Sun StorageTek™ RDAC Multipath Failover Driver Installation Guide

For Linux OS
Contents

Preface v

Sun StorageTek RDAC Multipath Failover Driver for Linux OS 1
About RDAC Multipath Drivers 1
Prerequisites 2
Downloading the RDAC Driver 3
Installing the RDAC Driver 5
Multipath Failover Driver Utilities 14
  mppUtil Utility 14
  lsdev Utility 17
  mppSupport Utility 17
Linux RDAC Configuration Settings 18
  Wait Time Settings 19
  Updating the initrd Image 21
Troubleshooting Linux RDAC Drivers 21
  Determining if a Path Failed 23
  Error Levels 24
  Fatal Errors 24
  Controller and Path Failover Events 26
Preface

The Sun StorageTek RDAC Multipath Failover Driver Installation Guide for Linux OS explains how to install the Redundant Dual Active Controller (RDAC) failover driver for Linux platforms. This document is written for technicians, system administrators, and users who have advanced experience with Linux OS.

Related Documentation

The online documentation related to this product is available at the Sun Microsystems Documentation site at:

http://docs.sun.com/app/docs/prod/6180.array#hic
http://docs.sun.com/app/docs/prod/6780.array#hic
http://docs.sun.com/app/docs/prod/6580.array#hic
http://docs.sun.com/app/docs/coll/st2500arrays
http://docs.sun.com/app/docs/coll/st6540array
Documentation, Support, and Training

<table>
<thead>
<tr>
<th>Sun Function</th>
<th>URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Documentation</td>
<td><a href="http://www.sun.com/documentation/">http://www.sun.com/documentation/</a></td>
</tr>
<tr>
<td>Support</td>
<td><a href="http://www.sun.com/support/">http://www.sun.com/support/</a></td>
</tr>
<tr>
<td>Training</td>
<td><a href="http://www.sun.com/training/">http://www.sun.com/training/</a></td>
</tr>
</tbody>
</table>

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. Submit comments about this document by clicking the Feedback[+] link at http://docs.sun.com. Please include the title and part number of your document with your feedback:

Sun StorageTek RDAC Multipath Failover Driver for Linux OS

The Sun StorageTek Redundant Dual Active Controller (RDAC) failover drivers support the following Linux OS distributions for the 2.6 kernel:

- SuSE SLES 10 SP1 and SLES 9 (x86, x86_64, IA64)
- SuSE Linux Enterprise Server 10 SP1 and 9 for IBM eServer pSeries
- Architectures (ppc64)
- Red Hat RHEL 5 (x86, x86_64, IA64)
- Red Hat Advanced Server 4 (x86, x86_64, IA64)

Note – The failover driver for Linux OS RDAC does not support any Linux OS 2.4 kernels such as SuSE SLES 8 OS on Intel architecture or AMD architecture, Red Hat 3 Linux OS on Intel or AMD architecture, and Linux OS SLES8 and Red Hat 3 on POWER (LoP) servers.

About RDAC Multipath Drivers

Redundant Dual Active Controller (RDAC) is the failover driver for the Sun Storage 6180 array, StorageTek 6000 array series (includes Sun Storage 6580 and 6780 arrays) and Sun StorageTek 2500 array series. The RDAC failover driver includes these features:

- On-the-fly path validation.
- Cluster support.
Automatic detection of path failure. The RDAC failover driver automatically routes I/O to another path in the same controller or to an alternate controller, in case all paths to a particular controller fail.

- Improved I/O retries. The RDAC driver handles vendor-specific SCSI error codes from the array controllers.
- Automatic rebalancing. When the failed controller obtains optimal status, storage array rebalance is performed automatically without user intervention.

**Note** – The RDAC driver cannot co-exist with an HBA-level failover driver such as the Emulex, QLogic, or LSI Logic HBA failover drivers.

### Prerequisites

Before you install the RDAC driver, be sure the following requirements are met:

1. **The HBAs connected to the host on which you are installing the RDAC driver are supported.** The driver supports QLogic, LSI, and Emulex HBAs. (For supported HBA model numbers, see the README file included in the RDAC driver package, as described in “Downloading the RDAC Driver” on page 3.)

2. **The compiler and kernel source for the appropriate OS are installed.**

3. **The HBAs connected to a single storage array are the same model.** Use your HBA utilities, such as HBAanywhere (Emulex) or `scli` (QLogic) to ensure that your storage area network (SAN) is configured properly for this requirement.

4. **The base HBA driver is installed and built.** The following base and host drivers are supported:

<table>
<thead>
<tr>
<th>HBA</th>
<th>Base Driver Name</th>
<th>Host Driver Name</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>QLogic</td>
<td>qla2xxx</td>
<td>qla2300</td>
<td></td>
</tr>
<tr>
<td>LSI</td>
<td>mptbase</td>
<td>mptscsi or mptscsih</td>
<td>mpt driver version earlier than 3.02.xx</td>
</tr>
<tr>
<td></td>
<td></td>
<td>mptspi, mptsas, and mptfc</td>
<td>3.02.xx or later for SCSI parallel interface transport, SAS transport, and FC transport respectively.</td>
</tr>
<tr>
<td>Emulex or IBM Emulex</td>
<td>lpfddd or lpfc</td>
<td>lpfc</td>
<td></td>
</tr>
</tbody>
</table>
5. To verify the driver is installed, use the following command:

```
lsmod | grep <driver_name>
```

The following sample output shows QLogic drivers (in bold text):

<table>
<thead>
<tr>
<th>Module</th>
<th>Size</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>.iscsi_tcp</td>
<td>56641</td>
<td>0</td>
</tr>
<tr>
<td>libiscsi</td>
<td>59329</td>
<td>2 ib_iser,iscsi_tcp</td>
</tr>
<tr>
<td>scsi_transport_iscsi</td>
<td>63569</td>
<td>4 ib_iser,iscsi_tcp,libiscsi</td>
</tr>
<tr>
<td>dm_mirror</td>
<td>61121</td>
<td>0</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
<tr>
<td>usb_storage</td>
<td>116257</td>
<td>0</td>
</tr>
<tr>
<td>qla2xxx</td>
<td>100766</td>
<td>0</td>
</tr>
<tr>
<td>intermodule</td>
<td>37508</td>
<td>2 qla2xxx_conf,qla2xxx</td>
</tr>
<tr>
<td>lpfc</td>
<td>222068</td>
<td>2</td>
</tr>
<tr>
<td>scsi_transport_fc</td>
<td>73161</td>
<td>1 lpfc</td>
</tr>
<tr>
<td>mptsas</td>
<td>59857</td>
<td>3</td>
</tr>
<tr>
<td>mptbase</td>
<td>87649</td>
<td>2 mptsas,mptscsih</td>
</tr>
<tr>
<td>scsi_transport_sas</td>
<td>64833</td>
<td>1 mptsas</td>
</tr>
<tr>
<td>.</td>
<td></td>
<td>.</td>
</tr>
</tbody>
</table>

---

**Downloading the RDAC Driver**

The Linux OS RDAC driver is available for the following storage products:

- Sun Storage 6180 array
- Sun Storage 6580 and 6780 arrays
- Sun StorageTek 6000 array series
- Sun StorageTek 2500 array series

The Linux OS RDAC driver is compatible with array controller firmware version 06.19.25.16 (minimum). Array firmware is bundled with Sun StorageTek Common Array Manager (CAM) and is also available at the Sun Download Center (see the URL in Step 1 of the following procedure).

RDAC drivers are available from the Sun Download Center.

---

**Caution** – MPIO and RDAC cannot coexist on the same Linux host server.
1. To download the RDAC driver, go to:
   http://www.sun.com/download/index.jsp?tab=2#S

2. Select the package for the Linux OS platform.

3. Place the download in a non-volatile location on the system to be installed.

4. Load the standard HBA non-failover driver.

Note – Emulex and Qlogic failover drivers are not compatible with the RDAC driver.

5. On the SuSE operating system, check the base HBA drivers are included in INITRD_MODULES in /etc/sysconfig/kernel for one of the following HBAs:

<table>
<thead>
<tr>
<th>TABLE 1 HBA Driver Names</th>
</tr>
</thead>
<tbody>
<tr>
<td>HBA</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>QLogic</td>
</tr>
<tr>
<td>LSI</td>
</tr>
<tr>
<td>Emulex or IBM Emulex</td>
</tr>
</tbody>
</table>
6. Make sure the kernel source tree is installed for the kernel version to be built against.

---

**Installing the RDAC Driver**

The following procedure describes how to install the RDAC driver for Red Hat RHEL 5, Red Hat Advanced Server 4, and Red Hat SuSE.

- **Red Hat RHEL 5 package:** rdac-LINUX-09.03.0B05.0042-source.tar.gz
- **Red Hat Advanced Server 4 package:** rdac-LINUX-09.03.0C02.0042-source.tar.gz

1. **Unpack the source code using the `tar` command:**
   ```
   # tar -zxvf rdac-LINUX-xx.xx.xxxx.xxxx-source.tar.gz
   ``
   where `xx.xx.xxxx.xxxx` is the RDAC version you downloaded.
   The files are copied to the `linuxrdac-xx.xx.xx` directory.

2. **Change to the directory containing the unpacked files.** For example,
   ```
   # cd linuxrdac-09.03.C2.13
   ``

3. **Remove any earlier version drivers that are in this directory by running the `make clean` command:**
   ```
   # make clean -r
   make V=0 -C /lib/modules/2.6.18-8.el5xen/build M=/PTS/linuxrdac-09.03.C2.13 MODVERDIR=/lib/modules/2.6.18-8.el5xen/build/.tmp_versions SUBDIRS=/PTS/linuxrdac-09.03.C2.13 clean
   make[1]: Entering directory `/usr/src/kernels/2.6.18-8.el5-xen-x86_64`
   make[1]: Leaving directory `/usr/src/kernels/2.6.18-8.el5-xen-x86_64`
   rm -f ./mpp_linux_sys_headers/mppLnx26p_spinlock.h
   mppLnx_Spinlock_Size
   rm -f Module.symvers
   rm -f mppUtil
   rm -f genuniqueid
   ```
4. Compile all driver modules and utilities by running the `make` command:

```
# make
make V=0 -C/lib/modules/2.6.18-8.el5xen/build M=/PTS/linuxrdac-09.03.C2.13 MODVERDIR=/lib/modules/2.6.18-8.el5xen/build/.tmp_versions SUBDIRS=/PTS/linuxrdac-09.03.C2.13 modules
make[1]: Entering directory '/usr/src/kernels/2.6.18-8.el5-xen-x86_64'
  CC [M] /PTS/linuxrdac-09.03.C2.13/MPP_hba.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_upper.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_sysdep.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppCmn_s2tos3.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppCmn_SysInterface.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhbamisc.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhbatask.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhba.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhbaproc.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhbalib.o
  CC [M] /PTS/linuxrdac-09.03.C2.13/mppLnx26p_vhbaio.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppUpper.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppVhba.o
Building modules, stage 2.
MODPOST
  CC /PTS/linuxrdac-09.03.C2.13/mppUpper.mod.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppUpper.ko
  CC /PTS/linuxrdac-09.03.C2.13/mppVhba.mod.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppVhba.ko
make[1]: Leaving directory '/usr/src/kernels/2.6.18-8.el5-xen-x86_64'
make V=0 -C/lib/modules/2.6.18-8.el5xen/build M=/PTS/linuxrdac-09.03.C2.13 MODVERDIR=/lib/modules/2.6.18-8.el5xen/build/.tmp_versions SUBDIRS=/PTS/linuxrdac-09.03.C2.13 modules
make[1]: Entering directory '/usr/src/kernels/2.6.18-8.el5-xen-x86_64'
Building modules, stage 2.
MODPOST
  CC /PTS/linuxrdac-09.03.C2.13/mppUpper.mod.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppUpper.ko
  CC /PTS/linuxrdac-09.03.C2.13/mppVhba.mod.o
  LD [M] /PTS/linuxrdac-09.03.C2.13/mppVhba.ko
make[1]: Leaving directory '/usr/src/kernels/2.6.18-8.el5-xen-x86_64'
gcc -D__KERNEL__ -I/lib/modules/2.6.18-8.el5xen/build/include mppLnx26p_spinlock_size.c -o mppLnx_Spinlock_Size
```
Sample output continued from previous page

```bash
make uninstall
MPP driver is currently not installed on the system
[root@va64-x4100d-sca11 linuxrdac-09.03.C2.13]
```

5. Uninstall any existing RDAC drivers using the `make uninstall` command:

```bash
# make uninstall
MPP driver is currently not installed on the system
[root@va64-x4100d-sca11 linuxrdac-09.03.C2.13]
```

6. Install the new RDAC driver using the `make install` command. This command:
   - Copies the driver modules to the kernel module tree.
- Builds the new RAMdisk image (mpp-`uname -r'.img) which includes the RDAC driver modules and all driver modules that are needed at boot time.

```bash
# make install
make V=0 -C/lib/modules/2.6.18-8.el5xen/build M=
/PTS/linuxrdac-09.03.C2.13 MODVERDIR=/lib/modules/2.6.18-
8.el5xen/build/.tmp_versions SUBDIRS=/PTS/linuxrdac-09.03.C2.13
modules
make[1]: Entering directory `/usr/src/kernels/2.6.18-8.el5-xen-
x86_64`
    Building modules, stage 2.
    MODPOST
make[1]: Leaving directory `/usr/src/kernels/2.6.18-8.el5-xen-
x86_64`
make V=0 -C/lib/modules/2.6.18-8.el5xen/build M=
/PTS/linuxrdac-09.03.C2.13 MODVERDIR=/lib/modules/2.6.18-
8.el5xen/build/.tmp_versions SUBDIRS=/PTS/linuxrdac-09.03.C2.13
modules
make[1]: Entering directory `/usr/src/kernels/2.6.18-8.el5-
xen-x86_64`
    Building modules, stage 2.
    MODPOST
make[1]: Leaving directory `/usr/src/kernels/2.6.18-8.el5-xen-
x86_64`
/bin/bash ./genfileattributes bld
gcc -I/PTS/linuxrdac-09.03.C2.13 -I/PTS/linuxrdac-
09.03.C2.13/mpp_linux_headers/ -I/PTS/linuxrdac-
09.03.C2.13/mpp_linux_sys_headers/ -c
./utility/mppUtil26p_sysdep.c -o mppUtilSysdep.o
gcc mppUtil.o mppUtilSysdep.o -o mppUtil
Checking Host Adapter Configuration...
Unsupported Host Adapter Model: Product ID fc10(rev01)
Unsupported Host Adapter Model: Product ID fc10(rev01)
Detected 2 Emulex Host Adapter Port(s) on the system
Detected 1 LSI Host Adapter Port(s) on the system
Detected 2 QLogic Host Adapter Port(s) on the system
Host Adapters from different supported vendors co-exists on
your system.
Unsupported model from Vendor Emulex  exists.
Please make sure that only one supported model of HBA is
connected to Storage Array.
```
7. Type **yes** when you are prompted to continue:

```plaintext
Do you want to continue (yes or no) ? **yes**  
Please wait while we modify the system configuration files. 
Your kernel version is 2.6.18-8.el5xen 
Preparing to install MPP driver against this kernel version... 
Generating module dependencies... 
Warning: Duplicate module options detected. 
  Option in /etc/modprobe.conf ( max_luns=256 ) takes 
  precedence over MPP default setting ( max_luns=512 ). 
Creating new MPP initrd image... 
  You must now edit your boot loader configuration file, 
  /boot/grub/menu.lst, to 
  add a new boot menu, which uses mpp-2.6.18-8.el5xen.img as 
  the initrd image. 
  Now Reboot the system for MPP to take effect. 
  The new boot menu entry should look something like this 
  (note that it may 
  vary with different system configuration): 
  ...
  
  title Red Hat Linux (2.6.18-8.el5xen) with MPP support 
  root (hd0,5) 
  kernel /vmlinuz-2.6.18-8.el5xen ro root=LABEL=RH9 
  initrd /mpp-2.6.18-8.el5xen.img 
  ...
MPP driver package has been successfully installed on your 
system.
```
8. Follow the on-screen instructions to modify the `grub.conf` file. Change the module and initrd names as follows:

<table>
<thead>
<tr>
<th>Original Name</th>
<th>Revised Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>module /initrd-2.6.18-8.el5xen.img</td>
<td>module /mpp-2.6.18-8.el5xen.img</td>
</tr>
<tr>
<td>initrd /initrd-2.6.18-8.el5.img</td>
<td>initrd /mpp-2.6.18-8.el5xen.img</td>
</tr>
</tbody>
</table>

The content of the `grub.conf` file looks similar to the following. (The revised module and initrd names are shown in bold text.)

```plaintext
# boot=/dev/sda
default=0
timeout=5
title Red Hat Enterprise Linux Server (2.6.18-8.el5xen)
   root (hd0,0)
   kernel /xen.gz-2.6.18-8.el5 com1=9600n8
   module /vmlinuz-2.6.18-8.el5xen ro root=LABEL=/ console=ttyS0,9600n8 rhgb quiet
   module /mpp-2.6.18-8.el5xen.img
   title Red Hat Enterprise Linux Server-base (2.6.18-8.el5)
   root (hd0,0)
   kernel /vmlinuz-2.6.18-8.el5 ro root=LABEL=/ console=ttyS0,9600n8 rhgb quiet
   initrd /mpp-2.6.18-8.el5xen.img
```

9. Reboot the Linux server.

```plaintext
# reboot
```

10. After the server is rebooted, log in and verify the driver stack is properly loaded using the `lsmod` command. The driver stack for Red Hat and SuSE platforms are as follows:

- Red Hat platforms—scsi_mod, sd_mod, sg, mppUpper, mppVhba, and base level HBA drivers (for example, qla2xxx, qla2300) are loaded.
SuSE platforms—sg, mppUpper, mppVhba, and base level HBA drivers (for example, qla2xxx, qla2300) are loaded.

```bash
# modinfo qla2xxx
filename: /lib/modules/2.6.18-8.el5xen/kernel/drivers/scsi/qla2xxx/qla2xxx.ko
version: 8.02.08
license: GPL
description: QLogic Fibre Channel HBA Driver
author: QLogic Corporation
srcversion: F368B42790B83A8F18E4151
alias: pci:v00001077d00002532sv*sd*sc*i*
```

```bash
# modinfo lpfc
filename: /lib/modules/2.6.18-8.el5xen/kernel/drivers/scsi/lpfc/lpfc.ko
version: 0:8.1.10.12
author: Emulex Corporation - tech.support@emulex.com
description: Emulex LightPulse Fibre Channel SCSI driver
license: GPL
srcversion: 13F4CC10783BE62B7822C3B
```

11. Verify the content of the `modprobe.conf.mppappend` file:

```bash
# more /opt/mpp/modprobe.conf.mppappend
### BEGIN OF MPP Driver Changes ###
    options scsi_mod max_report_luns=256 max_luns=256
    alias scsi_hostadapter99 mppVhba
    alias scsi_hostadapter98 mptsas
    options qla2xxx ql2xfailover=0 ql2xretrycount=3
    ql2xprocessnotready=0 qlport_down_retry=35
    alias scsi_hostadapter96 qla2xxx
    options lpfc lpfc_nodev_tmo=60
    alias scsi_hostadapter97 lpfc
### END OF MPP Driver Changes ###
```

12. Verify the RDAC driver discovered the available physical LUNs and created virtual LUNs by typing the following command:

```bash
# ls -lR /proc/mpp
/proc/mpp:
total 0
```
13. Verify all drivers are running.

```
# lsmod | grep qla
qla2xxx        1007660  0
intermodule    37508   1 qla2xxx
scsi_mod      184057   16 ib_iser,iscsi_tcp,
               libiscsi,scsi_transport_iscsi,sr_mod,mppVhba,lpfc,
               scsi_transport_fc,qla2xxx,usb_storage,mptsas,mptscsih,scsi_transp
               ort_sas,mppUpper,sg,sd_mod

# modinfo lpfc | grep version
version: 0:8.0.16.6_p3 37504CE1A0639D7C3FD1BAB
```
14. Verify RDAC is configured with the MPP drivers.

```
# more /etc/modprobe.conf
alias eth0 e1000
alias eth1 e1000
alias eth2 e1000
alias eth3 e1000
alias scsi_hostadapter mptbase
alias scsi_hostadapter1 mptsas
alias scsi_hostadapter2 lpfc
alias scsi_hostadapter3 qla2xxx
alias scsi_hostadapter4 usb-storage
install qla2xxx /sbin/modprobe qla2xxx_conf; /sbin/modprobe -
- ignore-install qla2xxx
remove qla2xxx /sbin/modprobe -r --first-time --ignore-remove
qla2xxx && ( /sbin/modprobe -r --ignore-remove qla2xxx_conf; )
alias qla2100 qla2xxx
alias qla2200 qla2xxx
alias qla2300 qla2xxx
alias qla2322 qla2xxx
alias qla2400 qla2xxx
options scsi_mod max_luns=256
alias scsi_hostadapter5 lpfc
### BEGIN MPP Driver Comments ###
remove mppUpper if [ 'ls -a /proc/mpp | wc -l' -gt 2 ]; then
  echo -e "Please Unload Physical HBA Driver prior to unloading
  mppUpper."; else /sbin/modprobe -r --ignore-remove mppUpper; fi
# Additional config info can be found in
/opt/mpp/modprobe.conf.mppappend.
# The Above config info is needed if you want to make mkinitrd
manually.
# Please read the Readme file that came with MPP driver for
building RamDisk manually.
# Edit the '/etc/modprobe.conf' file and run 'mppUpdate' to
create Ramdisk dynamically.
### END MPP Driver Comments ###
```

15. Verify the disk devices are mapped from the array to the host, as follows:

- Check the array mappings in Common Array Manager, to ensure that you have
  mapped a volume on the array to the Linux host.
- Use the `/usr/sbin/mppUtil -a <array_name_in_CAM>` command to get a
  report on how many devices can be seen by the MPP driver (see “Multipath
  Failover Driver Utilities” on page 14).
- View the `/proc/scsi/mpp` file for array devices.
View /proc/scsi/<base_hba_driver_name>/[0-9|a-z], for information about what devices can be seen.

See TABLE 1 for a list of HBA base driver names. The files in this directory are labeled 0 through 9 or A through Z. Each file represents a port on the HBA, which includes World Wide Names of storage devices and LUNs reported.

Multipath Failover Driver Utilities

The MPP RDAC driver installation creates three utilities for your use:

- mppUtil
- lsvdev
- mppSupport

mppUtil Utility

The mppUtil utility is a general purpose command-line driven utility that works only with MPP-based RDAC solutions. The utility instructs RDAC to perform various maintenance tasks, but you can also use the utility to troubleshoot failover problems.

To use the mppUtil utility, type this command and press enter.

```
mppUtil [-a target_name] [-c wwn_file_name] [-d debug_level] [-e error_level] [-g virtual_target_id] [-I host_num] [-o feature_action_name [=value][, SaveSettings]] [-s "failback" | "avt" | "busscan" | "forcerebalance"] [-S] [-U] [-V]
```

**Note** – The quotation marks must surround the parameters.
The mppUtil utility is a cross-platform tool. Some parameters might not have a meaning in a particular operating system environment. A description of each parameter follows.

### TABLE 2 mppUtil Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>-a target_name</code></td>
<td>Shows the RDAC driver’s internal information for the specified virtual target_name (storage array name). If a target_name value is not included, the -a parameter shows information about all the storage arrays that are currently detected by this host.</td>
</tr>
<tr>
<td><code>-c wwn_File_Name</code></td>
<td>Clears the WWN file entries. This file is located at /var/mpp with the extension .wwn.</td>
</tr>
</tbody>
</table>
| `-d debug_level` | Sets the current debug reporting level. This option only works if the RDAC driver has been compiled with debugging enabled. Debug reporting is comprised of two segments. The first segment refers to a specific area of functionality, and the second segment refers to the level of reporting within that area. The debug_level is one of these hexadecimal numbers:  
  • 0x20000000—Shows messages from the RDAC driver’s init() routine.  
  • 0x10000000—Shows messages from the RDAC driver’s attach() routine.  
  • 0x08000000—Shows messages from the RDAC driver’s ioctl() routine.  
  • 0x04000000—Shows messages from the RDAC driver’s open() routine.  
  • 0x02000000—Shows messages from the RDAC driver’s read() routine.  
  • 0x01000000—Shows messages related to HBA commands.  
  • 0x00800000—Shows messages related to aborted commands.  
  • 0x00400000—Shows messages related to panic dumps.  
  • 0x00200000—Shows messages related to synchronous I/O activity.  
  • 0x00000001—Debug level 1.  
  • 0x00000002—Debug level 2.  
  • 0x00000004—Debug level 3.  
  • 0x00000008—Debug level 4.  

These options can be combined with the logical or operator to provide multiple areas and levels of reporting as needed.
### TABLE 2  mppUtil Parameters (Continued)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>(-e) error_level</td>
<td>Sets the current error reporting level to error_level, which can have one of these values:</td>
</tr>
<tr>
<td></td>
<td><strong>0</strong>—Show all errors.</td>
</tr>
<tr>
<td></td>
<td><strong>1</strong>—Show path failover, controller failover, retryable, fatal, and recovered errors.</td>
</tr>
<tr>
<td></td>
<td><strong>2</strong>—Show path failover, controller failover, retryable, and fatal errors.</td>
</tr>
<tr>
<td></td>
<td><strong>3</strong>—Show path failover, controller failover, and fatal errors. This is the default setting.</td>
</tr>
<tr>
<td></td>
<td><strong>4</strong>—Show controller failover and fatal errors.</td>
</tr>
<tr>
<td></td>
<td><strong>5</strong>—Show fatal errors.</td>
</tr>
<tr>
<td>(-g) virtual_target_id</td>
<td>Display the RDAC driver’s internal information for the specified virtual_target_id.</td>
</tr>
<tr>
<td>(-I) host_num</td>
<td>Prints the maximum number of targets that can be handled by that host.</td>
</tr>
<tr>
<td></td>
<td>Here, host refers to the HBA drivers on the system and includes the RDAC driver. The host number of the HBA driver is given as an argument. The host numbers assigned by the Linux middle layer start from 0. If there are two ports on the HBA card, host numbers 0 and 1 would be taken up by the low-level HBA driver, and the RDAC driver would be at host number 2. Use /proc/scsi to determine the host number.</td>
</tr>
<tr>
<td>(-o) feature_action_name</td>
<td>Troubleshoots a feature or changes a configuration setting. Without the SaveSettings keyword, the changes only affect the in-memory state of the variable. The SaveSettings keyword changes both the in-memory state and the persistent state. You must run mppUpdate to reflect these changes in inird image before rebooting the server. Some example commands are:</td>
</tr>
<tr>
<td></td>
<td>* mppUtil (-o)—Displays all the available feature action names.</td>
</tr>
<tr>
<td></td>
<td>* mppUtil (-o) ErrorLevel=0x2—Sets the ErrorLevel parameter to 0x2 (affects only the in-memory state).</td>
</tr>
<tr>
<td>(-s) { &quot;failback&quot;</td>
<td>Manually initiates one of the RDAC driver’s scan tasks. A failback scan will cause the RDAC driver to reattempt communications with any failed controllers. An avt scan causes the RDAC driver to check whether AVT has been enabled or disabled for an entire storage array. A busscan scan causes the RDAC driver to go through its unconfigured devices list to see if any of them have become configured. A forcerebalance scan will cause the RDAC driver to move storage array volumes to its preferred controller and ignore the value of the DisableLunRebalance configuration parameter of the RDAC driver.</td>
</tr>
<tr>
<td>}</td>
<td></td>
</tr>
</tbody>
</table>
**lsdev Utility**

The `lsdev` utility provides a map of your array LUNs to their Linux block devices. To use the `lsdev` utility, type this command and press enter.

```
# /opt/mpp/lsdev
```

For example:

```
/usr/sbin # /opt/mpp/lsvdev
```

```
Array Name Lun sd device
------------------------
myarray 0 -> /dev/sdf
myarray 1 -> /dev/sdg
myarray 2 -> /dev/sdh
myarray 3 -> /dev/sdi
```

**mppSupport Utility**

The `mppSupport` utility provides information about your system, array connection, and configuration for Sun Services to assist you with any problems you might have.

To use the `mppSupport` utility, type this command and press enter.

```
# /opt/mppSupport
```

The collected support data is saved in the file:

```
/tmp/mppSupportdata_myhost_<RDACversion>_dateandtime.tar.gz
```
Linux RDAC Configuration Settings

The RDAC driver contains configuration settings that can modify the behavior of the driver. Any changes to the settings take effect on the next reboot of the host. The default values listed here are the platform-independent settings. Many of these values are overridden by the failover installer for Linux. For Linux, the configuration settings are located in the `/etc/mpp.conf` file.

After you change a configuration value, you must run the `mppUpdate` utility and reboot your Linux server for the change to take effect.

**Caution** – You might lose access to the storage array if you change these settings from their configured values.

<table>
<thead>
<tr>
<th>Setting</th>
<th>OS Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MaxPathsPerController</td>
<td>4</td>
<td>The maximum number of paths (logical endpoints) supported per controller. The total number of paths to the storage array is the MaxPathsPerController value multiplied by the number of controllers.</td>
</tr>
</tbody>
</table>
| ScanInterval        | 60               | The interval, in seconds, during which the failover driver will check for these conditions:  
|                     |                  | • A change in preferred ownership for a LUN  
|                     |                  | • Attempt to rebalance LUNs to their preferred paths  
|                     |                  | • A change in AVT enabled/disabled status |
| ErrorLevel          | 3                | Determines which errors to log. The valid range is from 0 to 4.  
|                     |                  | • 0—Display all errors  
|                     |                  | • 1—Display path failover errors, controller failover errors, retry errors, fatal errors, and recovered errors  
|                     |                  | • 2—Display path failover errors, controller failover errors, retry errors, and fatal errors  
|                     |                  | • 3—Display path failover errors, controller failover errors, and fatal errors (this is the default setting)  
|                     |                  | • 4—Display controller failover errors, and fatal errors |
**TABLE 3** RDAC Configuration Settings for the Linux OS (Continued)

<table>
<thead>
<tr>
<th>Setting</th>
<th>OS Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SelectionTimeoutRetry Count</td>
<td>0</td>
<td>The number of times a selection timeout is retried for an I/O request before the path fails. If another path to the same controller exists, the I/O is retried. If no other path exists, a failover takes place. If no valid paths exist to the alternate controller, the I/O is failed.</td>
</tr>
<tr>
<td>CommandTimeoutRetry Count</td>
<td>1</td>
<td>The number of times a command timeout is retried for an I/O request before the path fails. If another path to the same controller exists, the I/O is retried. If another path does not exist, a failover takes place. If no valid paths exist to the alternate controller, the I/O is failed.</td>
</tr>
<tr>
<td>UaRetryCount</td>
<td>10</td>
<td>The number of times a Unit Attention status from a LUN is retried. This parameter does not apply to UA conditions due to Quiescence In Progress.</td>
</tr>
<tr>
<td>SynchTimeout</td>
<td>120</td>
<td>The timeout, in seconds, for synchronous I/O requests generated internally by the failover driver. Examples of internal requests include those related to rebalancing, path validation, and issuing of failover commands.</td>
</tr>
<tr>
<td>DisableLunRebalance</td>
<td>0</td>
<td>Provides control over the LUN failback behavior of rebalancing LUNs to their preferred paths. The following values are possible:</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 0—LUN rebalance is enabled for both AVT and non-AVT modes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 1—LUN rebalance is disabled for AVT and enabled for non-AVT mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 2—LUN rebalance is enabled for AVT and disabled for non-AVT mode.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• 3—LUN rebalance is disabled for both AVT and non-AVT modes.</td>
</tr>
<tr>
<td>S2ToS3Key</td>
<td>Unique key</td>
<td>Value of the SCSI-3 reservation key generated during failover driver installation.</td>
</tr>
</tbody>
</table>

**Wait Time Settings**

When the failover driver receives an I/O request for the first time, the failover driver logs timestamp information for the request. If a request returns an error and the failover driver decides to retry the request, the current time is compared with the original timestamp information. Depending on the error and the amount of time that
has elapsed, the request is retried to the current owning controller for the LUN or a failover is performed and the request sent to the alternate controller. This process is known as a “Wait Time.”

**Caution** – Possible loss of data access—If you change these settings from their configured values, you might lose access to the storage array.

The configuration settings can be found in the `/etc/mpp.conf` file.

**TABLE 4  Wait Time Settings**

<table>
<thead>
<tr>
<th>Configuration Name</th>
<th>Default Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>NotReadyWaitTime</td>
<td>300</td>
<td>The time, in seconds, a Not Ready condition (SK 0x06, ASC/ASCQ 0x04/0x01) is allowed before failover is performed.</td>
</tr>
<tr>
<td>BusyWaitTime</td>
<td>300</td>
<td>The time, in seconds, a Busy condition is allowed for a failover is performed.</td>
</tr>
<tr>
<td>QuiescenceWaitTime</td>
<td>300</td>
<td>The time, in seconds, a Quiescence condition (SK 0x06, ASC/ASCQ 0x8b/0x02) is allowed before a failover is performed.</td>
</tr>
<tr>
<td>ControllerIoWaitTime</td>
<td>120</td>
<td>Provides an upper-bound limit, in seconds, an I/O is retried on a controller regardless of retry status before a failover is performed. If the limit is exceeded on the alternate controller the I/O is again attempted on the original controller. This process continues until the ArrayIoWaitTime limit is reached.</td>
</tr>
<tr>
<td>ArrayIoWaitTime</td>
<td>240</td>
<td>Provides an upper bound limit, in seconds, an I/O is retried to the storage array regardless of which controller the request is attempted to. Once this limit is exceeded, the I/O is returned with a failure status.</td>
</tr>
</tbody>
</table>
Updating the initrd Image

1. After you change a configuration value, run the `mppUpdate` utility and reboot the Linux server for the change to take effect. An MPP initrd image is created, as shown in the following example:

```
# mppUpdate
Unsupport Host Adapter Model: Product ID fc10(rev01)
Unsupported Host Adapter Model: Product ID fc10(rev01)
Detected 2 Emulex Host Adapter Port(s) on the system
Detected 1 LSI Host Adapter Port(s) on the system
Detected 2 QLogic Host Adapter Port(s) on the system
Host Adapters from different supported vendors co-exists on your system.
  Unsupported model from Vendor Emulex exists.
Warning: Duplicate module options detected.
  Option in /etc/modprobe.conf ( max_luns=256 ) takes precedence over MPP default setting ( max_luns=512 ).
  Creating new MPP initrd image...
```

2. Reboot the host server:

```
# reboot
```

Troubleshooting Linux RDAC Drivers

Question: How do I get logs from RDAC in the Linux OS?

Answer: Use the `mppSupport` command to obtain several logs related to RDAC. The `mppSupport` command is found in the `/opt/mpp/mppSupport` directory. The command creates a file named `mppSupportdata_hostname_RDAC version_datetime.tar.gz` in the `/tmp` directory.

Question: How does persistent naming work?

Answer: The Linux OS SCSI device names can change when the host system restarts. Use a utility, such as `devlabel`, to create user-defined device names that will map devices based on a unique identifier. The `udev` method is the preferred method for SLES10 and RHEL 5.

Question: What must I do after applying a kernel update?

Answer: After you apply the kernel update and start the new kernel, perform these steps to build the RDAC Initial Ram Disk image (initrd image) for the new kernel:
1. Change the directory to the Linux RDAC source code directory.

2. Type the following command, and press Enter.
   
   ```
   make uninstall
   ```

3. Reinstall RDAC.

**Question:** What is the Initial Ram Disk Image (*initrd* image), and how do I create a new *initrd* image?

**Answer:** The *initrd* image is automatically created when the driver is installed by using the `make install` command. The boot loader configuration file must have an entry for this newly created image.

The *initrd* image is located in the boot partition. The file is named `mpp-uname-r.img`.

For a driver update, if the system already has a previous entry for RDAC, the system administrator must modify the existing RDAC entry in the boot loader configuration file. In most of the cases, no change is required if the kernel version is the same.

To create a new *initrd* image, type the following command and press Enter.

```
# mppUpdate
```

The old image file is overwritten with the new image file.

If third-party drivers are needed to be added to the *initrd* image, change the `/etc/sysconfig/kernel` file (SuSE) with the third-party driver entries. Run the `mppUpdate` command again to create a new *initrd* image.

**Question:** How can I see if volumes have been added?

**Answer:** Run `hot_add -s` or `hot_add` to add the newly mapped volumes.

**Question:** How do I remove unmapped or disconnected devices from the existing host?

**Answer:** Run `hot_add -d` to remove all unmapped or disconnected devices.

**Question:** What if I remap a LUN from the storage array?

**Answer:** Run `hot_add -u` to update the host with the changed LUN mapping.

**Question:** What if I change the size of the LUN on the storage array?

**Answer:** Run `hot_add -c` to change the size of the LUN on the host.

**Question:** How do I make sure that RDAC finds the available storage arrays?
Answer: To make sure that the RDAC driver has found the available storage arrays and created virtual storage arrays for them, type the following commands, and press Enter after each command.

```
# ls -lr /proc/mpp
# mppUtil -a
# /opt/mpp/lsvdev
```

To show all attached and discovered volumes, type the following command and press Enter.

```
# cat /proc/scsi/scsi
```

Question: What must I do if I receive this message?

---

**Caution** — Changing the storage array name can cause host applications to lose access to the storage array if the host is running certain path failover drivers. If any of your hosts are running path failover drivers, please update the storage array name in your path failover driver’s configuration file before rebooting the host machine to insure uninterrupted access to the storage array. Refer to your path failover driver documentation for more details.

---

**Answer:** The path failover drivers that cause this warning are the RDAC drivers on Linux. The storage array user label is used for storage system-to-virtual target ID binding in the RDAC driver. For the Linux OS, change the following file to add the storage array user label and its virtual target ID.

```
# more /var/mpp/devicemapping
```

---

**Determining if a Path Failed**

With the failover driver, two cases determine if a path has failed:

- A path to a controller has failed, but the driver has other paths to the same controller that it can use. This situation causes a degraded condition and must be corrected before the controller can switch to an alternate path.

  An entry is made in the OS system log that shows that the failover driver has detected a path failure. CAM does not generate an alarm because no internal problems exist for the array.

- All of the paths to a controller have failed, or the controller itself has failed, and the other controller is now used to service the I/O.
CAM generates the “Volume Not on Preferred Path” alarm for all volumes affected by this scenario. If the array administrator has configured notifications in CAM, the administrator will receive email from CAM or a configured SNMP server. You also have the option of opening a service request using the Auto Service Request (ASR) feature of CAM. The resultant message and alarm will provide information about the fault, along with possible recovery instructions.

Error Levels

The failover driver has five error levels for messages that are logged to the Linux OS error log:

- Fatal errors
- Controller failover events
- Path failover events
- Retry errors
- Recovered errors

Fatal Errors

TABLE 5 lists the possible Linux OS fatal driver errors.

<table>
<thead>
<tr>
<th>Fatal Failover Driver Errors for the Linux OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Busy wait time exceeded</td>
</tr>
<tr>
<td>Busy wait time exceeded on failover command</td>
</tr>
<tr>
<td>Close failed on virtual bus node</td>
</tr>
<tr>
<td>Error trying to insert a new volume path</td>
</tr>
<tr>
<td>Discovered device is not a storage array</td>
</tr>
<tr>
<td>Duplicate storage array name found</td>
</tr>
<tr>
<td>Error trying to allocate data structures</td>
</tr>
<tr>
<td>Error trying to create virtual target</td>
</tr>
<tr>
<td>Error trying to determine controller slot</td>
</tr>
<tr>
<td>Error trying to determine state of volume</td>
</tr>
</tbody>
</table>

**Fatal Error Message**

- Busy wait time exceeded
- Busy wait time exceeded on failover command
- Close failed on virtual bus node
<table>
<thead>
<tr>
<th>Command aborted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Command timeout retry count exceeded</td>
</tr>
<tr>
<td>Controller unreachable without failback to current (which is disabled)</td>
</tr>
<tr>
<td>Discovered device is not a storage array</td>
</tr>
<tr>
<td>Duplicate storage array name found</td>
</tr>
<tr>
<td>Error trying to allocate data structures</td>
</tr>
<tr>
<td>Error trying to create virtual target</td>
</tr>
<tr>
<td>Error trying to determine AVT state of volume</td>
</tr>
<tr>
<td>Error trying to determine controller slot</td>
</tr>
<tr>
<td>Error trying to determine maximum number of volumes supported by storage array</td>
</tr>
<tr>
<td>Error trying to determine storage array name</td>
</tr>
<tr>
<td>Error trying to get World Wide Identifier (WWID) of volume</td>
</tr>
<tr>
<td>Error trying to insert a new controller path</td>
</tr>
<tr>
<td>Error trying to insert a new volume</td>
</tr>
<tr>
<td>Error trying to insert a new volume path</td>
</tr>
<tr>
<td>Error trying to match discovered controller to existing controller</td>
</tr>
<tr>
<td>Failover command failed</td>
</tr>
<tr>
<td>Failover failed, no path to volume</td>
</tr>
<tr>
<td>Failover failed, unable to allocate memory</td>
</tr>
<tr>
<td>Hardware error</td>
</tr>
<tr>
<td>Incorrect close type for virtual bus node</td>
</tr>
<tr>
<td>Incorrect open type for virtual bus node</td>
</tr>
<tr>
<td>Inquiry wait time exceeded</td>
</tr>
<tr>
<td>Maximum number of storage arrays exceeded</td>
</tr>
<tr>
<td>Maximum paths per controller exceeded</td>
</tr>
<tr>
<td>No paths available to start I/O</td>
</tr>
<tr>
<td>No paths available to start I/O, failing over</td>
</tr>
<tr>
<td>Not Ready wait time exceeded</td>
</tr>
<tr>
<td>Open failed on virtual bus node</td>
</tr>
<tr>
<td>Quiescence wait time exceeded</td>
</tr>
<tr>
<td>Received the Illegal Command error from the storage array</td>
</tr>
</tbody>
</table>
Controller and Path Failover Events

The following items are examples of failover driver controller events and path failover events:

- Change of AVT setting detected
- Failover command issued
- AVT failover invoked
- Path failure detected

TABLE 5  Fatal Failover Driver Errors for the Linux OS (Continued)

<table>
<thead>
<tr>
<th>Error Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request sense failure</td>
</tr>
<tr>
<td>Selection timeout retry count exceeded</td>
</tr>
<tr>
<td>Sense key hardware error received</td>
</tr>
<tr>
<td>Storage array found to be in state of Not Ready and Not Becoming Ready</td>
</tr>
<tr>
<td>Unit Attention retry count exceeded</td>
</tr>
<tr>
<td>Unrecognizable OS status</td>
</tr>
<tr>
<td>Unrecognized SCSI status</td>
</tr>
<tr>
<td>Unrecognizable sense key received</td>
</tr>
<tr>
<td>Volume number exceeds the maximum configured volume number</td>
</tr>
<tr>
<td>World Wide Identifier (WWID) for discovered volume does not match that of same</td>
</tr>
<tr>
<td>volume found on another path</td>
</tr>
</tbody>
</table>

World Wide Identifier (WWID) for discovered volume does not match that of same volume found on another path