FCP-2 Issues To Resolve

(T10/02-267r1)

Dave Peterson, Cisco Systems, Inc.

1. IBM INCITS No Vote

In reviewing the response to Veritas's FCP-2 Public Review comments concerning the clearing actions of link related functions, I do not believe that the proposed solution fully addresses the issues.

1) The solutions presented to the T10 only address the case of error recovery and reauthentication. It fails to address the issue of device discovery.

During FCP-2 recommended device discovery process, any new initiator on the SAN issues INQUIRY commands to all visible targets of interest. Issuing this command requires a PLOGI, because it will be the first time that the new initiator port has communicated with the target.

Many devices implement shared mode pages, so it is not possible to reset only the mode pages for the specific initiator. The result is that mode pages get reset for all initiators; thereby, crashing any backup application that is currently executing. Given that PLOGI is below the SCSI command layer, device reservations do not provide any protection against this problem.

2) It is also debatable as to whether saved mode pages would retain adequate information to restore the device. Specifically, the "Block Length" directly affects both the recording format and the SCSI command semantics on tape drives.

"Block Length" is part of the Mode Parameter Header and does not appear in any mode pages. The "Save Pages" bit in the mode select CDB is only specified to save mode "pages" which are reported as "savable". Therefore, parameters appearing outside of the mode pages won't be saved, except through some vendor-specific extension.

Suggested solution:

The problem of shared mode pages was already addressed for PRLI/PRLO through FCP-2 Rev 7, Table 4 footnote 12 and Table 6. If this same solution were applied to the columns for "Failed Discovery after LIP", "Failed Discovery after OLS" and "LOGO/PLOGI" the problem should be essentially solved without any need for treating sequential devices differently than non-sequential devices.

2. Time to wait for a response to REC (and SRR)

R_A_TOV or 2*R_A_TOV?

3. FCP-2: Lost Write Data, Not Last Frame of Sequence, Unacknowledged classes.

> The statements refer to the actions to be taken by the initiator.
> But, I don't know how FCP_DATA could be returned for a write operation
> either.
> The note is not correct and should be removed. I think the intent was to
> specify the behavior for out-of-order class 3.
>
> > Since the last frame of the sequence is assumed to have made it to the
> > target (lost write data iu was not the last frame of sequence), the
> > target has the sequence initiative and can complete the exchange with a
> > FCP_RSP indicating a CHECK CONDITION status with appropriate sk/asc/ascq
> > (as per Section 9.4.1). In this case, is the initiator expected to look
> > for CHECK CONDITION command completions of certain sk/asc/ascq
> > combinations and initiate a SLER operation on seeing those ? What
> > specific sk/asc/ascq combinations should the initiator look for while
> > parsing command completions to initiate SLER ?
> >
> > If this is not done, a write underrun condition as described above
> > cannot be recovered from, since SLER will not kick in and the
> > ULP/application will see the error.

> For this case:
>
> The target may or may not have sequence initiative
> (i.e., seq=1, seq_cnt=1 may not be the last frame of the sequence).
>
> For in-order delivery the target should have detected a Sequence error.
>
> Per clause 12.3.5 Additional error recovery by target:
> For unacknowledged classes of service, the target shall not attempt recovery
> for Sequence errors. The target shall depend on initiator time-outs for recovery.
>
> If SLER is enabled, the target should not return check condition.
>
> And we could use some words stating this in the doc...dap

4. RE: [Fwd: FCP-2: Lost FCP_CMND, Unacknowledged classes.]

> > Section 8.2 states :
> >
> > "If the destination FCP_Port of the REC request determines that the

> > originator S_ID, OX_ID, RX_ID or task retry id are inconsistent, it
> > shall respond with a FCP_RJT with a rsn_code of "unable to perform
> > command request" and rsn_expln of "invalid OXID-RXID combination".
> >
> > Annex C Fig C.2 states :
> >
> > "The LS_RJT (Logical Error, Invalid OXID-RXID combination) for the REC
> > indicates that the exchange is unknown."
> >
> > The 2 quoted sections above are inconsistent with the reason code of the
> > FCP_RJT to be used. From the target's perspective, when it receives a
> > REC with an OXID-RXID combination for which it has no exchange state,
> > both the above sections of FCP-2 hold good.
> >
> > Which is the reason code to be returned in this case ?
> >

Per the current definition in FC-FS (and FCP-2r7a):
‘00000011’ b Logical error
The request identified by the Command code is invalid or logically inconsistent for the conditions present.

‘00001001’ b Unable to perform command request
The Recipient of a Link Service command is unable to perform the request at this time.

Logical error should be the correct response in this case and is also specified response to an RRQ when the RX_ID, other than FFFFh, is unknown to the target.

Unable to perform command request is typically returned when the link service recipient has some sort of resource issue, not logged in, etc...

But I wouldn't be surprised if there is (other) inconsistent usage regarding Logical Error and Unable to Perform Command Request. More guidance surrounding FC "reason code" usage would be beneficial in FC-FS (at minimum).

Need to see what was actually implemented and then do the right thing.

> > Issue 2
> > =======
> > How should the initiator differentiate b/n a FCP_RJT from a target due
> > to a lost FCP_CMD (the scenario described in Annexe C Fig C.2) and the
> > case where the target has discarded exchange state due to the expiration
> > of RR_TOV after sending FCP_RSP.
> >
> > In both the above cases, our interpretation of FCP-2 is that the
> > initiator will see a FCP_RJT response to the REC with :
> > rsn_code = "Logical Error" or "Unable to perform command request"

> > rsn_expln = "Invalid OXID-RXID combination"
> >
> > In this case, the initiator cannot apply the same error recovery for the
> > 2 cases. In the lost FCP_CMND case, the initiator may safely re-issue
> > the command. The latter case could occur in the following manner :
> >
> > - Initiator issues a command which does not involve data xfer.
> > - Target sends FCP_RSP, FCP_RSP is lost.
> > - Initiator REC_TOV timer pops and initiators sends REC.
> > - REC times out after RA_TOVels (which is > RR_TOV, for fabric)
> > - Initiator aborts REC and issues another REC
> > - Target sends FCP_RJT response since it has discarded the exchange
> > state.
> >
> > In the above case, the initiator MUST NOT re-issue the FCP_CMND, since
> > this can potentially cause a data corruption with tape devices. (ex :
> > re-issuing a scsi command like SPACE, WRITE FILEMARKS when they had
> > previously been executed successfully can cause tape data corruption.)
> >
> > Can someone clarify on how FCP-2 differentiates these 2 cases ? Without
> > the ability to differentiate between these 2 cases, the use of SLER in a
> > lost FCP_CMND scenario can result in potential data corruption with tape
> > devices.
> >

Appears change will be required here to make this stuff work. Some options:

A. request an FCP_CONF on every exchange - not too attractive for non-tagged command operation (i.e., will affect performance).

B. use CRN and add more text regarding target behavior when a duplicate CRN is received - problem with this is that application clients today do not yet generate a CRN to use. I've always contended that (for the FC realm) the CRN should be a transport-based entity (i.e., the FCP driver should be generating the CRN), given that the SCSI protocol provides no built-in ordering mechanism:(I lost that battle and thus lobbied for a hook to provide a CRN in SAM-2. But, recent discussion of CRN as related to SAM-2 may cause the CRN hook to be removed, effectively forcing CRN back into the FCP driver. The fate of CRN is TBD, until at least next week:)

C. modify/review the pertinent error detection and recovery timers - this needs to be done anyway (per another issue that popped up regarding the time to wait for an REC response). The value of RR_TOV is plain wrong (e.g., it does not take into account the R_A_TOV/2*R_A_TOV wait time) and is (on the verge?) of being over-loaded.

This issue needs to be worked out. FCP-2 implementors feel free to speak up.

> > Issue 3
> > =======

> > Section 12.5.2 states that if a REC response is not received within
> > RA_TOV(els), the initiator shall abort the REC and send another REC in a
> > new exchange.
> >
> > Since the initiator detects the REC timeout only after RA_TOV (or 2 *
> > RA_TOV, as per proposed change in FCP-3) and this time value is larger
> > than RR_TOV, the target would have discarded exchange information after
> > RR_TOV.
> >
> > Hence, what is the point in retrying the REC ? It only exposes the
> > initiator to the issue described under "Issue 2".
> >
> > Any clarifications would be appreciated.
>
>

Again, the timer values needs to be reviewed...dap

5. Task Retry Identification functionality

> We have a question regarding the Task Retry Identification functionality
> described in Section 4.6.
>
> The re-use of OX_IDs within RR_TOV of their last use exposes the
> initiator to this condition described in Section 4.6.
>
> Since it is the initiator's OX_ID generation model which can determine
> the exposure to this problem, the Task Retry Identification
> functionality must be a "mandatory to implement, optional to use"
> feature and it should be left to the initiator's discretion on whether
> this should be enabled.
>
> Can someone clarify why this is not a mandatory feature of FCP-2 ?
> Without this feature, there is a risk of exposure to data corruption and
> FCP-2 sequence level error recovery (SLER) cannot be used in a reliable
> manner.
>
> Can the FCP-2 target implementors on this list comment on whether they
> support Task Retry Identification ? What is the extent of support for
> this feature among FCP-2 target implementations ?

6. 3rd party device identifier

5.2 SCSI third-party device identifier for the Fibre Channel protocol

The SCSI RESERVE commands that use the 64-bit THIRD-PARTY DEVICE ID defined by SPC-2 shall use the FCP third-party device id format defined in table 7 to identify the specified third-party target.

FCP_PORT IDENTIFIER:
The FCP_PORT IDENTIFIER field defines the address identifier of the target that shall be used by the target for thirdparty addressing.

Table 7 - FCP third-party device id format

Byte	
0	RESERVED
1–3	(MSB) FCP_PORT IDENTIFIER (LSB)
4–7	RESERVED

dap: should use the worldwide port name as the identifier.

7. FC-4 Link Service vs Extended Link Service for REC and SRR

Some implementations of early revisions of FCP-2 (See [14]) may have used the value hex '13' for REC (Read Exchange Concise) and hex '14' for SRR (Sequence Retransmission Request). These codes are made vendor specific in this standard to avoid conflicts with such noncompliant implementations. See FCP-2 for the standard implementation of REC and SRR as FC-4 Link Services. Noncompliant implementations may be identified by attempting to perform REC first as an FC-4 LS and then as an ELS. A compliant implementation will succeed on the FC-4 LS, obviating the need for the ELS. A noncompliant implementation will fail the FC-4 LS and succeed on the ELS.