NAME

cciss − HP Smart Array block driver

SYNOPSIS

modprobe cciss [cciss_allow_hpsa=1]

DESCRIPTION

cciss is a block driver for older HP Smart Array RAID controllers.

OPTIONS

cciss_allow_hpsa=1 This option prevents the cciss driver from attempting to drive any controllers which the hpsa driver is capable of controlling, which is to say, the cciss driver is restricted by this option to the following controllers:

Smart Array 5300
Smart Array 5i
Smart Array 532
Smart Array 5312
Smart Array 641
Smart Array 642
Smart Array 6400
Smart Array 6400 EM
Smart Array 6i
Smart Array P600
Smart Array P400i
Smart Array E200i
Smart Array E200
Smart Array E200i
Smart Array E200i
Smart Array E200i
Smart Array E500

SUPPORTED HARDWARE

The **cciss** driver supports the following Smart Array boards:

Smart Array 5300
Smart Array 5i
Smart Array 532
Smart Array 5312
Smart Array 641
Smart Array 642
Smart Array 6400
Smart Array 6400 U320 Expansion Module
Smart Array 6i
Smart Array P600
Smart Array P800
Smart Array E400
Smart Array P400i
Smart Array E200
Smart Array E200i
Smart Array E500
Smart Array P700m
Smart Array P212
Smart Array P410
Smart Array P410i

Smart Array P411
Smart Array P812
Smart Array P712m
Smart Array P711m

CONFIGURATION DETAILS

To configure HP Smart Array controllers, use the HP Array Configuration Utiltiy (either hpacuxe or hpacucli) or the Offline ROM-based Configuration Utility (ORCA) run from the Smart Array's option ROM at boot time.

FILES

DEVICE NODES

The device naming scheme is as follows:
Major numbers:

104 cciss0
105 cciss1
106 cciss2
105 cciss3
108 cciss4
109 cciss5
110 cciss6
111 cciss7

Minor numbers:

```
b7 b6 b5 b4 b3 b2 b1 b0
|----+----| |----+----|
   |         |
   |         +-------- Partition ID (0=wholedev, 1-15 partition)
   |
   +------------------- Logical Volume number
```

The device naming scheme is:

/dev/cciss/c0d0	Controller 0, disk 0, whole device
/dev/cciss/c0d0p1	Controller 0, disk 0, partition 1
/dev/cciss/c0d0p2	Controller 0, disk 0, partition 2
/dev/cciss/c0d0p3	Controller 0, disk 0, partition 3
/dev/cciss/c1d1	Controller 1, disk 1, whole device
/dev/cciss/c1d1p1	Controller 1, disk 1, partition 1
/dev/cciss/c1d1p2	Controller 1, disk 1, partition 2
/dev/cciss/c1d1p3	Controller 1, disk 1, partition 3

FILES IN /proc

The files /proc/driver/cciss/cciss[0-9]+ contain information about the configuration of each controller. For example:

```
someone@somehost:/proc/driver/cciss> ls -l
total 0
-rw-r--r-- 1 root root 0 2010-09-10 10:38 cciss0
-rw-r--r-- 1 root root 0 2010-09-10 10:38 cciss1
-rw-r--r-- 1 root root 0 2010-09-10 10:38 cciss2
someone@somehost:/proc/driver/cciss> cat cciss2
cciss2: HP Smart Array P800 Controller
Board ID: 0x3223103c
Firmware Version: 7.14
```

IRQ: 16
Logical drives: 1
Current Q depth: 0
Current # commands on controller: 0
Max Q depth since init: 1
Max # commands on controller since init: 2
Max SG entries since init: 32
Sequential access devices: 0

cciss/c2d0: 36.38GB RAID 0
someone@somehost:/proc/driver/cciss>

FILES IN /sys

/sys/bus/pci/devices/<dev>/ccissX/cXdY/model

Displays the SCSI INQUIRY page 0 model for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/rev

Displays the SCSI INQUIRY page 0 revision for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/unique_id

Displays the SCSI INQUIRY page 83 serial number for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/vendor

Displays the SCSI INQUIRY page 0 vendor for logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/block:cciss!cXdY

A symbolic link to /sys/block/cciss!cXdY

/sys/bus/pci/devices/<dev>/ccissX/rescan

Kicks off a rescan of the controller to discover logical drive topology changes.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/lunid

Displays the 8-byte LUN ID used to address logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/raid_level

Displays the RAID level of logical drive Y of controller X.

/sys/bus/pci/devices/<dev>/ccissX/cXdY/usage_count

Displays the usage count (number of opens) of logical drive Y of controller X.

SCSI tape drive and medium changer support

SCSI sequential access devices and medium changer devices are supported and appropriate device nodes are automatically created. (e.g. /dev/st0, /dev/st1, etc. See the "st" man page for more details.) You must enable "SCSI tape drive support for Smart Array 5xxx" and "SCSI support" in your kernel configuration to be able to use SCSI tape drives with your Smart Array 5xxx controller.

Additionally, note that the driver will not engage the SCSI core at init time. The driver must be directed to dynamically engage the SCSI core via the /proc filesystem entry which the "block" side of the driver creates as /proc/driver/cciss/cciss* at runtime. This is because at driver init time, the SCSI core may not yet be initialized (because the driver is a block driver) and attempting to register it with the SCSI core in such a case would cause a hang. This is best done via an initialization script (typically in /etc/init.d, but could vary depending on distribution). For example:

```
for x in /proc/driver/cciss/cciss[0-9]*
do
      echo "engage scsi" > $x
done
```

Once the SCSI core is engaged by the driver, it cannot be disengaged (except by unloading the driver, if it happens to be linked as a module.)

Note also that if no sequential access devices or medium changers are detected, the SCSI core will not be engaged by the action of the above script.

Hot plug support for SCSI tape drives

Hot plugging of SCSI tape drives is supported, with some caveats. The cciss driver must be informed that changes to the SCSI bus have been made. This may be done via the /proc filesystem. For example:

```
echo "rescan" > /proc/scsi/cciss0/1
```

This causes the driver to query the adapter about changes to the physical SCSI buses and/or fibre channel arbitrated loop and the driver to make note of any new or removed sequential access devices or medium changers. The driver will output messages indicating what devices have been added or removed and the controller, bus, target and lun used to address the device. It then notifies the SCSI mid layer of these changes.

Note that the naming convention of the /proc filesystem entries contains a number in addition to the driver name. (E.g. "cciss0" instead of just "cciss" which you might expect.)

Note: ONLY sequential access devices and medium changers are presented as SCSI devices to the SCSI mid layer by the cciss driver. Specifically, physical SCSI disk drives are NOT presented to the SCSI mid layer. The physical SCSI disk drives are controlled directly by the array controller hardware and it is important to prevent the kernel from attempting to directly access these devices too, as if the array controller were merely a SCSI controller in the same way that we are allowing it to access SCSI tape drives.

SCSI error handling for tape drives and medium changers

The linux SCSI mid layer provides an error handling protocol which kicks into gear whenever a SCSI command fails to complete within a certain amount of time (which can vary depending on the command). The cciss driver participates in this protocol to some extent. The normal protocol is a four step process. First the device is told to abort the command. If that doesn't work, the device is reset. If that doesn't work, the SCSI bus is reset. If that doesn't work the host bus adapter is reset. Because the cciss driver is a block driver as well as a SCSI driver and only the tape drives and medium changers are presented to the SCSI mid layer, and unlike more straightforward SCSI drivers, disk i/o continues through the block side during the SCSI error recovery process, the cciss driver only implements the first two of these actions, aborting the

command, and resetting the device. Additionally, most tape drives will not oblige in aborting commands, and sometimes it appears they will not even obey a reset command, though in most circumstances they will. In the case that the command cannot be aborted and the device cannot be reset, the device will be set offline.

In the event the error handling code is triggered and a tape drive is successfully reset or the tardy command is successfully aborted, the tape drive may still not allow i/o to continue until some command is issued which positions the tape to a known position. Typically you must rewind the tape (by issuing "mt -f /dev/st0 rewind" for example) before i/o can proceed again to a tape drive which was reset.

SEE ALSO

hpsa(4), hpacucli(8), hpacuxe(8), cciss_vol_status(8), http://cciss.sf.net, and from the linux kernel source, Documentation/blockdev/cciss.txt and Documentation/ABI/testing/sysfs-bus-pci-devices-cciss

AUTHORS

Don Brace, Steve Cameron, Chase Maupin, Mike Miller, Michael Ni, Charles White, Francis Wiran and probably some other people.