Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303-4900

March 15, 1999 T10/99-146 revision 1

To: John Lohmeyer
NCITS Technical Committee T10
From: Bob Snively
Subject: Correction in interaction of persistent and legacy reservations

1 Introduction to problem

In SPC, the standard clearly stated that the presence of a reservation caused by a RESERVE command would cause a reservation conflict with any persistent reservation command and that the presence of a persistent reserve registration would cause a reservation conflict with any RESERVE command. In both cases, the reservation conflict was caused regardless of the identity of the initiator creating the original state and the initiator attempting the conflicting command. The following text was included in SPC to specify that.

SPC, rev 11, section 7.21 describing the RESERVE command says:

> If a device server has any reservation keys registered (see 7.13.1.1) a RESERVE command shall be rejected with a RESERVATION CONFLICT status.

SPC, rev 11, section 7.17 describing the RELEASE command says:

> If a device server has any reservation keys registered (see 7.13.1.1) a RELEASE command shall be rejected with a RESERVATION CONFLICT status. Reservation conflicts shall not occur for the RELEASE(10) command, except when reservation keys are registered.

SPC, rev 11, section 7.13 describing the PERSISTENT RESERVE OUT command says:

> When a device server receives a PERSISTENT RESERVE OUT command and RESERVE(6) or RESERVE(10) logical unit or extent reservations or SMC element reservations are active (see 7.22), the command shall be rejected with a RESERVATION CONFLICT status.

SPC, rev 11, section 7.12 describing the PERSISTENT RESERVE IN command says:

> When a device server receives a PERSISTENT RESERVE IN command and RESERVE(6) or RESERVE(10) logical unit or extent reservations or SMC element reservations are active (see 7.22), the command shall be rejected with a RESERVATION CONFLICT status.

That makes the description pretty clear.

However, in the mapping to SPC-2, this critical set of interactions was not so well defined. In particular, the case where the same initiator presented both the reservation and the conflicting reservation command was not completely de-

fined. The following extracts outline all the cases I could identify in the new text.

RESERVE is rejected regardless of the initiator if a persistent reservation is present. This is described in section 5.3.2.4, pdf page 43, in the following manner. This information is partially duplicated in table 5, pdf page 37.

> If the target receives a RESERVE(10) or RESERVE(6) command when a persistent reservation exists for the logical unit then the command shall be rejected with a RESERVATION CONFLICT.

RELEASE is rejected if a persistent reserve from another initiator is present. This is described in the RELEASE(6)/RELEASE(10) row, columns B, of table 5, pdf page 37.

PERSISTENT RESERVE IN is rejected if a reservation from another initiator exists. This is described in the PERSISTENT RESERVE IN row, column A of table 5, pdf page 37.

PERSISTENT RESERVE OUT is rejected if a reservation from another initiator exists. This is described in all rows of table 6, column A on pdf page 38.

PERSISTENT RESERVE OUT other than register is rejected if the action is attempted before registration. This is described in figure 2, pdf page 45, and in the text of 5.3.2.3, pdf page 42 in the following manner.

> Any PERSISTENT RESERVE OUT command service action received from an unregistered initiator, other than the REGISTER service action, shall be rejected with a RESERVATION CONFLICT status.

In addition, section 5.3.2.4 on pdf page 42 indicates that only one persistent reservation can be created at a time.

> If the target receives a PERSISTENT RESERVE OUT command that attempts to create a persistent reservation when a persistent reservation already exists for the logical unit from an initiator other than the initiator that created the reservation, then the command shall be rejected with a RESERVATION CONFLICT status.

What has been left out of the intended set of restrictions is:

> RELEASE is not conflicting if a persistent reserve type of reservation is held by the same initiator.
>
> Neither RESERVE nor RELEASE conflicts if a persistent reserve registration is held by any initiator.
>
> Neither PERSISTENT RESERVE OUT nor PERSISTENT RESERVE IN conflicts if a reserve/release type of reservation is held by the same initiator.

■

2 Proposed correction of oversight

Several different possible changes can be installed to correct this, but the simplest and most compact would be to make the following changes in section 5.3 which introduces the reservation model.

5.3 Reservations

Reservations may be used to allow a device server to execute commands from a selected set of initiators. The device server shall reject commands from initiators outside the selected set of initiators by uniquely identifying initiators using protocol specific mechanisms.

Application clients may add or remove initiators 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 general description of reservations involves two groups of considerations;

a) the type of reservation established, and

b) the method used to establish, rescind, and manage the reservation.

There are limits on the combinations of reservation types available under some reservation management methods. See the reservations management commands descriptions for details.

The scope of a reservation shall be one of the following:

a) logical unit reservations - a logical unit reservation restricts access to the entire logical unit; and

b) element reservations - an element reservation restricts access to a specified element within a medium changer.

Reservations may be further qualified by restrictions on types of access (e.g., read, write, control). However, any restrictions based on the type of reservation are independent of the scope of the reservation. In addition, some methods of reservation management permit establishing reservations on behalf of another device in the same SCSI domain (third-party reservations).

The methods of managing reservations are identified by the commands associated with the methods. The methods of managing reservations are:

a) Reserve/Release - associated with the RESERVE(10), RELEASE(10), RESERVE(6), and RELEASE(6) commands (see 7.21, 7.16, 7.22, and 7.17, respectively); and

b) Persistent Reservations - associated with the PERSISTENT RESERVE OUT and PERSISTENT RESERVE IN commands (see 7.12 and 7.11, respectively).

The two methods are prevented from creating conflicting and undefined interactions using RESERVATION CONFLICT status in the following manner. If a logical unit has executed a PERSISTENT RESERVE OUT command with the REGISTER service ac-

tion and is still registered by any initiator, all RESERVE commands and all RELEASE commands regardless of initiator shall conflict and shall terminate with a RESERVATION CONFLICT status. If a logical unit has been reserved by any RESERVE command and is still reserved by any initiator, all PERSISTENT RESERVE IN and all PERSISTENT RESERVE OUT commands shall conflict regardless of initiator or service action.

Reservation restrictions are placed on commands as a result of access qualifiers associated with the type of reservation. The details of which commands are allowed under what types of reservations are described in table 5. For the reservation restrictions placed on commands for the Reserve/Release management method see table 5 column [A]. For the reservation restrictions placed on commands for the Persistent Reservations management method see the table 5 columns under [B].

If any element is reserved within a logical unit, that logical unit shall be considered reserved for the commands listed in table 5 and the accept/conflict information in the table shall apply.

In tables 5 and 6 the following key words are used:

allowed: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present should complete normally.

conflict: Commands issued by initiators not holding the reservation or by initiators not registered when a registrants only persistent reservation is present shall not be performed and the device server shall terminate the command with a RESERVATION CONFLICT status.

Commands from initiators holding a reservation should complete normally. The behavior of commands from registered initiators when a registrants only persistent reservation is present is specified in tables 5 and 6.

A command that does not explicitly write the medium shall be checked for reservation conflicts before the command enters the current task state for the first time. Once the command has entered the current task state, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

A command that explicitly writes the medium shall be checked for reservation conflicts before the device server modifies the medium or cache as a result of the command. Once the command has modified the medium, it shall not be terminated with a RESERVATION CONFLICT due to a subsequent reservation.

For each command, this standard or a related command standard (see 3.1.11) defines the conditions that result in RESERVATION CONFLICT. Depending on the particular command standard the conditions are defined in that standard's device model clause or in the clauses that define the specific commands. Annex B contains the RESERVATION CONFLICT information for some of the command sets.

■