To: T10 Technical Committee
From: Rob Elliott, HP (elliott@hp.com)
Date: 3 October 2003
Subject: T10/03-337r0 SPC-3 Persistent reservations read full status

Revision History
Revision 0 (3 October 2003) first revision

Related Documents
spc3r15 – SCSI Primary Commands – 3 revision 15

Overview
PERSISTENT RESERVE IN defines two service actions to retrieve status about the current reservations in the logical unit:

- READ KEYS - returns all keys that are registered (just the keys)
- READ RESERVATIONS - returns the key of the reservation holder and the type, scope, and scope-specific address of each reservation (there may be more than one if elements are being used)

However, what is really registered or the reservation holder is an I_T nexus, not just a key. Neither of these service actions identify the I_T nexus holding the key (or reservation).

Proposal
Add a PERSISTENT RESERVE IN service action that returns a list of all the I_T nexuses that are registered and identifies which ones are reservation holders. Initiator ports are described with Transport IDs; target ports are described with relative target port identifiers. An application can convert the relative target port identifier into a target port identifier by other means.

In case the logical unit is taking advantage of the All Target Ports feature and is not differentiating the Ts in the I_T nexus, it can return a single descriptor that covers all target ports.

Suggested Changes
5.6 Reservations
[5.6.1 Reservations overview]

[5.6.2 The Persistent Reservations management method]

[5.6.2.1 Overview of the Persistent Reservations management method]

[5.6.2.2 Preserving persistent reservations and registrations]

5.6.2.3 Finding persistent reservations and reservation keys

5.6.2.3.1 Summary of commands for finding persistent reservations and reservation keys

The application client may obtain information about the persistent reservation and the reservation keys (i.e., registrations) that are present within a device server by issuing PERSISTENT RESERVE IN commands with READ RESERVATION or READ KEYS service action.

5.6.2.3.2 Reporting reservation keys

An application client may issue a PERSISTENT RESERVE IN command with READ KEYS service action to determine if any I_T nexuses have been registered with a logical unit through any target port.

In response to a PERSISTENT RESERVE IN with READ KEYS service action the device server shall report the following:

a) The current PRgeneration value (see 6.11.3); and

b) The reservation key for every I_T nexus that is currently registered regardless of the target port through which the registration occurred.

The PRgeneration value allows the application client to verify that the configuration of the I_T nexuses registered with a logical unit has not been modified.

~~The application client may examine the reservation keys to identify relationships between I_T nexuses based on mechanisms that are outside the scope of this standard.~~ Duplicate keys shall be reported if multiple I_T nexuses are registered using the same reservation key.

If an application client uses a different reservation key for each I_T nexus the application client may use the reservation key to uniquely identify an I_T nexus.

[Editor's note: this proposal adds a mechanism to the standard, so the "outside the scope" line is no longer true. We can borrow a paragraph from 5.6.2.3.3 and use it instead (modified).]

5.6.2.3.3 Reporting persistent reservations

An application client may issue a PERSISTENT RESERVE IN command with READ RESERVATION service action to receive the persistent reservation information.

In response to a PERSISTENT RESERVE IN command with READ RESERVATION service action the device server shall report the ~~following as an uninterrupted series of actions~~current PRgeneration value (see 6.11.3) and, for every persistent reservation, the following information:

[Editor's note: the uninterrupted warning is usually on setting things, not reading. Need to account for more than one reservation in case elements are present.]

~~a) The current PRgeneration value (see 6.11.3);~~

b) The registered reservation key~~, if any,~~ associated with the I_T nexus that holds the persistent reservation;

c) The scope and type of ~~each~~ the persistent reservation~~, if any~~; and

d) The scope-specific address, if any (see 6.11.4.1).

If an application client uses a different reservation key for each I_T~~_L~~ nexus the application client may use the reservation key to associate the persistent reservation with the initiator port that holds the persistent reservation.

[Editor's note: most discussion is about I_Ts here; the _L is implicit in everything.]

This association is done using techniques that are outside the scope of this standard.

5.6.2.3.4 Reporting full status

An application client may issue a PERSISTENT RESERVE IN command with READ FULL STATUS service action to receive all information about registrations and persistent reservations.

In response to a PERSISTENT RESERVE IN with READ FULL STATUS service action the device server shall report the current PRgeneration value (see 6.11.3) and, for every I_T nexus that is currently registered, the following information:

a) The registered reservation key;

b) Whether the I_T nexus is a persistent reservation holder;

c) The scope and type of the persistent reservation, if the I_T nexus is a persistent reservation holder;

d) The scope-specific address, if any (see 6.11.4.1), if the I_T nexus is a persistent reservation holder;

e) the relative target port identifier; and

f) A TransportID identifying the initiator port.

[5.6.2.4 Registering]

[5.6.2.5 Reserving]

[5.6.2.6 Persistent reservation holder]

[**5.6.2.7 Releasing persistent reservations and removing registrations**]

6.11 PERSISTENT RESERVE IN command

6.11.1 PERSISTENT RESERVE IN command introduction

The PERSISTENT RESERVE IN command (see table 96) is used to obtain information about persistent reservations and reservation keys (i.e., registrations) that are active within a device server. This command is used in conjunction with the PERSISTENT RESERVE OUT command (see 6.12).

The PERSISTENT RESERVE IN parameter data includes a field that indicates the number of parameter data bytes available to be returned. The ALLOCATION LENGTH field in the CDB indicates how much space has been allocated for the returned parameter list. An allocation length that is not sufficient to contain the entire parameter list shall not be considered an error. If the complete list is required, the application client should send a new PERSISTENT RESERVE IN command with allocation length large enough to contain the entire list.

6.11.2 PERSISTENT RESERVE IN service actions

6.11.2.1 Summary of PERSISTENT RESERVE IN service actions

The service action codes for the PERSISTENT RESERVE IN command are defined in table 97.

Table 97 — PERSISTENT RESERVE IN service action codes

Code	Name	Description
00h	READ KEYS	Reads all registered reservation keys (i.e., registrations)
01h	READ RESERVATION	Reads the current persistent reservations
02h	REPORT CAPABILITIES	Returns capability information
03h	READ FULL STATUS	Reads complete information about all registrations and persistent reservations
03h - 1Fh	Reserved	Reserved

[Editor's note: the structure here seems odd. There are 3 short sections (6.11.2.x) on each service action, then 3 long sections (6.11.x) describing the parameter data for each service action. Shouldn't they be together?]

6.11.2.2 READ KEYS

The READ KEYS service action requests that the device server return a parameter list containing a header and a list of each currently registered I_T nexus' reservation key. If multiple I_T nexuses have registered with the same key, then that key value shall be listed multiple times, once for each such registration.

For more information on READ KEYS see 5.6.2.3.2.

6.11.2.3 READ RESERVATIONS

The READ RESERVATIONS service action requests that the device server return a parameter list containing a header and the persistent reservations, if any, that are present in the device server. Multiple persistent reservations may be returned only if element persistent reservations are present.

For more information on READ RESERVATION see 5.6.2.3.3.

6.11.2.4 REPORT CAPABILITIES

The REPORT CAPABILITIES service action requests that the device server return information on persistent reservation features.

6.11.2.4x READ FULL STATUS

The READ FULL STATUS service action requests that the device server return a parameter list describing the registration and persistent reservation status of each currently registered I_T nexus for the logical unit, and for each element, if any.

[6.11.3 PERSISTENT RESERVE IN parameter data for READ KEYS]
[6.11.4 PERSISTENT RESERVE IN parameter data for READ RESERVATION]
[6.11.5 PERSISTENT RESERVE IN parameter data for REPORT CAPABILITIES]
6.11.6 Format of PERSISTENT RESERVE IN parameter data for READ FULL STATUS

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ FULL STATUS service action is shown in table xx.

Table xx — PERSISTENT RESERVE IN parameter data for READ FULL STATUS

	7	6	5	4	3	2	1	0
0	(MSB)	PRGENERATION						
3								(LSB)
4	(MSB)	ADDITIONAL LENGTH (n - 7)						
7								(LSB)
8		Full status descriptor(s)						
n								

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN READ KEYS parameter data (see 6.11.3).

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow in reservation descriptor(s). If the allocation length specified by the PERSISTENT RESERVE IN command is not sufficient to contain the entire parameter list, then only the first portion of the list (byte 0 to the allocation length) shall be sent to the application client. The incremental remaining bytes shall be truncated, although the ADDITIONAL LENGTH field shall still contain the actual number of bytes of reservation descriptor(s) and shall not be affected by the truncation. This shall not be considered an error.

The format of the full status descriptors is defined in table yy. The device server shall return a full status descriptor covering each registered I_T nexus. The device server shall return additional full status descriptors covering each element that is reserved, if element reservations are present.

Table yy — PERSISTENT RESERVE IN full status descriptor

	7	6	5	4	3	2	1	0
0	(MSB)	RESERVATION KEY						
7								(LSB)
8	(MSB)	SCOPE-SPECIFIC ADDRESS						
11								(LSB)
12	Reserved						RESERVATION STATUS	
13	SCOPE				TYPE			
14	(MSB)	RELATIVE TARGET PORT						
17								(LSB)
18	(MSB)	ADDITIONAL DESCRIPTOR LENGTH (n - 21)						
21								(LSB)
22		TRANSPORTID						
n								

The RESERVATION KEY field is as defined in the READ RESERVATION parameter data (see 6.11.4).

The SCOPE-SPECIFIC ADDRESS field, SCOPE field, and TYPE field are as defined in the READ RESERVATION parameter data (see 6.11.4) if this I_T nexus is a reservation holder and are undefined otherwise.

The RESERVATION STATUS field is defined in Table nn.

Table nn — Reservation status

Code	Description
00b	The I_T nexus(es) described by this descriptor is registered
01b	The I_T nexus(es) described by this descriptor is a persistent reservation holder
all others	Reserved

The RELATIVE TARGET PORT field contains either:

a) a value of FFFFFFFFh if the reservation status is 01b, all target ports are registered with the same reservation key, and an all registrants type of persistent reservation is active;

b) a value of FFFFFFFFh if the reservation status is 00b and all target ports are registered with the same reservation key; or

c) the relative target port identifier (see 7.6.4.6) of the target port that is part of the I_T nexus.

The ADDITIONAL DESCRIPTOR LENGTH field contains a count of the number of bytes to follow in the descriptor (i.e., the size of the TransportID).

The TRANSPORTID field contains a TransportID (see 7.5.4) identifying the initiator port that is part of the I_T nexus.