Xsan for everyone

This a a guide on how to set up Xsan and use it without all the necessary equipment, potentially it can be used to simulate a live environment to test change procedures before implementing on a production system. It can be used to get familiar with the GUI and command line tools.

Getting the Software

Sign up for a free Mac Dev Center account at developer.apple.com
Log in to the Developer Downloads section
Download the Latest Version of Xsan and the Developer Serial Number

Network Configuration

Open a Terminal and switch user to root (this will just make the process easier)

```
sudo –s
```

Obtain your computer's IP address, easily done from the terminal the following command

```
ipconfig getifaddr interface–name
```

For the remainder of this document I will be using the IP Address 10.1.0.10

Edit your /etc/hosts file and add an entry for your computer

```
pico /etc/hosts

##
# Host Database
#
# localhost is used to configure the loopback interface
# when the system is booting.  Do not change this entry.
##
127.0.0.1       localhost
255.255.255.255 broadcasthost
::1             localhost
fe80::1%lo0     localhost
10.1.0.10       MacBook–002332bf12fa.private
```

Optionally you can configure the local DNS server on your computer

Edit /etc/named.conf and add the entries for the required zone files

```
pico /etc/named.conf

zone "MacBook–002332bf12fa.private." {
        type master;
        file "db.MacBook–002332bf12fa.private.";
        allow–transfer {none;};
        allow–update {none;};
};

zone "10.0.1.10.in–addr.arpa." {
        type master;
        file "db.10.0.1.10.in–addr.arpa.";
        allow–transfer {none;};
        allow–update {none;};
};
```

Create the zone files required

```
pico /var/named/db.1.0.1.10.in-addr.arpa.

$TTL 10800
10.0.1.10.in-addr.arpa. IN SOA MacBook-002332bf12fa.private. localhost.localdomain (
        2010090600      ;Serial
        86400           ;Refresh
        3600            ;Retry
        604800          ;Expire
        3600            ;Negative caching TTL
 )

10.0.1.10.in-addr.arpa. IN  NS MacBook-002332bf12fa.private.
10.0.1.10.in-addr.arpa. IN  PTR MacBook-002332bf12fa.private.


pico /var/named/db.MacBook-002332bf12fa.private

$TTL 10800
MacBook-002332bf12fa.private. IN SOA MacBook-002332bf12fa.private. localhost.localdomain
(
        2010091200      ;Serial
        86400           ;Refresh
        3600            ;Retry
        604800          ;Expire
        3600            ;Negative caching TTL
 )

MacBook-002332bf12fa.private. IN  NS MacBook-002332bf12fa.private.
MacBook-002332bf12fa.private. IN  A 10.1.0.10
```

Generate a `rndc.key` file for bind

```
rndc-confgen -a -c /etc/rndc.key
```

Edit the launchd item to enable DNS

```
defaults write /System/Library/LaunchDaemons/org.isc.named Disabled -bool false
```

Start DNS

```
launchctl load /System/Library/LaunchDaemons/org.isc.named.plist
```

Set your DNS server for the network interface in System Preferences to 127.0.0.1

In a terminal use the host command to verify your DNS resolves correctly

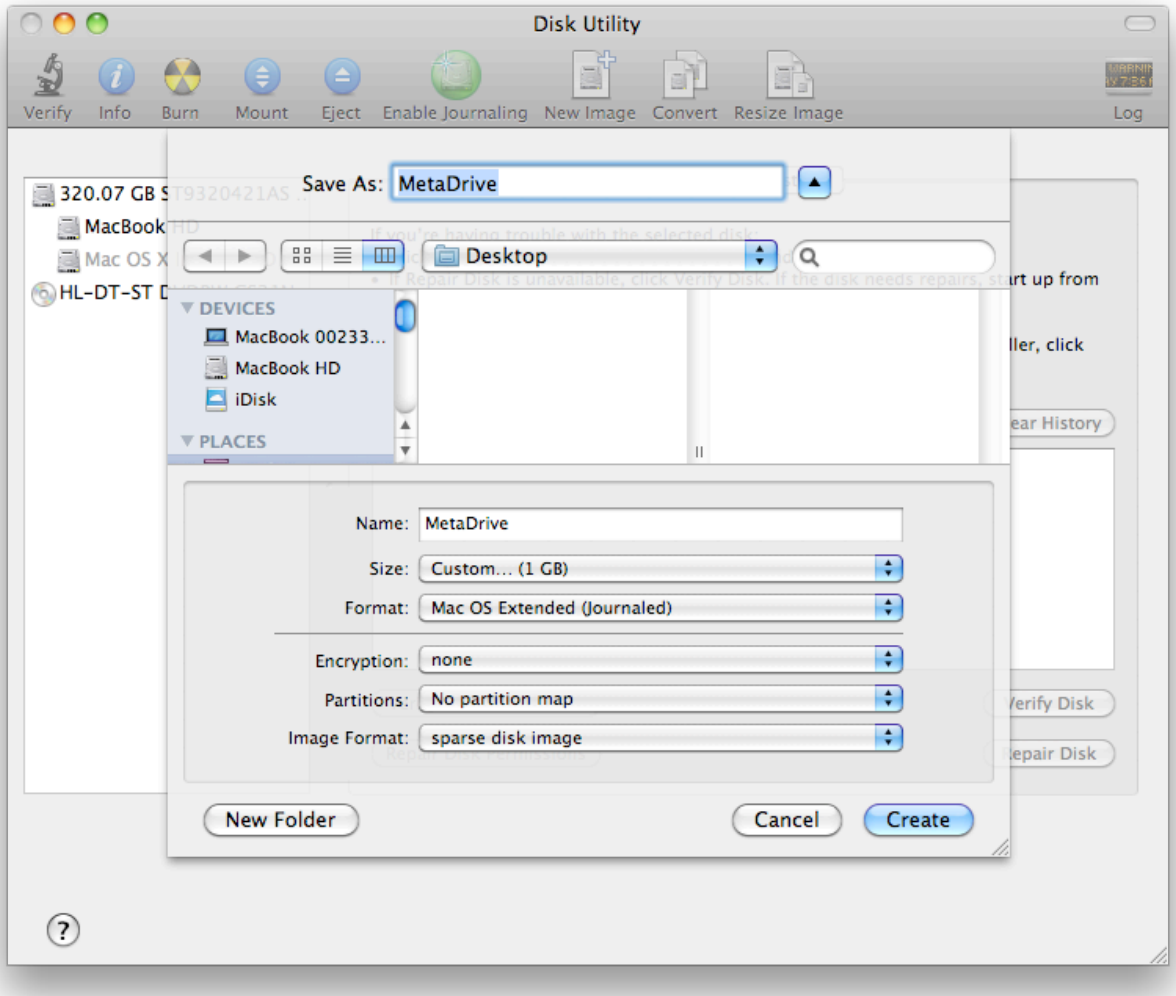
```
host 10.1.0.10
host MacBook-002332bf12fa.private
```

Set your computer's hostname to match your DNS or hosts record

```
scutil --set HostName MacBook-002332bf12fa.private
```

Preparing the 'LUNs'

Open Disk Utility and create the sparse images to use as your LUNs, note the settings used in the screenshot below:



I recommend the following sizes for this example:

MetaDrive	1GB
JournalDrive	1GB
VideoDrive	8GB
AudioDrive	4GB
DataDrive	4GB

Simply create one of each and then duplicate them in the Finder, you will need:

1x	MetaDrive
1x	JournalDrive
8x	VideoDrive
4x	AudioDrive
4x	DataDrive

Installing and Licensing Xsan

Install Xsan.

Launch Xsan Admin, you will be greeted with the Xsan Setup Assistant.
When prompted, select Configure new SAN.
Name your SAN, the default (My Xsan) will suffice.
Add and authenticate your computer.
Add the Serial Number from the Document you downloaded from the Mac Dev Center:

XSAN-020-000-N-6HF-RJY-FCM-VBS-G2S-•••-•

Choose your metadata network, this should be the network containing the manual address you set earlier (10.1.0.10/16)

When prompted, choose Create one or more volumes later

Exit the setup assistant and quit Xsan Admin.

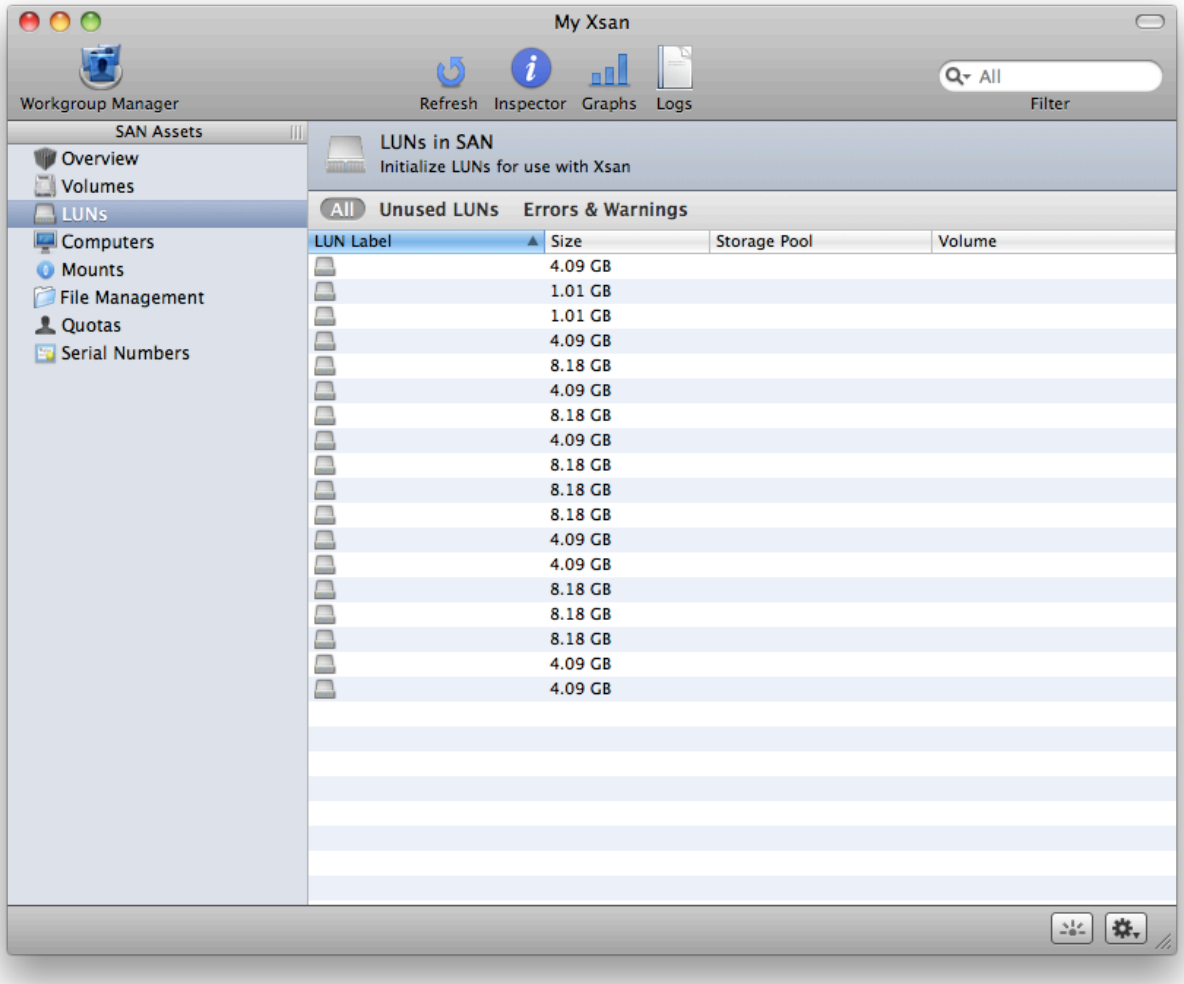
Creating a Volume via the GUI

Mount all of your sparse images in the Finder, be patient, wait for each image to mount before mounting the next.

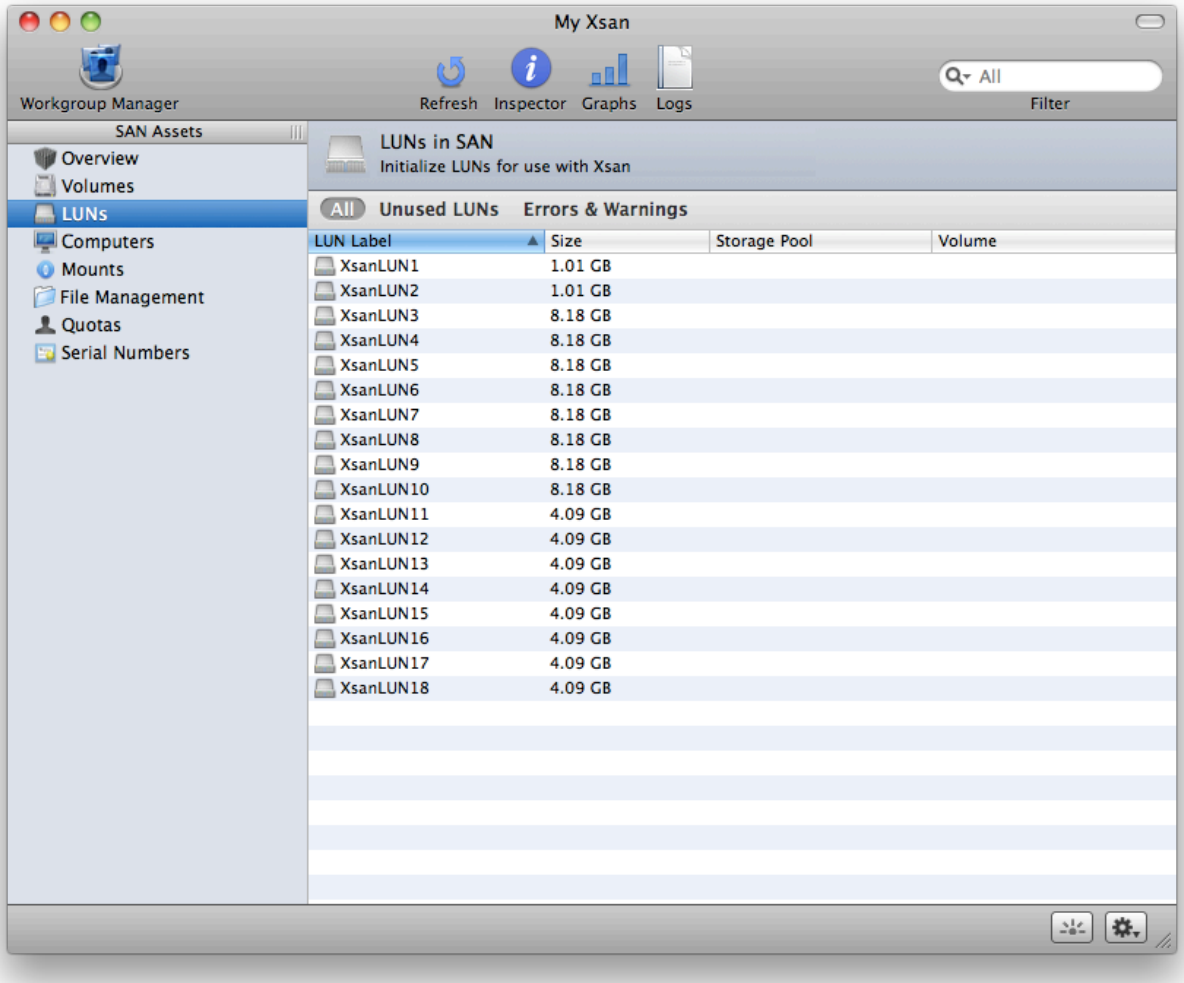
Tip: If you mount the images in a sequential order (Meta, Journ, Video, Audio, Data) it will make them easier to label from the command line.

Note: When doing this if you mount any additional volumes, LUNs tend to disappear in Xsan Admin, so ensure you create and mount all images before proceeding

Open Xsan Admin and click on LUNs, you will see all of the mounted images

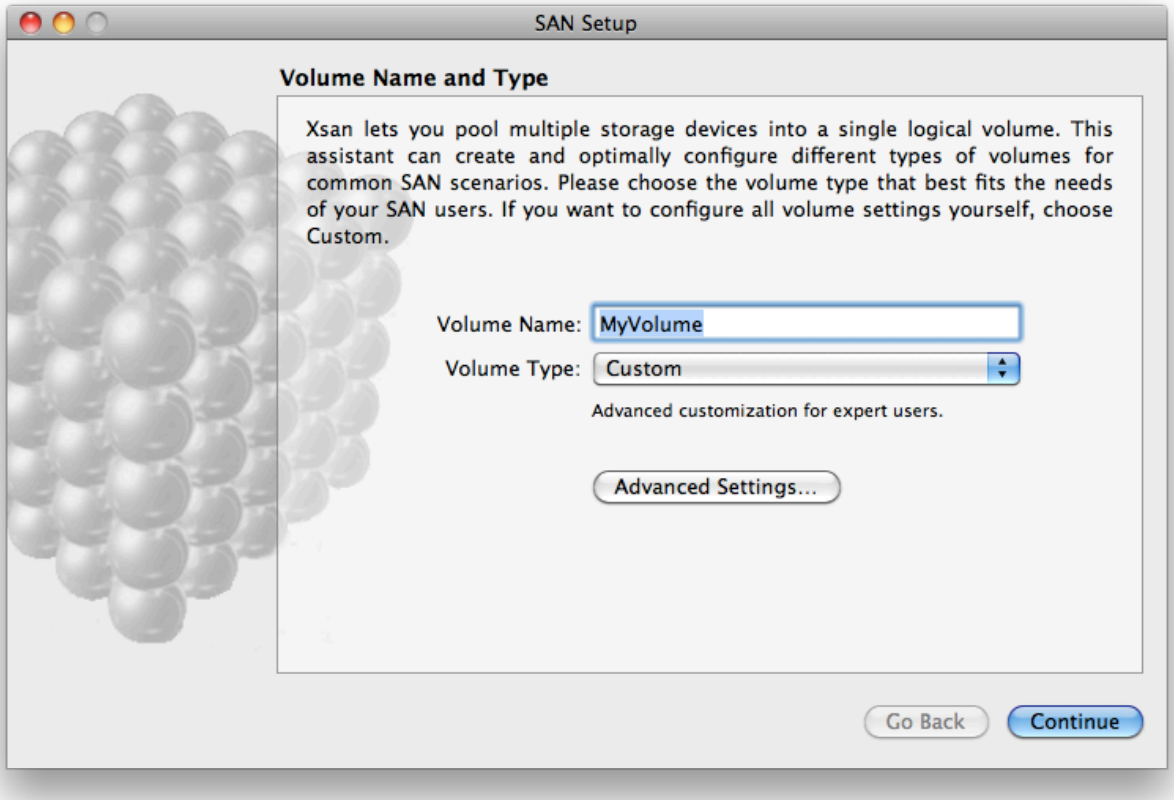


You may now proceed and label all the LUNs

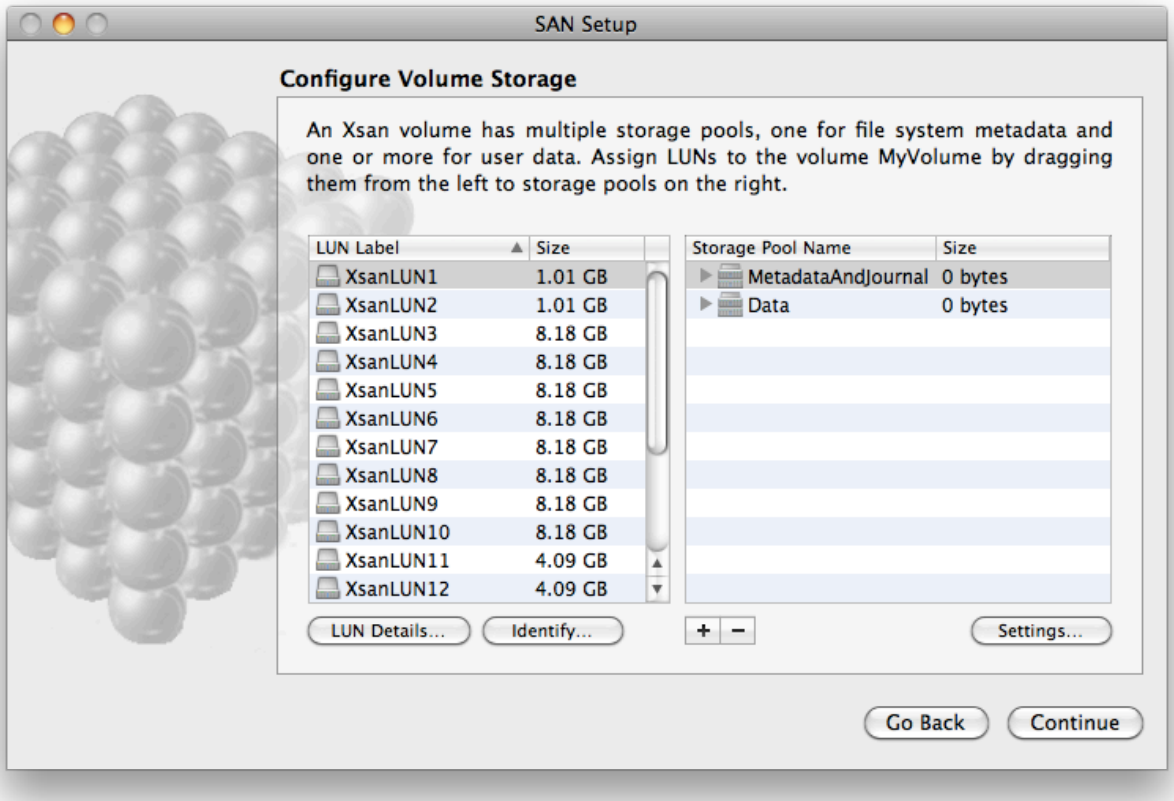


Click on Volumes and click the Add Volume button, proceed through the Setup Assistant.

In the SAN Setup window, name your volume (MyVolume) and select a type of Custom and click Continue.

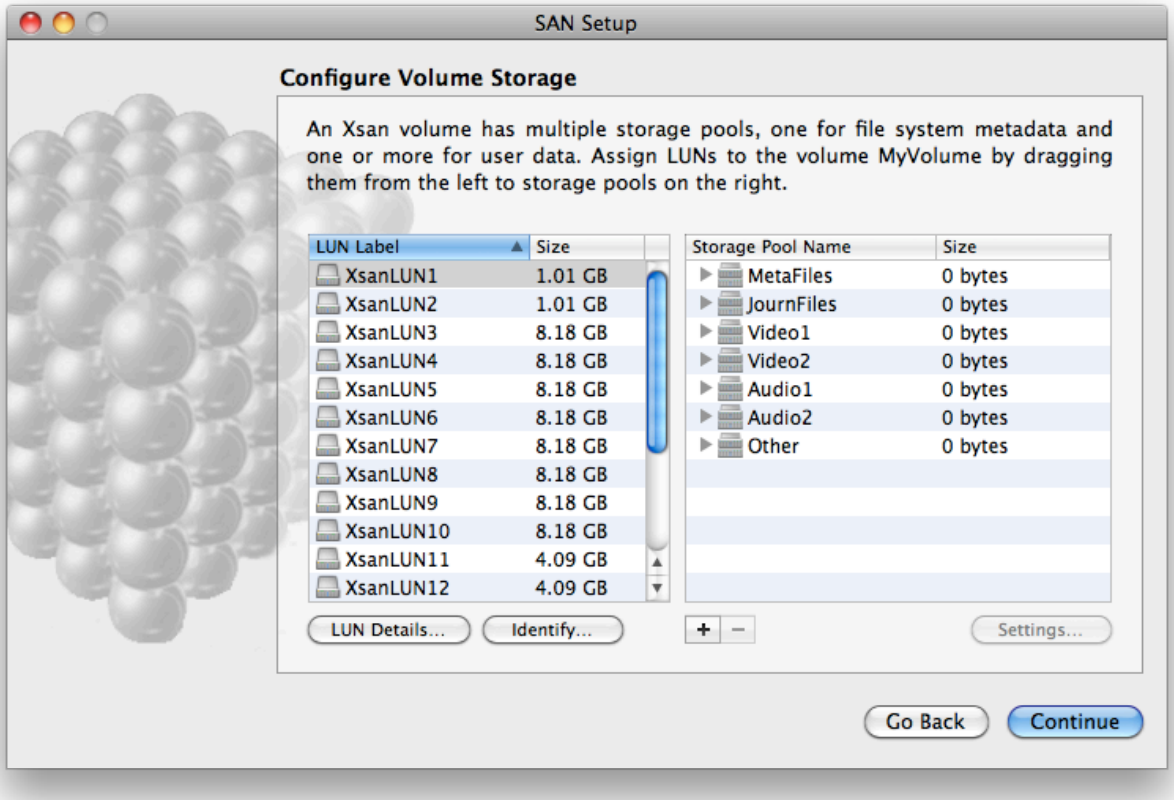


Select 'MetadataAndJournal' and click 'Settings...', adjust the Stripe Breadth to 64 blocks and change the name to MetaFiles. Rename the Data pool to JournFiles and adjust its Stripe Breadth to 64 blocks.



Create 5 more Storage Pools as follows:

Name	Stripe Breadth
Video1	43 blocks (688 KB)
Video2	51 blocks (816 KB)
Audio1	4 blocks (64 KB)
Audio2	15 blocks (240 KB)
Other	64 blocks (1MB)



Note: There are 2 methods for optimising your volume for performance, one is to match the frame size for the files you are writing out, the other is to match the write bursts for your RAID units (typically 1 MB).
The example we are creating is has 4 Storage pools tuned for specific file types and the others are matched to the RAID unit.

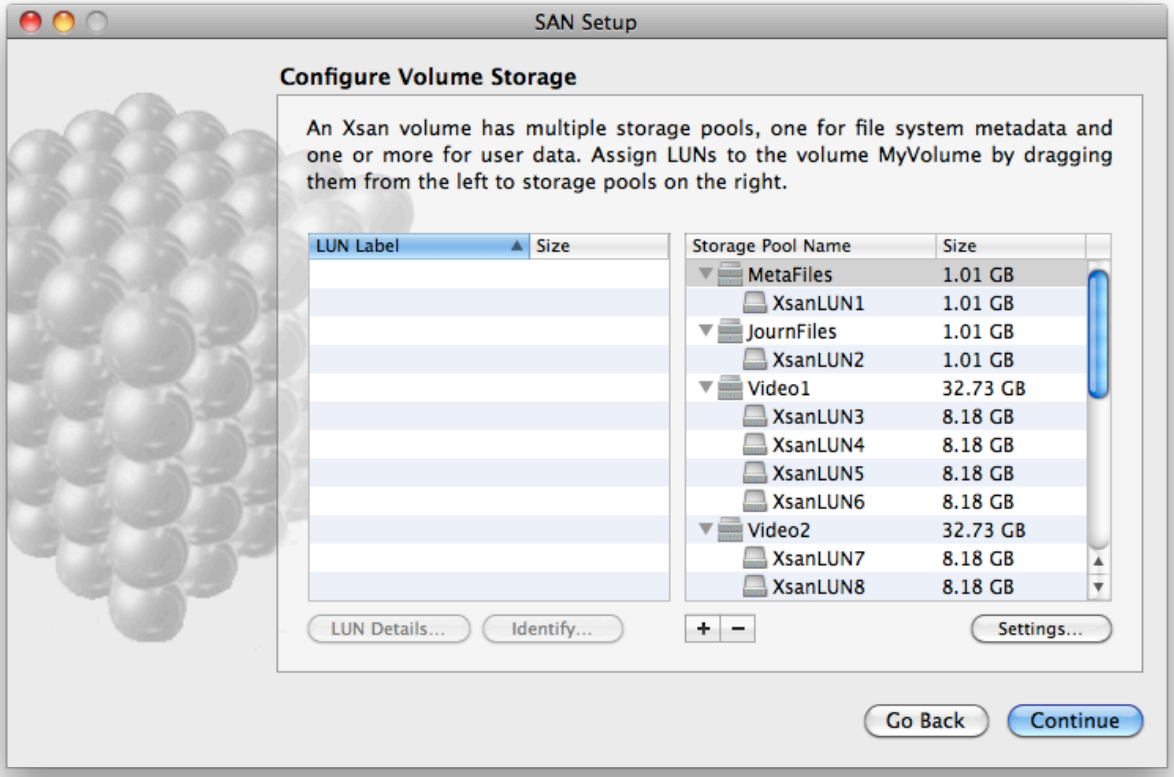
To match the RAID unit:
1024 KB / (Block Allocation Size KB * No of LUNs in Storage Pool) = Stripe Breadth
eg. MetaFiles: 1024 KB / (16 KB * 1) = 64 blocks
JournFiles: 1024 KB / (16 KB * 1) = 64 blocks
Other: 1024 KB / (16 KB * 4) = 16 blocks

To match the file frame size frame size:
Frame Size KB / Block Allocation Size = Stripe Breadth
eg. Video1 (NTSC): 688 KB / 16 KB = 43 blocks
Video2 (PAL): 816 KB / 16 KB = 51 blocks
Audio1 (NTSC): 64 KB / 16 KB = 4 blocks
Audio2 (PAL): 240 KB / 16 KB = 15 blocks

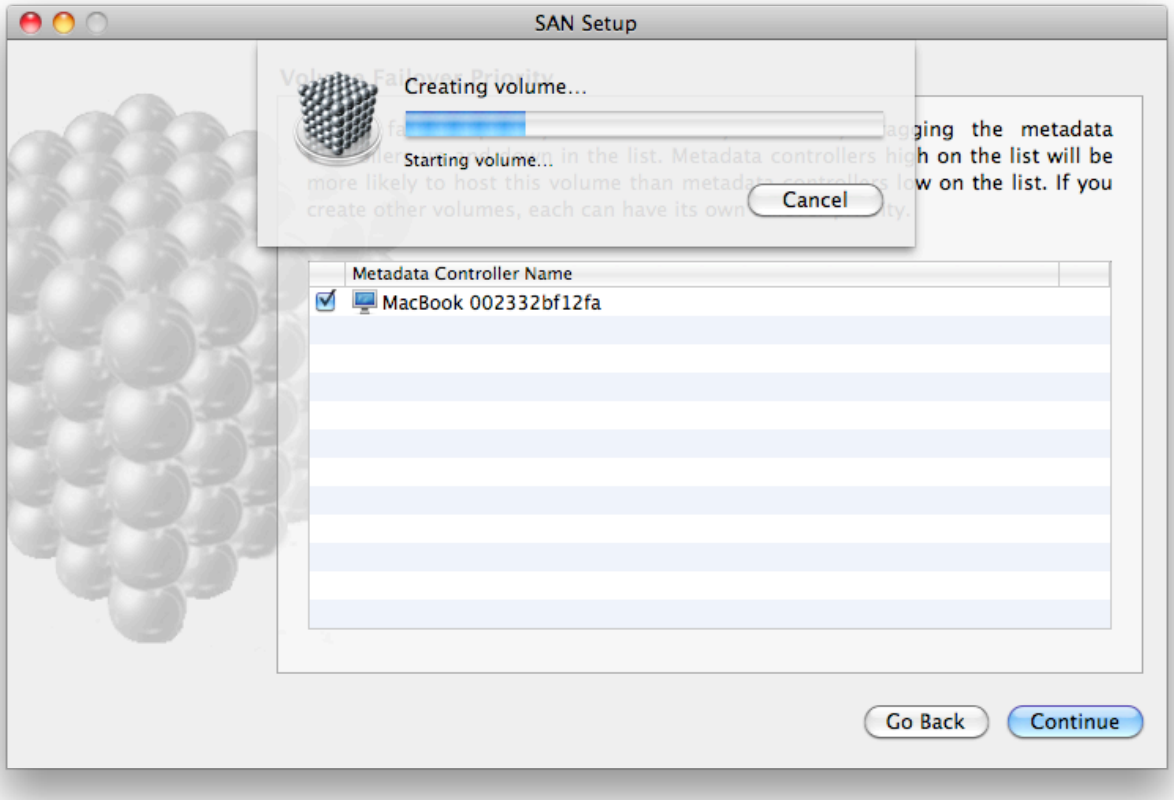
Tip: When using the pre-confingured volume settings in Xsan 2, it pays to adjust the Stripe Breadth of the MetadataAndJournal storage pool.

Add your LUNs to the Storage pools, as follows:

MetFiles	1 LUN
JournFiles	1 LUN
Video1	4 LUNs
Video2	4 LUNs
Audio1	2 LUNs
Audio2	2 LUNs
Other	4 LUNs



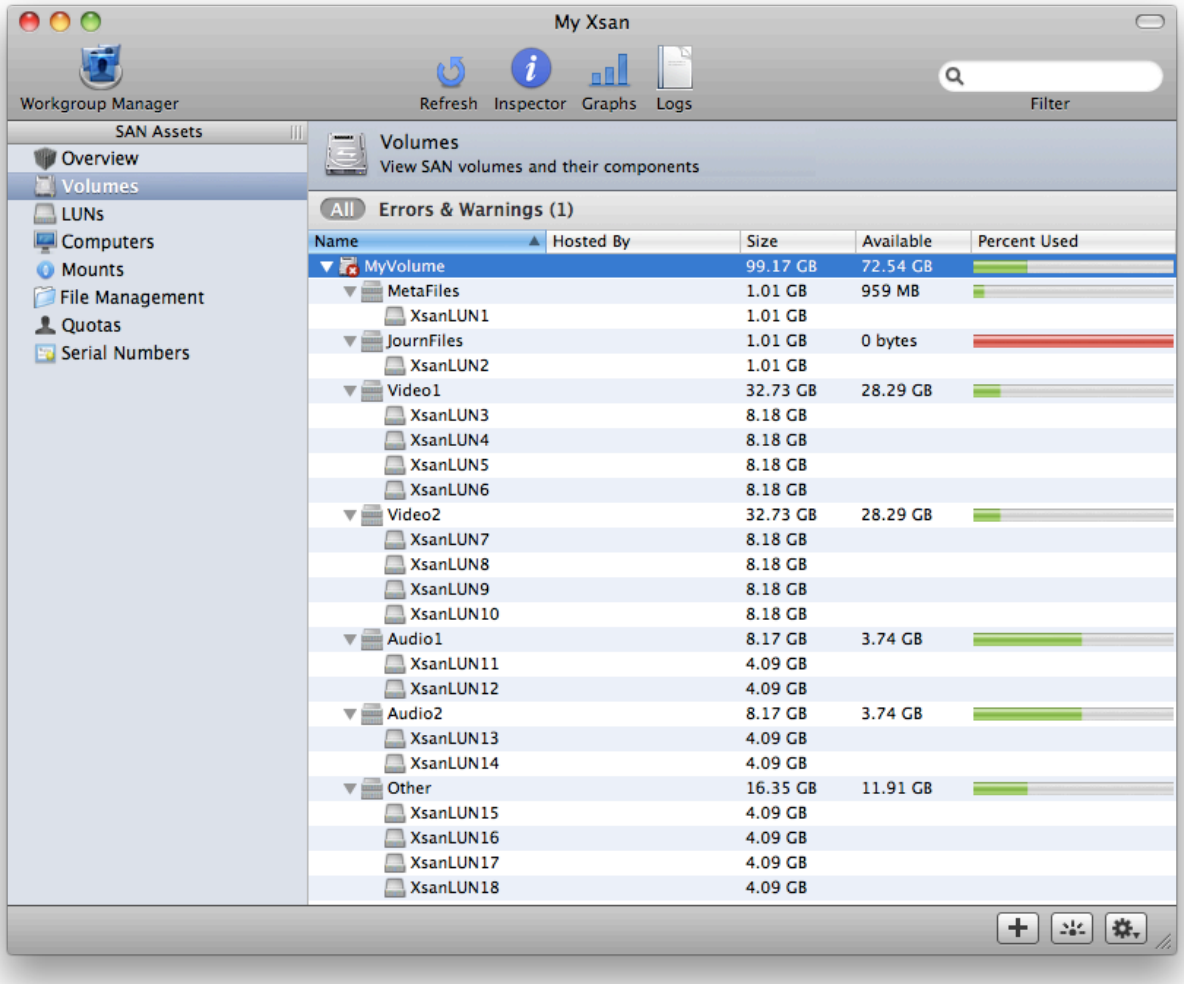
Ensure your computer is selected as a Metadata Controller and continue to create the volume.



When complete your volume will appear on the desktop!

Separating the Journal from Metadata

Stop the Volume in Xsan admin



Note: If you are doing this on a production system with multiple MDCs you should shut down the other MDCs at this point

Open a Terminal and type the following command to commit the journal file for the volume:

```
cvfsck -j MyVolume
```

The output should read:

```
Re-creating File System cvfsck Directory '/Library/Filesystems/Xsan/data/MyVolume/
trace'.

Attempting to acquire arbitration block... successful.


File System Journal Recovery completed successfully.
```

Perform a filesystem check using:

```
cvfsck -vw MyVolume
```

The output should read:

```
BUILD INFO:
#!@$ Server Revision 3.5.0 Build 7443 Branch branches_35X (412.3)
```

```
#!@$ Built for Darwin 10.0 i386
#!@$ Created on Mon Dec  7 12:52:39 PST 2009


Created directory /tmp/cvfsck20910a for temporary files.

Attempting to acquire arbitration block... successful.

Creating MetaFiles allocation check file.
Creating JournFiles allocation check file.
Creating Video1 allocation check file.
Creating Video2 allocation check file.
Creating Audio1 allocation check file.
Creating Audio2 allocation check file.
Creating Other allocation check file.

Recovering Journal Log.

Super Block information.
  FS Created On               : Thu Oct  7 18:31:45 2010
  Inode Version               : '2.5'
  File System Status          : Clean
  Allocated Inodes            : 5120
  Free Inodes                 : 5103
  FL Blocks                   : 1
  Next Inode Chunk            : 0xd06
  Metadump Seqno              : 0
  Restore Journal Seqno       : 0
  Windows Security Indx Inode : 0xa
  Windows Security Data Inode : 0xb
  Quota Database Inode        : 0xc
  ID Database Inode           : 0x10
  Client Write Opens Inode    : 0xd

Stripe Group MetaFiles                     (  0) 0xf1c0 blocks.
Stripe Group JournFiles                    (  1) 0xf1c0 blocks.
Stripe Group Video1                        (  2) 0x1e7a7c blocks.
Stripe Group Video2                        (  3) 0x1e7a34 blocks.
Stripe Group Audio1                        (  4) 0x79c98 blocks.
Stripe Group Audio2                        (  5) 0x79c7a blocks.
Stripe Group Other                         (  6) 0xf3900 blocks.

Building Inode Index Database 5120 (100%).

Verifying NT Security Descriptors

Verifying Free List Extents.

Scanning inodes 5120 (100%).

Sorting extent list for MetaFiles pass 1/1
Updating bitmap for MetaFiles extents 12 (100%).

Checking for dead inodes 5120 (100%).

Checking directories 3 (100%).

Scanning for orphaned inodes 5120 (100%).

Verifying link & subdir counts 5120 (100%).

Checking free list. 5120 (100%).
Checking pending free list.

Checking Arbitration Control Block.

Checking MetaFiles allocation bit maps (100%).
Checking JournFiles allocation bit maps (100%).
Checking Video1 allocation bit maps (100%).
Checking Video2 allocation bit maps (100%).
Checking Audio1 allocation bit maps (100%).
Checking Audio2 allocation bit maps (100%).
```

```
Checking Other allocation bit maps (100%).

File system 'MyVolume'. Blocks-6051970 free-6048674 Inodes-5120 free-5103.

File System Check completed successfully.
```

Open the volume configuration file in an editor:

```
pico /Library/Filesystems/Xsan/config/MyVolume.cfg
```

Scroll down to the Stripe Group definition section and adjust the MetaFiles and JournFiles stripe groups as below (changes are in bold)

```
# ****************************************************************************
# A stripe section for defining stripe groups.
# ****************************************************************************

[StripeGroup "MetaFiles"]
Status Up
Exclusive Yes
Metadata Yes
Journal No
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 64
Node "XsanLUN1" 0

[StripeGroup "JournFiles"]
Status Up
Exclusive Yes
Metadata No
Journal Yes
# Affinity "JournFil"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 64
Node "XsanLUN2" 0
```

Save the changes, exit the editor and issue the following command to update the volume configuration.

```
cvupdatefs MyVolume
```

The output should read:

```
The following changes have been detected in the configuration
Please review these changes carefully.

Stripe Group Name  Stripe Status  MetaData   Journal
================  =============  ========   =======
MetaFiles          No Change      No Change  Delete
JournFiles         No Change                 Create
Video1             No Change
Video2             No Change
Audio1             No Change
Audio2             No Change
Other              No Change


This will modify the file system "MyVolume".
Are you sure you want to continue? [y/N]
```

Confirm the change, the output should read

```
Attempting to acquire arbitration block... successful.

Flushing journal entries...  done
```

```
Freeing old journal space...
Allocating new journal space...
Flushing buffers...
Updating ICB information...
Updating SuperBlock information...
*Warning*: File system 'MyVolume' was modified.
```

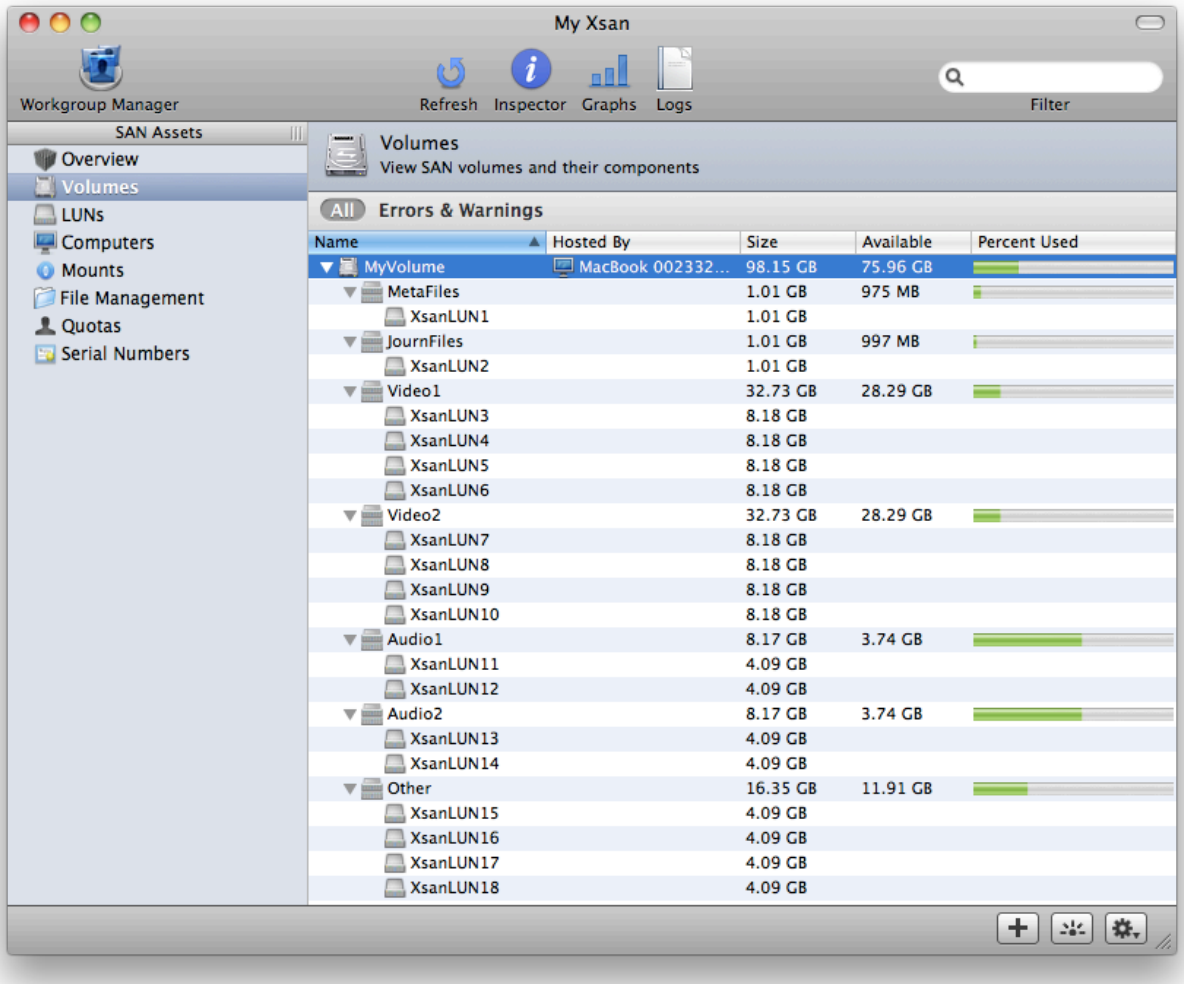
Perform a filesystem check using:

```
cvfsck –vw MyVolume
```

Note: If you are doing this on a production system with multiple MDCs you should start the other MDCs at this point and issue the following command on each of them:

```
cvadmin –e 'disks refresh'
```

Open Xsan Admin and start the volume.



Assigning Affinities for Data Types

Stop the Volume in Xsan Admin

Open a Terminal to edit the Volume configuration file

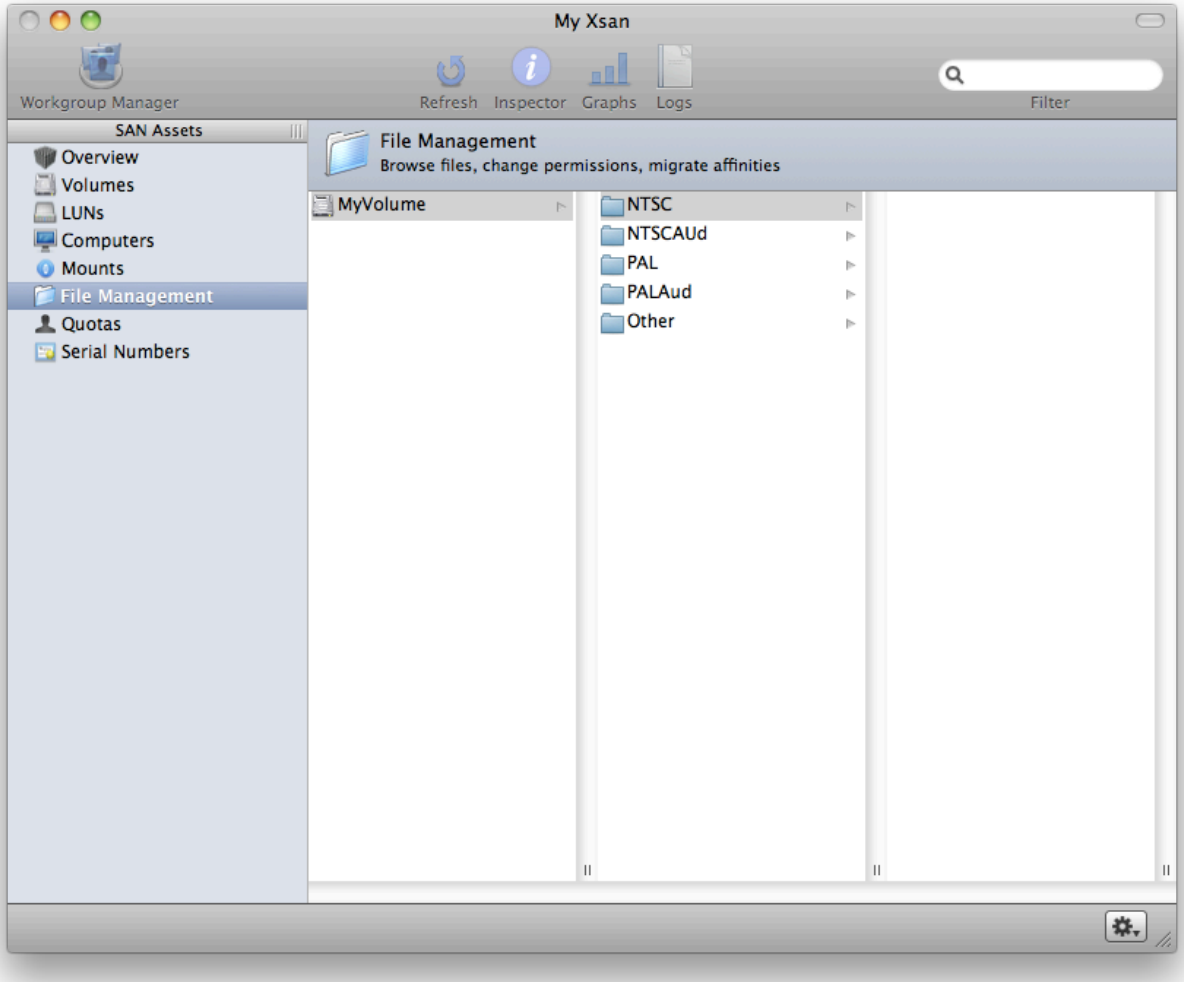
```
pico /Library/Filesystems/Xsan/config/MyVolume.cfg
```

Scroll down to the Stripe Group Definition for Video1 and adjust as below (changes are in bold)

Note: Affinity tags can't be any longer than 8 characters

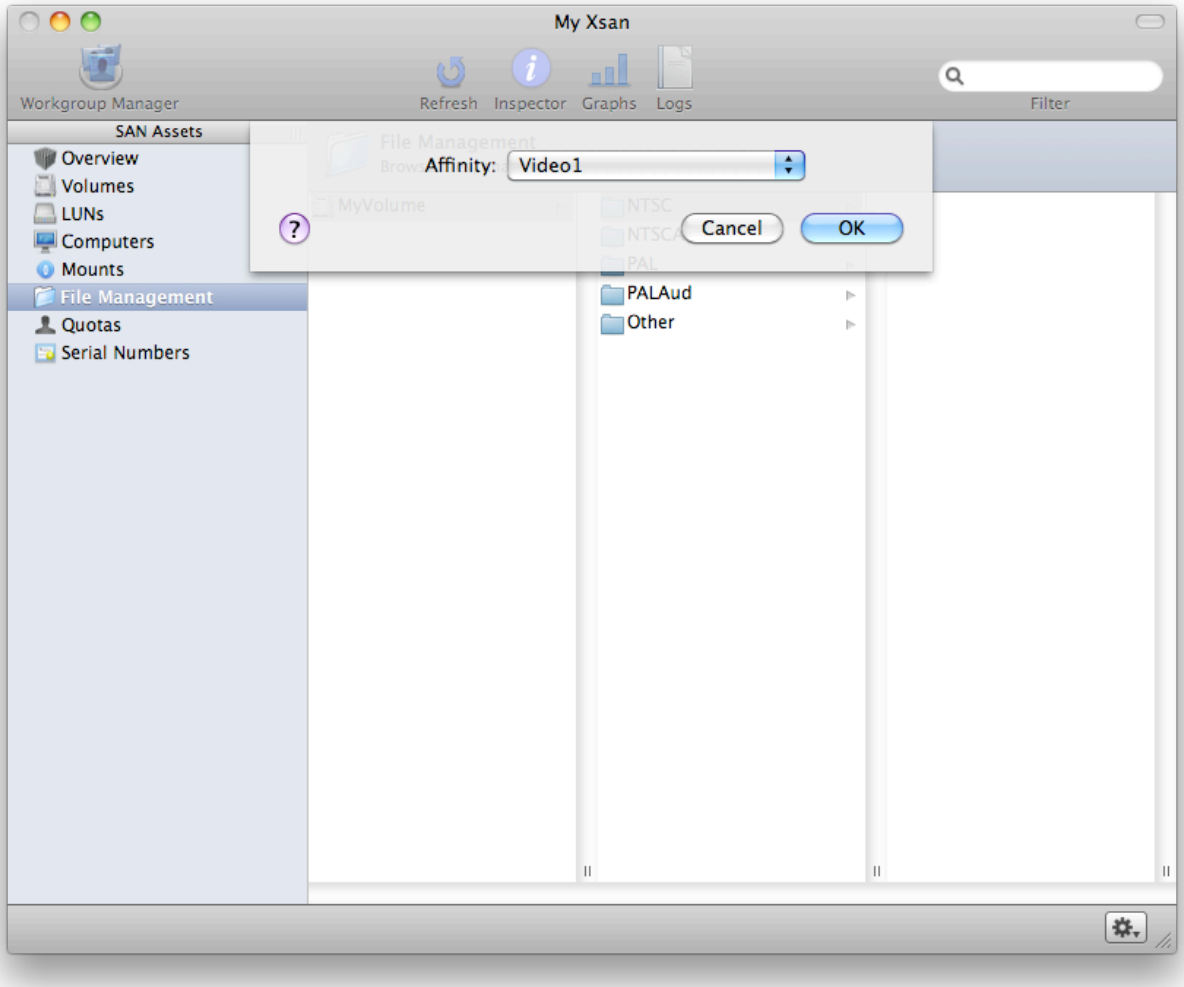
```
[StripeGroup "Video1"]
Status Up
Exclusive Yes
Metadata No
Journal No
Affinity "NTSC"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 43
Node "XsanLUN3" 0
Node "XsanLUN4" 1
Node "XsanLUN5" 2
Node "XsanLUN6" 3

[StripeGroup "Video2"]
Status Up
Exclusive Yes
Metadata No
Journal No
Affinity "PAL"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 51
Node "XsanLUN7" 0
Node "XsanLUN8" 1
Node "XsanLUN9" 2
Node "XsanLUN10" 3

[StripeGroup "Audio1"]
Status Up
Exclusive Yes
Metadata No
Journal No
Affinity "NTSCAud"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 4
Node "XsanLUN11" 0
Node "XsanLUN12" 1

[StripeGroup "Audio2"]
Status Up
Exclusive Yes
Metadata No
Journal No
Affinity "PALAud"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 15
Node "XsanLUN13" 0
Node "XsanLUN14" 1
```

Open Xsan Admin start the volume, mount it and select File Management.
Create 5 folders as below (NTSC, NTSCAud, PAL, PALAud)



Select each folder in turn and assign the affinity by clicking the gear in the lower right corner of the window, they should be:

NTSC = Video1
NTSCAud = Audio1
PAL = Video2
PALAud = Audio2
Other = Other



Congratulations, you now have configured a highly tuned Xsan volume!

Creating a Volume via the Command Line

Labelling the 'LUNs'

Create a cvlabel file, listing the decive(s), in the Terminal, type:

```
cvlabel –c > /tmp/cvlabel.txt
```

This will create a file listing all the available devices, which can be used, your recently mounted sparseimages should be present.

In a Terminal open the cvlabel.txt in an editor:

```
pico /tmp/cvlabel.txt
```

When the file is opened it will look something like this:

```
CvfsDisk_UNKNOWN /dev/rdisk18    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial a97f4efc36fc4f884ad5a4f4033dfc9d
CvfsDisk_UNKNOWN /dev/rdisk2    # host 0 lun 0 sectors 1981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 6304d65f8019b2230bbcf5eb27de8f81
CvfsDisk_UNKNOWN /dev/rdisk3    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 1a25a57ac4af9adcbe278bd1e3075d7d
CvfsDisk_UNKNOWN /dev/rdisk4    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial f4ed4a7a13872e9b1bab8d6d55326365
CvfsDisk_UNKNOWN /dev/rdisk5    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 1da1a48a1ace8c26b9a43923c78b3731
CvfsDisk_UNKNOWN /dev/rdisk6    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial ac467c760aa37fee4d5cdc883a8165c9
CvfsDisk_UNKNOWN /dev/rdisk7    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 40cba1eda156d698ab5be96c98bce3e2
CvfsDisk_UNKNOWN /dev/rdisk8    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 427407c941488a238a2423a24e443c2f
CvfsDisk_UNKNOWN /dev/rdisk9    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial ab2d54b182a8c707bee43515a76469a4
CvfsDisk_UNKNOWN /dev/rdisk10    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 0295d4655f3c4044a494305fd23b63b7
CvfsDisk_UNKNOWN /dev/rdisk11    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial c3526d004c293479803cbf6d2a143909
CvfsDisk_UNKNOWN /dev/rdisk12    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 5026c2b4d6afe559b998f93438d9562d
CvfsDisk_UNKNOWN /dev/rdisk13    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 2e6216e31cf21d5be5cfdb24874ddebe
CvfsDisk_UNKNOWN /dev/rdisk14    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 11f7499207fd558f7c3d965a0dd76e95
CvfsDisk_UNKNOWN /dev/rdisk15    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial a012e53057bbf05590ade83ffa85d6d6
CvfsDisk_UNKNOWN /dev/rdisk16    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial fd162b0d0ab7f72f45d648b4bcf212df
CvfsDisk_UNKNOWN /dev/rdisk17    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial ff1c5027729ff76d814e5795e5ad782f
CvfsDisk_UNKNOWN /dev/rdisk1    # host 0 lun 0 sectors 1981535 sector_size 512 inquiry
[Apple   Disk Image       0.6v28] serial 554f69bb5ba673335f03aaa7cb66dff6
```

Note: If you were patient and mounted your sparse images in sequential order after creating them the device numbers (/dev/rdisk1, etc) are a good reference for labeling your LUNs.

The 'LUNs' labels need to be changed, in this example we label our 'LUNs' according to their purpose, it should now read:

```
XsanLUN18 /dev/rdisk18    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial a97f4efc36fc4f884ad5a4f4033dfc9d
XsanLUN2 /dev/rdisk2    # host 0 lun 0 sectors 1981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 6304d65f8019b2230bbcf5eb27de8f81
XsanLUN3 /dev/rdisk3    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 1a25a57ac4af9adcbe278bd1e3075d7d
XsanLUN4 /dev/rdisk4    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial f4ed4a7a13872e9b1bab8d6d55326365
XsanLUN5 /dev/rdisk5    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 1da1a48a1ace8c26b9a43923c78b3731
XsanLUN6 /dev/rdisk6    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial ac467c760aa37fee4d5cdc883a8165c9
XsanLUN7 /dev/rdisk7    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 40cba1eda156d698ab5be96c98bce3e2
XsanLUN8 /dev/rdisk8    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 427407c941488a238a2423a24e443c2f
XsanLUN9 /dev/rdisk9    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial ab2d54b182a8c707bee43515a76469a4
XsanLUN10 /dev/rdisk10    # host 0 lun 0 sectors 15981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 0295d4655f3c4044a494305fd23b63b7
XsanLUN11 /dev/rdisk11    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial c3526d004c293479803cbf6d2a143909
XsanLUN12 /dev/rdisk12    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 5026c2b4d6afe559b998f93438d9562d
XsanLUN13 /dev/rdisk13    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 2e6216e31cf21d5be5cfdb24874ddebe
XsanLUN14 /dev/rdisk14    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 11f7499207fd558f7c3d965a0dd76e95
XsanLUN15 /dev/rdisk15    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial a012e53057bbf05590ade83ffa85d6d6
XsanLUN16 /dev/rdisk16    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial fd162b0d0ab7f72f45d648b4bcf212df
XsanLUN17 /dev/rdisk17    # host 0 lun 0 sectors 7981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial ff1c5027729ff76d814e5795e5ad782f
XsanLUN1 /dev/rdisk1    # host 0 lun 0 sectors 1981535 sector_size 512 inquiry [Apple
Disk Image       0.6v28] serial 554f69bb5ba673335f03aaa7cb66dff6
```

Note: The output doesn't give the disks any particular order, pay attention to the number of sectors in each 'LUN' and label accordingly.

Make a note of the number of sectors in each 'LUN' for reference when we create out config file.

In the example above, we have:

MetaDrive	sectors	1981535
JournalDrive	sectors	1981535
VideoDrive	sectors	15981535
AudioDrive	sectors	7981535
DataDrive	sectors	7981535

Exit and save cvlabel.txt

To label the 'LUNs', in a Terminal type:

```
cvlabel /tmp/cvlabel.txt
```

You will receive the warning below:

```
*WARNING* This program will over-write volume labels on the
          devices specified in the file "/tmp/cvlabel.txt".
```

```
          After execution, the devices will only be usable by the
          Xsan. You will have to re-partition the
          devices to use them on a different file system.


Do you want to proceed? (Y / N) ->
```

When prompted enter: `Y`

You will then receive another warning, relating to some OS compatibility issues, for each 'LUN':

```
/dev/rdisk14 [Apple   Disk Image       0.6v28] unknown  Controller
'2e6216e31cf21d5be5cfdb24874ddebe', Serial '2e6216e31cf21d5be5cfdb24874ddebe', Sector
Size 512, Sectors Max 15981535 (7.6GB)

Warning: Because it will use an EFI label, disk "/dev/rdisk18" will not be usable by
some Solaris systems, IRIX systems or earlier Xsan releases.

Do you want to label it acfs-EFI - Name: XsanLUN18 Sectors: 15981535 (Y / N) ->
```

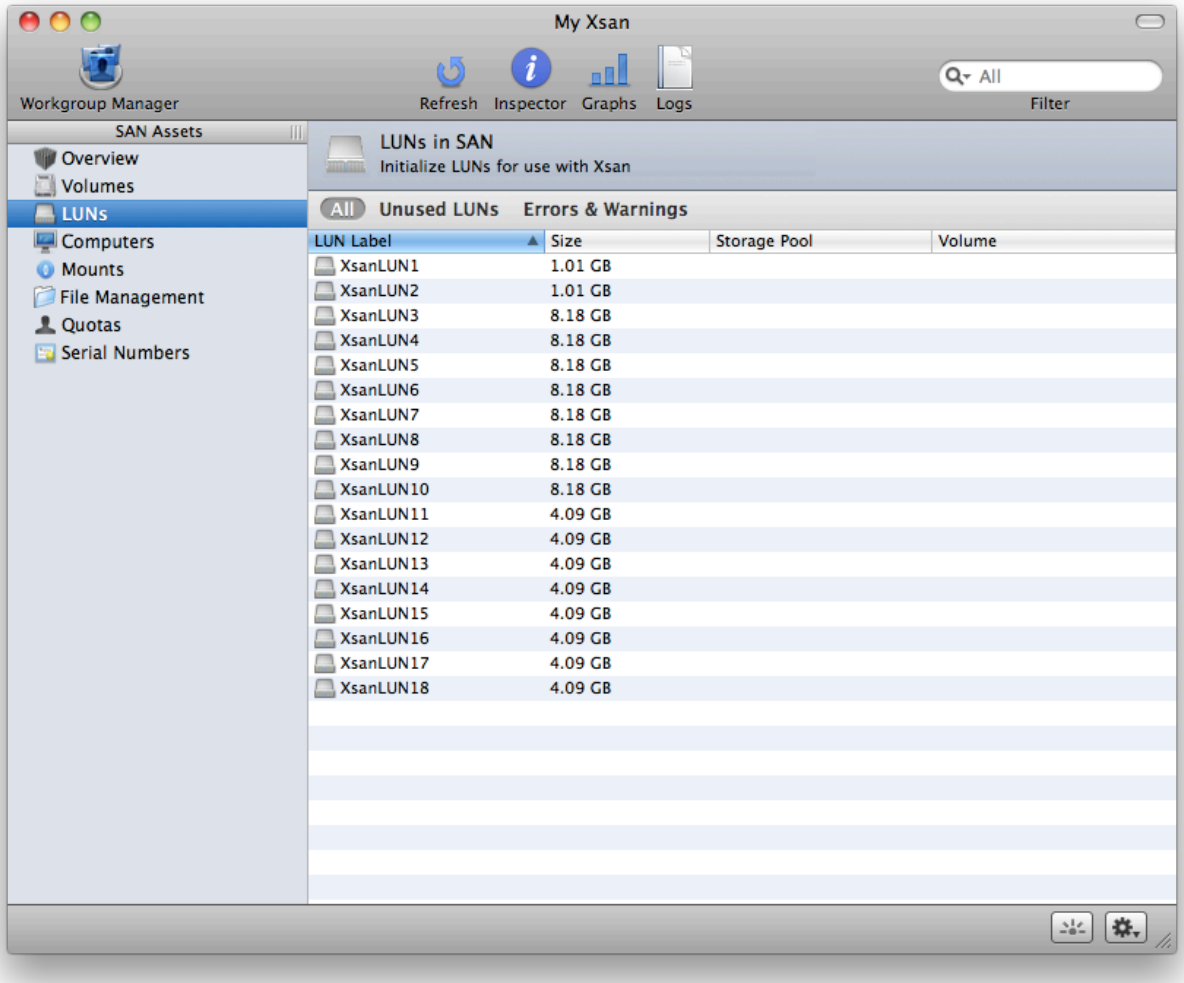
When prompted enter: `Y`

cvlabel should report back success:

```
New Volume Label -Device: /dev/rdisk18  acfs Label: XsanLUN18  Sectors: 15981535.

cvlabel: One or more disks with 2147483647 sectors or less has had an
EFI label applied to it, or has changed from an EFI label to a VTOC label.
Before they can use these disks, Solaris systems will need to be rebooted
with the following command:

    reboot -- -r

Done.  18 source lines.  18 labels.
```

This has read the modified cvlabel.txt file and assigned the LUN name(s) to the listed images(s), making them available to Xsan.

You may verify this by opening Xsan Admin and selecting LUNs from the sidebar.



Quit Xsan Admin.

Creating the Volume

fsmlist contains the names of all volumes hosted on your SAN.

To create a fsmlist file containing the name of your volume (MyVolume), in a Terminal type:

```
pico /Library/FileSystems/Xsan/config/fsmlist
```

Simply type MyVolume, save and exit

We now need a corresponding volume configuration file, the example config files have been removed from versions of Xsan greater than 2.0, so you will need to create one, The config file in this document is based on the sample in the cvfs_config man page (`man /Library/Filesystems/Xsan/man/man4/cvfs_config.4`), with the appropriate adjustments made for Xsan.

In a Terminal, create a MyVolume.cfg in an editor.

```
pico /Library/Filesystems/Xsan/config/MyVolume.cfg
```

Paste the following text in the editor in your Terminal.

```
#
# Xsan Volume Configuration File Example
#
#
# Names can be of [A-Z][a-Z][0-9] hyphen (-) and a under-bar (_)
#
# Other things, like user defined strings and pathnames must be enclosed
# by double quotes (").
#
# The comment character (#) may start anywhere and persists to the end
#  of line.
#


# ************************************************************************
# A global section for defining file system-wide parameters.
# ************************************************************************
FileLocks       Yes
GlobalSuperUser      Yes
Quotas               Yes
WindowsSecurity      No
EnforceACLs          Yes
EnableSpotlight      No
NamedStreams         Yes
ForceStripeAlignment Yes
UnixIdFabricationOnWindowsYes
Debug           0
AllocationStrategy   Round
SpotlightSearchLevel ReadWrite
InodeExpandMin       8
InodeExpandInc       32
InodeExpandMax       2048
BufferCacheSize      32M
JournalSize          16M
FsBlockSize          16K
InodeCacheSize       8K
MaxConnections       139
MaxLogSize      10M
ThreadPoolSize       256
UnixNobodyUidOnWindows    -2
UnixNobodyGidOnWindows    -2


# ************************************************************************
# A disktype section for defining disk hardware parameters.
# ************************************************************************
```

```
[DiskType "MetaDrive"]
Sectors 99999999
SectorSize 512

[DiskType "JournalDrive"]
Sectors 99999999
SectorSize 512

[DiskType "VideoDrive"]
Sectors 99999999
SectorSize 512

[DiskType "AudioDrive"]
Sectors 99999999
SectorSize 512

[DiskType "OtherDrive"]
Sectors 99999999
SectorSize 512


# ******************************************************************************
# A disk section for defining disks in the hardware configuration.
# ******************************************************************************

[Disk "XsanLUN1"]
Status Up
Type "MetaDrive"

[Disk "XsanLUN2"]
Status Up
Type "JournalDrive"

[Disk "XsanLUN3"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN4"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN5"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN6"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN7"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN8"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN9"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN10"]
Status Up
Type "VideoDrive"

[Disk "XsanLUN11"]
Status Up
Type "AudioDrive"

[Disk "XsanLUN12"]
Status Up
Type "AudioDrive"
```

```
[Disk "XsanLUN13"]
Status Up
Type "AudioDrive"

[Disk "XsanLUN14"]
Status Up
Type "AudioDrive"

[Disk "XsanLUN15"]
Status Up
Type "OtherDrive"

[Disk "XsanLUN16"]
Status Up
Type "OtherDrive"

[Disk "XsanLUN17"]
Status Up
Type "OtherDrive"

[Disk "XsanLUN18"]
Status Up
Type "OtherDrive"


# ****************************************************************************
# A stripe section for defining stripe groups.
# ****************************************************************************

[StripeGroup "Metadata"]
Status Up
Exclusive Yes
Metadata Yes
Journal No
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 64
Node "XsanLUN1" 0

[StripeGroup "Journal"]
Status Up
Exclusive Yes
Metadata No
Journal Yes
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 64
Node "XsanLUN2" 0

[StripeGroup "Video1"]
Status Up
Exclusive Yes
Affinity "NTSC"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 43
Node "XsanLUN3" 0
Node "XsanLUN4" 1
Node "XsanLUN5" 2
Node "XsanLUN6" 3

[StripeGroup "Video2"]
Status Up
Exclusive Yes
Affinity "PAL"
Read Enabled
Write Enabled
MultiPathMethod Rotate
```

```
StripeBreadth 51
Node "XsanLUN7" 0
Node "XsanLUN8" 1
Node "XsanLUN9" 2
Node "XsanLUN10" 3

[StripeGroup "Audio1"]
Status Up
Exclusive Yes
Affinity "NTSCAud"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 4
Node "XsanLUN11" 0
Node "XsanLUN12" 1

[StripeGroup "Audio2"]
Status Up
Exclusive Yes
Affinity "PALAud"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 15
Node "XsanLUN13" 0
Node "XsanLUN14" 1

[StripeGroup "Other"]
Status Up
Exclusive No
Affinity "Other"
Read Enabled
Write Enabled
MultiPathMethod Rotate
StripeBreadth 16
Node "XsanLUN15" 0
Node "XsanLUN16" 1
Node "XsanLUN17" 2
Node "XsanLUN18" 3

#
# End
#
```

We will need to edit the Disk Type Hardware Parameters section so it reflects the number of sectors in each of our 'LUNs'. Earlier we noted the number of sectors in each 'LUN' type when we labelled them, this is where we will use this information.

Scroll down to the section with the header, and make the appropriate changes to the Sectors for each disk type (noted in bold):

```
# ********************************************************************************
# A disktype section for defining disk hardware parameters.
# ********************************************************************************
```

Edit each `DiskType` parameter as appropriate:

```
[DiskType "MetaDrive"]
Sectors 1981535
SectorSize 512

[DiskType "JournalDrive"]
Sectors 1981535
SectorSize 512

[DiskType "VideoDrive"]
Sectors 15981535
SectorSize 512
```

```
[DiskType "AudioDrive"]
Sectors 7981535
SectorSize 512

[DiskType "DataDrive"]
Sectors 7981535
SectorSize 512
```

To make the filesystem, in a Terminal type:

```
cvmkfs MyVolume
```

You will see a warning:

```
Xsan File System Initializer.

** WARNING ** This will destroy all contents and data of the

              --- [MyVolume] ---

              file system. Re-initialization cannot be un-done!

Do you want to proceed? (Y/[N]) ->
```

Enter `Y` and the filesystem will be created:

```
Re-initializing file system 'MyVolume'.

Shared Meta Data File System.
Meta Data Root is on "MetaFiles".
Resetting Arbitration Block.
Writing Configuration Information Block.
Writing Root Directory Block.
Writing Disk Information Block.
Writing Stripe Group Index "Metadata".
Writing Stripe Group Index "Journal".
Writing Stripe Group Index "Video1".
Writing Stripe Group Index "Video2".
Writing Stripe Group Index "Audio1".
Writing Stripe Group Index "Audio2".
Writing Stripe Group Index "Other".
Resetting File System Journal.
Creating Free List.
Writing Initial 1024 Inodes.
Writing Initialization Control Block.
Setting "Force Stripe Alignment"
Resetting Super Block.
Writing Reserved Allocations.

File system 'MyVolume' successfully created.
```

We just now need to start and activate the volume, in a terminal type:

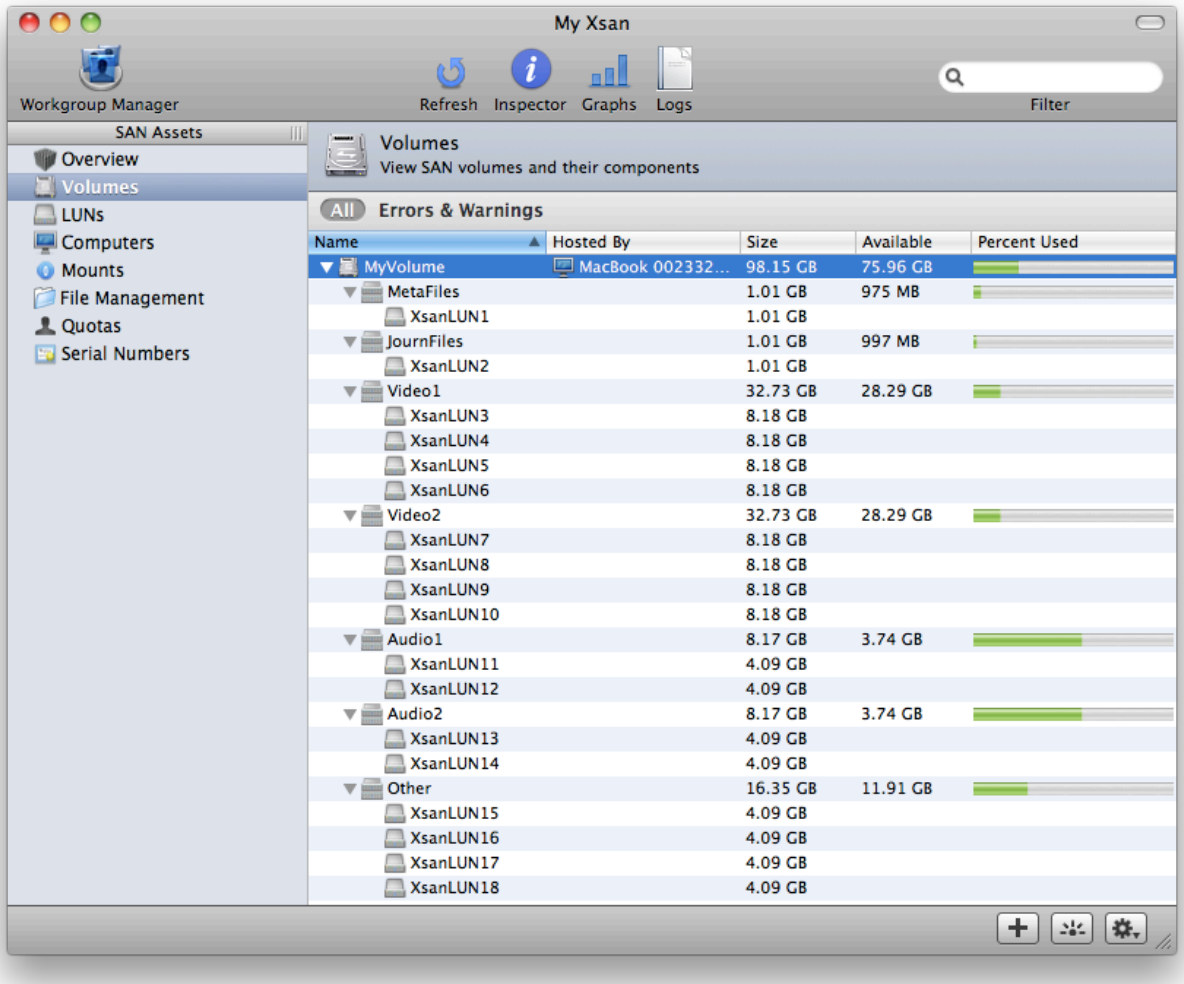
```
cvadmin -e "start MyVolume"
```

followed by:

```
cvadmin -e "activate MyVolume"
```

Finally to mount the volume on your desktop type:

```
xsanctl mount MyVolume
```

MyVolume should now appear in your computer, with the Xsan filesystem icon

Open Xsan Admin and verify the Volume is running.