To: INCITS Technical Committee T10
From: Kevin Butt
Date: Printed Wednesday, September 10, 2008 12:58 pm
Document: T10/08-342r1 — Reporting SPC-2 Reservation Holder

1. Revisions

1. 08-342r0 Initial revision (25 August 2008)
2. 08-342r1 Incorporated comments received from Gerry Holder (Seagate) and Ray Gilson (Symantec) (indicated by change bars)

2. Introduction

Persistent Reservations provides a method for determining which I_T nexus(es) are the reservation holder. We have found that customers have a need to know this information. We have also found that when one or more hosts in a SAN use SPC-2 RESERVE that this mehtod breaks down and customers cannot determine the reservation holder. This leads to frustration in those environments where a rogue device has taken over one or more targets (i.e, reserved them and left them reserved).

This proposal proposes to resolve this shortcomming by:

1. Adding the reporting an SPC-2 Reservation holder to the PERSISTENT RESERVE IN full status descriptor;
2. Adding a bit to the PERSISTENT RESERVE IN full status descriptor to indicate that the I_T nexus being reported is for an SPC-2 Reservation; and
3. Allowing a PERSISTENT RESERVE IN command to pass through an SPC-2 Reservation.

Key:

Gerry Holder Questions (Q)

My answers (A)

Q: The obvious complaint - why should we complicate a new feature by adding support for an obsolete feature? If we do this, will we need to "unobsolete" the legacy reservation so it is described in SPC-4? This might be ugly.

A: *We are not adding support for an obsolete feature. What we are attempted to do is to acknowledge that this obsolete feature is used in a high percentage of installations in which we deploy. Some of those applications are beyond our control. They are outside our circle of influence. Unfortunately they play with our systems - either intentionally or unintentionally. Customers resist change and in some cases are unable to update from these legacy systems to current systems. This causes problems in the SAN and there needs to be a way to DETECT this problem. This is a method of detections so the user can be told which is their malfunctioning system.*

We do not need to unobsolete the legacy reservation to do this. We already refer to it in allowing reservation interactions.

Q: 2nd obvious complaint - so to detect a single rogue initiator that still uses RESERVE, you want to require all target devices in the SAN to get updated firmware (or be replaced)? Shouldn't it be easier to update all initiators to use only persistent reservations? If you really want to update all the targets in the SAN, why not change them to reject RESERVE instead, so only persistent reservation can happen?

A: This feature is optional with an indication returned if supported. All target devices do not need to be updated, only those who have a need for this.

It is not easier to update all initiators to use persistent reserve. The majority of systems out there still use RESERVE and not Persistent Reserve. Customers are not going to update those systems easily.

Target devices cannot be updated to reject Reserve since the mojority of applicaitons and hosts still use the legacy reservations and not Persistent Reserve. At least in the tape drive market the majority of ISV's and device drivers are still using RESERVE/RELEASE. There is just now beginning to be a movement toeard Persistent Reserve.

3rd obvious complaint - Why not just do Logical Unit Reset to clear the reservation, establish a persistent reservation, and go on with real work?

A: A Logical Unit Reset is very disruptive and does not provide a means to discover which host is cuasing the problem. In some tape drives a Logical Unit Reset causes a rewind and changes the position of the medium. Also, this causes fabric events that are logged as errors and may be disruptive to many other devices (e.g., a fully populated FC Loop) In addition, there are several platforms that do not prived an API or IOCTL to perform a logical unit reset. In the customer sites where this problem has been seen it requires a person to manually find the drive that has the held reservation and a manual operation to power cycle the drive. This is not acceptable to the customer.

3. Proposal

Additions are shown in this font.

~~Deletions are shown in this font.~~

3.1.162 SPC-2 reservation : A reservation that is established with a RESERVE(6) or RESERVE(10) command as specified in SPC-2.

3.1.163 SPC-2 reservation holder: The I_T nexus for which the reservation was established with a RESERVE(6) or RESERVE(10) command as specified in SPC-2.

5.7 Reservations

5.7.1 Persistent Reservations overview

Reservations may be used to allow a device server to process commands from a selected set of I_T nexuses (i.e., combinations of initiator ports accessing target ports) and reject commands from I_T nexuses outside the selected set. The device server uniquely identifies I_T nexuses using protocol specific mechanisms.

Application clients may add or remove I_T nexuses from the selected set using reservation commands. If the application clients do not cooperate in the reservation protocol, data may be unexpectedly modified and deadlock conditions may occur.

The persistent reservations mechanism allows multiple application clients communicating through multiple I_T nexuses to preserve reservation operations across SCSI initiator device failures, which usually involve logical unit resets and involve I_T nexus losses. Persistent reservations persist across recovery actions. Persistent reservations are not reset by hard reset, logical unit reset, or I_T nexus loss.

The persistent reservation held by a failing I_T nexus may be preempted by another I_T nexus as part of its recovery process. Persistent reservations shall be retained by the device server until released, preempted, or

cleared by mechanisms specified in this standard. Optionally, persistent reservations may be retained when power to the SCSI target device is removed.

The PERSISTENT RESERVE OUT and PERSISTENT RESERVE IN commands provide the basic mechanism for dynamic contention resolution in systems with multiple initiator ports accessing a logical unit.

Before a persistent reservation may be established, the application client shall register a reservation key for each I_T nexus with the device server. Reservation keys are necessary to allow:

a) Authentication of subsequent PERSISTENT RESERVE OUT commands;
b) Identification of other I_T nexuses that are registered;
c) Identification of the reservation key(s) that have an associated persistent reservation;
d) Preemption of a persistent reservation from a failing or uncooperative I_T nexus; and
e) Multiple I_T nexuses to participate in a persistent reservation.

The reservation key provides a method for the application client to associate a protocol-independent identifier with a registered I_T nexus. The reservation key is used in the PERSISTENT RESERVE IN command to identify which I_T nexuses are registered and which I_T nexus, if any, holds the persistent reservation. The reservation key is used in the PERSISTENT RESERVE OUT command to register an I_T nexus, to verify the I_T nexus being used for the PERSISTENT RESERVE OUT command is registered, and to specify which registrations or persistent reservation to preempt.

Reservation key values may be used by application clients to identify registered I_T nexuses, using application specific methods that are outside the scope of this standard. This standard provides the ability to register no more than one reservation key per I_T nexus. Multiple initiator ports may use the same reservation key value for a logical unit accessed through the same target ports. An initiator port may use the same reservation key value for a logical unit accessed through different target ports. The logical unit shall maintain a separate reservation key for each I_T nexus, regardless of the reservation key's value.

An application client may register an I_T nexus with multiple logical units in a SCSI target device using any combination of unique or duplicate reservation keys. These rules provide the ability for an application client to preempt multiple I_T nexuses with a single PERSISTENT RESERVE OUT command, but they do not provide the ability for the application client to uniquely identify the I_T nexuses using the PERSISTENT RESERVE commands.

See table 170 in 6.14.2 for a list of PERSISTENT RESERVE OUT service actions. See table 158 in 6.13.1 for a list of PERSISTENT RESERVE IN service actions.

The scope (see 6.13.3.3) of a persistent reservation shall be the entire logical unit.

The type (see 6.13.3.4) of a persistent reservation defines the selected set of I_T nexuses for which the persistent reservation places restrictions on commands.

The details of which commands are allowed under what types of reservations are described in table 1.

In table 1 and table 2 the following key words are used:

allowed: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present should complete normally.

conflict: Commands received from I_T nexuses not holding the reservation or from I_T nexuses not registered when a registrants only or all registrants type persistent reservation is present shall not be performed and the device server shall complete the command with RESERVATION CONFLICT status.

Commands from I_T nexuses holding a reservation should complete normally. The behavior of commands from registered I_T nexuses when a registrants only or all registrants type persistent reservation is present is specified in table 1 and table 2.

A command shall be checked for reservation conflicts before the task containing that command enters the enabled task state. Once a task has entered the enabled task state, the command that comprises the task shall not be completed with RESERVATION CONFLICT status due to a subsequent reservation.

For each command, this standard or a command standard (see 3.1.27) defines the conditions that result in the command being completed with RESERVATION CONFLICT. Command standards define the conditions either in the device model or in the descriptions each of specific command.

Table 1 — SPC commands that are allowed in the presence of various reservations (part 1 of 2)

Command	Addressed logical unit has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From not registered I_T nexus	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
ACCESS CONTROL IN	Allowed	Allowed	Allowed	Allowed	Allowed
ACCESS CONTROL OUT	Allowed	Allowed	Allowed	Allowed	Allowed
CHANGE ALIASES	Conflict	Conflict	Allowed	Conflict	Conflict
EXTENDED COPY	Conflict	Conflict	Allowed	Conflict	Conflict
INQUIRY	Allowed	Allowed	Allowed	Allowed	Allowed
LOG SELECT	Conflict	Conflict	Allowed	Conflict	Conflict
LOG SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
MANAGEMENT PROTOCOL IN	Allowed	Conflict	Allowed	Allowed	Conflict
MANAGEMENT PROTOCOL OUT	Conflict	Conflict	Allowed	Conflict	Conflict
MODE SELECT(6) / MODE SELECT(10)	Conflict	Conflict	Allowed	Conflict	Conflict
MODE SENSE(6) / MODE SENSE(10)	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
PERSISTENT RESERVE IN	Allowed	Allowed	Allowed	Allowed	Allowed
PERSISTENT RESERVE OUT	*see table 2*				
READ ATTRIBUTE	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
READ BUFFER	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
READ MEDIA SERIAL NUMBER	Allowed	Allowed	Allowed	Allowed	Allowed
RECEIVE CREDENTIAL	Conflict	Conflict	Allowed	Conflict	Conflict
RECEIVE COPY RESULTS	Conflict	Conflict	Allowed	Conflict	Conflict
RECEIVE DIAGNOSTIC RESULTS	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
RELEASE(6)/ RELEASE(10)	As defined in SPC-2 [a]				
REPORT ALIASES	Allowed	Allowed	Allowed	Allowed	Allowed

Key: **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] Exceptions to the behavior of the RESERVE and RELEASE commands described in SPC-2 are defined in 5.7.3.

[b] Logical units claiming compliance with previous versions of this standard (e.g., SPC-2, SPC-3) may return RESERVATION CONFLICT in this case. Logical units may report whether certain commands are allowed in ALLOW COMMANDS field of the parameter data returned by the PERSISTENT RESERVE IN command with REPORT CAPABILITIES service action (see 6.13.4).

Table 1 — SPC commands that are allowed in the presence of various reservations (part 2 of 2)

Command	Addressed logical unit has this type of persistent reservation held by another I_T nexus				
	From any I_T nexus		From registered I_T nexus (RR all types)	From not registered I_T nexus	
	Write Excl	Excl Access		Write Excl RR	Excl Access – RR
REPORT IDENTIFYING INFORMATION	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT LUNS	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT PRIORITY	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT SUPPORTED OPERATION CODES	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
REPORT SUPPORTED TASK MANAGEMENT FUNCTIONS	Allowed[b]	Conflict	Allowed	Allowed[b]	Conflict
REPORT TARGET PORT GROUPS	Allowed	Allowed	Allowed	Allowed	Allowed
REPORT TIMESTAMP	Allowed	Allowed	Allowed	Allowed	Allowed
REQUEST SENSE	Allowed	Allowed	Allowed	Allowed	Allowed
RESERVE(6) / RESERVE(10)	As defined in SPC-2 [a]				
SECURITY PROTOCOL IN	Allowed	Conflict	Allowed	Allowed	Conflict
SECURITY PROTOCOL OUT	Conflict	Conflict	Allowed	Conflict	Conflict
SEND DIAGNOSTIC	Conflict	Conflict	Allowed	Conflict	Conflict
SET IDENTIFYING INFORMATION	Conflict	Conflict	Allowed	Conflict	Conflict
SET PRIORITY	Conflict	Conflict	Allowed	Conflict	Conflict
SET TARGET PORT GROUPS	Conflict	Conflict	Allowed	Conflict	Conflict
SET TIMESTAMP	Conflict	Conflict	Allowed	Conflict	Conflict
TEST UNIT READY	Allowed[b]	Allowed[b]	Allowed	Allowed [b]	Allowed [b]
WRITE ATTRIBUTE	Conflict	Conflict	Allowed	Conflict	Conflict
WRITE BUFFER	Conflict	Conflict	Allowed	Conflict	Conflict

Key: **Excl**=Exclusive, **RR**=Registrants Only or All Registrants, **<>** Not Equal

[a] Exceptions to the behavior of the RESERVE and RELEASE commands described in SPC-2 are defined in 5.7.3.

[b] Logical units claiming compliance with previous versions of this standard (e.g., SPC-2, SPC-3) may return RESERVATION CONFLICT in this case. Logical units may report whether certain commands are allowed in ALLOW COMMANDS field of the parameter data returned by the PERSISTENT RESERVE IN command with REPORT CAPABILITIES service action (see 6.13.4).

Table 2 — PERSISTENT RESERVE OUT service actions that are allowed in the presence of various reservations

Command Service Action	Addressed logical unit has a persistent reservation held by another I_T nexus	
	Command is from a registered I_T nexus	Command is from a not registered I_T nexus
CLEAR	Allowed	Conflict
PREEMPT	Allowed	Conflict
PREEMPT AND ABORT	Allowed	Conflict
REGISTER	Allowed	Allowed
REGISTER AND IGNORE EXISTING KEY	Allowed	Allowed
REGISTER AND MOVE	Conflict	Conflict
RELEASE	Allowed [a]	Conflict
RESERVE	Conflict	Conflict

[a] The reservation is not released (see 5.7.11.2).

The time at which a reservation is established with respect to other tasks being managed by the device server is vendor specific. Successful completion of a reservation command indicates that the new reservation is established. A reservation may apply to some or all of the tasks in the task set before the completion of the reservation command. The reservation shall apply to all tasks received by the device server after successful completion of the reservation command. Any persistent reserve service action shall be performed as a single indivisible event.

Multiple persistent reserve service actions may be present in the task set at the same time. The order of processing of such service actions is defined by the task set management requirements defined in SAM-4, but each is processed as a single indivisible command without any interleaving of actions that may be required by other reservation commands.

5.7.2 Third party persistent reservations

Editors Note 1 - KDB: No Change

5.7.3 Exceptions to SPC-2 RESERVE and RELEASE behavior

5.7.3.1 Overview

This subclause defines exceptions to the behavior of the RESERVE(6), RESERVE(10), RELEASE(6) and RELEASE(10) commands defined in SPC-2. The RESERVE(6), RESERVE(10), RELEASE(6) and RELEASE(10) commands are obsolete in this standard, except for the behavior defined in this subclause. Device servers that operate using the exceptions described in this subclause have indications that shall be set in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.13.4).

5.7.3.2 RESERVE and RELEASE behaviors

Editors Note 2 - KDB: Change all references to 5.7.3 to 5.7.3.2

~~This subclause defines exceptions to the behavior of the RESERVE and RELEASE commands defined in SPC-2. The RESERVE and RELEASE commands are obsolete in this standard, except for the behavior defined in this subclause.~~ Device servers that operate using the exceptions described in this subclause shall set the CRH bit to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.13.4).

A RELEASE(6) or RELEASE(10) command shall complete with GOOD status, but the persistent reservation shall not be released, if the command is received from:

a) An I_T nexus that is a persistent reservation holder (see 5.7.10); or
b) An I_T nexus that is registered if a registrants only or all registrants type persistent reservation is present.

A RESERVE(6) or RESERVE(10) command shall complete with GOOD status, but no reservation shall be established and the persistent reservation shall not be changed, if the command is received from:

a) An I_T nexus that is a persistent reservation holder; or
b) An I_T nexus that is registered if a registrants only or all registrants type persistent reservation is present.

In all other cases, a RESERVE(6) command, RESERVE(10) command, RELEASE(6) command, or RELEASE(10) command shall be processed as defined in SPC-2.

5.7.3.3 PERSISTENT RESERVE IN behaviors

Device servers that operate using the exceptions described in this subclause shall set the PIRH bit to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.13.4).

A PERSISTENT RESERVE IN command shall be allowed in the presence of an SPC-2 reservation.

5.7.4 Persistent reservations interactions with IKEv2-SCSI SA creation

Editors Note 3 - KDB: No Change

5.7.5 Preserving persistent reservations and registrations

Editors Note 4 - KDB: No Change

5.7.6 Finding persistent reservations and reservation keys

5.7.6.1 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 a PERSISTENT RESERVE IN command with a READ RESERVATION service action, a READ KEYS service action, or a READ FULL STATUS service action. If the PIRH bit is set to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.13.4) and an SPC-2 reservation is in effect, then the device server shall provide information about the SPC-2 reservation in the READ FULL STATUS parameter data.

5.7.6.2 Reporting reservation keys

An application client may send 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 6.13.2); 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.

Duplicate reservation 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.

5.7.6.3 Reporting the ~~persistent~~ reservation

An application client may send 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 information for the ~~persistent~~ reservation, if any:

a) The current PRgeneration value (see 6.13.2);
b) If the reservation is a persistent reservation, ~~T~~ the registered reservation key, if any, associated with the I_T nexus that holds the persistent reservation (see 5.7.10). If the persistent reservation is an all registrants type, the registered reservation key reported shall be zero; ~~and~~
c) If the reservation is a persistent reservation, ~~T~~ the scope and type of the persistent reservation, if any; and
d) If the reservation is a SPC-2 reservation, the SPC2_R bit.~~.~~

If an application client uses a different reservation key for each I_T nexus, the application client may use the reservation key to associate the persistent reservation with the I_T nexus that holds the persistent reservation. This association is done using techniques that are outside the scope of this standard.

5.7.6.4 Reporting full status

An application client may send a PERSISTENT RESERVE IN command with READ FULL STATUS service action to receive all information about registrations and the persistent reservation, if any. If the PIRH bit is set to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command (see 6.13.4), then information about an SPC-2 reservation, if any, shall be returned.

In response to a PERSISTENT RESERVE IN command with READ FULL STATUS service action the device server shall report the current PRgeneration value (see 6.13.2) and, for every I_T nexus that is currently registered or that is an SPC-2 reservation holder, the following information:

a) The registered reservation key;
b) Whether the I_T nexus is a persistent reservation holder;
c) If the I_T nexus is a persistent reservation holder, the scope and type of the persistent reservation;
d) If the PIRH bit is set to one in the parameter data returned by the REPORT CAPABILITIES service action of the PERSISTENT RESERVE IN command, whether the I_T nexus is an SPC-2 reservation holder;

e) The relative target port identifier identifying the target port of the I_T nexus; and
f) A TransportID identifying the initiator port of the I_T nexus.

Editors Note 5 - KDB: **Gerry Holder's comment:** If the PIRH bit is set to one and the SPC-2 R bit is set to one, there should be a separate paragraph defining what the reservation key field and all the other fields contain (or probably don't contain). By mixing this in with the persistent reservation fields there is an implication that all of the fields are relevant to an SPC-2 reservation. It seems like only one or two of the fields are relevant.

5.7.7 Registering

Editors Note 6 - KDB: No Change

6.13 PERSISTENT RESERVE IN command

6.13.1 PERSISTENT RESERVE IN command introduction

The PERSISTENT RESERVE IN command *(see table 3)* 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 6.14).

Table 3 — PERSISTENT RESERVE IN command

Bit / Byte	7	6	5	4	3	2	1	0
0	OPERATION CODE (5Eh)							
1	Reserved			SERVICE ACTION				
2	Reserved							
6								
7	(MSB)			ALLOCATION LENGTH				
8								(LSB)
9	CONTROL							

The ALLOCATION LENGTH field is defined in 4.3.5.6. The PERSISTENT RESERVE IN parameter data includes a length field that indicates the number of parameter data bytes available to be returned. The allocation length should be set to a value large enough to return the length field for the specified service action.

The service action codes for the PERSISTENT RESERVE IN command are defined in table 4.

Table 4 — PERSISTENT RESERVE IN service action codes

Code	Name	Description	Reference
00h	READ KEYS	Reads all registered reservation keys (i.e., registrations) as described in 5.7.6.2	6.13.2
01h	READ RESERVATION	Reads the current persistent reservations as described in 5.7.6.3	6.13.3
		Editors Note 7 - KDB: **Ray Gilson:** The new form of Read Reservation (if created), and/or the Read Full Status (if something isn't done about Read Reservation) must be mandatory for any target that reports the PIRH bit as one. It seems that the text makes all of the defined service action codes mandatory (I see "SHALL" return data for each). MANY devices do not support any service action code other than 0 and 1...	
02h	REPORT CAPABILITIES	Returns capability information	6.13.4
03h	READ FULL STATUS	Reads complete information about all registrations and the persistent reservations, if any	6.13.5
04h to 1Fh	Reserved	Reserved	

6.13.2 READ KEYS service action

Editors Note 8 - KDB: No Change

6.13.3 READ RESERVATION service action

Editors Note 9 - KDB: **Ray Gilson comment:** The Read Reservation should have a new set of parameter data for the case of a SPC-2 reservation. This could be a block of data with the additional length set to 4. The first byte following would include the SPC2_R flag. The three following bytes would be Reserved. I'm sure a Reserved bit could be found in the existing returned data block if a third form isn't desired. <<kdbutt: I think this section now accomplishes this>>

The Read Full Status doesn't report anything of particular value to an application -- target port / transport information is unknown to them. I would like to be able to use the Read Reservation command to find out if a reservation exists or not.

6.13.3.1 READ RESERVATION service action introduction

The READ RESERVATION service action requests that the device server return a parameter list containing a header and the persistent reservation, if any, that is present in the device server.

For more information on READ RESERVATION see 5.7.6.3.

6.13.3.2 Format of PERSISTENT RESERVE IN parameter data for READ RESERVATION

When no persistent reservation is held, the format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ RESERVATION service action is shown in table 5.

Table 5 — PERSISTENT RESERVE IN parameter data for READ RESERVATION with no reservation held

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (0)				(LSB)

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data (see 6.13.2).

The ADDITIONAL LENGTH field shall be set to zero, indicating that no persistent reservation is held.

When a persistent reservation is held, the format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the READ RESERVATION service action is shown in table 6.

Table 6 — PERSISTENT RESERVE IN parameter data for READ RESERVATION with reservation

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
3				PRGENERATION				(LSB)
4	(MSB)							
7				ADDITIONAL LENGTH (10h)				(LSB)
8	(MSB)							
15				RESERVATION KEY				(LSB)
16								
19				Obsolete				
20				Reserved				SPC2_R
21	SCOPE				TYPE			
22								
23				Obsolete				

The PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data.

The SPC-2 Reservation (SPC2_R) bit shall be set to one if the reservatioin is an SPC-2 reservation. The SPC2_R bit shall be set to zero if the reservation is not an SPC-2 reservation.

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow and shall be set to 16. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.5.6.

The RESERVATION KEY field shall contain the reservation key under which the persistent reservation is held (see 5.7.10).

If the SPC2_R bit is set to zero ~~T~~ the SCOPE field shall be set to LU_SCOPE (see 6.13.3.3). If the SPC2_R bit is set to zero the SCOPE field is undefined.

If the SPC2_R bit is set to zero ~~T~~ the TYPE field shall contain the persistent reservation type (see 6.13.3.4) specified in the PERSISTENT RESERVE OUT command that created the persistent reservation. If the SPC2_R bit is set to zero the TYPE field is undefined.

The obsolete fields in bytes 16 to 19, byte 22, and byte 23 were defined in a previous standard.

6.13.3.3 Persistent reservations scope

The SCOPE field *(see table 7)* shall be set to LU_SCOPE, specifying that the persistent reservation applies to the entire logical unit.

Table 7 — Persistent reservation SCOPE field

Code	Name	Description
0h	LU_SCOPE	Persistent reservation applies to the full logical unit
1h to 2h		Obsolete
3h to Fh		Reserved

The LU_SCOPE scope shall be implemented by all device servers that implement PERSISTENT RESERVE OUT.

6.13.3.4 Persistent reservations type

The TYPE field *(see table 8)* specifies the characteristics of the persistent reservation being established for all logical blocks within the logical unit. Table 44 (see 5.7.1) defines the persistent reservation types under which each command defined in this standard is allowed to be processed. Each other command standard (see 3.1.27) defines the persistent reservation types under which each command defined in that command standard is allowed to be processed.

Table 8 — Persistent reservation TYPE field (part 1 of 2)

Code	Name	Description
0h		Obsolete
1h	Write Exclusive	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for the persistent reservation holder (see 5.7.10). **Persistent Reservation Holder:** There is only one persistent reservation holder.
2h		Obsolete
3h	Exclusive Access	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for the persistent reservation holder (see 5.7.10). **Persistent Reservation Holder:** There is only one persistent reservation holder.
4h		Obsolete
5h	Write Exclusive – Registrants Only	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** There is only one persistent reservation holder (see 5.7.10).
6h	Exclusive Access – Registrants Only	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** There is only one persistent reservation holder (see 5.7.10).

Table 8 — Persistent reservation TYPE field (part 2 of 2)

Code	Name	Description
7h	Write Exclusive – All Registrants	**Access Restrictions**: Some commands (e.g., media-access write commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** Each registered I_T nexus is a persistent reservation holder (see 5.7.10).
8h	Exclusive Access – All Registrants	**Access Restrictions**: Some commands (e.g., media-access commands) are only allowed for registered I_T nexuses. **Persistent Reservation Holder:** Each registered I_T nexus is a persistent reservation holder (see 5.7.10).
9h to Fh	Reserved	

6.13.4 REPORT CAPABILITIES service action

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

The format for the parameter data provided in response to a PERSISTENT RESERVE IN command with the REPORT CAPABILITIES service action is shown in table 9.

Table 9 — PERSISTENT RESERVE IN parameter data for REPORT CAPABILITIES

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)							
1	LENGTH (0008h)							(LSB)
2	Reserved		PIRH	CRH	SIP_C	ATP_C	Reserved	PTPL_C
3	TMV	ALLOW COMMANDS			Reserved			PTPL_A
4	PERSISTENT RESERVATION TYPE MASK							
5								
6	Reserved							
7								

The LENGTH field indicates the length in bytes of the parameter data. The relationship between the LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.5.6.

A Permit Interactive Reservation Handling (PIRH) bit set to one indicates that the device server supports the exceptions to the SPC-2 RESERVE and RELEASE commands described in 5.7.3.3. A CRH bit set to zero indicates that RESERVE(6) command, RESERVE(10) command, RELEASE(6) command, and RELEASE(10) command are processed as defined in SPC-2.

A Compatible Reservation Handling (CRH) bit set to one indicates that the device server supports the exceptions to the SPC-2 RESERVE and RELEASE commands described in 5.7.3.2~~5.7.3~~. A CRH bit set to zero indicates that RESERVE(6) command, RESERVE(10) command, RELEASE(6) command, and RELEASE(10) command are processed as defined in SPC-2.

A Specify Initiator Ports Capable (SIP_C) bit set to one indicates that the device server supports the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data (see 6.14.3). An SIP_C bit set to zero indicates that the device server does not support the SPEC_I_PT bit in the PERSISTENT RESERVE OUT command parameter data.

An All Target Ports Capable (ATP_C) bit set to one indicates that the device server supports the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data. An ATP_C bit set to zero indicates that the device server does not support the ALL_TG_PT bit in the PERSISTENT RESERVE OUT command parameter data.

A Persist Through Power Loss Capable (PTPL_C) bit set to one indicates that the device server supports the persist through power loss capability (see 5.7.5) for persistent reservations and the APTPL bit in the PERSISTENT RESERVE OUT command parameter data. An PTPL_C bit set to zero indicates that the device server does not support the persist through power loss capability.

A Type Mask Valid (TMV) bit set to one indicates that the PERSISTENT RESERVATION TYPE MASK field contains a bit map indicating which persistent reservation types are supported by the device server. A TMV bit set to zero indicates that the PERSISTENT RESERVATION TYPE MASK field shall be ignored.

The ALLOW COMMANDS field *(see table 10)* indicates whether certain commands are allowed through certain types of persistent reservations.

Table 10 — ALLOW COMMANDS field

Code	Description
000b	No information is provided about whether certain commands are allowed through certain types of persistent reservations.
001b	The device server allows the TEST UNIT READY command (see table 44 in 5.7.1) through Write Exclusive and Exclusive Access persistent reservations and does not provide information about whether the following commands are allowed through Write Exclusive persistent reservations: a) the MODE SENSE, READ ATTRIBUTE, READ BUFFER, RECEIVE DIAGNOSTIC RESULTS, REPORT SUPPORTED OPERATION CODES, and REPORT SUPPORTED TASK MANAGEMENT FUNCTION commands (see table 44 in 5.7.1); and b) the READ DEFECT DATA command (see SBC-3).
010b	The device server allows the TEST UNIT READY command through Write Exclusive and Exclusive Access persistent reservations and does not allow the following commands through Write Exclusive persistent reservations: a) the MODE SENSE, READ ATTRIBUTE, READ BUFFER, RECEIVE DIAGNOSTIC RESULTS, REPORT SUPPORTED OPERATION CODES, and REPORT SUPPORTED TASK MANAGEMENT FUNCTION commands; and b) the READ DEFECT DATA command.
011b	The device server allows the TEST UNIT READY command through Write Exclusive and Exclusive Access persistent reservations and allows the following commands through Write Exclusive persistent reservations: a) the MODE SENSE, READ ATTRIBUTE, READ BUFFER, RECEIVE DIAGNOSTIC RESULTS, REPORT SUPPORTED OPERATION CODES, and REPORT SUPPORTED TASK MANAGEMENT FUNCTION commands; and b) the READ DEFECT DATA command.
100b to 111b	Reserved

A Persist Through Power Loss Activated (PTPL_A) bit set to one indicates that the persist through power loss capability is activated (see 5.7.5). A PTPL_A bit set to zero indicates that the persist through power loss capability is not activated.

The PERSISTENT RESERVATION TYPE MASK field *(see table 11)* contains a bit map that indicates the persistent reservation types that are supported by the device server.

A Write Exclusive – All Registrants (WR_EX_AR) bit set to one indicates that the device server supports the Write Exclusive – All Registrants persistent reservation type. An WR_EX_AR bit set to zero indicates that the device server does not support the Write Exclusive – All Registrants persistent reservation type.

Table 11 — Persistent Reservation Type Mask format

Bit / Byte	7	6	5	4	3	2	1	0
4	WR_EX_AR	EX_AC_RO	WR_EX_RO	Reserved	EX_AC	Reserved	WR_EX	Reserved
5	Reserved							EX_AC_AR

An Exclusive Access – Registrants Only (EX_AC_RO) bit set to one indicates that the device server supports the Exclusive Access – Registrants Only persistent reservation type. An EX_AC_RO bit set to zero indicates that the device server does not support the Exclusive Access – Registrants Only persistent reservation type.

A Write Exclusive – Registrants Only (WR_EX_RO) bit set to one indicates that the device server supports the Write Exclusive – Registrants Only persistent reservation type. An WR_EX_RO bit set to zero indicates that the device server does not support the Write Exclusive – Registrants Only persistent reservation type.

An Exclusive Access (EX_AC) bit set to one indicates that the device server supports the Exclusive Access persistent reservation type. An EX_AC bit set to zero indicates that the device server does not support the Exclusive Access persistent reservation type.

A Write Exclusive (WR_EX) bit set to one indicates that the device server supports the Write Exclusive persistent reservation type. An WR_EX bit set to zero indicates that the device server does not support the Write Exclusive persistent reservation type.

An Exclusive Access – All Registrants (EX_AC_AR) bit set to one indicates that the device server supports the Exclusive Access – All Registrants persistent reservation type. An EX_AC_AR bit set to zero indicates that the device server does not support the Exclusive Access – All Registrants persistent reservation type.

6.13.5 READ FULL STATUS service action

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.

For more information on READ FULL STATUS see 5.7.6.4.

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 12.

Table 12 — PERSISTENT RESERVE IN parameter data for READ FULL STATUS

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			PRGENERATION				
3								(LSB)
4	(MSB)			ADDITIONAL LENGTH (n-7)				
7								(LSB)
	Full status descriptors							
8				Full status descriptor [first] *(see table 13)*				
				. . .				
				Full status descriptor [last] *(see table 13)*				
n								

If the SPC2_R bit is set to zero in the first full status descriptor, then ~~T~~the PRGENERATION field shall be as defined for the PERSISTENT RESERVE IN command with READ KEYS service action parameter data (see 6.13.2). If the SPC2_R bit is set to one in the first full status descriptor, then the PRGENERATION field is undefined.

Editors Note 10 - KDB: **Ray Gilson comment:** Increment the PRgeneration for each SPC-2 Reserve command that establishes a reservation and for each SPC-2 Release command that removes a reservation. The PRgeneration is attempting to warn other Initiators that something odd has happened to their Registration/Reservation, or someone else has Registered. None of these odd things are important unless they have already Registered. The SPC-2 commands cannot "work" in this case, so a few extra increments won't cause anyone heartburn.

The ADDITIONAL LENGTH field contains a count of the number of bytes to follow in the full status descriptors. The relationship between the ADDITIONAL LENGTH field and the CDB ALLOCATION LENGTH field is defined in 4.3.5.6.

The format of the full status descriptors is shown in table 13. Each full status descriptor describes one or more registered I_T nexuses. The device server shall return persistent reservations status information for every registered I_T nexus.

Table 13 — PERSISTENT RESERVE IN full status descriptor format

Bit / Byte	7	6	5	4	3	2	1	0
0	(MSB)			RESERVATION KEY				
7								(LSB)
8				Reserved				
11								
12				Reserved		SPC2_R	ALL_TG_PT	R_HOLDER
13	SCOPE				TYPE			
14				Reserved				
17								
18	(MSB)			RELATIVE TARGET PORT IDENTIFIER				
19								(LSB)
20	(MSB)			ADDITIONAL DESCRIPTOR LENGTH (n-23)				
23								(LSB)
24				TRANSPORTID				
n								

If the SPC2_R bit is set to zero, then ~~T~~the RESERVATION KEY field contains the reservation key. If the SPC2_R bit is set to one, then the RESERVATION KEY field is undefined.

An SPC-2 Reservation (SPC2_R) bit set to one indicates that the I_T nexus described by this full status descriptor is an SPC-2 reservation holder (see 3.1.162). An SPC2_R bit set to zero indicates that the I_T nexus described by this full status descriptor is not an SPC-2 reservation holder. If the SPC2_R bit is set to one the R_HOLDER bit shall be set to zero.

A Reservation Holder (R_HOLDER) bit set to one and an SPC2_R bit set to zero indicates that all I_T nexuses described by this full status descriptor are registered and are persistent reservation holders (see 5.7.10). A R_HOLDER bit set to zero and an SPC2_R bit set to zero indicates that all I_T nexuses described by this full status descriptor are registered but are not persistent reservation holders. If the R_HOLDER bit is set to one the SPC2_R bit shall be set to zero.

An All Target Ports (ALL_TG_PT) bit set to zero indicates that this full status descriptor represents a single I_T nexus. An ALL_TG_PT bit set to one indicates that:

a) This full status descriptor represents all the I_T nexuses that are associated with both:
   A) The initiator port specified by the TRANSPORTID field; and
   B) Every target port in the SCSI target device;
b) All the I_T nexuses are registered with the same reservation key; and
c) All the I_T nexuses are either reservation holders or not reservation holders as indicated by the R_HOLDER bit.

The device server is not required to return an ALL_TG_PT bit set to one. Instead, it may return separate full status descriptors for each I_T nexus.

If the SPC2_R bit is set to one, then the ALL_TG_PT bit shall be set to zero.

If the R_HOLDER bit is set to one (i.e., if the I_T nexus described by this full status descriptor is a reservation holder), the SCOPE field and the TYPE field are as defined in the READ RESERVATION service action parameter data (see 6.13.3). If the R_HOLDER bit is set to zero or if the SPC2_R bit is set to one, the contents of the SCOPE field and the TYPE field are not defined by this standard.

If the ALL_TG_PT bit set to zero, the RELATIVE TARGET PORT IDENTIFIER field contains the relative port identifier (see 3.1.120) of the target port that is part of the I_T nexus described by this full status descriptor. If the ALL_TG_PT bit is set to one, the contents of the RELATIVE TARGET PORT IDENTIFIER field are not defined by this standard.

The ADDITIONAL DESCRIPTOR LENGTH field contains a count of the number of bytes that 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 or I_T nexuses described by this full status descriptor.