Building a SAN storage device using ISCSI on SuSe Linux

Part 1 – Installing and configuring the ISCSI Target

Wouter Rusman
wouter@rusman.net

Prerequisites and terminology....3
Review your kernel version....4
Building the iSCSI Target software....9
Configuring the file systems to export....10
Configuring the iSCSI Target software....15

Prerequisites and terminology

<version> = version you downloaded or you are using

Commands that have to be entered are written in a black background, for example

```
SAN1:~/build/ # <command>
```

This states that the commandline <command> has to be entered at the prompt.
The prompt shows that your current working directory is "/root/build/" on host "SAN1"

This document is based on SuSe 9.3, however this could also be used on version 10 and 10.1

Make sure you install a base system with the following options checked:
- Basic system
- wget
- mc

If you choose to build the ISCSI software on the ISCSI target itself, you can also check the following parts during the setup process.
- kernel development
- openssh-debuginfo
- openssl devel
- openssl-doc

If you plan to build the software on another box, make sure this uses the same software versions as the target machine. Otherwise Kernel-panics or invalid drivers could result in your SuSe box becoming defective.

This document assumes you have root privileges or you are logged in as root.
To change to the root user if you are logged in as a normal user use the command

```
# su -root
```

Review your kernel version

The standard kernel version on my SuSe 9.3 box was 2.6.11.4-20a-default
Developers of the iSCSI-Enterprise Target software always develop their product on the newest kernel, so it is wise to update the kernel to the latest available version to prevent compilation errors.

Check the kernel version with the command

```
SAN1:~ # uname -r
```

If the latest kernel available in RPM format is way off the latest release, or if the kernel-development RPM's are not available it is wise to build a kernel yourself.

Don't be scared !!

If you don't know how to compile a kernel, don't be afraid… it won't send information to Microsoft or blowup your server. Kernel-building is not something for the ultra-geek, even you can do it. So let's get going.

If you follow the steps for compiling the kernel below, nothing can go wrong. And even if something does go wrong you can always start the old kernel from the boot menu and start the build process all over.

In the text below you can choose either to use the ftp-server used in this document, or choose your nearest mirror to download the kernel-source from.

When downloading the kernel-source you can choose between 2 file-types, bz2 and gz. These files contain the same archive, but the bzip2 archives are usually a bit smaller, so if you don't have bandwidth at spare download the bz2 archive.

Choose a directory on your file system to download and compile the downloaded tarballs in. You can also choose to create a directory under your home-dir and work from there.
In this document I will use /root/build for this. Don't place downloaded files all over your system, this will clutter up the system, and create confusion when looking for the thing you just downloaded.

Create and go to /root/build by giving the commands

```
SAN1:~ # mkdir /root/build
SAN1:~ # cd /root/build
```

Now have a look what the latest kernel available is by issuing the command

```
SAN1:~/build/ # wget ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/LATEST*
```

You will get a response like the following :

```
SAN1:~/build/ # wget ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/LATEST*
--17:24:33--  ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/LATEST*
           => `.listing'
Resolving ftp.nluug.nl... 192.87.102.36, 2001:610:1:80aa:192:87:102:36
Connecting to ftp.nluug.nl|192.87.102.36|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /vol/3/linux-kernel/v2.6 ... done.
==> PASV ... done.    ==> LIST ... done.

    [ <=>                                 ] 44,610        --.--K/s

17:24:33 (463.79 KB/s) - `.listing' saved [44610]

Removed `.listing'.
--17:24:34--  ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/LATEST-IS-2.6.15.4
           => `LATEST-IS-2.6.15.4'
==> CWD /vol/3/linux-kernel/v2.6 ... done.
==> PASV ... done.    ==> RETR LATEST-IS-2.6.15.4 ... done.

    [ <=>                                 ] 0             --.--K/s

17:24:34 (0.00 B/s) - `LATEST-IS-2.6.15.4' saved [0]

SAN1:~/build/ #
```

As you can see the latest version is 2.6.15.4. This version-file can be deleted from your local hard disk with the command

```
SAN1:~/build/ # rm -f LATEST-IS-2.6.15.4
```

Now get the kernel-source using the command

```
SAN1:~/build/ # wget ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/linux-2.6.15.4.tar.gz
```

or

```
SAN1:~/build/ # wget ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/linux-2.6.15.4.tar.bz2
```

The download should start and when finished the archive will be saved to the directory /root/build.

```
SAN1:~/build/ # wget ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/linux-
2.6.15.4.tar.gz
--17:35:56--  ftp://ftp.nluug.nl/vol/3/linux-kernel/v2.6/linux-2.6.15.4.tar.gz
           => `linux-2.6.15.4.tar.gz'
Resolving ftp.nluug.nl... 192.87.102.36, 2001:610:1:80aa:192:87:102:36
Connecting to ftp.nluug.nl|192.87.102.36|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /vol/3/linux-kernel/v2.6 ... done.
==> PASV ... done.    ==> RETR linux-2.6.15.4.tar.gz ... done.
Length: 49,907,449 (48M) (unauthoritative)

100%[====================================>] 49,907,449   637.95K/s    ETA 00:00

17:37:13 (634.87 KB/s) - `linux-2.6.15.4.tar.gz' saved [49907449]

SAN1:~/build/ #
```

Now unpack the archive using

```
SAN1:~/build/ # tar -xvzf linux-<version>.tar.gz
```

or, if using the bz2 archive

```
SAN1:~/build/ # tar -xvjf linux-version.tar.bz2
```

Now move the unpacked archive-tree to the appropriate place using

```
SAN1:~/build/ # mv linux-<version> /usr/src/linux-<version>
```

And adjust the “linux”-symbolic link using

```
SAN1:~/build/ # rm -f /usr/src/linux
SAN1:~/build/ # ln -s /usr/src/linux-<version> /usr/src/linux
```

The defaults for building the kernel on a SuSe system are stored in the static kernel and are decompressed at run-time.
The file represents the settings for compiling a kernel with the same options as the one installed on your system.

To copy the default settings from the SuSe kernel to the new linux kernel source directory and decompress it using gzip give the commands

```
SAN1:~/build/ # cp /proc/config.gz /usr/src/linux/oldbuild.gz
SAN1:~/build/ # gzip -d /usr/src/linux/oldbuild.gz
```

Next step is to go into the source directory with the command

```
SAN1:~/build/ # cd /usr/src/linux
```

To get a clean start of the configuration and build process issue the command

```
SAN1:/usr/src/linux/ # make mrproper
```

This will remove all config files and already compiled options from the source tree.
Now start the kernel-configuration by issuing the command

```
SAN1:/usr/src/linux/ # make menuconfig
```

Load the configuration of the running kernel using the “Load alternate configuration file” and specify the just created “oldbuild” (without the .gz)

This configuration should be enough to re-compile the kernel, but if you know what you’re doing or just want to find out what goes into the config-file you can set extra options in this menu to compile the kernel the way you want it.

If you altered the configuration and wish to save it to a different file, use the option “Save configuration to an alternate file” and choose a name and write it down for later usage.
Exit the utility with the “exit” button and save the newly created configuration.

Now start the building process by giving the command

```
SAN1:/usr/src/linux/ # make
```

Sit back and enjoy a cup of your favorite beverage while the kernel builds, this process can take a while. Meanwhile I suggest you read the FAQ, wiki and readme of the iSCSI Target software at http://iscsitarget.sourceforge.net . You can read the guide for installing the iSCSI initiator on NetWare 6.5 at http://www.novell.com/documentation/iscsi1_nak/index.html

When the kernel build process is complete, install the new kernel and drivers with the commands

```
SAN1:/usr/src/linux/ # make modules
SAN1:/usr/src/linux/ # make modules_install
SAN1:/usr/src/linux/ # make install
```

Finally you have to build a new init ramdrive with the command

```
SAN1:/usr/src/linux/ # mkinitrd
```

Now the init ramdrive is created and your kernel is installed.
By default, the kernel is loaded using the default names also used by the old kernel. So unless you make this modification, when you have installed the new kernel there will be no way to boot the old one.
To modify the grub configuration use the command

```
SAN1: vi /boot/grub/menu.lst
```

Or when you have installed the midnight commander (mc)

```
SAN1: mcedit /boot/grub/menu.lst
```

In this file you see a section (in my case of SuSe 9.3)

```
color white/blue black/light-gray
default 1
timeout 8


title Linux-2.6.15.4-default
    kernel (hd0,0)/vmlinuz root=/dev/hda3 vga=0x317 selinux=0  splash=silent resume=/dev/hda2
showopts


###Don't change this comment - YaST2 identifier: Original name: linux###
title SUSE LINUX 9.3
    kernel (hd0,0)/vmlinuz root=/dev/hda3 vga=0x317 selinux=0  splash=silent resume=/dev/hda2
showopts
    initrd (hd0,0)/initrd
```

As you can see both entries rely on the “vmlinuz” kernel and the “initrd” ram drive.
Also the “default 1” line lets grub know to boot the old kernel as default.
To change this section to allow us to boot the old kernel and the new kernel separately, modify it using the kernel versions attached to the filenames found in the /boot directory

```
color white/blue black/light-gray
default 0
timeout 8


title Linux-2.6.15.4-default
    kernel (hd0,0)/vmlinuz-2.6.15.4-default root=/dev/hda3 vga=0x317 selinux=0  splash=silent
resume=/dev/hda2  showopts
    initrd (hd0,0)/initrd-2.6.15.4-default

###Don't change this comment - YaST2 identifier: Original name: linux###
title SUSE LINUX 9.3
    kernel (hd0,0)/vmlinuz-2.6.11.4-21.11-default root=/dev/hda3 vga=0x317 selinux=0
splash=silent resume=/dev/hda2  showopts
    initrd
```

With this altered like above, you can always refer to the stock kernel that came with your SuSe, and grub will boot the new kernel as default. (be sure to add the line for the new kernel defining the initrd)

Unfortunately the kernel can not be activated while running so for the new kernel to become active, the system has to be rebooted.
When the system is rebooted check if the new kernel is loaded using the command

```
SAN1:~/ # uname -r
```

Check the log file (/var/log/boot.msg) for errors, and if so try to resolve them by reviewing the build process for errors or typo's and build the kernel again using the above steps.

Building the iSCSI Target software

Go to http://iscsitarget.sourceforge.net/ and get the url for downloading the iSCSI enterprise target software package with wget. To download, go to the /root/build directory and give the command (using the url you found, or this url)

```
SAN1:~/build/ # wget http://heanet.dl.sourceforge.net/sourceforge/iscsitarget/iscsitarget-
0.4.13.tar.gz
```

When download has finished, unpack the tarball and go into the directory using

```
SAN1:~/build/ # tar -xvzf iscsitarget-<version>.tar.gz
SAN1:~/build/ # cd iscsitarget-<version>
```

specify what kernel-source to use when compiling the drivers

```
SAN1:~/build/iscsitarget-<version> # export KERNELSRC=/usr/src/linux-<version>
```

Build the kernel-modules using the command

```
SAN1:~/build/iscsitarget-<version> # make
```

If all goes well, no errors should occur and the build process will end with the line

```
make[1]: Leaving directory `/usr/src/linux-<version>`
```

now install the drivers using the command

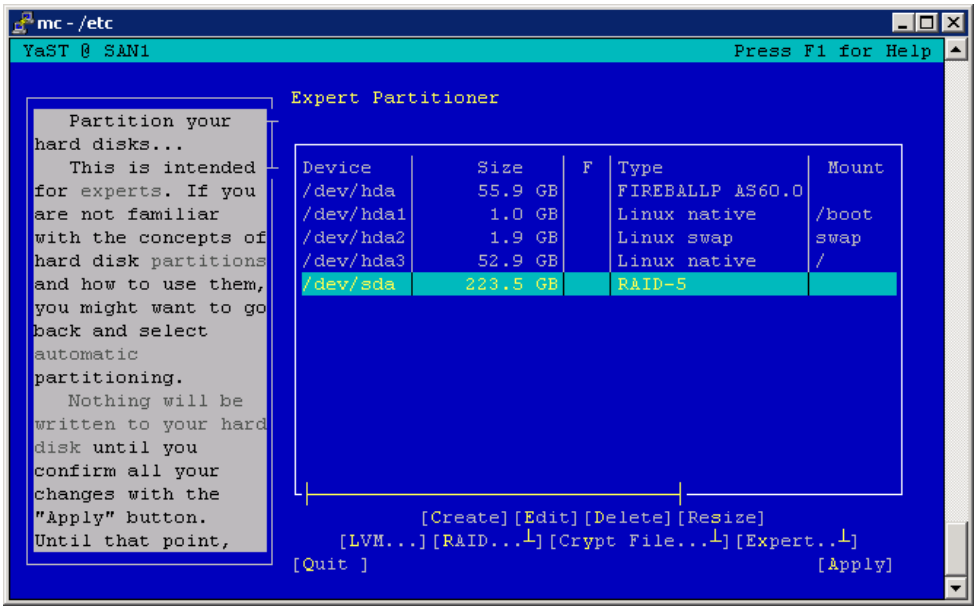
```
SAN1:~/build/iscsitarget-<version> # make install
```

The build process is now complete, and the configuration of the target software can begin.

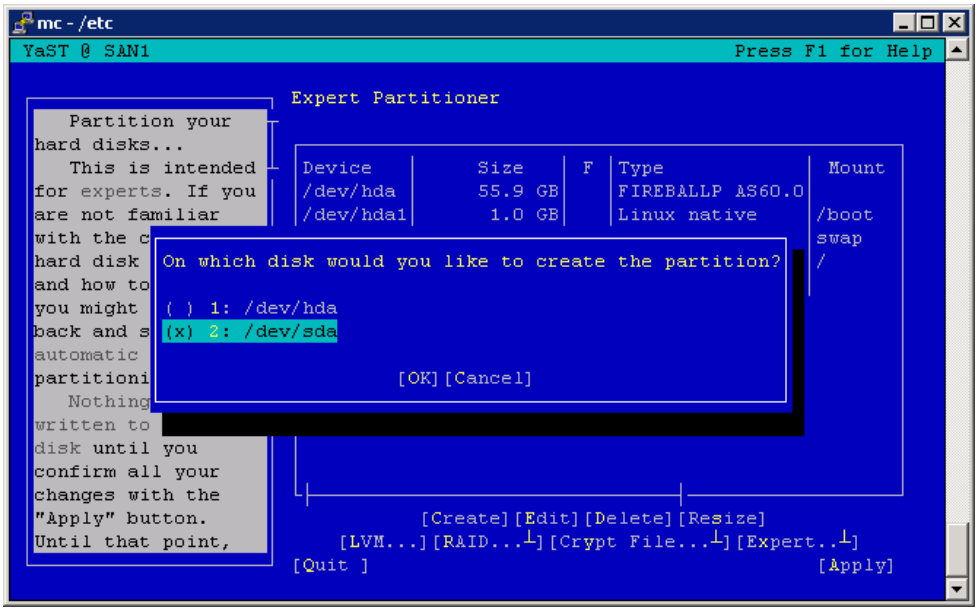
Configuring the file systems to export

If you wish to export multiple file systems or export file systems to multiple hosts rather than 1 big file system for 1 host, we have to create logical volumes on the raid array. You could choose to create standard partitions to export as iSCSI partitions, but with LVM you can add space or re-assign space on-the-fly without having to re-partition your disks.

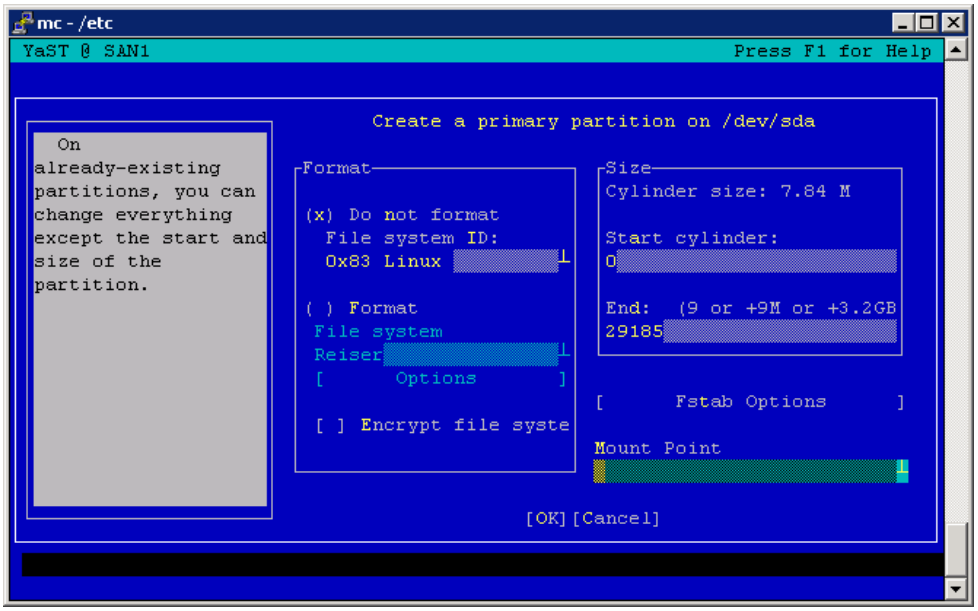
Open yast, and choose the menu “system → partitioner”



Create a new partition on your array using the “create” option, be sure to select the correct drive for adding the volume. Select the partition to be a primary partition.



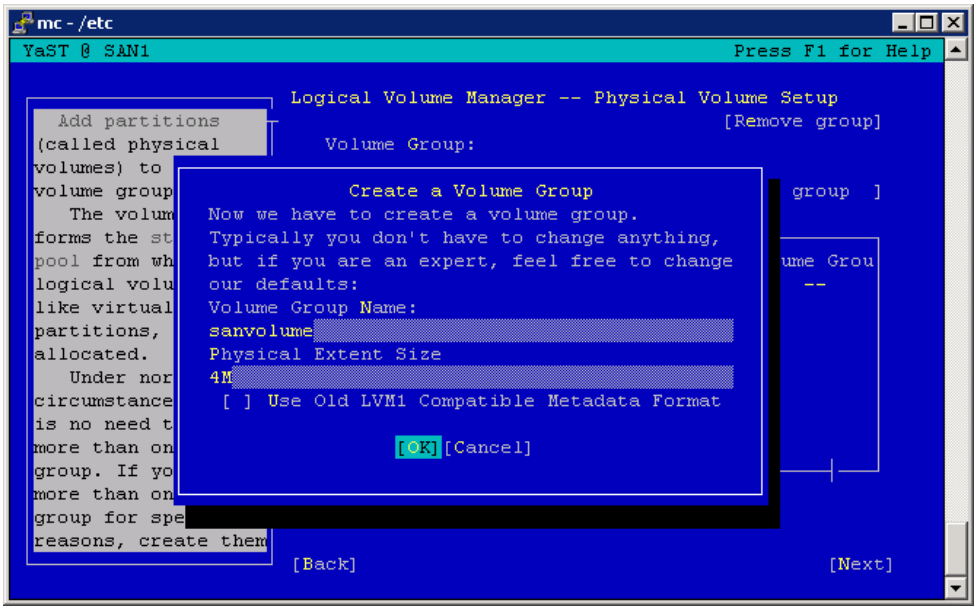
In the screen containing the options to create the partition select the
“Do not format” option, and select the “0x8d Linux LVM” partition type.
Leave the start and end selections as proposed, clear the mount point and select “ok”.



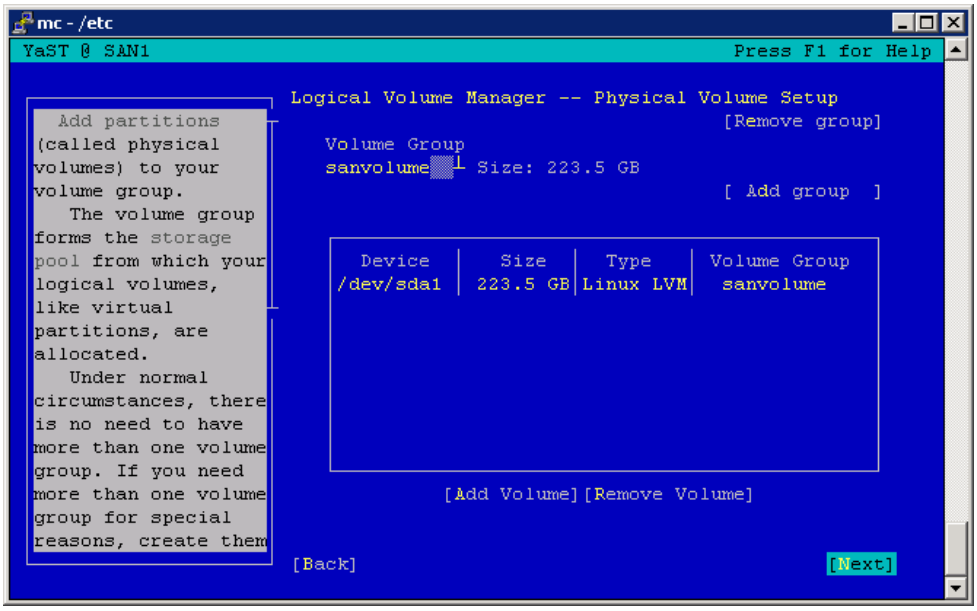
If you have more raid sets you would like to add to the LVM system, create partitions of the type LVW on these arrays too. You can choose to use all raid sets in a single volume group or to create different groups.
I advise you to create a single group for each array, otherwise the volumes will be spanned over multiple arrays creating a dangerous situation comparable with a RAID0 spanning set.

When the partition is created, select the option “LVM” from the bottom of the screen.

When Yast asks for a volume group name enter a new name like “sanvolume” or leave the default name “system”.

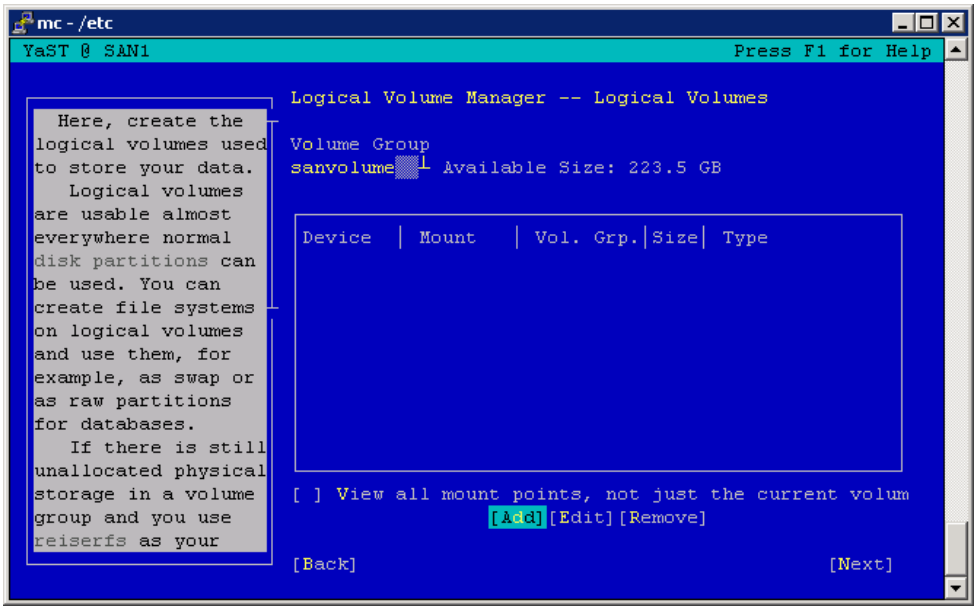


In the next screen select the “add volume” option to add the partition to the volume group. If there are more partitions to add, add them all here.

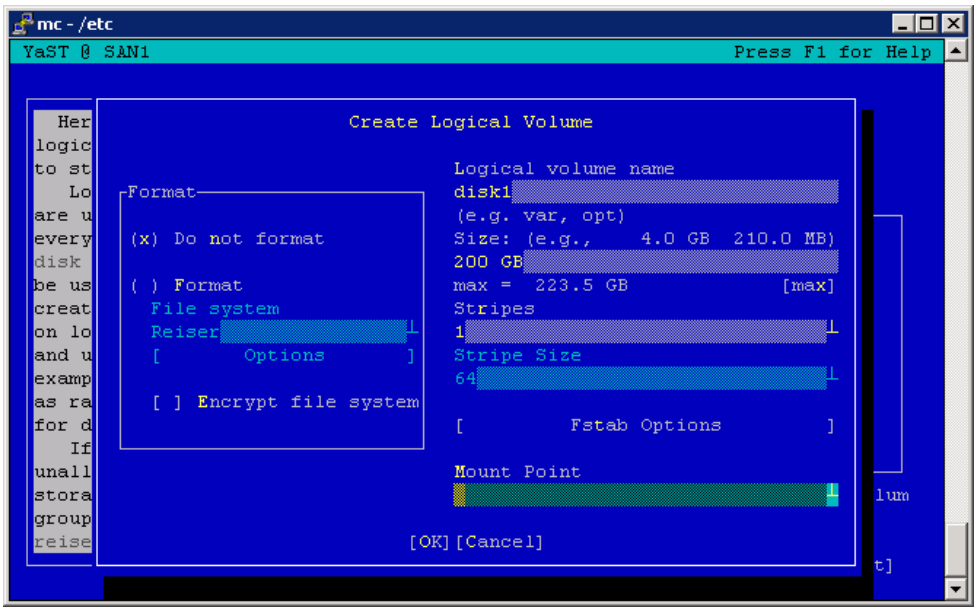


Select “next” to continue.

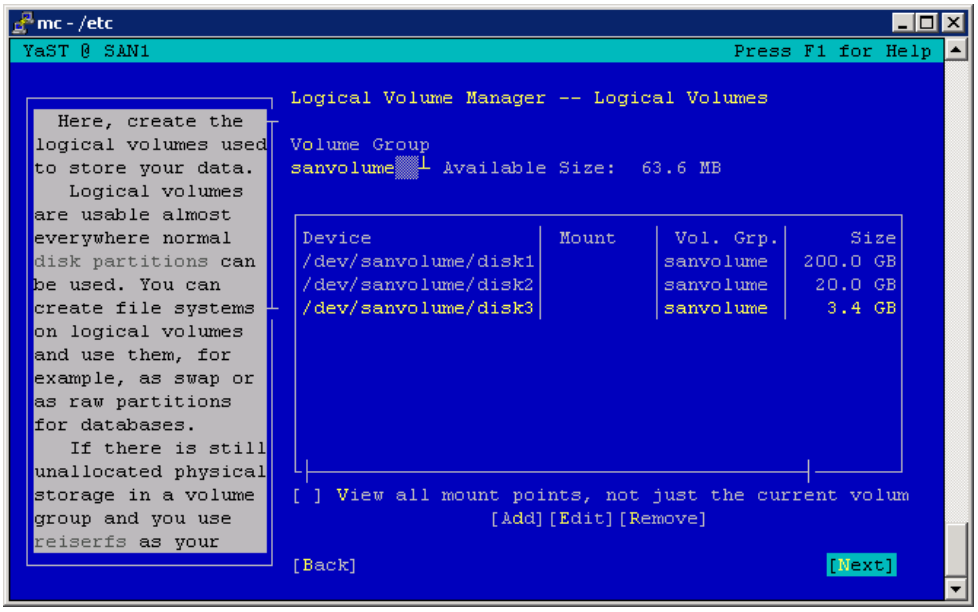
In the next screen you can create logical volumes
To prevent erasing or modifying the current and mounted volumes on the system I suggest you turn off the visibility of the active mount points with disabling the option “View all mount points, not just the current volumes”
In the top of the screen select the volume group you wish to create a volume in.



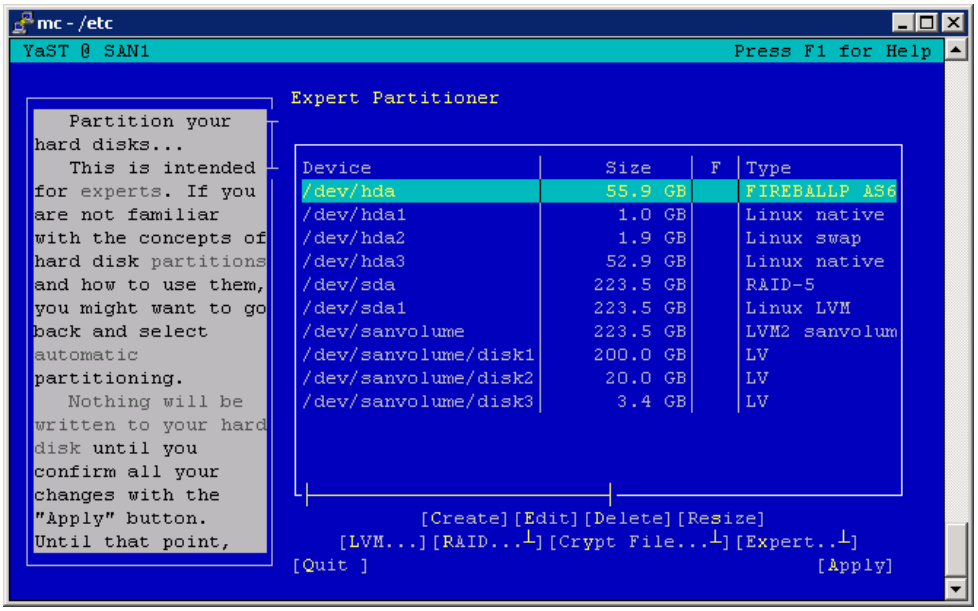
Select “add” to create a new logical volume.
Again select the option “Do not format” because the iSCSI initiator will do this with its own file system. Fill in the Logical volume name and the size of the volume. The size can be altered later when space is needed.
Clear the Mount point and select “ok” to continue.



You can create as many logical volumes as you want, until all space is filled up off course.
When you are done, select “next” from the volume manager screen



As you can see the Logical volumes are accessible as normal block devices.



Click “apply” to commit the changes to the system.
When it looks like nothing is happening don’t be scared, this process can take a while.
When all is done, exit the partitioner with the option “Quit” and exit Yast.

Configuring the iSCSI Target software

The first step in configuring the iSCSI Target is to prepare the host file for using a different domain name for the SAN network. Open the file /etc/hosts for editing using vi or mcedit

Change the line containing the SAN ip to another domain-name

```
192.168.10.1    SAN1.testnet.local
192.168.2.20    SAN1.testnet.local SAN1
```

Changes to

```
192.168.10.1    SAN1.sannet.local
192.168.2.20    SAN1.rusman.local SAN1
```

This change is to overcome broadcast traffic and iSCSI mounts from the normal “users” network.

Copy the sample file ietd.conf to the /etc directory using the command

```
SAN1:~/build/iscsitarget-<version> # cp etc/ietd.conf /etc/ietd.conf
```

Open the file for editing using vi or mcedit

First thing to change is the target name.
This name is built of the following items :

```
iqn.yyyy-mm.<reversed domain name>[:identifier]
```

So the domain name we just entered in the hostname has to be entered here also, but in reversed order : “sannet.local” becomes “local.sannet”.
The identifier can be anything you like, but for clarity use the name of the LVM logical volume that is exported in this target in the name.

Uncomment the LUN line, and specify the LVM volume you wish to export with this target.
You can also specify an alias, but this is optional.

```
Target iqn.2001-04.local.sannet:storage.disk1
        # Users, who can access this target. The same rules as for discovery
        # users apply here.
        # Leave them alone if you don't want to use authentication.
        #IncomingUser joe secret
        #OutgoingUser jim 12charpasswd
        # Logical Unit definition
        # You must define one logical unit at least.
        # Block devices, regular files, LVM, and RAID can be offered
        # to the initiators as a block device.
        Lun 0 Path=/dev/sanvolume/disk1,Type=fileio
        # Alias name for this target
        Alias sandisk1
        # various iSCSI parameters
        # (not all are used right now, see also iSCSI spec for details)
```

Create such a section for each target you will export on your SAN system.
Save the file and exit the editor

Next, open the file /etc/iscsi.conf for editing
Change any option as needed, or leave the standard options here for defaults.
Save the file and exit the editor

Now you are ready to start the iSCSI target daemon.
Issue the command

```
SAN1:/etc/ # /etc/init.d/iscsi-target start
```

Check the log file /var/log/messages for errors or type-mismatches with the command

```
SAN1:/etc/ # tail /var/log/messages
```

If no errors occur, this should look like this:

```
Feb 27 13:29:07 SAN1 kernel: iSCSI Enterprise Target Software - version 0.4.13
Feb 27 13:29:07 SAN1 kernel: iotype_init(90) register fileio
Feb 27 13:29:07 SAN1 kernel: iotype_init(90) register nullio
```

To automatically start the iSCSI target when the machine boots, open yast and choose
"system → System Services (Runlevel): Services"
At the iSCSI-Target service, choose "enable".
Save this, and exit yast.

Congratulations, you have successfully configured the iSCSI target on your linux box.
You can now configure the ISCSI initiator software on you Linux or Netware box to point at this machine.

This setup has been tested for 48 hours with the Novell Testsuite for certification of hardware and drivers, and has passed this test with success. The average load of the GBit network cards used for interconnecting the SAN and servers was about 90% and the CPU load of the 800 MHz Pentium III used in the SAN device was about 2 %.

Coming soon : Part 2 – Configuring the initiator software on Netware, Linux and windows servers.