6-gentoo # genkernel --menuconfig all
* Gentoo Linux Genkernel; Version 3.4.16
* Running with options: --menuconfig all

* Linux Kernel 3.1.6-gentoo for x86_64...
* kernel: Using config from /etc/kernels/kernel-config-x86_64-3.1.6-gentoo
*         Previous config backed up to .config--2012-02-12--14-10-05.bak
* kernel: >> Running mrproper...
```

These are the modules you need to select/change:

Select Networking support -> Networking options -> TCP/IP networking
Select Networking support -> Networking options -> TCP/IP zero-copy transfer completion notification
Select Device Drivers -> SCSI device support -> SCSI disk support
Select Enable the block layer -> IO Schedulers -> CFQ I/O Scheduler
Set Enable the Block layer -> IO Schedulers -> Default I/O Scheduler to 'CFQ'
Set Processor type and features -> Preemption Model to 'No Forced Preemption (Server)'



```
.config - Linux/x86_64 3.1.6-gentoo Kernel Configuration
                              Linux/x86_64 3.1.6-gentoo Kernel Configuration
 Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for
 Help, </> for Search.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

                                   [*] DMA memory allocation support
                                       General setup  --->
                                   [*] Enable loadable module support  --->
                                   [*] Enable the block layer  --->
                                       Processor type and features  --->
                                       Power management and ACPI options  --->
                                       Bus options (PCI etc.)  --->
                                       Executable file formats / Emulations  --->
                                   -*- Networking support  --->
                                       Device Drivers  --->
                                       Firmware Drivers  --->
                                   v(+)

                                          <Select>    < Exit >    < Help >
```

Exit the menu and say YES to save the changes. The kernel will now recompile with the options we have added to the kernel.

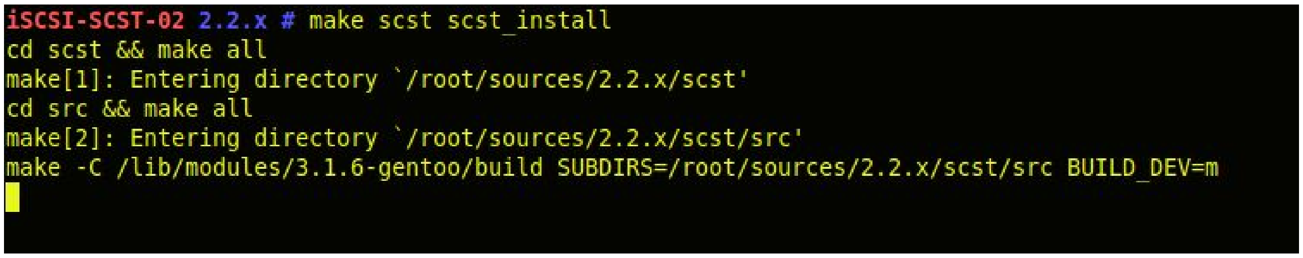
When the recompiling is done, reboot the system.

⟶ Type **reboot** and press Enter.

Now it is time to install SCST. Once logged in, navigate to the **scst** source directory.

Next, lets start installing.

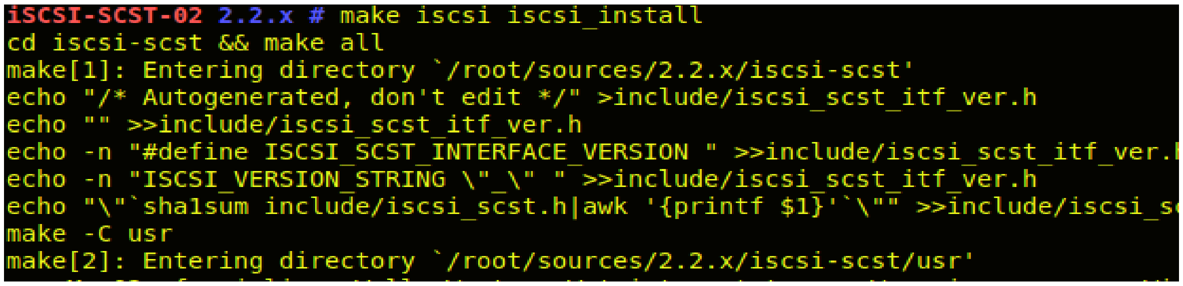
⟶ Type **make install** and press Enter.



```
iSCSI-SCST-02 2.2.x # make scst scst_install
cd scst && make all
make[1]: Entering directory `/root/sources/2.2.x/scst'
cd src && make all
make[2]: Entering directory `/root/sources/2.2.x/scst/src'
make -C /lib/modules/3.1.6-gentoo/build SUBDIRS=/root/sources/2.2.x/scst/src BUILD_DEV=m
```

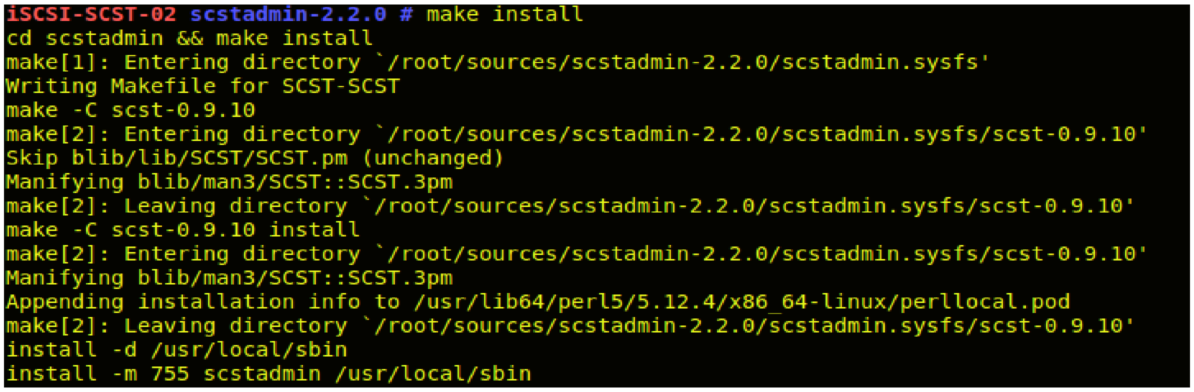
When that is done, navigate to your **iscsi-scst** source directory, and then:

⟶ Type **make install** and press Enter.



```
iSCSI-SCST-02 2.2.x # make iscsi iscsi_install
cd iscsi-scst && make all
make[1]: Entering directory `/root/sources/2.2.x/iscsi-scst'
echo "/* Autogenerated, don't edit */" >include/iscsi_scst_itf_ver.h
echo "" >>include/iscsi_scst_itf_ver.h
echo -n "#define ISCSI_SCST_INTERFACE_VERSION " >>include/iscsi_scst_itf_ver.
echo -n "ISCSI_VERSION_STRING \"_\" " >>include/iscsi_scst_itf_ver.h
echo "\""`sha1sum include/iscsi_scst.h|awk '{printf $1}'`\"" >>include/iscsi_s
make -C usr
make[2]: Entering directory `/root/sources/2.2.x/iscsi-scst/usr'
```

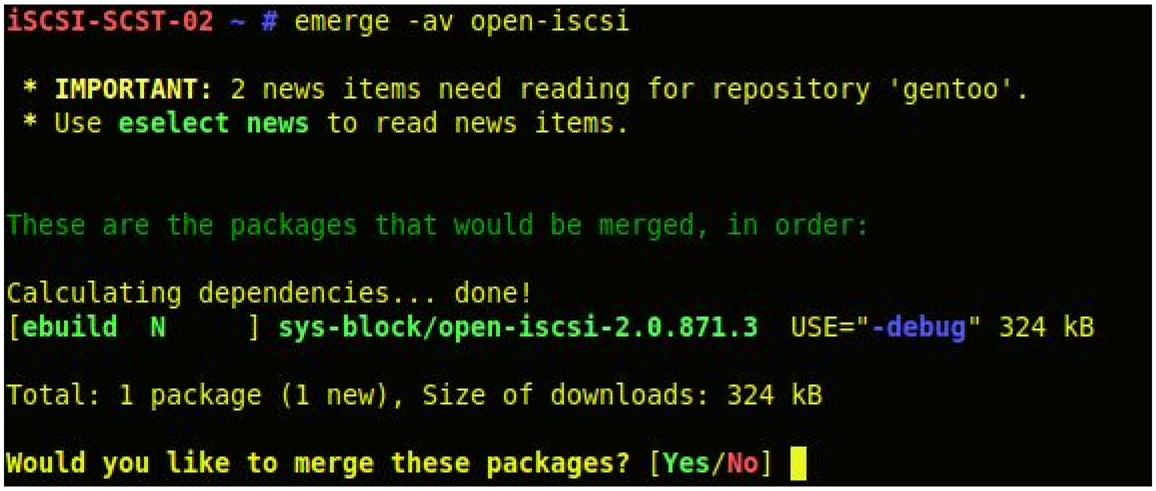
When that is done, navigate to your **scstadmin** source directory, and then:
⟶ Type **make install** and press Enter.



```
iSCSI-SCST-02 scstadmin-2.2.0 # make install
cd scstadmin && make install
make[1]: Entering directory `/root/sources/scstadmin-2.2.0/scstadmin.sysfs'
Writing Makefile for SCST-SCST
make -C scst-0.9.10
make[2]: Entering directory `/root/sources/scstadmin-2.2.0/scstadmin.sysfs/scst-0.9.10'
Skip blib/lib/SCST/SCST.pm (unchanged)
Manifying blib/man3/SCST::SCST.3pm
make[2]: Leaving directory `/root/sources/scstadmin-2.2.0/scstadmin.sysfs/scst-0.9.10'
make -C scst-0.9.10 install
make[2]: Entering directory `/root/sources/scstadmin-2.2.0/scstadmin.sysfs/scst-0.9.10'
Manifying blib/man3/SCST::SCST.3pm
Appending installation info to /usr/lib64/perl5/5.12.4/x86_64-linux/perllocal.pod
make[2]: Leaving directory `/root/sources/scstadmin-2.2.0/scstadmin.sysfs/scst-0.9.10'
install -d /usr/local/sbin
install -m 755 scstadmin /usr/local/sbin
```

Next we need to set up our initiator name. The most convienient way of doing this is by installing open-iscsi.

⟶ Type **emerge -v open-iscsi** and press Enter.



```
iSCSI-SCST-02 ~ # emerge -av open-iscsi

 * IMPORTANT: 2 news items need reading for repository 'gentoo'.
 * Use eselect news to read news items.


These are the packages that would be merged, in order:

Calculating dependencies... done!
[ebuild  N     ] sys-block/open-iscsi-2.0.871.3  USE="-debug" 324 kB

Total: 1 package (1 new), Size of downloads: 324 kB

Would you like to merge these packages? [Yes/No]
```

Next, type the following:

⟶ Type **/etc/init.d/iscsid stop** and press Enter.

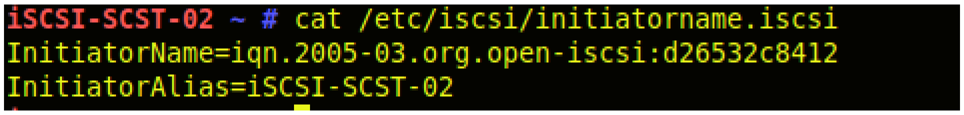
⟶ Type { **echo "InitiatorName=$(if [-e /usr/sbin/iscsi-iname]; then /usr/sbin/iscsi-iname; else /sbin/iscsi-iname; fi)";** and press Enter.

⟶ Type **echo "InitiatorAlias=$(hostname)"; } >/etc/iscsi/initiatorname.iscsi** and press Enter.

⟶ Type **/etc/init.d/iscsid start** and press Enter.

Next, lets see if the name has been correctly generated.

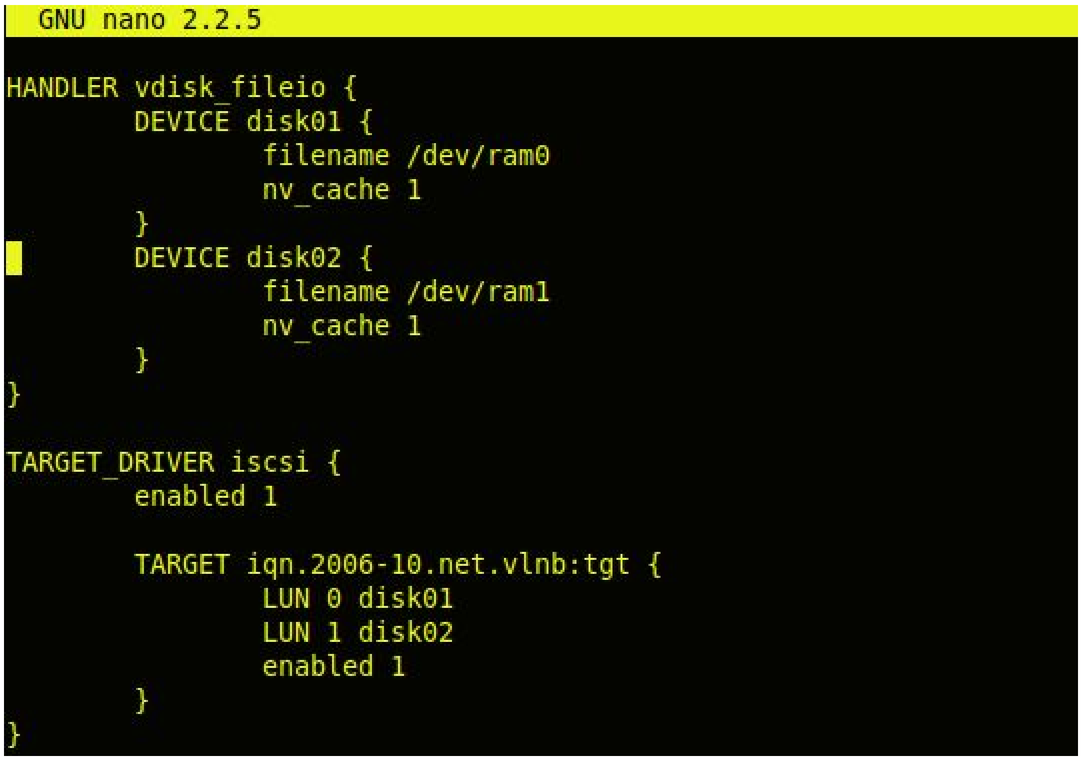
⟶ Type **cat /etc/iscsi/initiatorname.iscsi** and press Enter.



```
iSCSI-SCST-02 ~ # cat /etc/iscsi/initiatorname.iscsi
InitiatorName=iqn.2005-03.org.open-iscsi:d26532c8412
InitiatorAlias=iSCSI-SCST-02
```

Now let's set up a simple sample config for **scst.conf**. Make sure your file looks like the one in the screenshot.

⟶ Type **nano /etc/scst.conf** and press Enter.

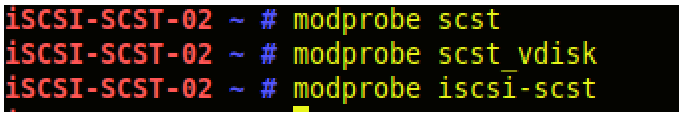


```
  GNU nano 2.2.5

HANDLER vdisk_fileio {
        DEVICE disk01 {
                filename /dev/ram0
                nv_cache 1
        }
        DEVICE disk02 {
                filename /dev/ram1
                nv_cache 1
        }
}

TARGET_DRIVER iscsi {
        enabled 1

        TARGET iqn.2006-10.net.vlnb:tgt {
                LUN 0 disk01
                LUN 1 disk02
                enabled 1
        }
}
```

Keep in mind that for each target, LUN0 must exist.

Now, lets load the kernel modules.

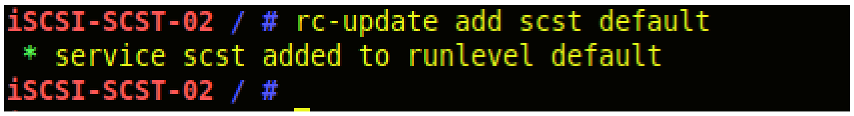
⟶ Type **modprobe scst** and press Enter.
⟶ Type **modprobe scst_vdisk** and press Enter.
⟶ Type **modprobe iscsi-scst** and press Enter



```
iSCSI-SCST-02 ~ # modprobe scst
iSCSI-SCST-02 ~ # modprobe scst_vdisk
iSCSI-SCST-02 ~ # modprobe iscsi-scst
```

Now, lets start scst on boot.

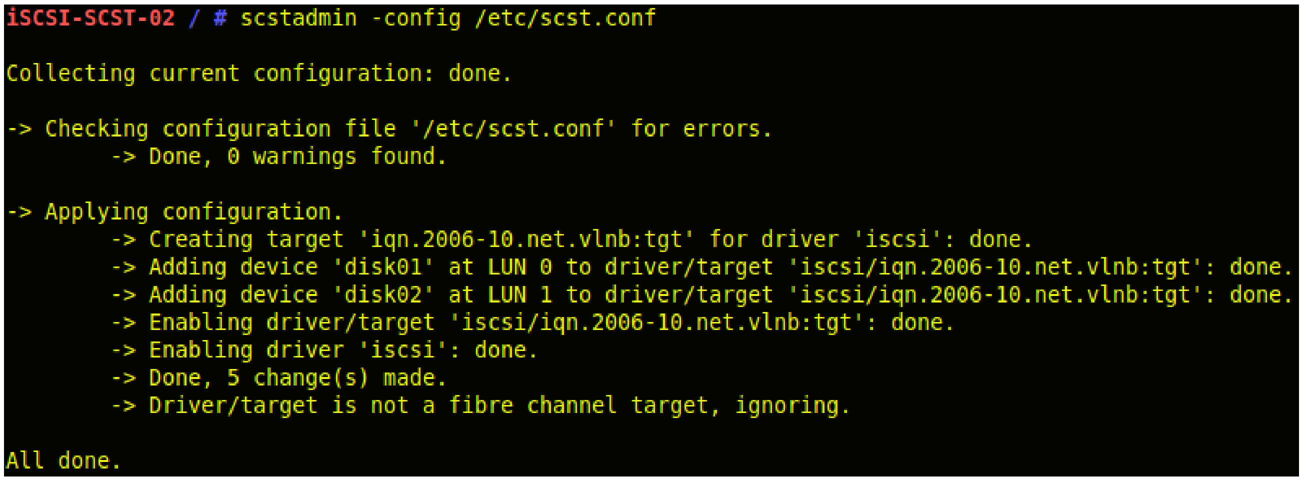
⟶ Type **rc-update add scst default** and press Enter.



```
iSCSI-SCST-02 / # rc-update add scst default
 * service scst added to runlevel default
iSCSI-SCST-02 / # _
```

Now let's apply our scst configuration.

⟶ Type **scstadmin -config /etc/scst.conf** and press Enter.



```
iSCSI-SCST-02 / # scstadmin -config /etc/scst.conf

Collecting current configuration: done.

-> Checking configuration file '/etc/scst.conf' for errors.
        -> Done, 0 warnings found.

-> Applying configuration.
        -> Creating target 'iqn.2006-10.net.vlnb:tgt' for driver 'iscsi': done.
        -> Adding device 'disk01' at LUN 0 to driver/target 'iscsi/iqn.2006-10.net.vlnb:tgt': done.
        -> Adding device 'disk02' at LUN 1 to driver/target 'iscsi/iqn.2006-10.net.vlnb:tgt': done.
        -> Enabling driver/target 'iscsi/iqn.2006-10.net.vlnb:tgt': done.
        -> Enabling driver 'iscsi': done.
        -> Done, 5 change(s) made.
        -> Driver/target is not a fibre channel target, ignoring.

All done.
```

And viola! SCST is now installed & up and running!! Now you just need to create your LUNs and update the configuration the way you need it.

4 Configuring SCST