Oracle Cluster File System (OCFS2) User's Guide

NOTE

This user's guide is for OCFS2 Release 1.2 that works only on the older RHEL4 and SLES9 distributions.

We have since released two new updates to the file system: OCFS2 Release 1.4 and OCFS2 Release 1.6 which work on newer Enterprise distributions.

The user's guide for OCFS2 Release 1.6 is available at the following link.
http://oss.oracle.com/projects/ocfs2/dist/documentation/v1.6/ocfs2-1_6-usersguide.pdf

The user's guide for OCFS2 Release 1.4 is available at the following link.
http://oss.oracle.com/projects/ocfs2/dist/documentation/v1.4/ocfs2-1_4-usersguide.pdf

Oracle Cluster File System (OCFS2) User's Guide

1. Introduction

A Cluster File System allows all nodes in a cluster to concurrently access a device via the standard file system interface. This allows for easy management of applications that need to run across a cluster.

OCFS (Release 1) was released in December 2002 to enable Oracle Real Application Cluster (RAC) users to run the clustered database without having to deal with RAW devices. The file system was designed to store database related files, such as data files, control files, redo logs, archive logs, etc.

OCFS2 is the next generation of the Oracle Cluster File System. It has been designed to be a general-purpose cluster file system. With it, one can store not only database related files on a shared disk, but also store Oracle binaries and configuration files (shared Oracle home) making management of RAC even easier.

2. Installation

The OCFS2 distributions comprises of two sets of RPMs. Namely, the kernel module and the tools.

The kernel module is available for download from http://oss.oracle.com/projects/ocfs2/files/ and the tools from http://oss.oracle.com/projects/ocfs2-tools/files/.

Download the appropriate RPMs. For the kernel module, download the one that matches the distribution, platform, kernel version and the kernel flavor (smp, hugemem, psmp, etc). For tools, simply match the platform and distribution.

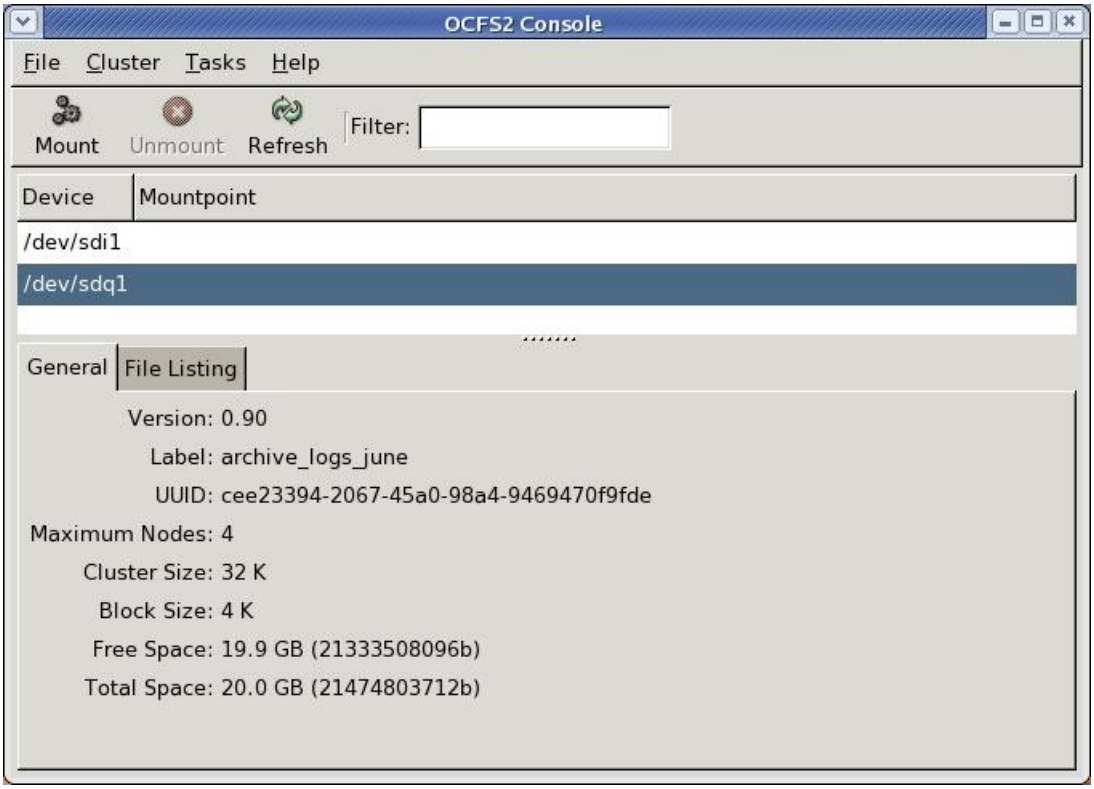
The tools RPMs are split into two. *ocfs2-tools* includes the command line tools whereas *ocfs2console* includes the GUI front end for the tools. One does not have to install the console, though it is recommended for ease-of-use.

Install the RPMs using the *rpm --install* or *rpm --upgrade* command.

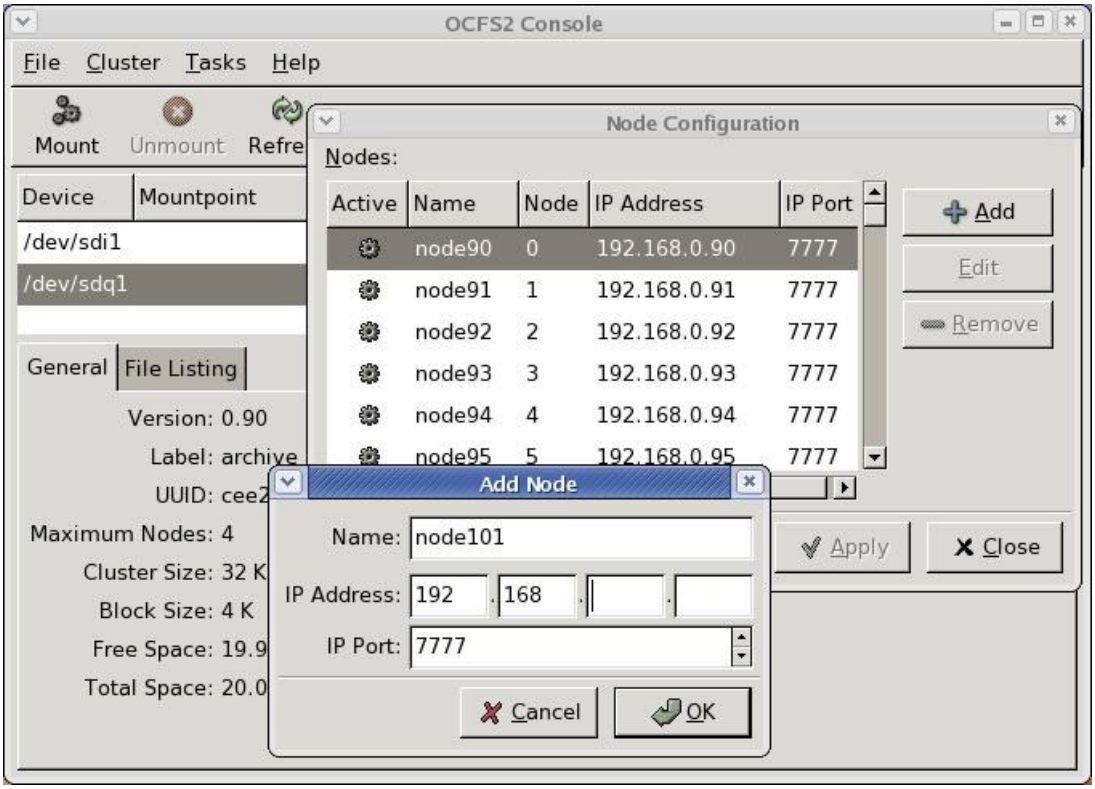
3. Configuration

OCFS2 has one configuration file, namely, */etc/ocfs2/cluster.conf*. In it, one needs to specify all the nodes in the cluster. This file should be the same on all the nodes in the cluster. Whereas one can add new nodes to the cluster dynamically, any other change, like name, ip address, requires the cluster to be restarted for the changes to take effect.

Users are strongly recommended to use *ocfs2console* to setup and propagate the cluster.conf to all the nodes in the cluster.



Start *ocfs2console* and click on menu item *Cluster* followed by *Configure Nodes*. If the cluster is offline, the console will start it and display a message to that effect. If cluster.conf is not present, the console will create one with a default cluster name *ocfs2*.



Click on *Add* to add nodes to the cluster. One needs to enter the node name (same as the hostname), the IP Address and Port. The console assigns node numbers sequentially from 0 to 254.

Once all the nodes are added, one can propagate the configuration to all the nodes by clicking on menu item *Cluster* followed by *Propagate Configuration*. As the console uses *ssh* to propagate the file, it requires the keys to all nodes be authorized for it to work without prompting for the password to connect to each node.

Please refer to the appendix A for a sample cluster.conf and the description of the layout.

4. O2CB Cluster Service

OCFS2 comes bundled with its own cluster stack, O2CB. The stack includes:

- NM: Node Manager that keep track of all the nodes in the cluster.conf
- HB: Heart beat service that issues up/down notifications when nodes join or leave the cluster
- TCP: Handles communication between the nodes
- DLM: Distributed lock manager that keeps track of all locks, its owners and status
- CONFIGFS: User space driven configuration file system mounted at /config
- DLMFS: User space interface to the kernel space DLM

All the cluster services have been packaged in the *o2cb* system service. As all the operations, like format, mount, etc., require the cluster to be online, start it before continuing to format.

To check the **status of the cluster**, do:

```
# /etc/init.d/o2cb status
Module "configfs": Not loaded
Filesystem "configfs": Not mounted
Module "ocfs2_nodemanager": Not loaded
Module "ocfs2_dlm": Not loaded
Module "ocfs2_dlmfs": Not loaded
Filesystem "ocfs2_dlmfs": Not mounted
```

To **load the modules**, do:

```
# /etc/init.d/o2cb load
Loading module "configfs": OK
Mounting configfs filesystem at /config: OK
Loading module "ocfs2_nodemanager": OK
Loading module "ocfs2_dlm": OK
Loading module "ocfs2_dlmfs": OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
```

To **online cluster *ocfs2***, do:

```
# /etc/init.d/o2cb online ocfs2
Starting cluster ocfs2: OK
```

To **offline cluster *ocfs2***, do:

```
# /etc/init.d/o2cb offline ocfs2
Cleaning heartbeat on ocfs2: OK
Stopping cluster ocfs2: OK
```

To **unload the modules**, do:

```
# /etc/init.d/o2cb unload
Unmounting ocfs2_dlmfs filesystem: OK
```

```
Unloading module "ocfs2_dlmfs": OK
Unmounting configfs filesystem: OK
Unloading module "configfs": OK
```

To **configure O2CB** to start on boot, do:

```
# /etc/init.d/o2cb configure
    Configuring the O2CB driver.

    This will configure the on-boot properties of the O2CB driver.
    The following questions will determine whether the driver is loaded on
    boot.  The current values will be shown in brackets ('[]').  Hitting
    <ENTER> without typing an answer will keep that current value.  Ctrl-C
    will abort.

    Load O2CB driver on boot (y/n) [n]: y
    Cluster to start on boot (Enter "none" to clear) []: ocfs2
    Writing O2CB configuration: OK
#
```

If the cluster is setup to load on boot, one could **start and stop cluster ocfs2**, as follows:

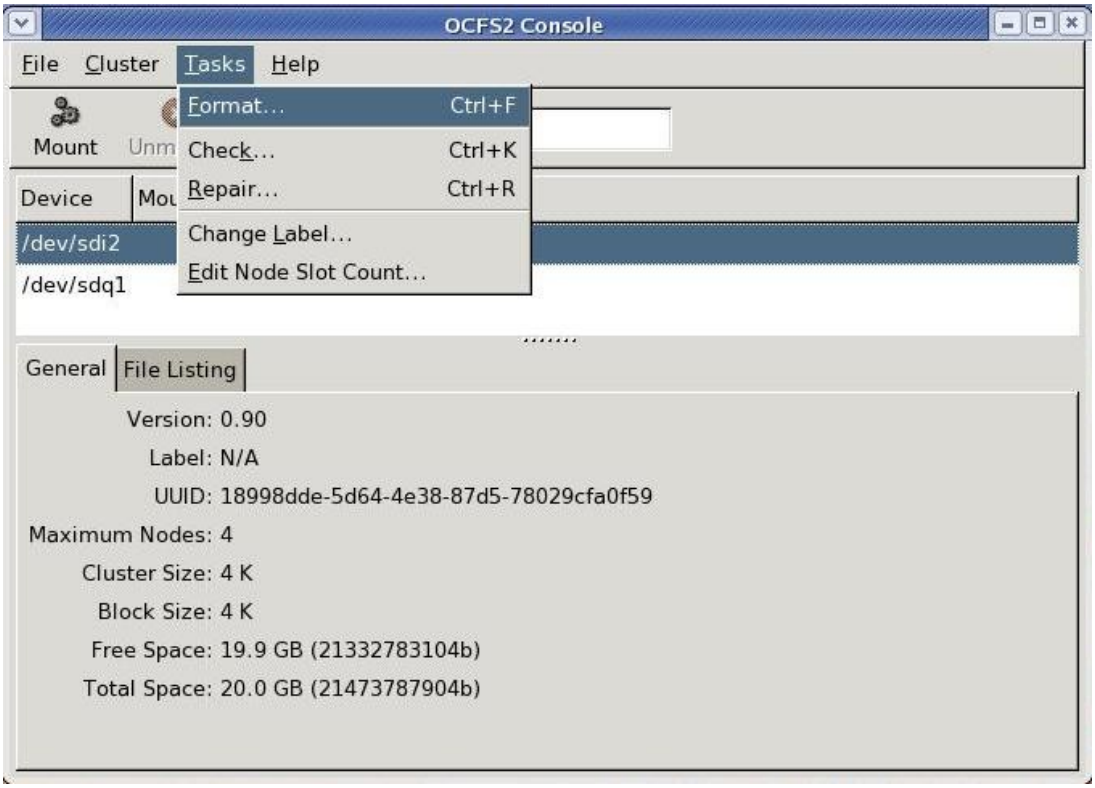
```
# /etc/init.d/o2cb start
Loading module "configfs": OK
Mounting configfs filesystem at /config: OK
Loading module "ocfs2_nodemanager": OK
Loading module "ocfs2_dlm": OK
Loading module "ocfs2_dlmfs": OK
Mounting ocfs2_dlmfs filesystem at /dlm: OK
Starting cluster ocfs2: OK


# /etc/init.d/o2cb stop
Cleaning heartbeat on ocfs2: OK
Stopping cluster ocfs2: OK
Unmounting ocfs2_dlmfs filesystem: OK
Unloading module "ocfs2_dlmfs": OK
Unmounting configfs filesystem: OK
Unloading module "configfs": OK
```

5. Format

If the O2CB cluster is offline, start it. The format operation needs the cluster to be online, as it needs to ensure that the volume is not mounted on some node in the cluster.

One can format a volume either using the console or using the command line tool, *mkfs.ocfs2*.

To format using the console, start *ocfs2console*, and click on menu item *Tasks* followed by *Format*.



Select a device to format in the drop down list *Available devices*. Wherever possible, the console will list the existing file system type.

Enter a label. It is recommended one label the device for ease of management. The label is changeable after the format.

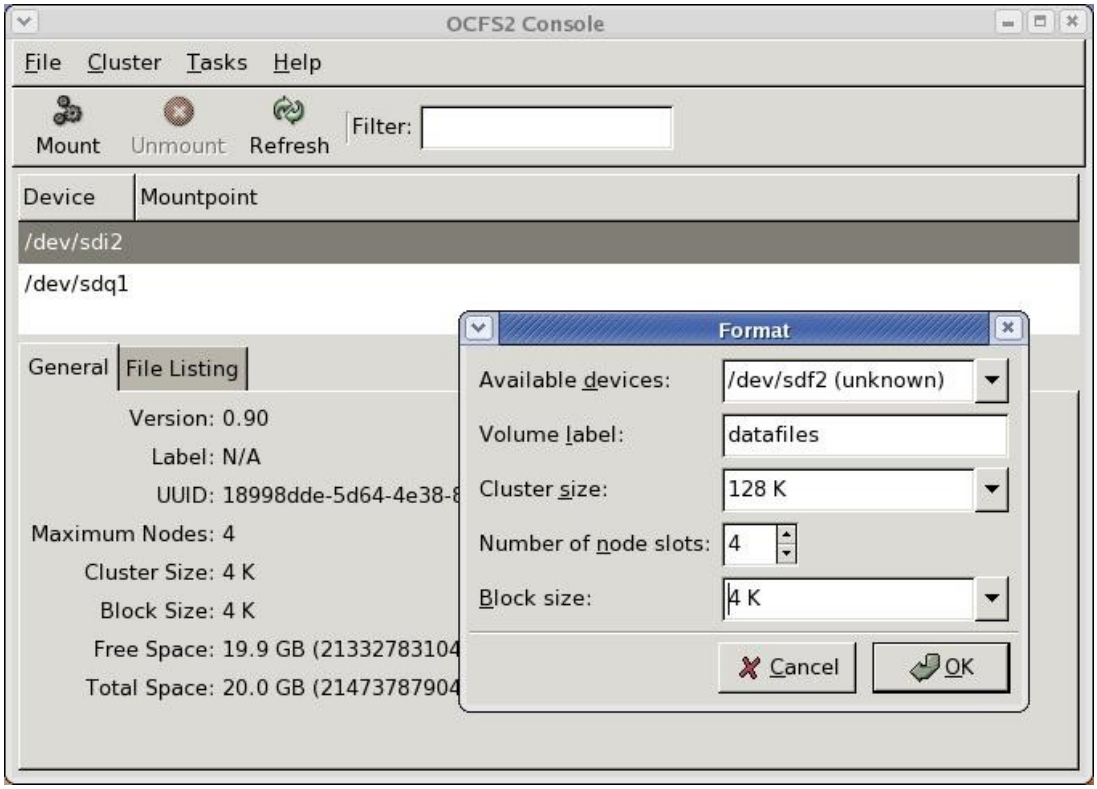
Select a cluster size. The sizes supported range from 4K to 1M. For a data files volume, large files, a cluster size of 128K or larger is appropriate.

Select a block size. The sizes supported range from 512 bytes to 4K. As OCFS2 does not allocate a static inode area on format, a 4K block-size is most recommended for most disk sizes. On the other hand, even though it supports 512 bytes, that small a block size is not recommended.

Both the cluster and blocks sizes are not changeable after the format.

Enter the number of node slots. This number determines the number of nodes that can concurrently mount the volume. This number can be increased, but not decreased, at a later date.

Click on *OK* to format the volume.



To **format a volume** with a 4K block-size, 32K cluster-size and 4 node slots using the command line tool, *mkfs.ocfs2*, do:

```
# mkfs.ocfs2 -b 4K -C 32K -N 4 -L oracle_home /dev/sdf2
mkfs.ocfs2 0.99.15-BETA16
Overwriting existing ocfs2 partition.
Proceed (y/N): y
Filesystem label=oracle_home
Block size=4096 (bits=12)
Cluster size=32768 (bits=15)
Volume size=21474820096 (655359 clusters) (5242872 blocks)
21 cluster groups (tail covers 10239 clusters, rest cover 32256 clusters)
Journal size=33554432
Initial number of node slots: 4
Creating bitmaps: done
Initializing superblock: done
Writing system files: done
Writing superblock: done
```

```
Writing lost+found: done
mkfs.ocfs2 successful
```

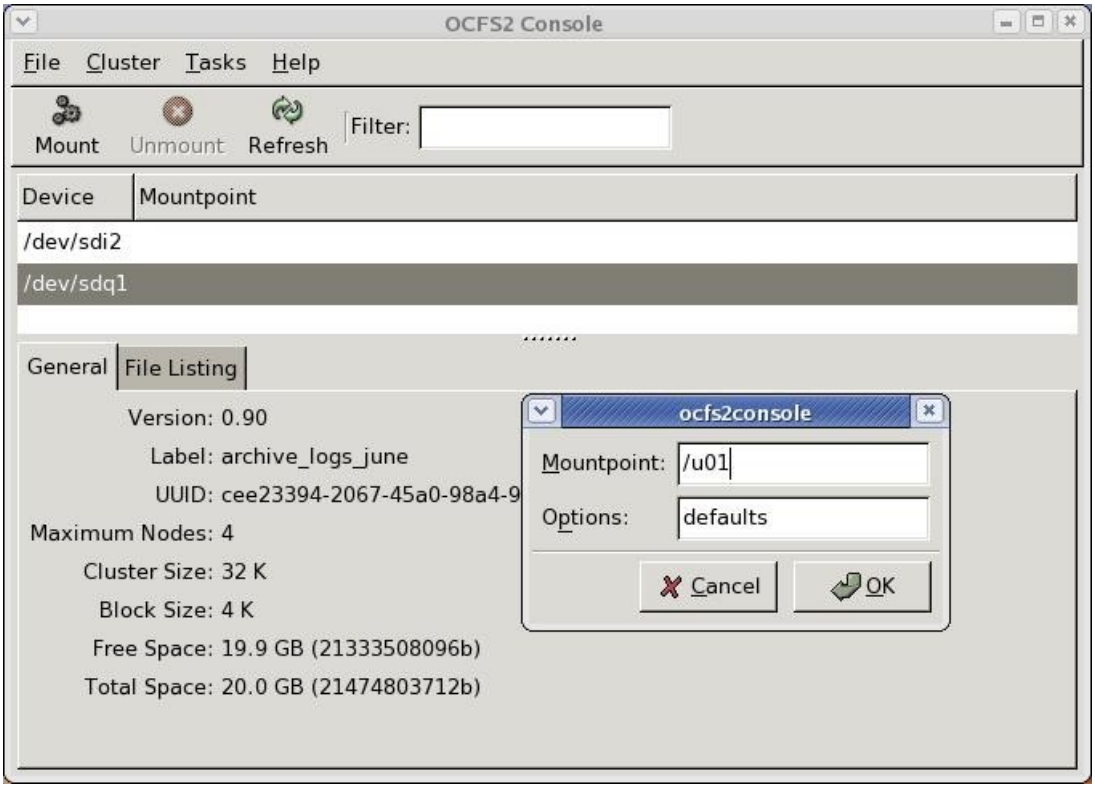
For the complete help on *mkfs.ocfs2*, refer to its man page.

6. Mount

If the O2CB cluster service is offline, start it. The mount operation requires the cluster to be online.

One can mount a volume using the console or the command line tool, *mount*.

To mount from the console, start *ocfs2console*, select the device and click on *Mount*. In the dialog, enter the directory mount point and, optionally, mount options. Click on *OK*. On a successful mount, the device list will show the mount point along with the device.



To **mount** from the command line, do:

```
# mount -t ocfs2 /dev/sdf2 /u01
```

To **unmount** a volume, one can select the desired volume in the console and click *Unmount* or do:

```
# umount /u01
```

Oracle database users must mount the volumes containing the Voting Disk file (CRS), Cluster Registry (OCR), Data files, Redo logs, Archive logs and Control files with the *datavolume* mount option so as to ensure that the Oracle processes open the files with the o_direct flag. All other volumes, including Oracle home, should not be mounted with this mount option.

To **mount volume containing Oracle data files, Voting disk**, etc., do:

```
# mount -t ocfs2 -o datavolume /dev/sdf2 /u01
# mount
/dev/sdf2 on /u01 type ocfs2 (rw,datavolume)
```

To **auto mount** OCFS2 volumes on boot, one needs to enable the *o2cb* service using *chkconfig*, configure *o2cb* to load on boot, and add the mount entries into */etc/fstab* as follows:

```
# cat /etc/fstab
...
/dev/sdf2   /u02  ocfs2      _netdev    0     0
...
```

The **_netdev** mount option is a must for OCFS2 volumes. This mount option indicates that the volume is to be mounted after the network is started and dismounted before the network is shutdown.

```
# chkconfig --add o2cb
o2cb                    0:off  1:off  2:on   3:on   4:off  5:on   6:off
#
#
# /etc/init.d/o2cb configure
...
    Load O2CB driver on boot (y/n) [n]: y
    Cluster to start on boot (Enter "none" to clear) []: ocfs2
    Writing O2CB configuration: OK
```

To **mount by label**, do:

```
# mount -L datafiles /u01
```

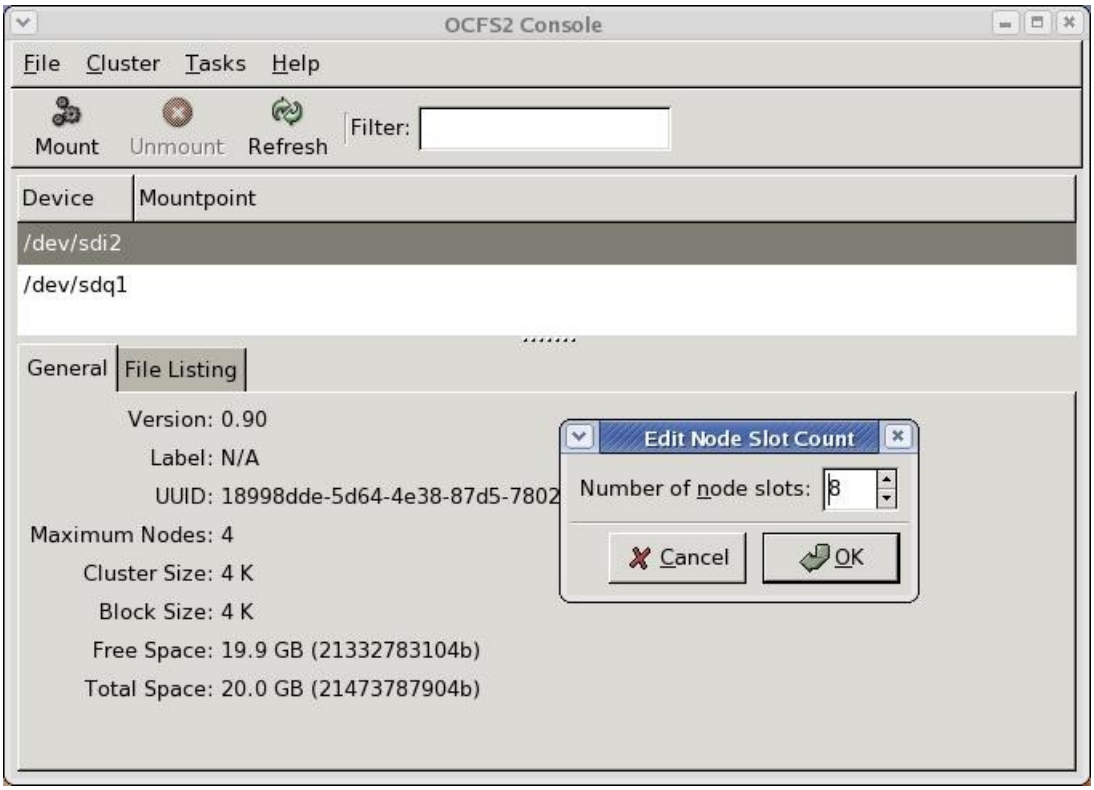
7. Tune

The tuning operation allows one to increase the number of node slots (to increase the number of nodes that can concurrently mount the volume), change the volume label and increase the size of the journal file.

tunefs.ocfs2 is the command-line tool that performs the tuning. One can also access this tool via the console.

If the O2CB cluster service is offline, start it. The tune operation requires the cluster service to be online.

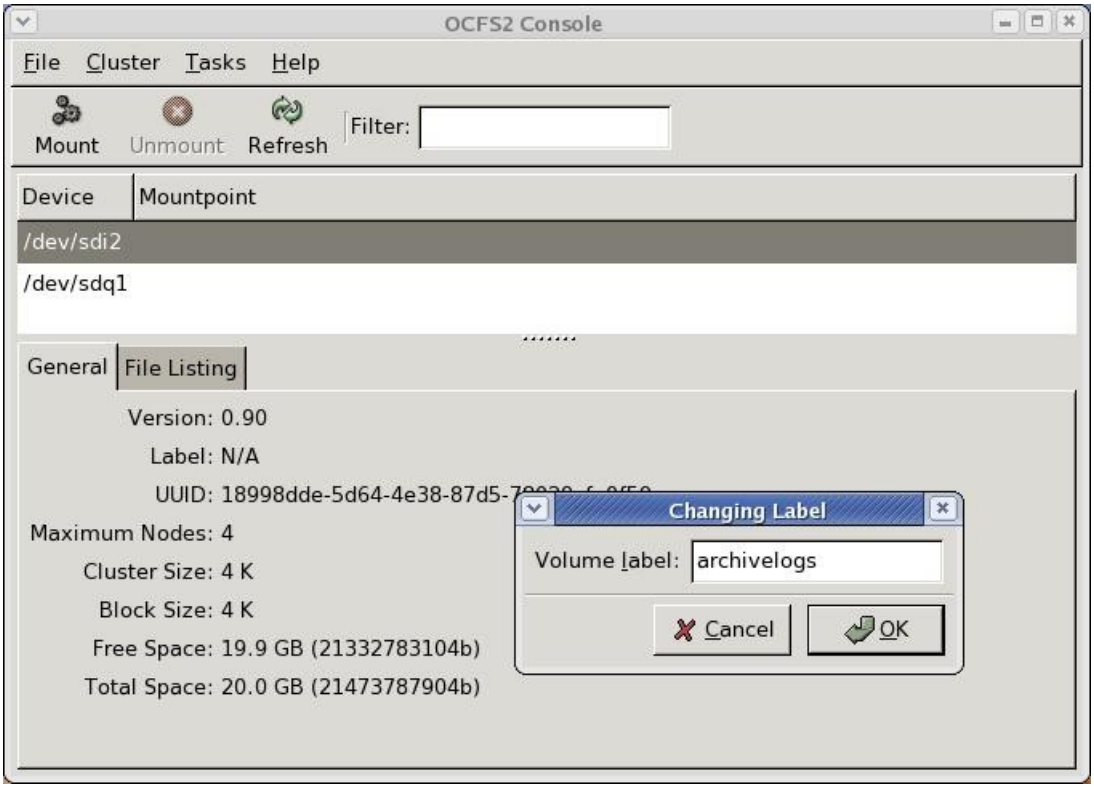
To **increase the number of node slots** in the volume using the console, click on the menu-item *Tasks* followed by *Edit Node Slot count*. Enter the new number and click on *OK*.



To **increase the number of node slots** using the command-line tool, *tunefs.ocfs*, do:

```
# tunefs.ocfs2 -N 8 /dev/sdf2
tunefs.ocfs2 0.99.15-BETA16
Changing number of node slots from 4 to 8
Proceed (y/N): y
Added node slots
Wrote Superblock
```

To **change the volume label** in the volume using the console, click on the menu item *Tasks* followed by *Change Label*. Enter the new label and click on *OK*.



To **change the volume label** using the command-line tool, *tunefs.ocfs*, do:

```
# tunefs.ocfs2 -L "old datafiles" /dev/sdf2
tunefs.ocfs2 0.99.15-BETA16
Changing volume label from datafiles to old datafiles
Proceed (y/N): y
Changed volume label
Wrote Superblock
```

For the complete help on *tunefs.ocfs2*, refer to its man page.

8. File System Check

File system check operation checks the health of the file system. It is recommended one run this utility regularly in order to detect and fix any irregularities in the volume.

fsck.ocfs2 is the command-line tool that performs the check. One can also access this tool via the console.

Start *ocfs2console*, and click on menu-item *Task* followed by *Check*. This performs the read-only check operation, and could be performed with the file system mounted on the cluster. The equivalent on the command-line would be,

```
# fsck.ocfs2 -n /dev/sdf2
Checking OCFS2 filesystem in /dev/sdf2:
  label:              oracle_home
  uuid:               dc a2 90 1b 24 1f 40 6e 80 e9 85 dd 6b 45 2d 1a
  number of blocks:   5242872
  bytes per block:    4096
  number of clusters: 655359
  bytes per cluster:  32768
  max slots:          4

/dev/sdf2 is clean.  It will be checked after 20 additional mounts.
```

To do the repair, click on the menu-item *Task* followed by *Repair*. This operation requires the O2CB cluster service to be online to ensure that the volume is not mounted on any node in the cluster. The equivalent for the command-line would be,

```
# fsck.ocfs2 -y /dev/sdf2
Checking OCFS2 filesystem in /dev/sdf2:
  label:              oracle_home
  uuid:               dc a2 90 1b 24 1f 40 6e 80 e9 85 dd 6b 45 2d 1a
  number of blocks:   5242872
  bytes per block:    4096
  number of clusters: 655359
  bytes per cluster:  32768
  max slots:          4

/dev/sdf2 is clean.  It will be checked after 20 additional mounts.
```

For the complete help on *fsck.ocfs2*, refer to its man page.

9. Context Dependent Symbolic Links (CDSL)

In a shared-disk clustered environment, there are instances when one needs to share just the file or the directory name across the cluster but not the contents. For example, in a shared Oracle home setup, `network/admin/listener.ora` is common to all the nodes but its contents differ from node to node. Similarly, in a shared root setup, one needs `/etc/` to be different for each node in the cluster.

In both these instances, the actual contents of the file or directory depend on the hostname. In other instances, one may require that the contents are dependent on the architecture (x86, x86-64 or IA64), or perhaps, the node number.

To handle this issue, OCFS2 implements Context Dependent Symbolic Links (CDSL). In short, it uses soft links to match a file or a directory name to its contents.

ocfs2cdsl is the command-line tool used to create these links.

To create a hostname dependent CDSL file, do,

```
root@node32:admin/# ocfs2cdsl listener.ora
root@node32:admin/# ls –l listener.ora
lrwxrwxrwx  1 root root   50 Aug  8 11:41 listener.ora ->
           ../.cluster/hostname/{hostname}/10g/network/admin/listener.ora
```

Edit this file and save the new content. To access the file from some other node in the cluster, create the CDSL on that node before editing. (As hostname is the default, one does not need to specify `–t hostname` in the command.)

```
root@node31:admin/# ocfs2cdsl listener.ora
```

To convert an existing file into a CDSL file, do,

```
root@node32:admin/# ocfs2cdsl –c sqlnet.ora
```

Once again, to access from some other node in the cluster, create the CDSL on that node before accessing the file.

```
root@node31:admin/# ocfs2cdsl sqlnet.ora
```

In this case, you will notice that the file has the same content as it did when the file was converted to a CDSL. Any new changes to the file on both nodes will not be visible to the other node.

To edit a CDSL file or directory belonging to another host or architecture, change directory to `.cluster` in the root (mount directory) of the OCFS2 volume and traverse to the directory holding the required copy. For example, to edit node32's copy of listener.ora on node31, do,

```
root@node31:/# ls -l /ohome/.cluster/hostname/node32/10g/network/admin/l*
-rw-r--r--  1 root root  3867 Aug  8 11:54 listener.ora
```

To delete a CDSL file, just unlink it, as,

```
root@node31:admin/# rm listener.ora
```

For the complete help on *ocfs2cdsl*, refer to its man page.

10. Other Tools

mounted.ocfs2

mounted.ocfs2 is a command-line tool used to list all the OCFS2 volumes on a node. This tool scans all the partitions listed in /proc/partitions as part of its detection process. If run in the full detect mode, it lists all the nodes that have mounted that volume.

To list all the OCFS2 volumes on a system, do,

```
# mounted.ocfs2 -d
Device           FS     UUID                                 Label
/dev/sdb1        ocfs2  e70c75a0-a08c-480a-bf50-ebda4191da30 mm_v2_dbf1
/dev/sdb2        ocfs2  f49163e8-6288-43c4-a792-e9401fde45fa mm_v2_ctrl
/dev/sdb3        ocfs2  2d441be2-adb6-4c52-9e19-9a7c2c485dc4 mm_v2_dbf2
/dev/sdb5        ocfs2  8607eae9-8e4f-4495-84e8-8db0bc9da60c mm_v2_log1
/dev/sdb6        ocfs2  acfb7b7d-a277-4741-9791-620ea9b82670 mm_v2_log2
/dev/sdf1        ocfs2  84749537-df45-4a97-aa28-fad63760b674 9ihome
/dev/sdq1        ocfs2  dca2901b-241f-406e-80e9-85dd6b452d1a oracle_home
/dev/sdcf1       ocfs2  663764bd-1eed-4b3c-aa48-a98f0be0e574 10ghome
/dev/sdcf2       ocfs2  8e2d9c21-ef47-4fea-8ac5-2306cc60455e mm_v2_log3
```

To list all the nodes that have mounted the OCFS2 volumes on a system, do,

```
# mounted.ocfs2 -f
Device           FS     Nodes
/dev/sdb1        ocfs2  node31, node32, node33, node34
/dev/sdb2        ocfs2  node31, node32, node33, node34
/dev/sdb3        ocfs2  node31, node32, node33, node34
/dev/sdb5        ocfs2  node31, node32, node33, node34
/dev/sdb6        ocfs2  node91, node90
/dev/sdf1        ocfs2  Not mounted
/dev/sdq1        ocfs2  node34, node35
/dev/sdcf1       ocfs2  Not mounted
/dev/sdcf2       ocfs2  Not mounted
```

Note: The node names are only listed if the O2CB cluster service is online. If not, it lists the global node numbers instead.

For the complete help on *mounted.ocfs2*, refer to its man page.

APPENDIX A

The configuration file is in a stanza format with two types of stanzas, namely, cluster and node. As OCFS2 does not support multiple clusters, a typical cluster.conf will have one cluster stanza and multiple node stanzas.

Sample /etc/ocfs2/cluster.conf

```
cluster:
node_count = 2
name = racdb

node:
ip_port = 7777
ip_address = 192.168.0.107
number = 7
name = node7
cluster = racdb

node:
ip_port = 7777
ip_address = 192.168.0.106
number = 6
name = node6
cluster = racdb
```

The **Cluster stanza** has two parameters:

- *node_count* - Total number of nodes in the cluster
- *name* - Name of the cluster

The **Node stanza** has five parameters:

- *ip_port* - IP Port
- *ip_address* - IP Address
- *number* - Unique node number from 0-254
- *name* - Hostname
- *cluster* - Name of the cluster should match that in the Cluster stanza