

Sun Datapath Policies for Disks and Disk Arrays

August 2007

Copyright © 2007 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, California 95054, U.S.A. All rights reserved.

Sun, Sun Microsystems, the Sun logo, Solaris, and Sun StorEdge are trademarks or registered trademarks of Sun Microsystems, Inc. in the U.S. and other countries.

All SPARC trademarks are used under license and are trademarks or registered trademarks of SPARC International, Inc. in the US and other countries. Products bearing SPARC trademarks are based upon an architecture developed by Sun Microsystems, Inc.

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Copyright 2006 Sun Microsystems, Inc., 4150 Network Circle, Santa Clara, Californie 95054, Etats-Unis. Tous droits réservés.

Sun, Sun Microsystems, le logo Sun, Solaris et Sun StorEdge sont des marques de fabrique ou des marques déposées de Sun Microsystems, Inc. aux Etats-Unis et dans d'autres pays.

Toutes les marques SPARC sont utilisées sous licence et sont des marques de fabrique ou des marques déposées de SPARC International, Inc. aux Etats-Unis et dans d'autres pays. Les produits portant les marques SPARC sont basés sur une architecture développée par Sun Microsystems, Inc.

LA DOCUMENTATION EST FOURNIE "EN L'ÉTAT" ET TOUTES AUTRES CONDITIONS, DECLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISEE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE A LA QUALITE MARCHANDE, A L'APTITUDE A UNE UTILISATION PARTICULIERE OU A L'ABSENCE DE CONTREFAÇON.

Contents

1 Introduction.....	1
1.1 Scope.....	1
1.2 Document Updates.....	1
1.3 References.....	2
2 Keywords.....	3
3 General Policies.....	4
3.1 Management Logical Units.....	4
3.2 Array/Disk Interface Specifications Requirement.....	5
3.3 Logical Unit Identifiers.....	5
3.4 SCSI Report LUNs Command.....	7
3.5 Asymmetric Multipath Support for the Solaris OS.....	8
3.6 Symmetric Multipath Support for the Solaris OS.....	9
3.7 Logical Unit Addressing Methods.....	10
3.8 Hierarchical Support.....	10
3.9 Sixteen-Byte CDBs.....	11
3.10 Report Logical Units & Inquiry Behavior for Masked Logical Units.....	11
3.11 Storage System (Target) Name for Multi-Unit Arrays.....	14
3.12 Storage System Management IP Addresses.....	14
3.13 Asynchronous Notification of Name or Management Address Changes.....	15
3.14 Device Name/Identifier Assumptions.....	15
3.15 SCSI-3 SPC Access Controls (ACLs).....	16
3.16 Logical Unit Reset Support.....	16
3.17 Mode Page Requirements.....	16
3.18 Notification of Capacity Change.....	17
3.19 CDB Control Byte – NACA or LINK Flags.....	18
3.20 Sun Cluster Support.....	18
4 Fibre Channel-Specific Policies.....	22
4.1 Multipath Support.....	22
4.2 Asynchronous Notification of Device Changes.....	22
4.3 Logical Unit Address Methods.....	22
5 iSCSI-Specific Policies.....	23
5.1 Support for Multiple LUNs.....	23
5.2 Multipath Support.....	23
5.3 Asynchronous Notification of Device Changes.....	23
5.4 Logical Unit Address Methods.....	23
5.5 Target Alias Handling.....	24
5.6 IQN Name Format.....	24
5.7 iSNS Support.....	24
5.8 RADIUS Support.....	25
5.9 iSCSI Features Related to RFC 3720 Parameters.....	25

6 Serial Attached SCSI (SAS) Specific Policies.....	26
6.1 Multipath Support.....	26
6.2 Asynchronous Notification of Device Changes.....	26
6.3 Logical Unit Address Methods.....	26
7 Behavior of Solaris OS Drivers.....	27
7.1 The Solaris OS Queue Throttle.....	27
7.2 Solaris OS Disk Driver Device Identifier Generation.....	27
7.3 Error Handling Behavior.....	28

1 Introduction

This document contains policies for disk and disk array implementations. This document is limited to *datapath* interfaces and does not address policies related to physical interfaces or management interfaces independent from the datapath. The software considered includes drivers for the Solaris™ Operating System (Solaris OS) version 9 and later, and management applications that issue SCSI, Fibre Channel, and iSCSI commands.

Although disks and arrays implement many standard interfaces, there are still numerous problems with lack of interoperability. These areas of non-interoperability stem from gaps in standards, standards that have changed over time, overlapping standards, standards that do not meet the needs of the industry, and differing interpretations of the standards. This document defines standards that apply to Sun products in order to minimize these areas of non-interoperability.

There are many costs associated with non-interoperability. Product-specific code in drivers and management software has engineering and test costs, and the additional complexity of any product-specific code adds to the complexity of the entire software component. The requirement for product-specific code in turn delays the ability to bring new disk and array products to market. One of the goals in creating this specification is to encourage interoperability to help reduce costs and speed time-to-market.

In the context of this document, the term “disk array” applies to RAID systems, JBOD (“Just a Bunch Of Disks”) arrays, and data services platforms that export virtual disks.

The intended target audiences for this document are:

- Sun development teams
- Partner developers
- Third-party array and disk developers
- Marketing and Program Management teams evaluating the cost of supporting certain features or platforms
- Review teams

In addition to the interface-related documentation, this document includes a section that describes the behavior of Solaris OS disk drivers. Although this information is not directly related to standards, the authors felt that this information would be valuable to many of the readers of this document.

Important Note:

The driver implementations described here do not constitute an interface and are subject to change without notice.

1.1 Scope

This document primarily addresses SCSI datapath messages. It also includes some requirements related to Fibre Channel, iSCSI, and SAS transport protocol.

1.2 Document Updates

Non-Sun readers should send mail to array-bestpractices-subscribe@sun.com and ask to be subscribed to announcements of array best practice document changes. This will add you to an announce-only list. You will not see any discussion on this list.

Sun readers can check for updates at [this Sun internal web-site](#) or subscribe to the Netadmin email list array-bestpractices@sun.com for announcements of changes. Note that array-bestpractices@sun.com is an internal Sun list and is open to discussion. All known requirements have been incorporated in this document, but project teams might have additional requirements. If you discover requirements not included, please use one of the email lists to send your comments.

1.3 References

Unless otherwise noted, the versions of standards specifications listed in the following table are those referenced in the specification.

SPC-3	SCSI Primary Commands – 3	T10	ANSI INCITS 408-2005
SBC-2	SCSI Block Commands – 2	T10	ANSI INCITS 405-2005
SAM-3	SCSI Architecture Model – 3	T10	ANSI INCITS 402-2005
FCP-3	SCSI Fibre Channel Protocol – 3	T10	2002/12/11, Rev: 04
FC-FS	Fibre Channel Framing and Signaling Interface	T11	ISO/IEC 14165-251 ANSI INCITS 373-2003
FC-GS-4	Fibre Channel Generic Services 4	T11	ISO/IEC 14165-414 ANSI INCITS 387-2004
RFC 3720	Internet Small Computer Systems Interface (iSCSI)	IETF	
RFC 4171	Internet Storage Name Service (iSNS)	IETF	
SMI-S	Storage Management Initiative Specification	SNIA	ANSI INCITS 388

Note that some documents are draft versions and might change before they are approved.

The [latest drafts of T10 documents](#) are available online, or [go to the main T10 web site](#) and look for links to drafts.

The [latest T11 FC drafts](#) are also available.

iSCSI RFCs (final and draft) are available at [the IETF IPS workgroup site](#).

2 Keywords

To paraphrase from T10 specifications: certain words in this document have a specific meaning beyond the normal English meaning.

may: Indicates flexibility of choice with no implied preference.

shall: Indicates a mandatory requirement. Designers are required to implement all such mandatory requirements to ensure interoperability with other standards-compliant products.

should: Indicates flexibility of choice with a strongly preferred alternative. Equivalent to the phrase “it is recommended.”

reserved: Refers to bits, bytes, words, fields, and code values that are set aside for future standardization. Their use and interpretation may be specified by future extensions to this or other standards. A reserved bit, byte, word, or field shall be set to zero, or in accordance with a future extension to this standard. The recipient may not check reserved bits, bytes, words, or fields. Receipt of reserved code values in defined fields may be treated as an error.

vendor-specific: An item (for example, a bit, field, or code value) that is not defined by this standard and may be vendor-defined.

3 General Policies

These policies are not specific interfaces, but describe general strategies that may apply to many interfaces.

3.1 Management Logical Units

Management logical units are logical units used solely for the purpose of management commands. There are two potential problems:

- If management logical units are exposed as disks (SCSI INQUIRY response dtype = 0), then they show up in utilities that list disks, such as format. But these management logical units are not general-purpose disks, so utility operations may fail. This causes confusion and frustration with customers.
- Regardless of the device type, management logical units typically create a promiscuous management interface where any attached host can inadvertently (or deliberately) change or delete logical units allocated to other hosts. In-band management interfaces generally do not require any authentication or authorization at the device, so any application granted the ability to issue a CDB on a host can issue management commands.

Both of these problems can be solved if the management commands are sent to data logical units. For the second issue above, device LUN masking comes into play. The effect of LUN masking is that different hosts see different subsets of an array's logical units. Unless the array administrator has granted access, management applications running on one host are not able to delete or modify another host's logical units.

This approach (using data logical units for management) is appropriate for a variety of management tasks, but not all. In particular, if no data LUN is available, then there is no handle for in-band management commands. Some device vendors try to mitigate this by having a management LUN that "morphs" into a data LUN. This approach is problematic for the Solaris OS. Typically the morphing is accomplished by changing SCSI Inquiry information, such as the device type or Vendor/Product identifiers. If these values are changed in a way not anticipated by Solaris OS drivers, then the transition may not cause the expected updates. Other problems are possible if the change to SCSI properties causes the Solaris OS to think the device does or does not support multipath.

Sun uses the following policies:

1. Sun prefers that devices not expose management logical units. Sun prefers that data logical units be used for in-band management commands
2. If management LUNs are exposed, they should be SES devices (SCSI INQUIRY dtype 0Dh).
3. The Solaris OS prefers that SES management logical units be left as permanent entries in the device tree and not change INQUIRY information just because data logical units are added.
4. If the logical unit (management or data) vendor/product is configured as multipath (see 3.6.4), there is no SCSI passthrough support to select a path. The Solaris OS uses a round-robin algorithm to select the path, and SCSI passthrough commands may be routed to separate paths. If an application using SCSI passthrough requires control over path selection, the device should be configured as follows:
 - The management logical unit should have a different vendor/product ID combination than the data logical units

- The management logical unit vendor/product ID should not be configured as multipath-capable
 - The management logical unit should not change characteristics (vendor/product ID or device type) when data logical units are created
5. If a disk type (SCSI INQUIRY dtype 0) management LUN is used:
- It may be left as a permanent entry in the device tree.
 - If the device vendor wishes to morph a management LUN to a usable data LUN, they shall provide a unique page 83h LUN ID for the management LUN and a separate page 83h ID for the data LUN and (for FC devices) send a LIP or RSCN after the characteristics change.
 - The vendor may change the VID/PID when the management LUN is morphed to a real data LUN. This allows different multipath configurations.

3.2 Array/Disk Interface Specifications Requirement

All array and disk products shall provide a specification that includes the following information:

- The version of the relevant T10/T11 specifications that the product implements.
- All usage of T10/T11 reserved fields in requests/responses. It is preferred that reserved fields never be used, but if they are used, they shall be specified.
- The actual implementation of any T10 commands and fields that the specified standards define as optional.
- All vendor-specific SCSI commands and response formats that relate to the issues specified in this document (that is, usage of values that are specified as vendor-specific as opposed to reserved).
- A description of all other behavior that relates to the issues discussed in the rest of this document (for example, queue-full and logical unit Reset behavior).
- How to correlate identifiers and names between in-band and out-of-band interfaces and between different out-of-band protocols.
- How service and customer configuration roles are isolated.

The array/disk interface specification **may** include documentation on standard or other vendor-specific interfaces. The intent of this document is not to copy parts of other specifications. Wherever appropriate, this specification should reference materials from existing standards and product specifications.

3.3 Logical Unit Identifiers

3.3.1 Background

Disks are assigned and arrays generate VPD page 83h identifiers with association equal to 0 (indicating that the identifier is associated with a logical unit) to uniquely identify each logical unit/volume. These identifiers are used by multipath software to differentiate multipath logical units from multiple single-path logical units. The identifiers are also used by volume management and file-system software to uniquely identify data. Data loss may occur if different volumes report the same identifier.

These identifiers typically are based on a sixteen-byte IEEE Registered Extended format (or eight-byte IEEE Registered format) that includes a three-byte vendor identifier. Most of the identifier format is vendor-specific. The vendor is expected to manage the use of these vendor-specific bits and to maintain a registry of released products to assure that collisions do not occur.

Sun does not release SCSI products frequently. Rather than creating an additional registry, the ARC case log will serve as a registry for these identifiers.

Page 83h is a recent addition to SCSI standards. Prior to page 83h, page 80h provided a Unit Serial Number. Older versions of the Solaris OS still rely on page 80h.

Sun multipath drivers (Sun StorEdge™ Traffic Manager) require implementation of VPD page 83h identifiers with association equal to 0 (indicating that the identifier is associated with a logical unit) and type equal to 2 (EUI) or 3 (NAA). Prior to the Solaris OS version 10, type 1 (T10 vendor ID based) was also acceptable. This compatibility was removed from the Solaris OS version 10 because type names may exceed 32 characters when encoded as text and the identifier is used to generate device tree names.

3.3.2 Related Standards

The VPD page 83h identifier format is defined in SCSI SPC. A page 83h response consists of a list of identification descriptors. Each descriptor contains metadata and an identifier. The metadata includes an *association* (whether the identifier is associated with a logical unit [0], target port [1], or the entire target device [2]) and a *type* (the format of the identifier).

Originally this VPD page was optional, but SPC now specifies that a page 83h logical unit identifier is required:

For logical units that are not well-known logical units (see clause 8), at least one identification descriptor shall contain 1h, 2h, or 3h in the IDENTIFIER TYPE field and 0h in the ASSOCIATION field. At least one identification descriptor should contain 2h or 3h in the IDENTIFIER TYPE field and 0h in the ASSOCIATION field.

SPC recommends the IEEE Registered Extended name format (16 bytes, type 3 with NAA 6) for virtual volumes:

In the case of virtual logical units (e.g., volume sets as defined by SCC-2), the IDENTIFIER field should be in the NAA IEEE Registered Extended name format as defined in (the section titled "NAA IEEE Registered Extended identifier format").

[IEEE Tutorial for SCSI Use of IEEE company_id](#) is clearer in recommending this format for RAID volumes:

For logical units that are virtual devices (e.g., RAID virtual volumes), it is recommended that the FC-PH IEEE Registered Extended name be used.

The advantage of the IEEE Registered Extended name format is that it is 16 rather than 8 bytes long, making it more appropriate for use in RAID storage where identifiers are generated dynamically.

Page 80h was defined before page 83h. There are different interpretations of the use of this identifier and it was too short for practical use in RAID arrays, so page 83h was added in SCSI-3 standards.

3.3.3 Sun Standards

All RAID arrays and disks (including SCSI-2 JBOD disks):

- Shall include at least one identifier with association equal to 0 (logical unit) and type equal to 1 (T10 vendor ID), 2 (EUI), or 3 (NAA) in VPD page 83h responses, as specified here:
 - RAID Arrays should include at least one type 3 identifier with a 16 byte IEEE Identifier with NAA 6
 - JBOD Arrays should include at least one type 3 identifier with an IEEE Identifier with NAA 2, 5, or 6
 - The type 3 (NAA) descriptor should be the first descriptor in the response from the target for use with AIX
- If the “Code Set” field is 1 (binary), the identifier length shall be less than or equal to 16
- If the “Code Set” field is 2 (ASCII), the identifier length shall be less than or equal to 32
- Shall implement VPD page 00h (Supported VPD Pages) and include 83h as a supported page

RAID arrays (and JBOD disks) with a Sun Vendor ID:

- Shall include at least one NAA format (Identifier type equal to 3) with NAA 6 (16-byte identifier) and association equal to 0 (logical unit)
- Shall format the identifier as follows (each row represents 4 bytes):

<i>NAA = 6</i>	<i>Company ID = 080020 (or other Sun identifiers)</i>	<i>ProductID</i>
<i>ProductID</i>	<i>Volume ID (unique for the specific Sun product)</i>	
<i>Volume ID (continued)</i>		
<i>Volume ID (continued)</i>		

The SunID/ProductID combination shall be unique across Sun products.

- The SunID/ProductID combination shall be clearly documented in an ARC case. The string “ProductID” shall be included in the ARC materials to allow easy searching.
- This policy applies to company_ids registered to Sun Microsystems and recent acquisitions. (It may take a while to get identifiers transferred to Sun.) Check the [current IEEE Company_ID list](#).
- This policy does not apply to products that expose a third-party company identifier.
- If a company_id is used in a released product, it may not be feasible to follow this policy (if the ProductID portion of the identifier is used for a different purpose).

Applicability: All disks and arrays

3.4 SCSI Report LUNs Command

3.4.1 Related Standards

REPORTS LUNS is defined in SPC.

3.4.2 Sun Standards

All targets supporting logical unit numbers other than zero shall support the REPORT LUNS command.

3.5 Asymmetric Multipath Support for the Solaris OS

3.5.1 Background

This background applies to asymmetric and symmetric (see 3.6) multipath access.

“Asymmetric Logical Unit Access” (ALUA) was included in SPC-2 and updated in the SPC-3 specification. This interface allows an initiator to discover *Target Port Groups* - groups of ports expected to provide common failover behavior for specific logical units. The Target Port Group Support (TPGS) field in the standard INQUIRY response describes the logical unit's adherence to the standard, whether the logical unit provides symmetric or asymmetric access, and whether the logical unit uses explicit or implicit failover. The standard provides a standard explicit failover command, and commands to determine which ports are members of a target port group and other information about the multipath configuration.

Solaris 9 (and Later) OS support

If the device supports T10 SPC TPGS implicit failover (INQUIRY response TPGS field set to 01b or 11b), the Solaris OS will automatically configure the device with multipath support. Explicit failover (INQUIRY response TPGS field set to 10b) is also supported starting in Solaris 11.

The built-in multipath support in the Solaris OS works with Fibre Channel devices (using Sun's fcp driver), iSCSI devices (using Sun's iSCSI driver), and InfiniBand (using Sun's ibsrp driver).

The Solaris OS does not have built-in support for multipath Parallel SCSI or SAS devices.

3.5.2 Related Standards

SPC-2 and SPC-3

3.5.3 Sun Standards

Disks or arrays that expose asymmetric path access shall:

- Specify the appropriate non-zero value in the TPGS field in the standard INQUIRY response
- Implement the INQUIRY command Device Identifier VPD page (see SPC 7.6.4) identifier types 4h and 5h
- Support the REPORT TARGET PORT GROUPS command as described in SPC 6.24
- If the target supports explicit logical unit failover, it shall support the SET TARGET PORT GROUPS command
- When the device has a change to path access states, it shall establish a unit attention condition for the initiator ports associated with all I_T nexuses with the additional sense code set to ASYMMETRIC ACCESS STATE CHANGED
- Accept a 2-byte allocation length for Inquiry commands

Applicability: All disks and arrays that expose asymmetric path behavior

3.6 Symmetric Multipath Support for the Solaris OS

3.6.1 Background

Solaris OS versions 9 and later support symmetric logical units via either of two techniques:

- The logical unit supports the T10 standard (see 3.6.2)
- The customer manually configures the device by vendor/product identifiers (see 3.6.4)

The Solaris OS version 9 (and version 10 before update 1) only support manual configuration (see 3.6.4).

3.6.2 Related Standards

SCSI Primary Commands 3 (SPC-3) defines an optional interface for devices to report that the logical unit supports symmetric multipath access. The following is a direct quote:

“Symmetrical logical unit access should be represented as follows:

- The TPGS field in the standard INQUIRY data indicates that implicit asymmetric access is supported;
- The REPORT TARGET PORT GROUPS command is supported; and
- The REPORT TARGET PORT GROUPS parameter data indicates that the same state (e.g., active/optimized state) is in effect for all target port groups.”

3.6.3 Sun Standards

The Sun standard for symmetric logical units is the same as the T10 standard. - See 3.6.2.

3.6.4 Manual Configuration of Symmetric Multipath Access

To manually configure symmetric multipath access:

1. Determine the Vendor ID and Product ID string as returned in the INQUIRY command for the device.
2. Modify the configuration file `/kernel/drv/scsi_vhci.conf` to include the third-party symmetric device as shown in the following example.

Example for a single symmetric VID/PID. The device has a vendor ID of "EMC" and a product ID of "SYMMETRIX":

```
device-type-scsi-options-list =
"EMC      SYMMETRIX", "symmetric-option";
symmetric-option = 0x1000000;
```

Example showing multiple VID/PIDs for symmetric devices:

```
device-type-scsi-options-list =
"SUN      SENA", "symmetric-option",
"SUN      SESS01", "symmetric-option",
"HITACHI OPEN", "symmetric-option";
symmetric-option = 0x1000000;
```

3.7 Logical Unit Addressing Methods

3.7.1 Related Standards

The SCSI Architecture Model (SAM) 3 specification defines an eight-byte logical unit number structure and several addressing methods. Although SAM describes using this eight-byte structure as four two-byte levels, in practice, only the first level (bytes 0 and 1) is used. The high-order two bits (the Address Method Field) of the logical unit structure define the addressing method. Two addressing methods are in common use.

The Solaris OS supports the peripheral device address method (00 binary) for all device types.

For parallel SCSI and SAS devices, byte one is used as an eight-bit LU number with values between 0 and 255.

Byte	Bit	7	6	5	4	3	2	1	0
0		0	0	zero					
1		LU number							

For Fibre Channel and iSCSI devices, the Solaris OS treats six bits from byte zero and all of byte one as a fourteen-bit LU number with values between 0 and 16,383.

Byte	Bit	7	6	5	4	3	2	1	0	
0		0	0	(MSB)						
1		LU number								(LSB)

For Fibre Channel devices, the Solaris OS supports the same fourteen-bit LU number with the logical unit (10 binary) address method.

For iSCSI devices, the Solaris OS supports the same fourteen-bit LU number with flat space (01 binary), or logical unit (10 binary) address methods.

3.7.2 Sun Standards

Disks and arrays shall provide one of these address methods for Solaris OS support. Logical Units with addresses between 256 and 16383 shall use flat space addressing. Logical units with addresses less than 256 should use peripheral device addressing, but may use flat space addressing.

Disks and arrays shall support peripheral device addressing mode.

Applicability: All arrays that expose multiple logical units (in practice this is RAID arrays)

3.8 Hierarchical Support

3.8.1 Related Standards

The HISUP flag in the standard INQUIRY response is set to indicate the logical unit addressing methods described (see 3.7). If this flag is not set, the addressing method is not defined.

3.8.2 Sun Standards

Any disk or array shall set the HISUP flag in the standard INQUIRY response.

Applicability: All disks and arrays

3.9 Sixteen-Byte CDBs

3.9.1 Background

With 512 bytes blocks, 10-byte CDBs in common use have a volume maximum capacity of 2 terabytes. T10 has added 16-byte CDBs to recent versions of the standard. 16-byte CDBs allow access to larger volumes.

3.9.2 Related Standards

Sixteen-byte CDBs are included in SCSI Block Commands (SBC) specifications.

3.9.3 Sun Standards

Support for 16-byte CDBs is recommended for all RAID/virtualization hardware with more than two terabytes of storage. The following command formats are required to allow software support of larger volumes:

READ(16)

WRITE(16)

READ CAPACITY (long LBA)

READ DEFECT DATA (long LBA)

LONG LBA support in MODE SENSE and MODE SELECT

Applicability: All disks and arrays, but at this time, only RAID arrays provide multi-terabyte volumes.

3.10 Report Logical Units & Inquiry Behavior for Masked Logical Units

3.10.1 Background

A SCSI target may expose a subset of logical units to some initiators. Different vendors take different approaches to handling these masked logical units. They may or may not show up in a REPORT LUNS response. If they do, they may use the standard INQUIRY peripheral qualifier or device type to give hints to drivers that the masked logical units should not be added to the device tree.

The SCSI SPC-3 specification is clear on the majority of device response behavior for the standard INQUIRY and REPORT LUNS commands. For these two commands, there are areas in the specification that are vague enough to have caused varying behavior within the arrays that Sun currently supports. These areas are the peripheral qualifier in the standard INQUIRY page, and the logical unit inventory in the REPORT LUNS command within the context of masked or undefined logical units.

The area in both the standard INQUIRY and REPORT LUNS commands that tends to be interpreted differently among vendors is the behavior with respect to logical unit number 0 in the following scenarios:

- A SCSI target with multiple logical units has no logical units defined.

- Referencing the specification, the default report of the logical unit inventory shall contain at least logical unit number 0. There appears to be some differing behavior on how logical unit number 0, in this case, reports the peripheral qualifier and the peripheral device type via the standard INQUIRY command.
- A SCSI target with multiple logical units has one or more logical units defined, yet one or more logical units is masked from the initiator.

The specification indicates that the default report of the logical unit inventory shall contain at least logical unit number 0. This has not always been interpreted to mean that the logical unit inventory shall always contain logical unit number 0. For example, when logical unit number 0 is masked from the initiator and there are other logical units accessible to the initiator, what does the 'default report' mean?

3.10.2 Related Standards

The following excerpts are from the REPORT LUNS commands defined in the SCSI SPC-3 specification and are relevant to this issue:

The REPORT LUNS command requests that the peripheral device logical unit inventory accessible to the initiator via the addressed SCSI target port be sent to the application client.

LUN 0 shall always respond to a REPORT LUNS command on devices which have more than a single logical unit. For devices having only a single logical unit, support of REPORT LUNS is optional.

The logical unit inventory is a list that shall include the logical unit numbers of all logical units having a PERIPHERAL QUALIFIER value of 000b. Logical unit numbers for logical units with PERIPHERAL QUALIFIER values of 100b, 101b, 110b or 111b may optionally be included in the logical unit inventory.

The REPORT LUNS command ... requests that the peripheral device logical unit inventory accessible to the I_T nexus be sent to the application client. The logical unit inventory is a list that shall include the logical unit numbers of all logical units having a PERIPHERAL QUALIFIER value of 000b (see 6.4.2). Logical unit numbers for logical units with PERIPHERAL QUALIFIER values other than 000b and 011b may be included in the logical unit inventory. Logical unit numbers for logical units with a PERIPHERAL QUALIFIER value of 011b shall not be included in the logical unit inventory.

...

A non-empty peripheral device logical unit inventory that does not contain either LUN 0 or the REPORT LUNS well-known logical unit is valid.

Additionally, the Peripheral Qualifier table in the standard INQUIRY command description lists the defined peripheral qualifier values, as shown in the following table.

Qualifier	Description
000b	A peripheral device having the specified peripheral device type is connected to this logical unit. If the device server is unable to determine whether a peripheral device is connected, it also shall use this peripheral qualifier. This peripheral qualifier does not mean that the peripheral device connected to the logical unit is ready for access.
001b	A peripheral device having the specified peripheral device type is not connected to this logical unit. However, the device server is capable of supporting the specified peripheral device type on this logical unit.
010b	Reserved.
011b	The device server is not capable of supporting a peripheral device on this logical unit. For this peripheral qualifier, the peripheral device type shall be set to 1Fh. All other peripheral device type values are reserved for this peripheral qualifier.
1xxb	Vendor-specific.

3.10.3 Sun Standards

The logical unit inventory of the REPORT LUNS command addressed to logical unit number 0 shall contain logical unit number 0 when there are no logical units accessible to the initiator via the addressed SCSI target port. In this case, the peripheral qualifier in the standard INQUIRY data for logical unit number 0 shall be set to 001b.

The logical unit inventory of the REPORT LUNS command addressed to logical unit number 0 should contain logical unit number 0 when logical unit number 0 is masked from the initiator via the addressed SCSI target port, or when there is no physical device connected to logical unit number 0 and there are logical units other than logical unit number 0 accessible to the initiator. In this case, the peripheral qualifier in the standard INQUIRY data for logical unit number 0 shall be set to 001b.

If a logical unit is not supported on a device server, the device server may respond to a standard INQUIRY command addressed to that logical unit. In this case, the peripheral qualifier in the standard INQUIRY data should be set to 011b (it shall not be set to 000b, 001b or 010b, but may be set to a vendor-specific value). For this peripheral qualifier, the peripheral device type shall be set to 1Fh.

When a logical unit is masked from the initiator via the addressed SCSI target port, or when the physical device on a logical unit is not connected, the peripheral device type in the standard INQUIRY response should be set to the device type supported on the addressed logical unit. For device servers that support multiple peripheral device types on a logical unit, the peripheral qualifier should be set to 001b and the peripheral device type should be set to one of the device types supported on the addressed logical unit.

For purposes of device enumeration on the host, the host driver shall only attempt enumeration of logical units that have a peripheral qualifier of 000b in the standard INQUIRY data.

Applicability: All disks and arrays supporting multiple logical units

3.10.4 Sun Standards

Arrays that support multiple logical units and provide LUN Masking shall implement LUN Mapping so that different initiators see different volumes at the same unit number. In particular, arrays shall allow customers to configure the array so that different initiators see different volumes as logical unit number zero.

Applicability: All arrays supporting multiple logical units

3.11 Storage System (Target) Name for Multi-Unit Arrays

3.11.1 Background

Management software needs to correlate all the components in an array. In arrays with no LUN masking, this is simple; but when an array supports LUN masking, it may appear as multiple smaller arrays. Distinct volume and port identifiers are well defined (VPD page 83h and pWWN). In addition to these, a SCSI target Name shall be provided, and the same Name shall be exposed for every logical unit when accessed from any port.

This is generally not a driver issue, but is important to management and diagnostic applications.

3.11.2 Related Standards

T10 SPC-3 has added an association value to Inquiry Device Identification VPD response. (This response consists of a list of identifiers/names with descriptive metadata.) This association value (2) indicates that the name describes a target, as opposed to a logical unit or port. This is the preferred approach for future implementations.

The SNIA SMI-S specification discusses *durability* of names: a desired characteristic in which names tend to stay the same across configuration and FRU changes.

3.11.3 Sun Standards

All arrays supporting multiple logical units or multiple ports should provide a Storage System Name. New arrays shall provide this name as a VPD page 83 identifier with association equal to 2 (target system). Existing arrays provide this name using a vendor-specific command, and this command shall be included in the array interface specification.

Storage system names shall be world-wide unique.

Storage system names should be durable, that is, they should not easily change. An IP address is not a durable identifier: a customer may need to change it when the array is redeployed. Worse yet, if DHCP is used, it may change routinely. Using a Field Replaceable Unit (FRU) identifier as the System identifier should be avoided. If this is not possible, the identifier of the least likely to be replaced FRU shall be selected.

Applicability: All arrays supporting multiple logical units

3.12 Storage System Management IP Addresses

3.12.1 Background

Management applications need an in-band command to get the IP addresses used to manage the array. This capability allows an application to create URLs to applications that manage the discovered arrays.

Since multiple management interfaces (e.g., SNMP, telnet, http) are sometimes available, a list of URLs would be preferable.

3.12.2 Related Standards

The FC-GS Platform interfaces allow an FC node to expose a list of management URLs. FC-FS RNID also provides a single IP address and TCP/IP port number.

SPC-3 defines VPD page 85h, which returns a list of URLs for the device's network management services. VPD page 85h allows the reported management URLs to be hosted on the devices, on a Service Processor, or as a management proxy on the customer LAN.

3.12.3 Sun Standards

New arrays shall implement INQUIRY VPD page 85h. All arrays should implement INQUIRY VPD page 85h.

Applicability: All arrays with IP management capabilities

3.13 Asynchronous Notification of Name or Management Address Changes

3.13.1 Background

Management software relies on asynchronous notification to minimize dependency on polling all devices. The target and logical unit names are the primary mechanisms for identifying storage. If these names change, the management software is notified to assure consistent information for customers. Similarly, management software adjusts its representations of management addresses when they change.

Names may change due to FRU replacement or when a name is dynamically changed by a remote management or service processor.

3.13.2 Related Standards

SCSI standards use Unit Attention for devices to inform initiators of configuration changes.

3.13.3 Sun Standards

The array shall use a Unit Attention to notify host software whenever the logical unit name (see 3.3) or storage system name (see 3.11) change. The Unit Attention shall have ASC 3Fh and ASCQ 05h (DEVICE IDENTIFIER CHANGED). Changes to a logical unit name are discouraged because they may cause host-based software to break.

Applicability: RAID arrays

3.14 Device Name/Identifier Assumptions

This section documents driver assumptions regarding device identifiers.

Solaris OS Driver Behavior

The Solaris OS uses an algorithm involving several name and identifier fields in determining a "devid", a unique representation of disk volumes that is specific to the Solaris OS.

This algorithm is included in 7.2 "Solaris OS Disk Driver Device Identifier Generation" in this document.

3.15 SCSI-3 SPC Access Controls (ACLs)

3.15.1 Background

SPC-2 defines an in-band interface for LUN Masking (see ACCESS CONTROL and REPORT ACL commands). Most vendors have implemented out-of-band alternatives and do not appear to be supporting this interface.

3.15.2 Related Standards

This has been part of SPC starting in SPC-2.

3.15.3 Sun Standards

Use of this interface is discouraged.

Arrays should comply to the SNIA (<http://www.snia.org>) SMI-S standard for out-of-band array management.

Applicability: Arrays

3.16 Logical Unit Reset Support

3.16.1 Background

Logical unit reset is preferable to a target reset, because it affects only the given logical unit, not all logical units on a device.

Sun drivers currently support logical unit reset on Sun devices and devices which report that logical unit reset is supported via the REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS command.

3.16.2 Related Standards

Logical unit reset and REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS are part of SCSI standards.

3.16.3 Sun Standards

Logical unit reset shall be supported on devices that support multiple logical units.

REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS shall be supported on devices that support multiple logical units.

Applicability: Arrays supporting multiple logical units

3.17 Mode Page Requirements

3.17.1 Background

SCSI mode pages allow a disk or array to report parameters to clients. Solaris OS disk drivers require certain mode pages. Although geometry is largely irrelevant in SCSI disks (and virtual disks), it is still used by some host software.

3.17.2 Related Standards

SCSI Block Commands Specification (SBC-2) defines the format of mode sense pages reported by disk devices (including RAID arrays and virtualization systems).

3.17.3 Sun Standards

The Solaris OS requires that disk devices implement the following Mode Sense Pages:

Format Device mode page	page code 03h
Rigid Disk Geometry mode page	page code 04h
Caching mode page	page code 08h

Earlier versions of the Solaris OS require that the SBC 24-bit "Number of Cylinders" field in the Rigid Disk Geometry Mode Sense response shall fit in 16 bits (i.e., not exceed 65,535). This restriction has been fixed in the Solaris OS version 10, and patches are available for earlier versions.

Geometry-Related Policies

Some Solaris OS applications attempt to optimize data placement around the reported geometry. For example, the Solaris OS format utility defaults to creating partitions that start on cylinder boundaries. Because host I/O does not span partition (stripe) boundaries, it is beneficial to avoid RAID 5 stripes that span partitions (i.e., avoid a known situation that guarantees partial stripe updates). For this reason, Sun recommends that RAID/virtualization targets report geometry so that RAID stripe boundaries always fall on cylinder boundaries.

If a target allows a logical unit to be replicated or to migrate to a different type of backing store, the geometry information should not be changed. When a Solaris OS format creates a volume label, it creates a backup copy in a reserved area at the end of the disk. The location of the backup copy is determined based on the reported geometry. If the format utility is unable to read sector zero, it may be able to recover by using the backup copy. If the target changes the reported geometry, the recovery capability may fail.

Applicability: All disks and arrays

3.18 Notification of Capacity Change

3.18.1 Background

Arrays that support logical unit expansion should report capacity changes. Solaris OS storage applications (e.g., file systems) are being updated to support dynamic capacity changes based on unit attentions.

3.18.2 Related Standards

SPC-3 includes "CAPACITY DATA HAS CHANGED" ASC/ASCQ (2Ah/09h). Prior to this, MODE PARAMETERS CHANGED (2Ah/01H) was the best approximation.

3.18.3 Sun Standards

Management Applications require that arrays shall return CAPACITY DATA HAS CHANGED or MODE PARAMETERS CHANGED unit attention when volume capacity changes. Arrays should return CAPACITY DATA HAS CHANGED (the current standard).

Applicability: All disks and arrays

3.19 CDB Control Byte – NACA or LINK Flags

Solaris OS drivers do not set either the NACA or LINK bits in CDB control bytes.

Other platforms do use these flags. Notably, AIX uses NACA, and targets without NACA support perform suboptimally under AIX.

3.20 Sun Cluster Support

3.20.1 Background

Sun Cluster relies on SCSI reservations when cluster nodes are not able to communicate over the network.

3.20.2 Related Standards

SCSI-2 and SCSI-3 Primary Commands

3.20.3 Sun Standards

There are four SCSI versions of interest now: SCSI-2 and three versions of SCSI-3, documented in SPC (SCSI Primary Commands), SPC-2, and SPC-3. The Solaris OS does not base reservations-related decisions on the SCSI version number returned in the standard INQUIRY response. It appears that most arrays/disks support a subset of the SPC-2 PERSISTENT RESERVE I/OUT command set even if they claim conformance to SPC or SCSI-2. The Solaris OS use of this command set is documented below.

The Solaris OS ships with built-in multipath support (MPxIO) for Fibre Channel and iSCSI. MPxIO is enabled by default on recent Solaris versions. MPxIO is *not* enabled by default for SPARC® systems. MPxIO automates handling of reservations across all paths (Initiator/Target Nexi). In particular, MPxIO merges key lists from PERSISTENT RESERVATION IN responses, and removes duplicates before handing the list back to Sun Cluster.

Sun Cluster uses either SCSI-2 RESERVE/RELEASE or SCSI-3 persistent reservations commands, depending on the number of paths it sees connected to disks. (In this context, disk means either a disk or virtual disk from a RAID system.) If Sun Cluster sees two paths to a disk, Sun Cluster uses SCSI-2 RESERVE/RELEASE. If it sees more than two paths, Sun Cluster uses SCSI-3 persistent reservations.

- If multipath access to disks is done through FC/iSCSI drivers bundled with the Solaris OS and MPxIO, Sun Cluster sees one path per computer connected to the disk. If there are two computers connected to the disk, Sun Cluster uses SCSI-2 RESERVE/RELEASE. If there are more than two computers, Sun Cluster uses SCSI-3 persistent reservations.
- Sun Cluster may see multiple paths per computer with other multipath drivers. In this case, Sun Cluster may choose to use SCSI-3 persistent reservations with fewer computers.

Sun Cluster requires that devices that support SCSI-3 persistent reservations commands will support APTPL flag (persist across power failures).

Sun Cluster expects targets to conform to a subset of the SPC-2 standard. This subset is defined in the following SCSI interfaces.

3.20.3.1 WRITE(10) Command With Zero Data Length

WRITE(10) command with zero data length is used to test the status of reservations. This command shall return GOOD status if either the LUN is not reserved or the current system/initiator owns the reservation. This command shall return RESERVATION CONFLICT if the LUN is reserved by another system/initiator.

3.20.3.2 TEST UNIT READY

TEST UNIT READY returns GOOD status if either the LUN is not reserved or the current system/initiator owns the reservation. This was valid in SPC and SPC-2 and invalid in SPC-3.

3.20.3.3 SCSI-2 RESERVE and RELEASE

As defined in SPC-2.

In addition, Sun recommends that targets clear SCSI-2 reservations on Fibre Channel login and logout.

3.20.3.4 PERSISTENT RESERVE OUT With REGISTER Service Action

The Solaris interface is the MHIOCGRP_REGISTER ioctl.

The CDB is set up as follows:

- Scope is zero (ignored per SPC-3)
- Type is zero (ignored per SPC-3)

The PERSISTENT RESERVE OUT parameter list is set up as follows:

- APTPL (persist across power failures) is populated by Sun Cluster
- SPC-3 SPEC_I_PT is always zero, PERSISTENT RESERVE OUT parameter list additional parameter data is not used
- ALL_TG_PT is zero

There are four scenarios in which this command is used:

1. Sun Cluster believes that the host is already registered with the device – PERSISTENT RESERVE OUT parameter list RESERVATION KEY and SERVICE ACTION RESERVATION KEY are set to the existing key.
2. Sun Cluster believes that the host is not registered with the device - PERSISTENT RESERVE OUT parameter list RESERVATION KEY is zero. SC sets SERVICE ACTION REGISTRATION KEY to “XXXXXXXX” (corresponds to SPC-3).
3. Sun Cluster wishes to unregister with the device - PERSISTENT RESERVE OUT parameter list RESERVATION KEY is the existing key and PERSISTENT RESERVE OUT parameter list SERVICE ACTION REGISTRATION KEY is zero (corresponds to SPC-3r21 top of page 176 - “b”).
4. Sun Cluster believes that the host is registered with a different key, and tries to register with each of the registration keys returned by MHIOCGRP_INKEYS as the old key until it gets a successful registration. This can happen when a cluster is reinstalled or when storage is moved from one

cluster to another, because PGR keys are persistent and the key used by a cluster is specific to that cluster installation.

3.20.3.5 PERSISTENT RESERVE OUT With RESERVE Service Action

The Solaris interface is the MHIOCGRP_RESERVE ioctl.

The CDB is set up as follows:

- Scope is zero (logical unit)
- Type is Write Exclusive, Registrants Only
- APTPL is set to true (ignored per SPC-3)

The PERSISTENT RESERVE OUT parameter list is set up as follows:

- Scope Specific Address is zero (Sun Cluster does not use this)
- RESERVATION KEY is provided by Sun Cluster
- SERVICE ACTION REGISTRATION KEY is zero (ignored per SPC-3)

3.20.3.6 PERSISTENT RESERVE OUT With PREEMPT AND ABORT Service Action

The Solaris interface is the MHIOCGRP_PREEMPTANDABORT.

The CDB is set up as follows:

- Scope is zero (logical unit)
- Type is Write Exclusive, Registrants Only (from caller)

The PERSISTENT RESERVE OUT parameter list is set up as follows:

- Scope Specific Address is zero (Sun does not use this)
- RESERVATION KEY is provided by caller
- SERVICE ACTION REGISTRATION KEY is the reservation to be removed

3.20.3.7 PERSISTENT RESERVE OUT With REGISTER AND IGNORE EXISTING KEY Service Action

The Solaris interface is the MHIOCGRP_REGISTERANDIGNOREKEY ioctl.

The CDB is set up as follows:

- Scope is zero (ignored per SPC-3)
- Type is zero (ignored per SPC-3)

The PERSISTENT RESERVE OUT parameter list is set up as follows:

- APTPL (persist across power failures) is populated by Sun Cluster
- SPC-3 SPEC_I_PT is always zero, PERSISTENT RESERVE OUT parameter list additional parameter data is not used
- ALL_TG_PT is zero
- SERVICE ACTION RESERVATION KEY is the key to register
- RESERVATION KEY is zero (ignored per SPC-3)

- REGISTRATION KEY is zero (corresponds to SPC-3r21 top of page 176 - “b”)

3.20.3.8 PERSISTENT RESERVE IN With READ KEYS Service Action

Used in two contexts:

1. Whenever a disk is added to the device tree:

If the SENSE KEY is ILLEGAL REQUEST, Sun assumes that a SCSI-3 PGR is not supported, but SCSI-2 RESERVE/RELEASE is supported.
2. Used directly by Sun Cluster (Solaris ioctl MHIOCGRP_INKEYS):
 - Response shall include PRgeneration (per SPC).
 - Response shall include a list of reservation keys (per SPC):
 - If the number of keys would cause the key list to exceed the ALLOCATION LENGTH from the CDB, the number of keys should be truncated.
 - The ADDITIONAL LENGTH field (in the response) shall contain the actual number of bytes in the reservation key list.

3.20.3.9 PERSISTENT RESERVE IN With READ RESERVATIONS Service Action

The Solaris interface is the MHIOCGRP_INRESVS.

- Response shall include PRgeneration (per SPC).
 - Response shall include:
 - No reservation held – Response ADDITIONAL LENGTH is zero.
 - Reservation held – Response shall contain:
 - ADDITIONAL LENGTH field is 16.
 - RESERVATION KEY field holds the reservation key.
- Note:** With Element scope support in SPC-2, it was possible to have multiple reservation keys. Element scope is not used by SC (and is deprecated in SPC-3), so more than one reservation key would not be expected.

Applicability: All disks and arrays

3.20.3.10 Removing Keys

Sun Cluster removes (“scrubs”) keys as follows:

1. Get the existing keys (see 3.20.3.8).
2. Register and ignore keys (see 3.20.3.7).
3. If that fails, register with old key (see 3.20.3.4).
4. If that fails, register with a key of zero.
5. If that fails, register with another node's existing key.
6. Preempt everyone else's keys (see 3.20.3.6).
7. Remove my key (see 3.20.3.4 scenario 3).

4 Fibre Channel-Specific Policies

4.1 Multipath Support

Solaris OS Fibre Channel (FC) drivers support multipath devices that conform to the T10 standards as documented in 3.5 and 3.6.

4.2 Asynchronous Notification of Device Changes

FC targets shall establish a unit attention condition for the initiator ports associated with all affected I_T nexuses with sense codes defined in this document or SPC when configuration changes occur. FC targets should send LIP or RSCN a few seconds after the configuration change to any initiator that does not issue a CDB that receives the unit attention.

4.3 Logical Unit Address Methods

Solaris OS drivers use the addressing method in REPORT LUNS responses. For FC devices, the Solaris OS supports the T10 *peripheral device* address method with up to 14-bit logical unit addresses (see 3.7 and SAM).

5 iSCSI-Specific Policies

This section describes policies related to target devices attached to the iSCSI initiator bundled with the Solaris OS version 10. In this section, the words “node”, “portal”, “session”, and “connection” are used consistently with IETF iSCSI RFC 3720.

5.1 Support for Multiple LUNs

The target may support a single LUN per target node or multiple LUNs per target node.

5.2 Multipath Support

iSCSI drivers in the Solaris OS support several different approaches for multipath access to devices:

- Multiple Connections per Session (as defined in iSCSI RFC 3720).
- SCSI LUN Multipath with a single iSCSI target with multiple Target Portal Group Tags (TPGTs).
- SCSI LUN Multipath with a single iSCSI target with a single Target Portal Group Tag requiring the customer to manually configure the initiator with multiple ISIDs for the targets.
- Login Redirect – The target supports a “redirection portal” IP address that provides the IP addresses Link Aggregation Control Protocol as defined in IEEE specification 802.3ad.

Any iSCSI target that provides more than one iSCSI network portal (i.e., NIC) should support at least one of these multipath approaches. Sun's preference for targets is the order of the list above (i.e., the most preferred approach is Multiple Connections per Session).

Note: The Solaris OS iSCSI initiator does not support Multiple Connections per Session until Solaris OS version 10, Update 4. With previous versions of the Solaris OS, a target supporting MC/S should be able to describe a procedure using approach 3 with appropriate `iscsiadm` commands. With Solaris OS version 10, Update 4, the Solaris iSCSI initiator should be able to operate with an MC/S target without this extra manual step.

5.3 Asynchronous Notification of Device Changes

iSCSI targets shall establish a unit attention condition for the initiator ports associated with all affected I_T nexuses with sense codes defined in this document or SPC when configuration changes occur.

iSCSI targets should support iSNS (see RFC 4171) and send appropriate Asynchronous messages through iSNS when the target configuration changes.

The Solaris OS iSCSI initiator also accepts iSCSI and SCSI asynchronous events as described in IETF RFC 3720 3.5.2.1 and 10.4.7. Asynchronous SCSI sense may be used to inform the Solaris OS of configuration changes such as LUN creation/ deletion or capacity changes.

5.4 Logical Unit Address Methods

Drivers in the Solaris OS use the addressing method in REPORT LUNS responses. For iSCSI devices, the Solaris OS supports the T10 *peripheral device*, *flat space*, or *logical unit* address methods with up to 14-bit logical unit addresses (see 4.6 and SAM).

5.5 Target Alias Handling

If the target management software allows the customer to provide a name referred to as an “alias”, it should be related to a target node and included as part of the TargetAlias parameter as defined in IETF RFC 3720.

5.6 IQN Name Format

IQN names should use the following format:

Per IETF RFC 3721, the iSCSI qualified name (iqn) format contains (in order):

1 - The string “iqn.”

2 - A date code specifying the year and month in which the organization registered the domain or subdomain name used as the naming authority string.

3 - The organizational naming authority string, which consists of a valid, reversed domain or subdomain name.

Optionally, a ':', followed by a string of the assigning organization's choosing, which shall make each assigned iSCSI name unique.

For Sun-branded iSCSI devices, the date code and organization shall be “1986-03.com.sun”.

For Sun-branded iSCSI devices, the remainder of the name should use one of these two formats:

```
:01:<MAC address>.<timestamp>[.<friendly name>]
:02:<UUID>.<local target name>
```

The square brackets around “.<friendly name>” indicate that it is optional.

The value between the first two colons indicates the format used by the remainder of the string.

<MAC address> is the 12 lowercase hexadecimal characters encoding a 6-byte MAC address.

<timestamp> is the lowercase hexadecimal representation of the number of seconds since January 1, 1970 and the time the name is created. The number of seconds value may be attained from the UNIX time() function.

<friendly name> is an extension that provides a name that is meant to be meaningful to users.

<UUID> is a unique identifier consisting of 36 in the format xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx where x represents a lowercase, hexadecimal character. The UUID may be acquired from libuuid, an open-source UUID library bundled with the Solaris platform and other platforms.

<local target name> is a name used by the target.

All these fields shall conform to the IQN name requirements of IETF RFCs 3720, 3721, and 3722.

5.7 iSNS Support

Target devices should support iSNS as defined in IETF RFC 4171. Targets should act as iSNS clients and register with an iSCSI server.

Targets should allow manual entry of iSNS server name/IP address. The Solaris OS does not currently support advertisement of the iSNS server address through DHCP, though support may be added in the future. The Solaris OS does not support SLP discovery of the iSNS server address.

Targets should not provide their own iSNS servers and shall not have an iSNS server enabled by default.

The Solaris OS Initiator has been tested with the Microsoft iSNS server.

5.8 RADIUS Support

Support for RADIUS (IETF RFC 2865) is optional.

The Solaris OS Initiator has been tested with the FreeRadius server.

5.9 iSCSI Features Related to RFC 3720 Parameters

The following table provides notes on iSCSI features related to parameters defined in IETF RFC 3720. If Sun does not have any requirements beyond those in the RFC, the parameter is omitted.

Parameter	Notes
HeaderDigest	Sun prefers that targets support HeaderDigest.
DataDigest	Sun prefers that targets support DataDigest.
SendTargets	Sun prefers that targets support SendTargets.
TargetName	Required (see 5.6).
TargetAlias	Recommended (see 5.5).
MaxConnections	See 5.2.
InitialR2T	Set as optimal for the target.
ImmediateData	Sun strongly recommends support for immediate data, because this contributes to lower latency and higher utilization.
MaxBurstLength	Sun prefers support for MaxBurstLength of 64K or greater, because larger values minimize utilization of host resources.
FirstBurstLength	Sun prefers support for FirstBurstLength of 64K or greater, because larger values minimize utilization of host resources.
MaxRecvDataSegment Length	Sun prefers support for MaxRecvDataSegmentLength of 64K or greater, because larger values minimize utilization of host resources.
MaxOutStandingR2T	Sun prefers MaxOutStandingR2T to be 1.
DataPDUInOrder	Sun prefers that targets support DataPDUInOrder.
DataSequenceInOrder	Sun prefers that targets support DataSequenceInOrder.
ErrorRecoveryLevel	At this time, the Solaris OS Initiator doesn't support level 1 or 2. But support is planned for Solaris 10, Update 4.
Authentication	Should support CHAP bidirectional, CHAP unidirectional, or None.
OFMarker, IFMarker, OFMarkInt, IFMarkInt	At this time, the Solaris OS iSCSI Initiator does not support markers.

6 Serial Attached SCSI (SAS) Specific Policies

6.1 Multipath Support

The Solaris OS does not have built-in support for multipath access to SAS devices.

6.2 Asynchronous Notification of Device Changes

SAS targets shall establish a unit attention condition for the initiator ports associated with all impacted I_T nexuses with sense codes defined in this document or SPC when configuration changes occur.

6.3 Logical Unit Address Methods

Solaris OS drivers use the addressing method in REPORT LUNS responses. For SAS devices, the Solaris OS supports the T10 *peripheral device* address method with up to 8-bit logical unit addresses (see 3.7 and SAM).

7 Behavior of Solaris OS Drivers

This chapter provides information about SCSI-protocol related behavior used in the Solaris OS array drivers. This provides some information about how the Solaris OS uses some of the standard interfaces described above. This information is likely to be useful to array implementers.

This documentation is a snapshot of the implementation as of this writing, which will change as the I/O subsystem is further enhanced and updated. There is no formal or implied commitment to maintain this document as the drivers are updated.

7.1 The Solaris OS Queue Throttle

The Solaris OS disk drivers throttle the number of outstanding I/O requests to each logical unit to prevent overloading the device. The default throttle value is 256. This may be overridden system-wide by setting `sd_max_throttle` in `/etc/system`. The system-wide setting may be overridden for specific types of disk products in `/kernel/drv/[s]sd.conf` as described in 7.3.4.2. See 7.3.4 for cautionary advice regarding changes to the default device driver behavior. If the number of outstanding requests is equal to or greater than the throttle, subsequent requests are queued by the driver.

The throttle value is reduced when the transport is busy or the target returns a queue full status. The algorithm sets the throttle value to the number of outstanding requests. On subsequent requests, if the number of outstanding requests is greater than or equal to the throttle value, the request is queued.

The throttle value gets reset to the default `sd_max_throttle` value after it falls below `sd_min_throttle` or when the device is closed. The default value of `sd_min_throttle` is 8 and may be overridden similarly to `sd_max_throttle` as described above. The lowest possible value for `sd_min_throttle` is 2. When the throttle algorithm is activated due to the transport being busy (as opposed to a QUEUE FULL event), the driver also starts a background task that waits 60 seconds and then restores the original throttle value.

An adaptive throttle algorithm was introduced in Solaris OS version 10 update 3. The new adaptive throttle behavior provides a method to restore the throttle value back to the maximum in an increasing step function. When QFULL is received by the drivers, the throttle is reduced as before. With the new behavior there's an initial 60 second delay (by using a background task the same as for transport busy), and then the throttle value is increased by 10% of the current value. This is repeated every 10 seconds until the throttle value reaches `sd_max_throttle` value.

With Sun StorEdge Traffic Manager enabled, the throttle is applied to requests across all paths.

7.2 Solaris OS Disk Driver Device Identifier Generation

The Solaris OS target disk driver uses unique information acquired from a disk device to generate a device identifier. If unique device information is unavailable, the disk driver will use a Solaris OS system call to generate a unique device identifier which is then stored by the disk driver on the alternate cylinders of the disk.

There are three types of unique information associated with SCSI and Fibre Channel devices that the disk driver attempts to acquire.

Device Identification Vital Product Data Page (83h) - Provides the means to retrieve zero or more identification descriptors applying to the logical unit (SCSI-3 SPC 8.4.3). This is the preferred disk/volume identifier.

Unit Serial Number Vital Product Data Page (80h) - Provides a product serial number for the target or logical unit (SCSI-3 SPC 8.4.6).

INQUIRY command contents - Sun Microsystems provides product specs for all vendor drives which indicate that vendors shall populate the vendor identification (vid) and product identification (pid) fields of the Inquiry data (SCSI-3 SPC 7.5.1), with unique per logical unit values. In the past, Sun Microsystems required that the Inquiry data include 11 bytes of vendor-specific data consisting of a unique per logical unit serial number.

As the Solaris OS and SCSI have evolved, along with the introduction of Fibre Channel technology, the disk driver algorithm for generating a unique device identifier has changed. The following sections and table describe the algorithm used by the SPARC disk driver for the most recent update of Solaris OS version 9.

In addition, the disk driver was split with the introduction of Fibre Channel and there existed a parallel SCSI version (sd) and Fibre Channel version (ssd) until Solaris OS version 9, at which point the code bases were merged back together. As a result, some of the Solaris OS releases have different algorithms depending upon the bus type (i.e., parallel SCSI or Fibre Channel).

Solaris OS Version 9 (and later) sd & ssd Algorithm:

1. Issue an INQUIRY command for the list of supported VPD pages.
2. Determine from the VPD page list if the device identification VPD page (83h) is supported.
 - a. If page 83h is available, issue an INQUIRY command for VPD page 83h.
 - b. Inspect the returned identification descriptor list to determine if descriptor type 03h (NAA) is available. If descriptor type 03h is available, it is used as the device identifier.
3. If VPD page 83h with descriptor type 03h is not available, determine from the VPD page list if the unit serial number VPD page (80h) is available.
 - a. If page 80h is available, issue an INQUIRY command for VPD page 80h.
 - b. Inspect the VPD page 80h contents to determine if they are valid.
 - c. Create a device identifier by concatenating the contents of VPD page 80h with the INQUIRY vid and pid: `<vid><pid><VPD page 80h serial #>`
4. If VPD pages 83h and 80h are not available with valid data, determine if the disk has been qualified by Sun as indicated by the string "SUN" in the contents of the INQUIRY pid contents.
 - a. If the disk is Sun qualified, create a device identifier by concatenating the contents of the INQUIRY vid, pid, and serial number: `<vid><pid><INQUIRY serial #>`
 - b. If the disk is not Sun qualified, fabricate a device identifier using the Solaris OS device identifier fabrication routine. This OS routine uses the machine's host identifier and a timestamp to create a device identifier.

7.3 Error Handling Behavior

7.3.1 Introduction

The Solaris OS versions 9/10 SPARC (and Solaris version 10 x86) SCSI disk drivers consist of two drivers compiled from a single source base. This document addresses the error handling behavior and configurability of the Solaris 9/10 SPARC SCSI disk drivers. The driver is sd for parallel SCSI disks

and ATAPI/SCSI removable media devices, and `ssd` is the driver for SPARC storage Array and Fibre Channel disk devices.

7.3.2 Error Handling Policy

The Solaris OS target disk drivers implement a fairly simple error handling policy. The primary method of dealing with an error is to retry the failed command a fixed number of times. The I/O is failed after the retry limit has been reached. Each I/O is independently retried. Other than retrying, the only error recovery actions implemented by the disk driver are an attempt to spin up the device (via the `START_STOP_UNIT` command) in response to a “Not Ready” sense key and resets.

7.3.3 Retry/Reset Parameters

The two most requested pieces of information regarding the Solaris OS target disk drivers are how are retries/resets implemented and how to configure them. This section details the various parameters used by the implementation and how they are initialized and configured. Error Handling Implementation (7.3.5) is a comprehensive overview of the implementation. Parameter Configuration (7.3.4) details configuration of the driver behavior.

7.3.3.1 I/O Retry Counts

Each I/O OPERATION processed by the driver has an associated retry count (contained in the `buf` extension for the Solaris OS versions 9/10 implementation). Every time the command is retried this I/O retry count is incremented. The I/O retry count is used for comparison against the counts described in the following sections. Each I/O also has an associated victim retry count, which applies when a command fails due to a SCSI bus reset issued by another initiator on the SCSI bus.

7.3.3.2 Global Retry Counts

This set of global variables form the core of the driver reset and retry implementation. The value of `sd_io_time` is used as the command timeout (in seconds) for all I/O. If a command does not complete within this interval it is retried based upon the retry count. The value of `sd_retry_count` is used as the retry count for most error conditions (see 7.3.5). The victim retry count is most commonly applied when a command fails due to a bus device reset associated with a recovery operation of another device on the chain. The reset retry count is the number of retries attempted before this initiator issues a bus or device reset.

The following table shows that most commands are retried 3 or 5 times with a reset issued after the first or second command failure for specific error cases. If the device is completely unresponsive, each retry takes as long as the current setting of `sd_io_time` (default is 60 seconds) to complete, but in most cases the retries will be issued after a minimal delay of 0-100msec (see 7.3.3.4 and 7.3.3.5).

Global Name*	Fibre Channel (ssd)	Parallel SCSI (sd)	Parallel SCSI Removable Media (sd)
<code>sd_io_time</code>	60	60	60
<code>sd_retry_count</code>	3	5	5
<code>sd_victim_retry_count</code>	<code>sd_retry_count*2</code>	<code>sd_retry_count*2</code>	<code>sd_retry_count*2</code>
<code>sd_reset_retry_count</code>	<code>sd_retry_count/2</code>	<code>sd_retry_count/2</code>	<code>sd_retry_count/2</code>
*Fibre Channel versions of these global variables use the “ <code>ssd_</code> ” prefix.			

7.3.3.3 Per Device Type Retry Counts

Over time, the driver retry and reset behavior has been enhanced for unique error handling. The retry behavior for “Not Ready” and “Busy” status have been modified and the reset behavior is configurable on a per-device type basis (determination based upon the SCSI INQUIRY data). The following table shows that most commands that fail with a “Not Ready” status will be retried one or two times for disks and six to 10 times for a parallel SCSI removable media device. In addition, the driver implements unique retry behavior for targets manufactured by LSI (or Symbios) in response to a “Not Ready” status. Special handling is also implemented for “Busy” status on the Sun StorEdge T3. This special handling for certain target devices ensures that there is sufficient time to complete a failover (which takes up to three minutes) without being reset during the failover.

Variable Name	Fibre Channel (ssd)	Parallel SCSI (sd)	Parallel SCSI Removable Media (sd)
un_notready_retry_count	sd_retry_count/2	sd_retry_count/2	sd_retry_count*2
LSI*	24	24	24
un_busy_retry_count	sd_retry_count	sd_retry_count	sd_retry_count
Sun StorEdge T3*	60	60	60
un_reset_retry_count	sd_reset_retry_count	sd_reset_retry_count	sd_reset_retry_count
Sun StorEdge T3*	36	36	36

*Set via the driver device configuration table (see 7.3.4).

7.3.3.4 Global Timeouts/Intervals

The following set of hard-coded timeouts and intervals is used to initialize driver global and per instance data and is part of the default driver retry behavior. See 7.3.5 for details on their use.

Variable Name*	Fibre Channel (ssd)	Parallel SCSI (sd)	Parallel SCSI Removable Media (sd)
SD_RETRY_DELAY	100 (msec)	0	0
SD_BSY_TIMEOUT	5 (sec)	5 (sec)	5 (sec)
SD_IO_TIME	60 (sec)	60 (sec)	60 (sec)

*Fibre Channel versions of these global variables use the “SSD_” prefix.

7.3.3.5 Per-Instance Timeouts/Intervals

Like certain driver retry and reset counts, the driver uses the following set of timeouts and intervals that are configurable on a per-instance basis. The values of cmd_timeout and uscsi_timeout are used for the majority of error conditions. This is an interesting configuration hook in that the command timeout could potentially be configured on a per-instance basis, although the driver does not currently leverage this capability. The reserve_release_timeout is used by the driver to determine how long a period the target should take to release a reservation before a failure is indicated and a bus reset should be issued. The Sun StorEdge T3 takes up to 60 seconds to complete this action.

Variable Name	Fibre Channel (ssd)	Parallel SCSI (sd)	Parallel SCSI Removable Media (sd)
un_cmd_timeout	sd_io_time	sd_io_time	sd_io_time
un_uscgi_timeout	un_cmd_timeout	un_cmd_timeout	un_cmd_timeout*2
un_reserve_release_time	5	5	5
Sun StorEdge T3*	60	60	60
*Set via the driver device configuration table (see 7.3.4).			

7.3.4 Parameter Configuration

A word of warning is necessary before beginning any discussion of how the driver error handling parameters are configured. *All of the disk behavior in all configurations has been qualified with the default values documented above and the implementation documented in 7.3.5 Error Handling Implementation. Modifying these parameters can have a serious negative impact.* Before any parameters are modified, a comprehensive understanding of the driver retry behavior and the system configuration is required. If the specific error mode of the configuration is unknown, it is not advised to modify the parameters in the hope that things will get better. The configuration behavior must be thoroughly understood before any attempt is made to change the driver retry behavior.

In addition to the potential system issues identified above, the Solaris OS configuration mechanisms themselves are incomplete. There is no mechanism for understanding which parameters are configurable. The current mechanism makes it equally easy to set a well-known parameter, or an arbitrary integer within a kernel module. The administrator isn't presented with any way of discerning the difference between safe and unsafe parameters. It is also not possible to obtain a list of parameters that can be tuned. There is no common mechanism for ascertaining which system parameters have been set different to the default. During upgrade or system setup, it is almost impossible to gather this information. There isn't any mechanism to understand what a reasonable default should be for a parameter. No central administration command(s) exist to check the validity of system parameters, potentially leading to downtime after the next reboot. The Solaris OS update mechanism should understand and upgrade/delete settings of parameters that are required to change. There is no mechanism for regressing back to a known set of working parameters. With the Solaris OS, it is possible to set parameters which render a system unbootable. A common framework to allow these common administrative practices to be completed, with minimal effort and/or risk to availability of the system, is required before broad exposure to driver parameters can be granted.

7.3.4.1 Driver Device Table

The device driver contains a static configuration table. Driver parameters are configurable for specific vendor equipment based on the SCSI INQUIRY data. An example of how this is implemented is the un_notready_retry_count values for LSI targets as well as several parameters mentioned in the previous sections for the Sun StorEdge T3 array. The majority of the current entries in the device configuration table are used to set the throttle value. This type of configuration is appropriate when a very specific device needs unique handling, independent of the system configuration. Introducing this type of configuration is not always feasible because the source code must be updated and the driver binaries must be recompiled. The device driver static configuration table is only available to the development community.

7.3.4.2 *sd.conf* File

The Solaris OS target disk driver .conf files are the supported method for tuning the driver behavior for a specific system configuration. The .conf entry for sd and ssd takes the following form:

```
sd-config-list=
    "VENDOR1 PRODUCT1", "sd-ver1-tst-data";
sd-ver1-tst-data= 1,0x1F01F,0,12,1,5,0,0,8,0,0,0,0,0,0,5,60,0,2,0;
```

where:

```
"VENDOR1 "      is the contents of the vendor identification field of the
                  device response to a SCSI INQUIRY.
"PRODUCT1"      is the contents of the product identification field of the
                  device response to a SCSI INQUIRY.
"sd-ver1-tst-data" is the property array name
"sd-ver1-tst-data" is the property array definition.
```

Note: The vendor field must be 8 bytes in length. If the device response is less than 8 bytes, the .conf file entry must be padded with spaces. The product field may be up to 16 bytes in length. If fewer than 16 bytes are supplied, the comparison is limited to the length supplied.

Note: The bit fields specified in the following tables are based on the current driver implementation for each release. These fields have been updated in legacy releases through patches. If the following error occurs, it means an older version of the driver is installed on the system which does not support all of the parameters. The user may either update to the latest driver patch, or change the field and flags based to the length specified in the message (i.e. **n**).

```
data property sd-ver1-tst-data list size is incorrect. No props set.
Size expected: version + 1 flagword + n props.
```

The property array is defined by `version`, `flag`, `integer array`. The version is always set to 1 and the flag is a bit mask indicating the members of the integer array to be used during device configuration. *The user should set only those bits in the mask that correspond to the configurable parameters being set.* For example if the user wishes to configure the maximum throttle for sd in Solaris OS version 9, the flag bit mask should be set to 0x00001. Similarly if the user wishes to configure max throttle, and min throttle for ssd in Solaris OS version 9, the flag bit mask should be set to 0x8001.

The following tables show the set of configurable parameters for the sd/ssd drivers and example .conf file entries.

Solaris OS Versions 9/10		
flag value	sd	ssd
0x00001	throttle (max)	throttle (max)
0x00002	controller type	not ready retries
0x00004	not ready retries	busy retries
0x00008	fabricate device id	fabricate device id
0x00010	disable caching	disable caching
0x00020	busy retries	controller type
0x00040- 0x00800	*	*
0x01000	reset retries	reset retries
0x02000	reservation release time	reservation release time
0x04000	TUR check	TUR check
0x08000	throttle (min)	throttle (min)
0x10000	disable disksort	disable disksort
0x20000	enable logical unit reset	enable logical unit reset

* These bit positions are used by the Solaris OS version 9 x86 sd driver and have been reserved on the SPARC platform to ease the Solaris OS version 10 merge of the x86 sd driver with the SPARC driver.

```
#This is a sample sd config list entry for Solaris 9+ SPARC
sd-config-list=
    "VENDOR1 PRODUCT1", "sd-ver1-tst-data";
sd-ver1-tst-data= 1,0x3F01F,12,1,5,0,0,8,0,0,0,0,0,0,5,60,0,2,0,1;
#
#NOTE: Field Definitions
#sd-ver1-tst-data= version, flags, throttle (max), ctype,
#               not_ready_retry, fabricate devid, nocache,
#               busy_retries, rsvd, rsvd, rsvd, rsvd, rsvd, rsvd,
#               reset_retries, reservation_release_time, TUR_check,
#               throttle (min), disable_disksort, logical unit reset

# This is a sample ssd config list entry for Solaris 9+ SPARC
ssd-config-list=
    "VENDOR1 PRODUCT1", "ssd-ver1-tst-data";
ssd-ver1-tst-data= 1,0x3F01F,12,8,5,0,0,1,0,0,0,0,0,0,5,60,0,2,0,1;
#
# NOTE: Field Definitions
#ssd-ver1-tst-data= version, flags, throttle (max), not_ready_retry,
#                  busy_retries, fabricate devid, nocache, ctype,
#                  rsvd, rsvd, rsvd, rsvd, rsvd, rsvd,
#                  reset_retries, reservation_release_time, TUR_check,
#                  throttle (min), disable_disksort, logical unit reset
```

7.3.4.3 /etc/system File

The SolarisOS specification file may be used to set an integer variable in the kernel or a kernel component such as the disk drivers. This facility is not recommended and should be used with caution (if at all). The following driver global variables are configurable via /etc/system:

```
sd_io_time
sd_retry_count
sd_victim_retry_count
sd_reset_retry_count
sd_max_throttle
sd_min_throttle
```

See 7.3.3 for details and dependencies of the retry and timeout global variables.

7.3.5 Error Handling Implementation

The device driver error handling implementation begins in the interrupt processing routine. This routine (sdintr) uses the SCSI packet reason, status, and sense key to determine how to properly handle the I/O. In general four retry algorithms are implemented as noted in 7.3.5.1 with exceptions noted in the sections detailing the specific SCSI packet reason, status, and sense keys.

7.3.5.1 Basic Retry Algorithms

Standard Retry

The standard retry consists of a check of the I/O retry count against sd_retry_count. If the I/O has been retried more than the maximum count it is failed, otherwise it is immediately retried (after a 100msec delay for ssd). This results in most failures being retried 3-5 times with minimal delay. The driver retry implementation take substantially longer for the case of commands timing out due to an unresponsive or unavailable target device (see 7.3.6.1).

Victim Retry

The victim retry count is applied when a command fails due to a SCSI bus reset; most commonly due to a recovery operation of another device on the chain. The victim retry consists of a check of the I/O victim retry count against sd_victim_retry_count. If the I/O has been retried more than the maximum count it is failed, otherwise it is immediately retried (after a 100msec delay for ssd).

Busy Retry

The busy retry consists of a check of the I/O retry count against un_busy_retry_count. If the I/O has been retried more than the maximum count it is failed, otherwise it is immediately retried (after a 100msec delay for ssd). The busy retry is applied when the device returns a SCSI status of "Busy"

Nocheck Retry

A nocheck retry consists of immediately retrying a command (after a 100msec delay for ssd) without checking any of the global driver retry counts.

7.3.5.2 SCSI Packet Reason Handling

The SCSI packet reason is a piece of information specific to the Solaris OS that is returned by the HBA layer, indicating whether the command was successfully transported. The reason codes are shown in the following table.

Reason Code	Description	Retry Type
CMD_CMPLT	No transport errors - normal completion.	N/A
CMD_INCOMPLETE	Transport stopped with abnormal state.	Reset Tgt*, Standard Retry
CMD_TRAN_ERR	Unspecified transport error.	Reset Tgt*, Standard Retry
CMD_RESET	SCSI bus reset destroyed command.	Reset Tgt*, Victim Retry
CMD_ABORTED	Command transport aborted on request.	Reset Tgt*, Victim Retry
CMD_TIMEOUT	Command timed out.	Reset Tgt*, Standard Retry (see 7.3.6.1)
CMD_TAG_REJECT	Target rejected the tag message.	Disable tagged queuing, Nocheck Retry
CMD_UNX_BUS_FREE	Unexpected Bus Free Phase.	Standard Retry
CMD_DMA_DERR	DMA direction error.	Reset Tgt*, Standard Retry

Reason Code	Description	Retry Type
CMD_CMD_OVR	Command Overrun.	Reset Tgt*, Standard Retry
CMD_STS_OVR	Status Overrun.	Reset Tgt*, Standard Retry
CMD_BADMSG	Message not Command Complete.	Reset Tgt*, Standard Retry
CMD_NOMSGOUT	Target refused to go to Message Out phase.	Reset Tgt*, Standard Retry
CMD_XID_FAIL	Extended Identify message rejected.	Reset Tgt*, Standard Retry
CMD_IDE_FAIL	Initiator Detected Error message rejected.	Reset Tgt*, Standard Retry
CMD_ABORT_FAIL	Abort message rejected.	Reset Tgt*, Standard Retry
CMD_REJECT_FAIL	Reject message rejected.	Reset Tgt*, Standard Retry
CMD_NOP_FAIL	No Operation message rejected.	Reset Tgt*, Standard Retry
CMD_PER_FAIL	Message Parity Error message rejected.	Reset Tgt*, Standard Retry
CMD_BDR_FAIL	Bus Device Reset message rejected.	Reset Tgt*, Standard Retry
CMD_ID_FAIL	Identify message rejected.	Reset Tgt*, Standard Retry

* The target is unconditionally reset for these error cases, which would result in a reset occurring between each retry if the error is persistent.

7.3.5.3 SCSI Status Handling

The SCSI status returned by the target indicates whether the command was successfully completed, as shown in the following table.

SCSI Status Code	Description	Retry Type
GOOD	The target has successfully completed the command.	Standard if non-zero residual for READ/WRITE commands.
CHECK	A contingent allegiance condition has occurred.	Standard Retry.
TERMINATED	The target terminated the current I/O process after receiving a terminate I/O process message.	Standard Retry.
BUSY	The target is busy.	Reset target after checking per instance reset retry count, Busy Retry.
RESERVATION CONFLICT	This status is returned whenever an initiator attempts to access a reserved logical unit.	Standard for non-clustering configurations.
QFULL	The command queue in the target is full.	Reduce the driver throttle, Nocheck Retry.
SCSI-2	This is the SCSI-2 modifier bit.	Unexpected Status, Fail I/O.
MET	The requested operations are satisfied.	Unexpected Status, Fail I/O.
INTERMEDIATE	Returned for every successfully completed command in a series of linked commands.	Unexpected Status, Fail I/O.
INTERMEDIATE MET	This status is a combination of STATUS_MET and STATUS_INTERMEDIATE.	Unexpected Status, Fail I/O.

7.3.5.4 SCSI Sense Key Handling

The SCSI sense keys that can be returned by the target are shown in the following table.

SCSI Extended Sense Key	Description	Retry Type
NO SENSE	Indicates that there is no specific sense key information to be reported.	Standard Retry.
RECOVERABLE ERROR	Indicates that the last command completed successfully with some recovery action performed by the target.	Standard Retry if a non-zero residual for READ/WRITE commands.
NOT READY	Indicates that the logical unit addressed cannot be accessed.	See 7.3.6.2.
MEDIUM ERROR	Indicates that the command terminated with a non-recovered error condition that was probably caused by a flaw on the medium or an error in the recorded data.	Reset target after checking per instance reset retry count, Busy Retry.
HARDWARE ERROR	Indicates that the target detected a non-recoverable hardware failure while performing the command or during a self-test.	Reset target after checking per instance reset retry count, Busy Retry.
ILLEGAL REQUEST	Indicates that there was an illegal parameter in the CDB or in the additional parameters supplied as data for some commands.	Fail I/O.
UNIT ATTENTION	Indicates that the removable medium may have been changed or the target has been reset.	See 7.3.6.3.
WRITE PROTECT DATA PROTECT	Indicates that a command that reads or writes the medium was attempted on a block that is protected from this operation.	Fail I/O.
VOLUME OVERFLOW	Indicates that a buffered peripheral device has reached the end-of-partition and data may remain in the buffer that has not been written to the medium.	Fail I/O.
MISCOMPARE	Indicates that the source data did not match the data read from the medium.	Fail I/O.
BLANK CHECK	Indicates that a write-once device or a sequential access device encountered blank medium or format-defined end-of-data indication while reading or a write-once device encountered a non-blank medium while writing.	Fail I/O.
ABORTED COMMAND	Indicates that the target aborted the command.	Standard Retry.
VENDOR UNIQUE	This sense key is available for reporting vendor-specific conditions.	Standard Retry.
COPY ABORTED	Indicates that a COPY, COMPARE, and COPY AND VERIFY command was aborted.	Standard Retry.
EQUAL	Indicates that a SEARCH DATA command has satisfied an equal comparison.	Standard Retry.
RESERVE	Indicates that the target is currently reserved by a different initiator.	Standard Retry.

7.3.6 Special Case Error Handling

The following error cases either have a unique implementation in the driver, or are frequently encountered and require special attention in this document.

7.3.6.1 Command Timeout Handling

The driver sets a timeout for each I/O of `sd_io_time` (see 7.3.3.4). The HBA driver uses this parameter to determine when a command is to be timed out. When a target device is no longer responding to commands, every I/O takes up to $sd_io_time * sd_retry_count$ to be failed. For the default values of these global variables **each I/O** takes from 3 to 5 minutes to fail. The failover process in a system configuration that includes volume management and failover support can take a substantial amount of time based upon the failover implementation and I/O workload at the time of the failover.

7.3.6.2 Not Ready Handling

Processing of the Not Ready SCSI status begins with a check of the I/O retry count against `un_notready_retry_count`. If the I/O has been retried more than the maximum count, it is failed. Following the retry count check, the SCSI additional sense code and SCSI additional sense code qualifier is processed as shown in the following table. The START STOP UNIT is issued in an attempt to spin up the drive.

ASC	Description	ASCQ	Description	Implementation
04h	Logical Unit Not Ready	00h	Cause Not Reportable	Reset Target, Issue START STOP UNIT, Nocheck Retry
		01h	Logical Unit Coming Ready	Nocheck Retry
		02h	Initializing CMD Required	Issue START STOP UNIT, Nocheck Retry
		03h	Manual Intervention Required	Fail I/O
		04h	Format in Progress	Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry
		05h	Rebuild in Progress	Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry
		06h	Recalculation in Progress	Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry
		07h	Operation in Progress	Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry
		08h	Long WRITE in Progress	Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry
		other		Fail I/O for removable-media devices, Issue START STOP UNIT, Nocheck Retry

ASC	Description	ASCQ	Description	Implementation
05h	No Response to Selection			Nocheck Retry
3Ah	Medium Not Present			Track media state change, Fail I/O
Other				Nocheck Retry

7.3.6.3 Unit Attention Handling

The driver implements infinite retries for Unit Attention SCSI status conditions for the majority of SCSI additional sense codes. This is a legacy implementation that some target devices are based upon. These devices may return UNIT ATTENTION for over one minute under certain conditions, which can consume many retry attempts. The driver uniquely handles the following SCSI additional sense codes.

Failure Prediction Threshold Exceeded

This event is uniquely tracked via driver error statistics and the I/O is retried using the standard implementation.

Power On, Reset, or Bus Device Reset Occurred

The driver reservation state is updated to indicate that a reservation has been lost and the I/O is retried using the Nocheck implementation.

Not Ready to Ready Change, Medium May Have Changed

When the driver gets a unit attention from a removable-media device, the device may be in a state that will take a long time to recover (e.g., from a reset). Because the event is processed in the interrupt context, the driver cannot wait for the device to come back. So the command is handed off to the `sd_media_change_task()` for deferred processing under `taskq` thread context. (Note that the command still may fail if a problem is encountered at a later time.) When a reset is issued on a CD-ROM, it takes a long time to recover. The first few attempts to read capacity and other things related to handling unit attention fail (with a ASC 04h and ASCQ 01h). In that case, the driver guarantees enough retries and limits the retries in other cases of genuine failures such as no media in drive (see 7.3.6.2).