Class 3 Error Detection and Recovery for Sequential and Random Access Devices

Preliminary ANSI X3T11 Working Document 97-189R3

Scope

Problems exist in PLDA in detecting and correcting error conditions on sequential access devices (tapes). The basic causes of these problems are due to the lack of a guaranteed delivery protocol and the implicit state information intrinsic to sequential access devices. More specifically, lost frames in FCP can result in FC information units being lost. Clearly this leads to compromised customer data. If one relies solely on upper layer protocol (ULP) timeouts to detect and correct these errors, then lengthy recovery is a reality. For a variety of reasons, ULP recovery is lengthy, including an inability to detect errors quickly at the Fibre Channel level, the effort required to implement recovery mechanisms, and the extended time at the operating system (OS) level left open to detect and recover from error conditions. Therefore, a detection mechanism that discovers Fibre Channel errors in a timely manner and recovers before ULP intervention is desired. The characteristics of such a mechanism are described below in the Requirements section. Further analysis of the problem is given, then a general solution is described.

Draft Release Notes : Document number 97-189Rn

97-189R0 - Presented at the May 1997 X3T10 meeting.

97-189R1 - Presented at the June 1997 X3T11 meeting after review and update from the FCL error recovery SWIG

97-189R2 - Released for review after additional FCL conference calls and the X3T11 meeting in June

97-189R3 - Presented at the July X3T10 meeting with agreements reached during the 7/8 FCL error recovery SWIG conference call

Requirements

An ideal solution will incorporate the following characteristics:

Provide the ability to recover from lost frames in FCP for sequential access devices
Provide the ability to recover from lost frames in FCP for random access devices without loss of performance - e.g. Free resources rapidly in the drives
Interoperability with block and sequential access devices
No or minimal changes to FC-PH and PLDA
No additional protocol overhead for normal operation
Can be implemented with existing silicon
Don’t turn fiber transport errors into tape drive recovery
Optimize for single sequence errors
Don’t add inefficiencies for multiple-sequence errors
Utilize Relative Offset for retransmission of READ or WRITE data

Problem Analysis

On stream and media changer devices there are two classes of commands for which it is critical to know whether the command was accepted by the target, and then whether successful completion of the command occurred.

The first class, unique to these devices, are those that alter the media state or content in a way that simply re-executing the command will not recover the error. These include read/write/position/write filemarks (the tape is repositioned past the referenced block(s) or files only if the operation started; how far the operation continued is critical to proper recovery) and move medium/load/unload medium (which may have actually changed the medium in the target). Unfortunately, these comprise most of the commands issued during normal operation of the subsystem.

The second class, which is not unique to these devices, are those in which information is lost if it is presumed sent by the target, but not received by the initiator. These commands include request sense and read/reset log. Loss of sense data also may affect error recovery from failed commands of the aforementioned media move/change class, but it may also affect proper error recovery for cached/RAID disk controllers as well.

On a parallel SCSI bus, the host adapter has positive confirmation that the target accepted the command by the fact that the target requested all bytes of the CDB and continued to the next phase without a Restore Pointers message. Such confirmation is only implicit in a serial protocol by receipt of a response message, such as Transfer_Ready or Response. In cases of some commands, this implicit confirmation may require a lengthy period of time, during which mechanical movement requiring several multiples of E_D_TOV occurs (in FLA environments, R_A_TOV may be the appropriate value). Similarly, the target has positive confirmation that the host has accepted sense or log data immediately upon completion of the data and status phases; this data may now be reset. In a serial environment, this is only implicit by receipt of the next command. Note that a change to the target to only clear sense/log data on receipt of a command other than request sense or read/reset log would eliminate this problem.

In summary, the errors that are of concern are where FCP information units are lost in transit between an FCP initiator and target. The cause for such loss is not specific, but is assumed to be cases where a link level connection is maintained between the target and initiator, and some number of FCP IU's are dropped. Other cases are either handled by PLDA through existing methods, or may be generally classified as unrecoverable and treated in a fashion similar to a SCSI bus reset.

In order to meet the defined requirements, any proposed solution must enable the initiator to make the following determinations:

An error condition occurred (an FCP IU is expected and not received, or not responded to)
If FCP_CMND, was it received by the target
If FCP_DATA, was it received or sent by target
If FCP XFER_RDY or FCP_RSP, was it sent by target

Note that the solution must work in a Class 3 environment, preferably with no change to existing hardware.

Tools For Solution

The tools prescribed in FC-PH for FC-2 recovery are the Read Exchange Status (RES), and Read Sequence Status (RSS) Extended Link Services, and the Abort Sequence (ABTS) Basic Link Service.

We have identified several functions providing some of the needed functionality, these are listed below, along with some deficiencies we have identified in each of the existing mechanisms. We are proposing additional ELS functions to provide the required functionality.

RES is an appropriate tool for the host adapter to use; its function is to inquire of the status of an operation during and for some period of time after its life. Unfortunately, in several of the cases of interest, the RX_ID is unknown to the exchange initiator. In these cases, the initiator must use an RX_ID of 0xFFFF, which, combined with the FC_PH wording that "...the Responder destination N_PORT would use RX_ID and ignore the OX_ID", means that if the Responder had not received the command frame, the RES would be rejected, and if the Responder had received the command and sent an FCP_RSP response frame, the RES would be rejected, in both cases with the same reason code; only in the case where the command was in process but no FCP_RSP response frame had been sent by the Responder would a useful response be sent. Real implementations appear to search for the S_ID - OX_ID pair when the RX_ID is set to 0xFFFF in the RES request, and this behavior needs to become required.

Further, even if this change is implemented, in the case of a non-transfer command, it is impossible to detect the difference between a command that was never received and a command whose response was lost unless the target retains ESB information for a period of RA_TOV after the exchange is closed.

Further clarification of the text in the standard (FC-PH) is required, and requirements specified in profile documents.

In view of these difficulties, we are proposing the addition of the Read Exchange Concise (RES), a new ELS which returns all of the required information, without including superfluous information at a frame level. In light of this, this proposed solution assumes that the RES becomes part of the standard. RES can be used instead of RES given that FC-PH is changed to reflect the tightened description required to make the RES completely useful.

Similar arguments apply to the use of the RSS, though the wording of the applicable section uses the word "may" rather than "would".

ABTS, while recommended in FC-PH for use in polling for sequence delivery, is always interpreted as an abort of the exchange in FC-PLDA, and is therefore not useful for this purpose.

Additionally, there needs to be a mechanism for requesting retransmission of sequences that were not received at the destination. We are proposing the addition of the Sequence Retransmission Request (RR), a new ELS which provides information to the sequence initiator about which sequences were not received by the sequence recipient.

Proposed Solution

A method is proposed where the initiator determines the state of an exchange and initiates appropriate recovery. A timer is used in conjunction with internal driver state information to determine if a target response is overdue, indicating that packet information may have been lost. The initiator will then request exchange state information from the target from which it can be determined if corrective action is necessary. The initiator can then resend information, request that the target resend information , or provide early indication to the ULP that an error has occurred.

The timer is based on the maximum frame propagation delivery time through the fabric. This is significantly less than typical ULP time out values, providing the capability to detect and correct errors before ULP actions take effect. The suggested time out for FLA environments is twice R_A_TOV (currently 2 seconds).

The suggested method of determining target state is by using a combination of the RES extended link service, one word the target keeps for each initiator, and an ability for the initiator to request that the target resend data at a given relative offset. For WRITEs, the target will resend FCP_XFR_RDY after lost data is detected. RES does need to be tightened in its FC-PH description, and this will be described later.RES.

Details of the recovery mechanism are as follows:

Rules:

1) Initiators shall use monotonically increasing OX_IDs inside the (S_ID, D_ID) pair.

2) Targets shall keep any bad status, sense data, etc. until positive confirmation has occured between the host and target. Freeing of resources for good status and sense data can be done immediately by the target.

3) Targets shall retain the last OX_ID for the last monotonically increasing OX_ID successfully completed with the transmission of valid FCP_RSP for the given (S_ID, D_ID) pair. This can be kept as a WORD where the target keeps its login parameters. This is a nominal requirement and it has a fixed known size, e.g. = 4 bytes * # of initiators supported.

Flow of Events:

After an agreed upon timeout, (either E_D_TOV as the first timeout then 2 * R_A_TOV thereafter, or 2 * R_A_TOV for all) when no reply sequence is received for the FCP_CMND_IU:

Issue RES for the exchange containing the FCP_CMND. The RES is issued in a new exchange. If there is no ACC or LS_RJT response to the RES within 2*R_A_TOV, resend the RES. The RES shall (optionally?) be retried once per 2*R_A_TOV until successful completion of RES or the exchange, or until the ULP timer has expired and ULP intervention has occured.

If the response is an LS_RJT, with a reason code indicating that the function is not supported, as is required in PLDA for block devices, treat the target as a disk or other device not supporting this proposal and allow normal ULP recovery to occur.

If the FCP_CMND was not received by the target (i.e., the initiator receives an LS_RJT for the RES, with a reason code indicating that the OX_ID is unknown), send the QUERY LAST OX_ID ELS.

If the ACC for an RES indicates that the FCP_CMND was received by the target, and that no reply sequence has been sent, the command is in process and no recovery is needed at this time. At intervals of 2*R_A_TOV the RES may optionally be retransmitted to insure proper operation of the exchange. This is to ensure that no reply sequences have been lost.

If the ACC for an RES indicates that an FCP_XFER_RDY was sent by the target, but not received by the initiator, issue an RR Extended Link Service (see below for details) frame to request sequence retransmission. The target retransmits the FCP_XFER_RDY, with F_CTL bit 9 set, indicating that this is a retransmitted frame. When the FCP_XFER_RDY is successfully received, the data is sent, and the operation continues normally. No error is reported to the ULP, though the error counters in the LESB should be updated. If the RR receives a LS_RJT, perform sequence/exchange error recovery as documented in PLDA section 9.1, 9.3.

If an ACC for an RES indicates that an FCP_DATA sequence was sent by the target, but not successfully received by the initiator, issue an RR Extended Link Service frame to request retransmission of the data at the given relative offset that was not successfully received. The target retransmits the FCP_DATA, with F_CTL bit 9 set in each corresponding data frame, indicating that these are retransmitted frames. The received data is delivered to the ULP, and no error is reported. Note that the sequence that was partially received in error is not delivered to the ULP. If the target responds to the RR with an LS_RJT and a reason code indicating that the function could not be performed, the target shall present an FCP_RSP IU with an appropriate error status (e.g., Sense key 4, ASC/ASQ of 48/00 (initiator detected error)).

If an LS_RJT occurs to an RES for a lost FCP_RSP sequence sent by the target, but not received by the initiator, issue the QUERY LAST OX_ID ELS. The target transmits the OX_ID of the exchange it last sent a valid FCP_RSP. If the OX_ID is for the exchange in question, then valid status can be inferred at the initiator because of the OX_ID being reported by the target. See Rules above.

If the ACC for an RES indicates that an FCP_DATA sequence was sent by the initiator, but not successfully received by the target, the initiator can do one of two things. The first option is, send an RSI Extended Link Service to request sequence initiative. As documented in PLDA Sec. 9.2, the target discards the sequence in error, but does not initiate any recovery action. When the ACC is received for the RSI, the data sequence is retransmitted by the initiator with F_CTL bit 9 set in each frame, indicating that this is a retransmitted frame. The operation should complete with no error indication to the ULP.

It is the responsibility of the initiator to determine the appropriate action (retry, allow ULP time out, or return status to ULP) required based on the information determined by RES and other internal state. As described in PLDA, the target does not initiate recovery action.

Note that link recovery should be treated as the equivalent of a bus reset. All open exchanges will be terminated and a unit attention condition shall be generated.

RR Extended Link Service

The RR (Resend Request) Extended Link Service sequence follows the rules for extended link services as defined in FC-PH Rev 4.3, Section 23.1. A new Link Service command code in R_CTL needs to be added to FC_PH. The next available value is 0001 0011b.

In the event that the target cannot accept this request, the target shall present a check condition as if it had not responded to an Initiator Detected Error with a Restore Pointers message (i.e., Sense Key = 4, ASC/ASQ = 48/00). The target shall not reject requests for retransmission of FCP_XFER_RDY frames unless the RR is not supported.

The RR payload and reject codes are defined below. The Accept does not require a payload. The direction flag indicates to the target that the initiator is requesting sequence data transfer to (0) or from (1) the target. All other fields are as defined in FC-PH.

Item	Size Bytes
SEQ_ID	1
Direction	1
OX_ID	2
RX_ID	2
Low Relative Offset	2
High Relative Offset	2

RR Payload

Encoded Value	**LS_RJT Reason code explanation**
0x00052A00	Can’t resend requested information
Reserved	

RR LS_RJT Reason Codes

Read Exchange Status (RES)

The RES Extended Link Service requests an N_Port to return information on completed sequences for the FX_ID originated by the S_ID specified in the Payload of the request Sequence. The specification of OX_ID and RX_ID may be useful or required information for the destination N_Port to locate the status information requested. A Responder destination N_Port would use the RX_ID and ignore the OX_ID, unless the RX_ID was undetermined (i.e., RX_ID = 0xFFFF). An Originator N_Port would use the OX_ID and ignore the RX_ID. This function provides the N_Port transmitting the request with information regarding the current status of the Exchange specified.

If the destination N_Port of the RES request determines that the SEQ_ID, Originator S_ID , OX_ID, or RX_ID are inconsistent, then it shall reply with an LS_RJT Sequence with a reason code that it is unable to perform the command request.

Protocol:

Read Exchange Concise request Sequence
Accept (ACC) reply Sequence

Format: FT_1

Addressing:

The S_ID field designates the source N_Port requesting the Exchange information. The D_ID field designates the destination N_Port to which the request is being made.

Payload:

The format of the Payload is shown in the following table. The Payload shall include an Association Header for the Exchange if the destination N_Port requires X_ID reassignment.

RES Payload

Item	Size -Bytes
Hex ‘13000000’	4
Reserved	1
Originator S_ID	3
OX_ID	2
RX_ID	2
Association Header (optionally required)	32

Reply Link Service Sequence

Service Reject (LS_RJT)

Signifies rejection of the RES command.

ACC

Signifies that the N_Port has transmitted the requested data.

- Accept payload:

- The format of the Accept Payload is shown in the table below. The format of the Concise Exchange Status is specified in below.

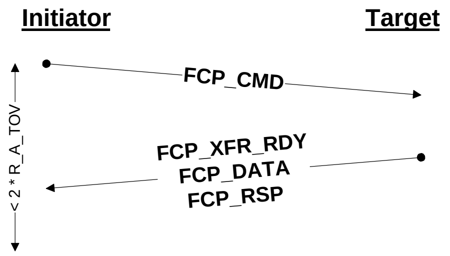
Note that for a sequence to be reported as received, the entire sequence must have been successfully received. For a sequence to be reported as transmitted, the entire sequence must have been successfully transmitted.

RES Accept Payload	
Item	Size -Bytes
Hex ‘02000000’	4
Concise Exchange Status (see 24.8.xx)	N
Association Header (optionally required)	32

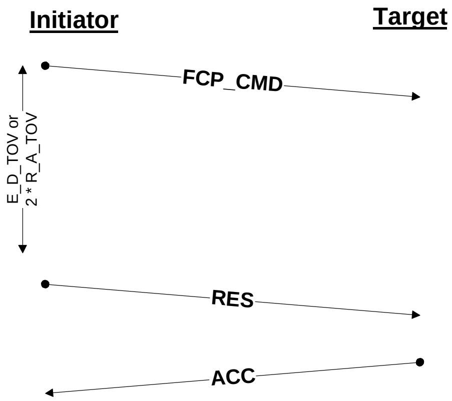
Concise Exchange Status	
Item	Size -Bytes
OX_ID	2
RX_ID	2
Originator Address Identifier (High order byte – reserved)	4
Responder Address Identifier (High order byte – reserved)	4
E_STAT	4
Number of sequences received (m)	4
Number of sequences transmitted (n)	4
R_SEQ_ID 0	1
:	:
R_SEQ_ID m-1	1
X_SEQ_ID 0	1
:	:
X_SEQ_ID n-1	1

Class 3 Operation for Tape Devices on FC-AL using FCP

Operational Case



Lengthy Command Case



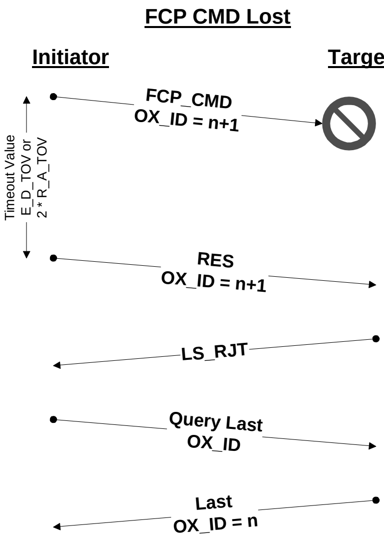
LS_RJT Codes for RES command:

- 0x00051700 - Invalid Exchange
- 0x00052A00 - Cannot provide Sequence Information
- 0x000B0000 - Don't Support Command

LS_RJT Codes for RR command:

- 0x00052A00 - Cannot provide Sequence Information

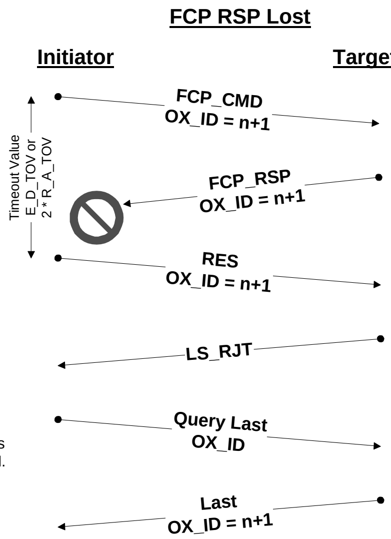
Initiator understands that the Target received the FCP_CMD



The target remembers the last OX_ID that was successfully completed. e.g. the last exchange that the target sent FCP_RSP back to the initiator

At this point the Initiator knows that the FCP_CMD was not received at the target. It is up to the initiator to invoke the appropriate error recovery mechanism. It is believed that if the FCP_CMD was sent over 2*R_A_TOV ago that it has been lost or discarded and will never return. Therefore, the FCP_CMD can be sent again.

NOTE: RES shall be used first to discover whether the FCP_CMD was actually received at the target. If the exchange is not open, then utilize the Query OX_ID ELS to discover if the exchange has completed. If it has not completed then the FCP_CMD never made it, otherwise, FCP_RSP was sent and was lost.



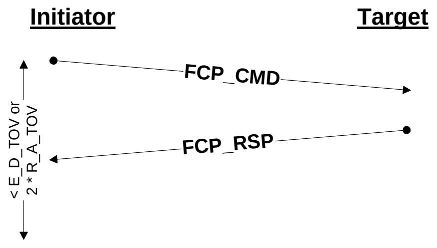
The target remembers the last OX_ID that was successfully completed. e.g. the last exchange that the target sent FCP_RSP back to the initiator

At this point the Initiator knows that the FCP_CMD was received at the target and that the FCP_RSP was sent and the status was good. It is up to the initiator to continue operation or invoke an error recovery mechanism if required (this is not necessary).

As a rule of thumb, the target shall always keep bad status until it is confirmed that the initiator has received that bad status.

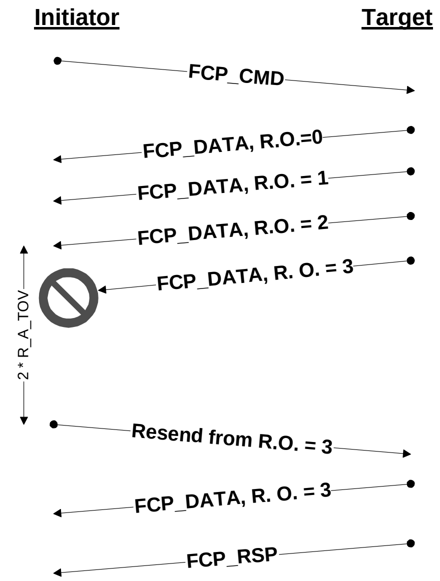
More on disks in the next revision. At first blush though, it appears that disks can just ignore or utilize this error recovery mechanism. Further work needs to be detailed on out of order. The author will produce this in Revision 4 along with comments received while presenting Revision 3.

Relative Offset Recovery - Task Management



No need for Relative Offset Recovery mechanism for task management functions

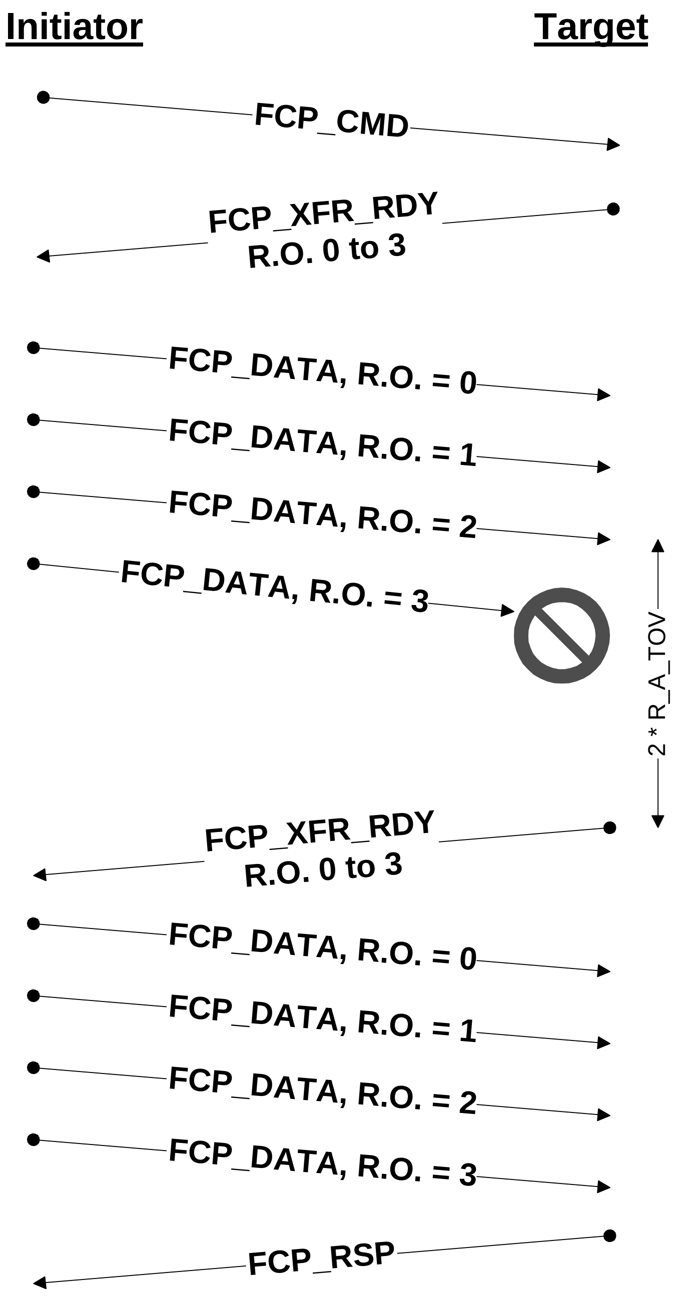
Relative Offset Recovery - READ



Resend from R.O. = x to be defined later

NOTE: Utilizing R.O. retransmission is exactly like the recovery mechanism that TCP utilizes today in the TCP/IP protocol stack.

Relative Offset Recovery - WRITE



The target could ask for the entire FCP_XFR_RDY chunk, or be smarter and ask for just R.O. 3