Readme File

FC HBA Driver for Linux Kernel 2.6.x

Table of Contents

1. Package Contents
2. OS Support
3. Supported Features
4. Using the Driver Package
   4.1 Download the Driver Source
   4.2 Installing the Driver
5. Driver System Parameters
6. SNIA API Library Package (combo package)
   6.1 Installing SNIA API Library
   6.2 Uninstalling SNIA API Library
7. Additional Notes
   7.1 Failover Support
   7.2 Persistent Binding
   7.3 Configuration Data
   7.4 Booting from SAN
   7.5 Compilation Messages
   7.6 Dynamically Modifying SCSI Black-list Entries
8. IP Support
   8.1 Load the Driver Manually Using insmod or modprobe
   8.2 Configuring Interfaces
9. Contacting Support

1. Package Contents

The following table describes the contents provided in the FC HBA Driver for Linux package.

Filename	Description
drvrsetup	Linux installation script.
libinstall	Script file to install/setup HBA API library.
libremove	Linux script to remove libqlsdm.so from `/usr/lib` or/and # `# /usr/lib64`.
qla2x00src-v*x.yy.zz*.tgz	Compressed package that contains the driver and all support documents. **NOTE**: *`x.yy.zz`* represents the driver package version number.
qlapi-v*x.yy*build`-rel.tgz	Compressed package that contains the HBA API software and all support documents. **NOTE**: *`x.yy`* represents the version and *bb* represents the build number.
README.qla2xxx	Text version of this package readme file.

2. OS Support

The FC HBA Driver for Linux is compatible with the following OS platforms.

Operating Systems		
OS Name	**OS Type**	**Hardware Platform**
Red Hat RHEL AS 4.0	32-bit/64-bit	Intel x86, Intel EM64T, AMD64, Intel IA64, and PPC64 platforms
Novell SLES 9	32-bit/64-bit	32-bit/64-bit on Intel x86, Intel EM64T, AMD64, Intel IA64, and PPC64 platforms
Novell SLES 10	32-bit/64-bit	64-bit on Intel x86, Intel EM64T, AMD64, Intel IA64, and PPC64 platforms

NOTE: For specific OS service packs (SP) and updates, refer to the descriptions where this software version is posted on the QLogic website (http://support.qlogic.com/support/drivers_software.asp).

3. Supported Features

The FC HBA Driver for Linux supports the following:

- FCAL - direct attach loop
- Point-to-point
- Fabric support
- Initiator mode only
- Fault recovery on down loops
- Persistent binding
- Extended LUN support up to 255 LUNs
- FC tape support
- Non-Failover and Failover capability

4. Using the Driver

This section provides procedures for downloading and installing the driver.

4.1 Download the Driver Source

To download the driver source:

1. Download the driver distribution file (`qla2x00-vx.yy.zz-dist.tgz`) from QLogic's website.
2. If prompted "`What would you like to do with this file?`" choose **Save this file to disk**.
3. Create a directory in the local hard drive and download the driver.

NOTE: Because the driver distribution file is now larger than 1.44 Mb, it cannot fit on a 1.44 Mb floppy disk; therefore, you must use a USB drive or local hard disk to download the file.

4.2 Installing the Driver

Driver installation makes extensive use of the `build.sh` script located in driver source (`extras/build.sh`). The following subsections describe driver installation:

- 4.2.1 Building the FC HBA Driver for Linux
- 4.2.2 Manually Loading the FC HBA Driver for Linux Using insmod or modprobe
- 4.2.3 Making a ramdisk Image to Load the Driver

4.2.1 Building the FC HBA Driver for Linux

From the source code, you can build a `qla2xxx.ko`, `qla2xxx_conf.ko`, `qla2300.ko`, `qla2322.ko`, and `qla2400.ko` for your host and load the driver manually using `modprobe` or automatically using a ramdisk image during system boot time.

1. In the directory that contains the source driver file (`qla2xxx-vx.yy.zz-dist.tgz`) use following commands:
   ```
   # tar -xvzf *.tgz
   # cd qlogic
   ```
   `# ./drvrsetup` (this extracts the source files directory in to the current directory)
   ```
   # cd qla2xxx-x.yy.zz
   ```
2. Build and install the driver modules from the source code by executing the `build.sh` script:
   ```
   # ./extras/build.sh install
   ```

This build script:

- Builds the driver .ko files.
- Copies the `.ko` files to the appropriate `/lib/modules/2.6.../kernel/drivers/scsi/qla2xxx` directory.
- Adds the appropriate directive in the `modprobe.conf[.local]` to remove the `qla2xxx_conf` module when unloading the qla2xxx modules.
- Updates the newly built `qla2xxx_conf.ko` module with any previously saved data in `/etc/qla2xxx.conf`.

NOTE: To load the driver manually, see section 4.2.2. To make a ramdisk image to load the driver during system boot time, see section 4.2.3.

4.2.2 Manually Loading the FC HBA Driver for Linux Using insmod or modprobe

Before loading the driver manually, build the driver binary from the driver source files as described in section 4.2.1.

- To load the driver directly from the local build directory, use the following commands in order:
  ```
  # insmod qla2xxx_conf.ko
  # insmod qla2xxx.ko
  ```
  `# insmod qla2300.ko` (for QLA23xx, QLA200 HBAs) or,
  `# insmod qla2322.ko` (for QLE23xx, QLA210 HBAs) or,
  `# insmod qla2400.ko` (for QLx24xx, QLE22x, QMx24xx, and QEM24xx HBAs)

  NOTE: To determine the HBA type installed based on the PCI Device ID, enter the command:
  ```
  # lspci | grep QLogic
  ```
- To load the driver using `modprobe`:
  `# modprobe -v qla2300` (for QLA23xx, QLA200 HBAs)
  or
  `# modprobe -v qla2322` (for QLE23xx, QLA210 HBAs)
  or
  `# modprobe -v qla2400` (for QLx24xx, QLE22x, QMx24xx, and QEM24xx HBAs)

  NOTE: The `modprobe -v qla2300` (or qla2322/qla2400) command automatically loads the `qla2xxx.ko` and `qla2xxx_conf` components.
- To unload the driver using `modprobe`:
  ```
  # modprobe -r qla2300
  ```
  This command unloads the `qla2300.ko` and `qla2xxx.ko` modules. If there are additional firmware-loader modules (e.g. `qla2400.ko`) which depend on `qla2xxx.ko`, then the `qla2xxx.ko` is not unloaded.
  ```
  # modprobe -r qla2xxx_conf
  ```

This command unloads qla2xxx_conf.ko.

4.2.3 Making a ramdisk Image to Load the Driver

Use the following steps to build a ramdisk image on RHEL 4 and SLES.

For **RHEL 4 ramdisk Image:**

To build the ramdisk image, follow these steps:

1. Follow the steps in section 4.1.
2. Install the driver module (*.ko) files to the appropriate kernel module directory:
   ```
   # ./extras/build.sh install
   ```
3. Edit the /etc/modprobe.conf file and add the following entries if they are not present:
   alias scsi_hostadapter1 qla2xxx_conf (SANsurfer use only)
   alias scsi_hostadapter2 qla2xxx
   alias scsi_hostadapter3 qla2322 (QLE236X) or qla2300 (QLA234X)
   or
   alias scsi_hostadapter4 qla2400 (QLA24XX/QLE24XX/QLE220)
4. Change to the /boot directory.
5. Back up the current ramdisk image:
   ```
   # cp -f initrd-2.6.[kernel_version].img initrd-2.6.[k_v].img.bak
   ```
6. Build the ramdisk image with the following command:
   ```
   # mkinitrd -f initrd-2.6.[kernel_version].img [kernel_version]
   ```
7. Reboot the system to load the ramdisk image with the QLogic driver.

For **SLES ramdisk Image:**

To build the ramdisk image on SLES, follow these steps:

1. Follow the steps in section 4.1.
2. Install the driver module (*.ko) files to the appropriate kernel module directory:
   ```
   # ./extras/build.sh install
   ```
3. Edit the /etc/sysconfig/kernel file and modify the INITRD_MODULES directive as follows:
   ```
   ...
   INITRD_MODULES=".... qla2xxx_conf qla2xxx qla2300 qla2322 qla2400"
   ...
   ```
   NOTE: Add the first module qla2xxx_conf (for SANSurfer) followed by qla2xxx. After qla2xxx, add the modules specific to the HBA installed in the system. For example:

   qla2xxx_conf: SANsurfer use only
   qla2xxx: Common module
   qla2300: For QLA234X
   qla2322: For QLE236X
   qla2400: For QLA24XX and QLA2XX
4. Change to the /boot directory.
5. Back up the current ramdisk image:
   ```
   # cp -f initrd-2.6.[kernel_version] initrd-2.6.[k_v].bak
   ```
6. Build the ramdisk image with the following command:
   ```
   # /sbin/mk_initrd
   ```
7. Reboot the system to load the ramdisk image with the QLogic driver.

5. Driver System Parameters

The driver gets its parameters when specified with the insmod command. For example:

```
insmod qla2xxx.ko ql2xextended_error_logging=1
```

If using the modprobe command, you must specify the parameters under the /etc/modprobe.conf[.local] file.

For example:

```
options qla2xxx ql2xextended_error_logging=1
```

To view a comprehensive list of parameters, enter the following command:

```
# /sbin/modinfo qla2xxx.ko
```

Driver Command Parameter Descriptions

Parameter	Description	Default

ql2xextended_error_logging	This parameter defines whether the driver prints verbose logging information. 0 to disable; 1 to enable.
	This parameter can also be updated dynamically (on kernels > 2.6.8) while the driver is loaded.
	To enable logging (depending on kernel version):
	$ echo 1 > /sys/module/qla2xxx/ql2xextended_error_logging
	or
	$ echo 1 >/sys/module/qla2xxx/parameters/ql2xextended_error_logging
	To disable (again depending on kernel version):
	$ echo 0 > /sys/module/qla2xxx/ql2xextended_error_logging
	or
	$ echo 0 > /sys/module/qla2xxx/parameters/ql2xextended_error_logging
		0

ql2xfailover	Defines whether failover mode is enabled or disabled. 0 to disable; 1 to enable.	1
ql2xmaxqdepth	This parameter defines the maximum queue depth reported to SCSI Mid-Level per device. The Queue depth specifies the number of outstanding requests per LUN	32
ql2xlogintimeout	This parameter defines the login timeout value in seconds during the initial login.	20 seconds
qlport_down_retry	This parameter defines how long to wait for a port that returns a PORT-DOWN status before returning I/O back to the OS.	0 (use value specified in NVRAM)
ql2xretrycount	This parameter defines the maximum number of SCSI mid-level retries allowed per command.	20 (standard mode value) 30 (failover mode value)
displayConfig	This parameter defines whether to display the current configuration. 0 - do not display the configuration; 1 - display the configuration.	0
Bind	This parameter defines the target persistent binding method to use. 0 - bind by Portname; 1 - bind by PortID.	0 (Portname binding)
ConfigRequired	This parameter defines how to bind devices. 0 - Present all devices discovered to the OS; 1 - Present only configured devices (i.e. the device defined in /etc/qla2xxx.conf) to the OS.	0
MaxPathsPerDevice	This parameter defines the maximum number of paths to a device at compile time only.	8
MaxRetriesPerPath	This parameter defines how many retries to perform on the current path before failing over to the next path in the path list.	3
MaxRetriesPerIo	This parameter defines total retries to do before failing the command and returning to the OS with selection timeout (DID_NO_CONNECT).	(MaxRetriesPerPath * MaxPathsPerDevice) + 1
QlFailoverNotifyType	This parameter defines type of failover notification mechanism to use when a failover or failback occurs. Certain storage systems require special CDBs to be issued to do failover or failback.	0 (none)
FailbackTime	This parameter defines the delay in seconds before a failback is performed to ensure all paths are available.	5 seconds

RecoveryTime	This parameter defines the time in seconds required before commands can be sent to the restored path.	10 seconds
ql2xautorestore	This parameter enables or disables the logic that restores the previous failed preferred path and/or controller for a given LUN. This option toggles the default state. Combine one or more of the following model numbers into an inclusion mask: 0x80 - MSA A/A (auto-restore disabled) 0x20 - HSV111, HSV101, HSV200, HSV210 (auto-restore disabled) 0x10 - DSXXX (auto-restore disabled) 0x04 - HSV110, HSV100 (auto-restore disabled) 0x02 - MSA1000 (auto-restore disabled) 0x01 - XP (auto-restore enabled)	0
ql2xlbType	This parameter defines the load balance method for the driver as static or dynamic. 0 (None) - Exposes LUNs on the first active path and make them the preferred path, or the first active optimize path and make them the preferred path (storages: MSA A/A and EVA A/A). 1 (Static load balancing) - Distributes and exposes the LUNs across the active optimize port(s) or active un-optimize port(s) and HBA(s). 2 (Least outstanding I/O) - Sends command to the path with the lowest I/O count. 3(Least Service time) - Sends request to the path with the shortest execution time.	0 (None)
ql2xexcludemodel	This parameter excludes device models from being a failover capable target. Combines one or more of the following model numbers into an exclusion mask: 0x80 - MSA A/A 0x20 - HSV111, HSV101, HSV200, HSV210 0x10 - DSXXX 0x04 - HSV110, HSV100 0x02 - MSA1000 0x01 - XP	0
ql2xtgtemul	This parameter enables or disables target level grouping emulation. This option is necessary for the GUI to work correctly if the driver is set for LUN level grouping of paths by LUNId. The following storages uses this method of combining paths: HSV210, DSXXX, HSV110, MSA1000, XP. 1 - Enables target level grouping emulation 0 - Disables target level grouping emulation	0 (enable)

6. SNIA API Library Package (combo package)

The SNIA API library package (qlapi-<api_version>-rel.tgz) is included in the driver combo package (qla2x00-vx.yy.zz-dist.tgz) or (qla2x00-vx.yy.zz-fo-dist.tgz).

Using the files you downloaded in section 4.1 copy the qla2xxx-vx.yy.zz-dist.tgz distribution file to /qla2x00. Follow these steps from the / (root) directory:

```
# mkdir qla2x00
# cd qla2x00
# mount /mnt/floppy
# cp /mnt/floppy/*.tgz . (the period at the end is required)
# tar -xvzf *.tgz
# cd qlogic
```

6.1 Installing SNIA API Library

To install and set up the API library, enter the following command in current directory:

```
# ./libinstall (this installs/sets up HBA API library)
```

6.2 Uninstalling SNIA API Library

To remove the API library, enter the following command in current directory:

```
# ./libremove (Script file to remove HBA API library)
```

7. Additional Notes

This section provides the following additional information:

- 7.1 Failover Support
- 7.2 Persistent Binding
- 7.3 Configuration Data
- 7.4 Booting from SAN
- 7.5 Compilation Messages
- 7.6 Dynamically Modifying SCSI Black-list Entries

7.1 Failover Support

This section describes how to provide failover support. For details, see the following topics:

- 7.1.1 How to Disable the Failover Support in the Driver
- 7.1.2 Configuration Changes Made via (LUN Masking) SANsurfer FC HBA Manager

7.1.1 How to Disable the Failover Support in the Driver

- To disable failover support in the qla2xxx driver, use the `ql2xfailover` module parameter:
  ```
  # insmod qla2xxx.ko ql2xfailover=0 ; insmod qla2300.ko
  ```
- To disable the failover in the configuration file, add the following in `modprobe.conf` (for RH) or `modprobe.conf.local` (for SLES):
  ```
  options qla2xxx ql2xfailover=0
  ```

NOTE: Failover is enabled by default when the 8.x driver is built.

7.1.2 Configuration Changes Made via (LUN Masking) SANsurfer FC HBA Manager

For the new LUN masking configuration to take effect, the driver must be reloaded. Use the following procedure to load, remove, and reload the driver.

1. Load the driver:
   ```
   # modprobe qla2300
   ```
   or
   ```
   # modprobe qla2400
   ```
2. Load the qlremote agent:
   ```
   #/usr/local/bin/qlremote
   ```
   or
   ```
   # qlremote start
   ```
3. Start the SANsurfer FC HBA Manager (GUI) and connect it to the destination system:
   ```
   # SANsurfer
   ```
4. Make LUN masking changes.
5. Disconnect the host from GUI and stop the `qlremote` agent.
6. Unload the driver:
   ```
   # modprobe -r qla2300 ; modprobe -r qla2xx_conf
   ```
7. Reload the driver:
   ```
   # modprobe qla2300
   ```
8. Load `qlremote` agent again.
9. Start the GUI and connect it to the destination system.
10. View the updated LUN masking configuration.

NOTE: The Linux SCSI mid-layer requires communication to a target using LUN 0; therefore, the driver does not mask LUN 0.

7.2 Persistent Binding

The persistent binding information consists of some HBA configuration entries along with some target entries.

You can specify persistent binding in two ways: manually or using SANsurfer FC HBA Manager/CLI. We recommend using SANsurfer FC HBA Manager/CLI for ease of use. The following is the procedure shows how to add persistent binding commands manually.

The driver displays the current configuration when the `displayConfig` command line option is specified. The persistent binding configuration is found in `/var/log/messages` file. It prints the configuration information in the format required by the driver.

To extract configuration messages, use the grep command and direct the output to a file. You need to remove the Linux timestamp at the beginning of each message and combine them together on single line. For example:

```
# insmod qla2300.ko displayConfig=1
# grep "scsi-qla" /var/log/messages > /tmp/info.cfg
```

The format of the persistent binding commands is as follows:

```
scsi-qla<#>-adapter-port=<adapter port name value>;
```

where the `qla<#>` expression and <#> is the HBA instance number.

This parameter specifies the FC port name used for the HBA, where <*adapter port name value*> is the FC port name value in hexadecimal format. If this entry is not specified in the configuration file, the default value is the HBA's port name as saved in NVRAM.

For example:

```
scsi-qla0-adapter-port=210000e08b01158d\;
```

Host HBA instance 0 has a portname of `210000e08b01158d`.

```
scsi-qla<#1>-tgt-<#2>-di-<#3>-node=<device FC name>;
```

This parameter associates the specified <*device FC name*> with the SCSI target ID value specified by <*#2*> and a device ID value specified by <*#3*>. The <*device FC name*> type is the FC nodename of the device, and <*#2*> is the SCSI target ID to be assigned to the device and <*#3*> is the device unique ID.

where:
<*#1*> specifies the HBA instance number
<*#2*> specifies the SCSI ID of Target
<*#3*> specifies the path/device ID

```
scsi-qla<#1>-tgt-<#2>-di-<#3>-port=<device FC name>;
```

This parameter associates the specified <*device FC name*> with the SCSI target ID value specified by <*#2*> and a device ID value specified by <*#3*>. The <*device FC name*> type is the FC port.

where:
<*#1*> specifies the HBA instance number
<*#2*> specifies the SCSI ID of the target
<*#3*> Specifies the path/device ID (always 0 for non-failover)

```
scsi-qla<#1>-tgt-<#2>-di-<#3>-disabled=<256 bit mask>;
```

This parameter associates the specified <*256 bit mask*> with the SCSI target ID value specified by <*#2*> and a device ID value specified by <*#3*>.

where:

<*#1*> specifies the HBA instance number
<*#2*> specifies the SCSI ID of Target
<#3> specifies the path/device ID

```
<256 bit mask>
msb                                                            lsb
 000000000000000000000000000000000000000000000000000000000000000F
```

The mask above makes the first four LUNs (3, 2, 1, and 0) of a given target disabled on that target/path.

This mask specification is heavily type-checked to be a sequence of 64 hex digits.

7.3 Configuration Data

Configuration/persistent data loads in the driver automatically when installing and loading the driver. Normally this information passes to the driver using the command line, but due to the constraints inherent in using the command line, you may prefer using an alternate method: `QLA_OPTS`. For details, see the following topics:

- 7.3.1 QLA_OPTS
- 7.3.2 Compatibility with SANsurfer FC HBA Manager

7.3.1 QLA_OPTS

QLA_OPTS reads the configuration data from qla2x00.conf and updates the binary module qla2x00_conf.o. The driver automatically tries to load the binary module qla2x00_conf.o at driver initialization time. Once loaded, the module passes the configuration information directly to the driver.

The configuration data is located in /etc/qla2xxx.conf.

NOTE: Approximately 300K of configuration space has been pre-allocated within the qla2200_conf/ qla2300_conf module for configuration/persistent data.

7.3.2 Compatibility with SANsurfer FC HBA Manager

QLA_OPTS works seamlessly with updated SANsurfer FC HBA Manager applications. It writes all appropriate configuration data to the _conf.ko module.

7.4 Booting from SAN

Booting from SAN means booting to the OS from a Fibre Channel target device. We recommend using the QLogic inbox driver to install the OS to a Fibre Channel target device which is attached to a QLogic HBA. If there is no QLogic inbox driver that supports the HBA, you must create a DD-kit to boot from SAN.

For details, see the following topics:

- 7.4.1 Creating a Driver Disk (DD Kit) Image
- 7.4.2 Using the Driver Disk (DD Kit) image
- 7.4.3 RHEL 4 - Boot from SAN Installation on Non-LUN 0 LUN

7.4.1 Creating a Driver Disk (DD Kit) Image

The driver disk (DD Kit) image enables installation of the Red Hat 4/SLES OS and QLogic driver using the Fibre Channel boot disk connected to the FC HBAs.

To build the driver disk floppy image:

1. Download an appropriate version of the DD-kit.tgz file from the QLogic web site:
   http://www.qlogic.com/
2. Untar the tgz file:
   ```
   # tar -xvzf *.tgz
   ```
3. Enter the following command:
   ```
   # dd if=qla2xxx-8.0x.0y<platform>-dd-<kernel-ver>.img of=/dev/fd0 bs=1440k
   ```
   (Requires a 1.44MB Floppy)

To build the driver disk CD from an ISO image:

1. Download an appropriate version of the DD-kit .tgz file from the QLogic web site:
   http://www.qlogic.com/
2. Untar the tgz file:
   ```
   # tar -xvzf *.tgz
   ```
3. Use a CD recorder to burn the iso image to a CD.

7.4.2 Using the Driver Disk (DD Kit) image

Using DD-kits is different with each Linux OS. Please refer to the readme file that ships with each DD-kit for instructions.

7.4.3 RHEL 4 - Boot from SAN Installation on Non-LUN 0 LUN

NOTE: The following procedure is for the RHEL 4 distribution only.

When installing to a LUN other than LUN 0 and LUN 0 is present, follow these steps to boot from the LUN successfully:

1. At the Boot Loader Configuration screen, select the **Configure Advance Boot Loader** option and click **Next**.
2. At the Advanced Boot Loader Configuration screen, select **Change Driver Order**.
3. At the Edit Drive Order window, move the entry for the boot LUN.
4. Click **OK** to continue.
5. Continue with the OS installation as usual.

7.5 Compilation Messages

During driver compilation you may see a message similar to the following:

```
WARNING: 'inter_module_*' is deprecated (declared at /usr/src/...)
```

This message is not fatal and does not impede the functionality of the driver.

7.6 Dynamically Modifying SCSI Black-list Entries

On 2.6.x kernels, you can dynamically change the SCSI blacklist, either by writing to a `/proc` entry or using the `scsi_mod` module parameter, which allows persistence across reboot.

This requires the SCSI Vendor/Model information for the SCSI device, available at `/proc/scsi/scsi`.

Black list entries are in the form:

```
vendor:model:flags[,v:m:f]

where flags can be the following integer values:

 0x001   /* Only scan LUN 0 */
 0x002   /* Known to have LUNs, force scanning,
            deprecated: Use max_luns=N */
 0x004   /* Flag for broken handshaking */
 0x008   /* unlock by special command */
 0x010   /* Do not use LUNs in parallel */
 0x020   /* Buggy Tagged Command Queuing */
 0x040   /* Non-consecutive LUN numbering */
            -- -> value need to be passed to "flags" variable for
            sparse lun
 0x080   /* Avoid LUNS >= 5 */
 0x100   /* Treat as (removable) CD-ROM */
 0x200   /* LUNs past 7 on a SCSI-2 device */
 0x400   /* override additional length field */
 0x800   /* ... for broken inquiry responses */
 0x1000  /* do not do automatic start on add */
 0x2000  /* do not send ms page 0x08 */
 0x4000  /* do not send ms page 0x3f */
 0x8000  /* use 10 byte ms before 6 byte ms */
 0x10000 /* 192 byte ms page 0x3f request */
 0x20000 /* try REPORT_LUNS even for SCSI-2 devs
            (if HBA supports more than 8 LUNs) */
 0x40000 /* don't try REPORT_LUNS scan (SCSI-3 devs) */
 0x80000 /* don't use PREVENT-ALLOW commands */
 0x100000 /* device is actually for RAID config */
 0x200000 /* select without ATN */
 0x400000 /* retry HARDWARE_ERROR */
```

For example:

```
# echo <VENDOR>:<MODEL>:040 > /proc/scsi/device_info
```

To enable persistence across reboots, add the following line:

```
options scsi_mod dev_flags=<VENDOR>:<MODEL>:<FlAGS>
```

to the following file (based on distribution):

`/etc/modprobe.conf` for RHEL 4.0 and above and

`/etc/modprobe.conf.local` for SLES

and rebuild the ramdisk. Please refer to section 4.2.3, "Making a ramdisk Image to Load the Driver."

8. IP Support

This section describes how to load the IP driver and configure the network interface to allow TCP/IP applications to communicate with IP-enabled FC HBAs. For details, see:

- 8.1 Loading the IP Driver
- 8.2 Configuring Interfaces

8.1 Loading the IP Driver

Before loading the driver manually, build the driver binary from the driver source files as described in section 4.2.1. Enter the insmod or modprobe commands to load the IP driver:

- To load the driver directly from the local build directory, enter the following in order:

  ```
  # insmod qla2xxx_conf.ko
  # insmod qla2xxx.ko
  # insmod qla2300.ko
  # insmod qla2xip.ko
  ```

- To load the driver using modprobe:

  a. Install the driver module (*.ko) files to the appropriate kernel module directory:

  ```
  # ./extras/build.sh install
  ```

  b. Enter the following to load the driver for qla23xx HBAs:

  ```
  # modprobe -v qla2xxx_conf
  # modprobe -v qla2300
  # modprobe -v qla2xip
  ```

  The `modprobe -v qla2300` command automatically loads the `qla2xxx.ko` component.

- To unload the driver using modprobe:

  a. Enter the following to unload the `qla2300.ko` and `qla2xxx.ko` modules.

  ```
  # modprobe -r qla2xip
  # modprobe -r qla2300
  ```

  b. Enter the following to unload `qla2xxx_conf.ko`:

  ```
  # modprobe -r qla2xxx_conf
  ```

  The `qla2xip` driver creates network-interface binding to each IP-capable recognized HBA. You can view binding entries from the messages file after the IP driver has loaded:

  ```
  qla2xip: QLogic IP using Fibre Channel Network Driver
  qla2xip: Driver Version 1.0b2, Entry point: e08e5060
  qla2xip: Mapping interface fc0 to HBA 210100e08b20a15b
  qla2xip: Mapping interface fc1 to HBA 210200e08b40a25b
  ```

8.2 Configuring Interfaces

To allow TCP/IP applications running on the host to communicate with other IP-capable FC HBAs, you must configure the network interfaces:

1. Compile the following basic host information to allow the IP driver to pass TCP/IP data over FC HBAs in your host:

Interface name:	fc0	(From above)
IP Address:	192.168.1.*x**	(A non-routable address)
Netmask:	255.255.255.0	(Standard class C mask)
Interface name:	fc1	(From above)
IP Address:	192.168.2.*x**	(A non-routable address)
Netmask:	255.255.255.0	(Standard class C mask)

   ***x** in the IP address is a unique number between (1 and 254)

2. Use the `ifconfig` program to configure an interface with the compiled host information:
   ```
   # ifconfig fc0 192.168.1.x up
   ```

3. Configure other interfaces (if multiple IP-capable HBAs are present in the host):
   ```
   # ifconfig fc1 192.168.1.x up
   ```

4. Verify the configured interfaces:
   ```
   # ifconfig
   ```
   The screen displays details on the newly configured interfaces, as shown in the following example:

   ```
   fc0 Link encap:Ethernet  HWaddr 00:E0:8B:20:A1:5B
      inet addr:192.168.1.1  Bcast:192.168.1.255  Mask:255.255.255.0
      UP BROADCAST RUNNING  MTU:4096  Metric:1
      RX packets:1214577458 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1214213174 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:32
      RX bytes:3081095492 (2938.3 Mb) TX bytes:2751945609 (2624.4 Mb)

   fc1 Link encap:Ethernet  HWaddr 00:E0:8B:40:A2:5B
      inet addr:192.168.2.1  Bcast:192.168.2.255  Mask:255.255.255.0
      UP BROADCAST RUNNING  MTU:4096  Metric:1
      RX packets:1204464697 errors:0 dropped:0 overruns:0 frame:0
      TX packets:1194873236 errors:0 dropped:0 overruns:0 carrier:0
      collisions:0 txqueuelen:32
      RX bytes:1454694706 (1387.3 Mb) TX bytes:991094469 (945.1 Mb)
   ```

   This completes configuration.

5. Verify that basic networking is possible between two hosts connected via two HBAs using a simple ping command:
   ```
   # ping 192.168.1.2
   ```

For more details on Linux networking, refer to the Linux Networking "HOWTOs" available at http://www.tldp.org.

9. Contacting Support

Please feel free to contact your QLogic approved reseller or QLogic Technical Support at any phase of integration for assistance. QLogic Technical Support can be reached by the following methods:

Web: http://support.qlogic.com

North America Contact Information
Email: support@qlogic.com

Phone: (952) 932-4040

Support contact information for other regions of the world is available at the QLogic website:
http://support.qlogic.com

Go to Top