iSCSI Consortium
Full Feature Phase Test Suite
For iSCSI Initiators

Version 0.1

Last Update: July 3, 2003

iSCSI Consortium
InterOperability Laboratory
Research Computing Center
University of New Hampshire
http://www.iol.unh.edu

121 Technology Drive Suite 2
Durham, NH 03824-3525
Phone: (603) 862-1908
Fax: (603) 862-4181

MODIFICATION RECORD

1. Currently on Version 0.1. Version 1.0 is awaiting publication of iSCSI RFC

ACKNOWLEDGMENTS

The University of New Hampshire would like to acknowledge the efforts of the following individuals in the development of this test suite.

David Woolf University of New Hampshire

INTRODUCTION

Overview

The University of New Hampshire's InterOperability Laboratory (IOL) is an institution designed to improve the interoperability of standards based products by providing an environment where a product can be tested against other implementations of a standard. This suite of tests has been developed to help implementers evaluate the functioning of their iSCSI products. The tests do not determine if a product conforms to the iSCSI draft standard, nor are they purely interoperability tests. Rather, they provide one method to isolate problems within an iSCSI device. Successful completion of all tests contained in this suite does not guarantee that the tested device will operate with other iSCSI devices. However, combined with satisfactory operation in the IOL's semi-production environment, these tests provide a reasonable level of confidence that the Device Under Test (DUT) will function well in most multivendor iSCSI environments.

Organization of Tests

The tests contained in this document are organized to simplify the identification of information related to a test and to facilitate in the actual testing process. Each test contains an identification section that describes the test and provides cross reference information. The detailed section discusses the background information and specifies how the test is to be performed. Tests are grouped in order to reduce setup time in the lab environment. Each test contains the following information:

Test Label

The Label associated with each test is a title that is used to refer to the test. The attached number is an internal reference number dealing with an internal reference to the test.

Purpose

The purpose is a short statement describing what the test attempts to achieve. The test is written at the functional level.

References

The references section lists cross references to the iSCSI draft standard and other documentation that might be helpful in understanding and evaluating the test and results.

Resource Requirements

The requirements section specifies the software, hardware, and test equipment that will be needed to perform the test. The items contained in this section are special test devices,

software that must reside on the DUT, or other facilities which may not be available on all devices.

Last Modification
This specifies the date of the last modification to this test.

Discussion
The discussion covers the assumptions made in the design or implementation of the test as well as known limitations. Other items specific to the test are covered here.

Test Setup
The setup section describes in detail the configuration of the test environment and includes a block diagram for clarification as well as information such as the interconnection of devices, what monitoring equipment should capture, what the generation equipment should send, and any other configuration information vital to carrying out the test. Small changes in the configuration should be included in the test procedure.

Procedure
The procedure section of the test description contains the step-by-step instructions for carrying out the test. It provides a cookbook approach to testing, and will often be interspersed with observable results.

Observable Results
The observable results section lists observables that can be examined by the tester to verify that the DUT is operating properly. When multiple values are possible for an observable, this section provides a short discussion on how to interpret them. Note that complete delineation between the observables in the **Procedure** and **Observable Results** is virtually impossible. As such a careful note should be made of the requirements in both sections. In certain cases, it may be necessary to modify certain steps in the **Procedure** section while doing the actual tests so as to be able to perform the tests. In such cases, the modifications will be noted in the summary report.

Possible Problems
This section provides some clues to look for if the test does not yield the expected results.

REFERENCES

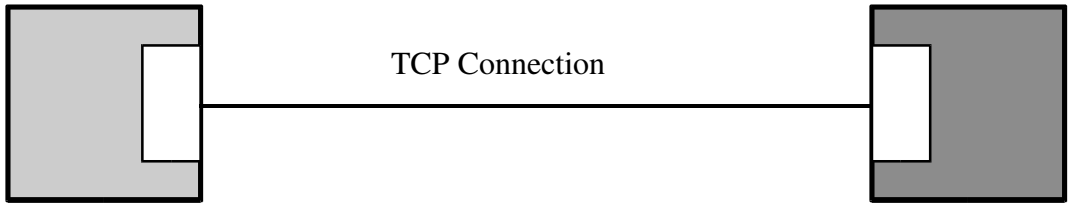
The following documents are referenced in this text:

IETF IPS Working Group iSCSI draft 20

TEST SETUPS

The following test setups are used in this test suite:

Test Setup 1:



Testing Station/ Monitor DUT

TABLE OF CONTENTS

Test #1.1: Command Numbering

Purpose: To verify that an initiator performs command numbering properly.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:12:15 2003

Discussion: Command numbering starts with the first login request on the first connection of a session (the leading login on the leading connection) and command numbers are incremented by 1 for every non-immediate command issued afterwards.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command.

Observable Results:
· Verify that all commands issued by the DUT follow the rules for Command Numbering. The CmdSN field should be incremented by 1 for every NonImmediate Command or Request, starting with the first Login Request.

Possible Problems: None.

Test #2.1: Immediate Delivery

Purpose: To verify that a device transmitting a command marked for immediate delivery does so properly

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Thu Jun 19 16:21:30 2003

Discussion: Commands meant for immediate delivery are marked with an immediate delivery flag; they MUST also carry the current CmdSN. CmdSN does not advance after a command marked for immediate delivery is sent.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Perform a standard login and proceed to the Full Feature Phase.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for a non task management command marked for Immediate Delivery. The DUT may choose to set a regular READ or WRITE command for Immediate Delivery.

Observable Results:
· Verify that the command sent by the DUT intended for Immediate Delivery is marked with the Immediate Delivery flag (I bit = 1) and that current CmdSN is included. Verify that the command transmitted after the Immediate Command has the same CmdSN as the Immediate Command.

Possible Problems: It may be difficult to get the DUT to transmit a command for immediate delivery.

Test #2.2: Immediate Delivery

Purpose: To verify that an initiator performs command numbering properly for task management commands.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:12:31 2003

Discussion: Commands meant for immediate delivery are marked with an immediate delivery flag; they MUST also carry the current CmdSN. CmdSN does not advance after a command marked for immediate delivery is sent.If immediate delivery is used with task management commands, these commands may reach the target before the tasks on which they are supposed to act. However their CmdSN serves as a marker of their position in the stream of commands. The initiator and target must ensure that the task management commands act as specified by [SAM2]. For example, both commands and responses appear as if delivered in order. Whenever CmdSN for an outgoing PDU is not specified by an explicit rule, CmdSN will carry the current value of the local CmdSN variable (see later in this section).

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any command. For the purpose of this test, this command must be set for Immediate Delivery. The DUT may choose to send a READ or WRITE Command and then immediately cancel this command and transmit an ABORT TASK command set for immediate delivery. Another method would be for the Testing Station to wait for any command from the DUT, and not respond at all. The DUT may attempt to transmit a

SNACK, the Testing Station should reject this SNACK. The Testing Station should reject any attempts by the DUT to reassign or retry the command. Eventually the command should time out and the DUT may transmit an ABORT TASK set for immediate delivery. The only requirement for this is that the DUT send a task management command intended for immediate delivery.

Observable Results:
· Verify that the task management command marked for immediate delivery is given the CmdSN that would be given to the next non-immediate command. The command after the Task Management command should have the same CmdSN as the Task Management command.

Possible Problems: In all Error Recovery classes, the implementer has the choice of deferring errors to the SCSI initiator. This may result in a timeout elapsing before a Task Management command is sent.

Test #3.1: MaxCmdSN-ExpCmdSN

Purpose: To verify that an initiator properly handles the received MaxCmdSN and ExpCmdSN fields.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:12:39 2003

Discussion: For the numbering mechanism, the initiator and target maintain CmdSN, ExpCmdSn, MaxCmdSN. The initiator's ExpCmdSN and MaxCmdSN are derived from target-to-initiator PDU fields. Comparisons and arithmetic on ExpCmdSN and MaxCmdSN MUST use Serial Number Arithmetic where SERIAL_BITS = 32. The target MUST NOT transmit a MaxCmdSN that is less than ExpCmdSN-1. For non-immediate commands, the CmdSN field can take any value from ExpCmdSN to MaxCmdSN inclusive. The target MUST silently ignore any non-immediate command outside of this range or non-immediate duplicates within the range. The CmdSN carried by immediate commands may lie outside the ExpCmdSN to MaxCmdSN range. For example, if the initiator has previously sent a non-immediate command carrying the CmdSN equal to MaxCmdSN, the target window is closed.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command. Transmit response data and status with a MaxCmdSN equal to ExpCmdSN-1. At this point the command window on the DUT should be at zero.
· Initiate a second READ or WRITE command on the DUT.

· After 2 seconds the Testing Station should transmit a NOP-In PDU with MaxCmdSN = ExpCmdSN+3.

Observable Results:
· Verify that the DUT does not transmit any SCSI Command PDUs to the Testing Station, until after the testing station has transmitted a NOP-In PDU with MaxCmdSN > ExpCmdSN-1, indicating that the target is ready to receive more commands.

Possible Problems: None

Test #3.2: MaxCmdSN-ExpCmdSN

Purpose: To verify that an initiator properly handles the received MaxCmdSN and ExpCmdSN fields.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:12:45 2003

Discussion: For the numbering mechanism, the initiator and target maintain CmdSN, ExpCmdSn, MaxCmdSN. The initiator's ExpCmdSN and MaxCmdSN are derived from target-to-initiator PDU fields. Comparisons and arithmetic on ExpCmdSN and MaxCmdSN MUST use Serial Number Arithmetic where SERIAL_BITS = 32. The target MUST NOT transmit a MaxCmdSN that is less than ExpCmdSN-1. For non-immediate commands, the CmdSN field can take any value from ExpCmdSN to MaxCmdSN inclusive. The target MUST silently ignore any non-immediate command outside of this range or non-immediate duplicates within the range. The CmdSN carried by immediate commands may lie outside the ExpCmdSN to MaxCmdSN range. For example, if the initiator has previously sent a non-immediate command carrying the CmdSN equal to MaxCmdSN, the target window is closed.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command. Transmit response data and status with a MaxCmdSN is 100 units greater than ExpCmdSN-1. At this point the command window on the DUT should be at the difference between MaxCmdSN and ExpCmdSN-1.
· Initiate multiple READ or WRITE commands on the DUT.

Observable Results:
· Verify that if the DUT supports multiple outstanding commands, it transmits multiple SCSI Command PDUs to the Testing Station without waiting for a response to indicate a larger command window size. This indicates that the DUT is aware of the number of commands that the target is willing to accept.

Possible Problems: None

Test #3.3: MaxCmdSN-ExpCmdSN

Purpose: To verify that an initiator properly handles the received MaxCmdSN and ExpCmdSN fields.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:12:54 2003

Discussion: For the numbering mechanism, the initiator and target maintain CmdSN, ExpCmdSn, MaxCmdSN. The initiator's ExpCmdSN and MaxCmdSN are derived from target-to-initiator PDU fields. Comparisons and arithmetic on ExpCmdSN and MaxCmdSN MUST use Serial Number Arithmetic where SERIAL_BITS = 32. The target MUST NOT transmit a MaxCmdSN that is less than ExpCmdSN-1. For non-immediate commands, the CmdSN field can take any value from ExpCmdSN to MaxCmdSN inclusive. The target MUST silently ignore any non-immediate command outside of this range or non-immediate duplicates within the range. The CmdSN carried by immediate commands may lie outside the ExpCmdSN to MaxCmdSN range. For example, if the initiator has previously sent a non-immediate command carrying the CmdSN equal to MaxCmdSN, the target window is closed.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command. Transmit response data and status with a MaxCmdSN equal to ExpCmdSN. At this point the command window on the DUT should be at one.
· Initiate a second READ or WRITE command on the DUT.

Observable Results:
· Verify that the DUT does not transmit more than one SCSI Command PDU to the Testing Station, until after the testing station has transmitted a NOP IN PDU with MaxCmdSN > ExpCmdSN-1, indicating that the target is ready to receive more commands.

Possible Problems: None

Test #4.1: Command Retry

Purpose: To verify that if an initiator chooses to retry a command it does so properly.

Reference: 3.2.2.1, 6.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:13:06 2003

Discussion: A numbered iSCSI request will not change its allocated CmdSN, regardless of the number of times and circumstances in which it is reissued.By resending the same iSCSI command PDU ("retry") in the absence of a command acknowledgement (by way of an ExpCmdSN update) or a response, an initiator attempts to "plug" (what it thinks are) the discontinuities in CmdSN ordering on the target end. Discarded command PDUs, due to digest errors, may have created these discontinuities. Retry MUST NOT be used for reasons other than plugging command sequence gaps, and in particular, cannot be used for requesting PDU retransmissions from a target. Any such PDU retransmission requests for a currently allegiant command in progress may be made using the SNACK mechanism described in section 9.16, although the usage of SNACK is OPTIONAL.When an iSCSI command is retried, the command PDU MUST carry the original Initiator Task Tag and the original operational attributes (e.g., flags, function names, LUN, CDB etc.) as well as the original CmdSN. The command being retried MUST be sent on the same connection as the original command unless the original connection was already successfully logged out.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attempt to negotiate support for ErrorRecoveryLevel=1.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, do not transmit response data and status.

· The DUT may time out and retry the command. The DUT may transmit a SNACK before retrying the command. This is more likely to occur if ErrorRecoveryLevel 1 is supported.

Observable Results:
· Verify that if the DUT chooses to Retry a command, it appears exactly as the original was with the same Initiator Task Tag, Flags, LUN and CDB.Verify that it is transmitted on the same connection as the original was.
· If the DUT chooses to transmit a SNACK then verify that this also appears on the same connection as the original command.

Possible Problems: In all Error Recovery classes, the implementer has the choice of deferring errors to the SCSI initiator.

Test #5.1: ExpStatSN

Purpose: To verify that if an initiator recognizes a large absolute difference between StatSN and ExpStatSN.

Reference: 3.2.2.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:13:14 2003

Discussion: Responses in transit from the target to the initiator are numbered. The StatSN (Status Sequence Number) is used for this purpose. StatSN is a counter maintained per connection. ExpStatSN is used by the initiator to acknowledge status. The status sequence number space is 32-bit unsigned-integers and the arithmetic operations are the regular mod(2**32) arithmetic. A large absolute difference between StatSN and ExpStatSN may indicate a failed connection. Initiators MUST undertake recovery actions if the difference is greater than an implementation defined constant that MUST NOT exceed 2**31-1. Initiators and Targets MUST support the response-numbering scheme.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command. Transmit response data and status with a StatSN which has an absolute difference of 2**31 from the received ExpStatSN value.

Observable Results:
· Verify that the DUT undertakes recovery actions. If ErrorRecoveryLevel=0 the DUT may choose to close the session.

Possible Problems: In all Error Recovery classes, the implementer has the choice of deferring errors to the SCSI initiator.

Test #6.1: DataSN

Purpose: To verify that the initiator properly increments the DataSN field when transmitting SCSI Data-out PDUs.

Reference: 3.2.2.3, 10.7.5

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:13:27 2003

Discussion: Data and R2T PDUs transferred as part of some command execution MUST be sequenced. The DataSN field is used for data sequencing. For output data PDUs, DataSN starts with 0 for the first data PDU of a sequence (the initial unsolicited sequence or any data PDU sequence issued to satisfy an R2T) and advances by 1 for each subsequent data PDU.For output (write) data PDUs, the DataSN is the Data-Out PDU number within the current output sequence. The current output sequence is either identified by the Initiator Task Tag (for unsolicited data) or is a data sequence generated for one R2T (for data solicited through R2T).

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase. Negotiate ImmediateData=No and InitialR2T=Yes. Continue into Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any WRITE Command iSCSI request.
· Transmit an R2T in response to the iSCSI request.
· Wait for the sequence of Data Out PDUs.
· Wait for a second WRITE Command iSCSI request. This request should have a new InitiatorTaskTag.
· Transmit an R2T in response to the iSCSI request.

· Wait for the sequence of Data Out PDUs.

Observable Results:
· Verify that in each sequence of Data Out PDUs the DataSN field started at 0 and was incremented by one for each subsequent Data PDU.

Possible Problems: None.

Test #7.1: Connection Reassignment

Purpose: To verify that if an initiator recognizes a failed connection, that it correctly performs connection reassignment if that is supported.

Reference: 6.2.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:13:36 2003

Discussion: If a connection fails before a command over that connection completes, the connection allegiance of the command may be explicitly reassigned to a different transport connection. By issuing a "task reassign" task management request the initiator signals its intent to continue an already active command (but with no current connection allegiance) as part of connection recovery. This means that a new connection allegiance is requested for the command, which seeks to associate it to the connection on which the task management request is being issued. Before the allegiance reassignment is attempted for a task, an implicit or explicit Logout with the reason code "remove the connection for recovery" MUST be successfully completed for the previous connection to which the task was allegiant.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Negotiate support for ErrorRecoveryLevel=2.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for any READ or WRITE Command. Transmit response data.
· Before the command has been completed (i.e. before sending Status data) , the Testing Station should drop the connection.

Observable Results:
· Verify that the DUT starts a new connection, and transmits a task reassign task management command.

Possible Problems: The device may choose not to perform Connection Reassignment for the procedure described above. If a means of causing the device to perform Connection Reassignment cannot be found, it will not be possible to complete this test.

Test #8.1: Data Transmission

Purpose: To verify that an iSCSI initiator uses the parameters negotiated during the Login Phase to govern how it sends data to the target.

Reference: 3.2.4.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:13:44 2003

Discussion: Outgoing SCSI data (initiator to target user data or command parameters) is sent as either solicited data or unsolicited data. Solicited data are sent in response to R2T PDUs. Unsolicited data can be sent as part of an iSCSI command PDU ("immediate data") or in separate iSCSI data PDUs.An initiator may send unsolicited data up to FirstBurstLength as immediate (up to the negotiated maximum PDU length), in a separate PDU sequence or both. All subsequent data MUST be solicited. The maximum length of an individual data PDU or the immediate-part of the first unsolicited burst MAY be negotiated at login. The maximum amount of unsolicited data that can be sent with a command is negotiated at login through the FirstBurstLength key.If any non-immediate unsolicited data is sent, the total unsolicited data MUST be either FirstBurstLength, or all of the data if the total amount is less than the FirstBurstLength.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following parameters: FirstBurstLength = 512; MaxBurstLength = 1024; InitialR2T = No; ImmediateData = Yes.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for WRITE commands from the DUT.

Observable Results:
· Verify that if the DUT chose to transfer ImmediateData with its initial WRITE

command it only included data up to the negotiated FirstBurstLength.
· Verify that if the DUT chose to transfer its initial data as a separate Data-out PDUs (as opposed to immediate data) it did not transfer more than the negotiated FirstBurstLength, before receiving an R2T.
· Verify that all subsequent Data-out PDUs are not transmitted until an R2T has been received from the Testing Station.
· Verify that the DUT honors any received R2T data requests for a valid outstanding command.

Possible Problems: Some devices may not support the Operational Parameter values to be negotiated in this test. If so, other values can be substituted. The objective of this test is to verify that after an Initiator sends the maximum amount of Unsolicited data that the target is willing to accept, it will send only solicited data.

Test #8.2: Data Transmission

Purpose: To verify that an iSCSI initiator does not transmit Unsolicited data when a target is operating in R2T mode.

Reference: 3.2.4.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:14:04 2003

Discussion: Outgoing SCSI data (initiator to target user data or command parameters) is sent as either solicited data or unsolicited data. Solicited data are sent in response to R2T PDUs. It is considered an error for an initiator to send unsolicited data PDUs to a target that operates in R2T mode (only solicited data are allowed). It is also an error for an initiator to send more unsolicited data, whether immediate or as separate PDUs, than FirstBurstLength.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following parameters: FirstBurstLength = 512; MaxBurstLength = 1024; InitialR2T = Yes; ImmediateData = No.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for WRITE commands from the DUT.

Observable Results:
· Verify that if the DUT does not transmit any ImmediateData.
· Verify that the DUT does not transmit any Data Out PDUs until an R2T has been received from the Testing Station.

Possible Problems: Some devices may not support the Operational Parameter values to be negotiated in this test. If so, other values can be substituted. The objective of this test

is to verify that after an Initiator does not send unsolicited data when this is not supported by the target.

Test #8.3: Data Transmission

Purpose: To verify that an iSCSI initiator does not transmit Unsolicited data when a target is operating in R2T mode.

Reference: 3.2.4.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:14:12 2003

Discussion: Outgoing SCSI data (initiator to target user data or command parameters) is sent as either solicited data or unsolicited data. Solicited data are sent in response to R2T PDUs. Unsolicited data can be sent as part of an iSCSI command PDU ("immediate data") or in separate iSCSI data PDUs.An initiator may send unsolicited data up to FirstBurstLength as immediate (up to the negotiated maximum PDU length), in a separate PDU sequence or both. All subsequent data MUST be solicited. The maximum length of an individual data PDU or the immediate-part of the first unsolicited burst MAY be negotiated at login. The maximum amount of unsolicited data that can be sent with a command is negotiated at login through the FirstBurstLength key.If any non-immediate unsolicited data is sent, the total unsolicited data MUST be either FirstBurstLength, or all of the data if the total amount is less than the FirstBurstLength.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following parameters: FirstBurstLength = 512; MaxBurstLength = 1024; InitialR2T = No; ImmediateData = No.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for WRITE commands from the DUT.

Observable Results:
· Verify that the DUT chooses to send a separate PDU of Unsolicited Data and that it

does not transmit more than the negotiated value for FirstBurstSize.

Possible Problems: Some devices may not support the Operational Parameter values to be negotiated in this test. If so, other values can be substituted. The objective of this test is to verify that after an Initiator does not send unsolicited data greater than the amount that the target has indicated it can accept.

Test #9.1: Target Transfer Tag

Purpose: To verify that an iSCSI initiator properly uses the Target Transfer Tag provided by the iSCSI target.

Reference: 3.2.4.3, 10.7.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Thu Jun 19 16:21:00 2003

Discussion: Target tags are not strictly specified by the protocol. It is assumed that target tags are used by the target to tag (alone or in combination with the LUN) the solicited data. Target tags are generated by the target and "echoed" by the initiator.On outgoing data, the Target Transfer Tag is provided to the target if the transfer is honoring an R2T. In this case, the Target Transfer Tag field is a replica of the Target Transfer Tag provided with the R2T.The Target Transfer Tag values are not specified by this protocol except that the value 0xffffffff is reserved and means that the Target Transfer Tag is not supplied. If the Target Transfer Tag is provided, then the LUN field MUST hold a valid value and be consistent with whatever was specified with the command; otherwise, the LUN field is reserved.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following parameters: FirstBurstSize = 512; MaxBurstSize = 1024; InitialR2T = Yes; ImmediateData = No.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status. Wait for WRITE commands from the DUT.
· Transmit an R2T PDU ,with the Target Transfer Tag set, to the DUT to allow it to begin transmitting Data-out PDUs.

Observable Results:

· Verify that if the DUT includes the Target Transfer Tag supplied by the Testing Station, in its Data-out PDUs.

Possible Problems: Some devices may not support the Operational Parameter values to be negotiated in this test. If so, other values can be substituted.

Test #10.1: Data-in Status

Purpose: To verify that an iSCSI initiator properly supports status accompanying a Data-in PDU.

Reference: 10.7, 10.7.3

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:14:33 2003

Discussion: Status can accompany the last Data-in PDU if the command did not end with an exception (i.e., the status is "good status" - GOOD, CONDITION MET or INTERMEDIATE CONDITION MET). The presence of status (and of a residual count) is signaled though the S flag bit. Although targets MAY choose to send even non-exception status in separate responses, initiators MUST support non-exception status in Data-In PDUs. The S bit is set to indicate that the Command Status field contains status. If this bit is set to 1, the F bit MUST also be set to 1.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status. Wait for one READ command from the DUT.
· Transmit a Data-in PDU up to the size for MaxRecvDataSegmentLength declared by the DUT. Continue to do this until the READ command is complete.
· In the final Data-in PDU of the command include non-exception (GOOD) status information. The S bit an the F bit should both be set to 1.

Observable Results:
· Verify that the DUT accepts the status information included in the Data-in PDU, and will initiate new commands without waiting for a SCSI Response PDU for the original command.

Possible Problems: None.

Test #10.2: Data-in A bit

Purpose: To verify that an iSCSI initiator properly responds when the target chooses to set the A bit.

Reference: 10.7.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:14:39 2003

Discussion: For sessions with ErrorRecoveryLevel 1 or higher, the target sets this bit to 1 to indicate that it requests a positive acknowledgement from the initiator for the data received.On receiving a Data-In PDU with the A bit set to 1 on a session with ErrorRecoveryLevel greater than 0, if there are no holes in the read data until that Data-In PDU, the initiator MUST issue a SNACK of type DataACK except when it is able to acknowledge the status for the task immediately via ExpStatSN on other outbound PDUs if the status for the task is also received. In the latter case (acknowledgement through ExpStatSN), sending a SNACK of type DataACK in response to the A bit is not mandatory, but if it is done, it must not be sent after the status acknowledgement through ExpStatSN.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one READ command from the DUT.
· Transmit a Data-in PDU up to the size for MaxRecvDataSegmentLength declared by the DUT. Continue to do this until the READ command is complete.
· In the final Data-in PDU of the command set the A bit. Do not include status information in this PDU.

Observable Results:

· Verify that the DUT transmits a SNACK of type DataACK to the Testing Station upon receiving a Data-in PDU with the A bit set. THe DUT may also acknowledge the command through ExpStatSN. Verify that if the DUT sends a DataACK, it does so before any acknowledgement through ExpStatSN.

Possible Problems: None.

Test #11.1.1: Data-out DataSegmentLength

Purpose: To verify that an iSCSI initiator properly uses the MaxRecvDataSegmentLength parameter declared by the target.

Reference: 10.7.7

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:02 2003

Discussion: This is the data payload length of a SCSI Data-In or SCSI Data-Out PDU. The sending of 0 length data segments should be avoided, but initiators and targets MUST be able to properly receive 0 length data segments. The Data Segments of Data-in and Data-out PDUs SHOULD be filled to the integer number of 4 byte words (real payload) unless the F bit is set to 1. The Data Segments of Data-in and Data-out PDUs SHOULD be filled to the integer number of 4 byte words (real payload) unless the F bit is set to 1.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· The Testing Station should declare a MaxRecvDataSegmentLength of 512
· Negotiate the following parameters: ImmediateData=Yes; InitialR2T=No; FirstBurstLength=65536; MaxBurstLength=262144.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one WRITE command from the DUT.

Observable Results:
· Verify that the DUT does not transmit any Data-out PDU with greater than MaxRecvDataSegmentLength.

Possible Problems: None.

Test #11.1.2: Data-In DataSegmentLength

Purpose: To verify that an iSCSI initiator supports a DataSegmentLength of 0.

Reference: 10.7.7

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:08 2003

Discussion: The sending of 0 length data segments should be avoided, but initiators and targets MUST be able to properly receive 0 length data segments.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· The Testing Station should declare a MaxRecvDataSegmentLength of 256.
· Negotiate the following parameters: ImmediateData=Yes; InitialR2T=No; FirstBurstLength=65536; MaxBurstLength=262144.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one READ.
· Transmit a Data-in PDU with a DataSegmentLength=0.

Observable Results:
· Verify that the DUT does not interpret DataSegmentLength=0 as an error.

Possible Problems: None.

Test #11.2.1: Data-out F bit

Purpose: To verify that an iSCSI initiator properly sets the F bit for the last PDU of unsolicited Data.

Reference: 10.7.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:17 2003

Discussion: For outgoing data, this bit is 1 for the last PDU of unsolicited data or the last PDU of a sequence that answers an R2T.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following values: ImmediateData=Yes; InitialR2T=No; FirstBurstLength=65536; MaxBurstLength=262144.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one WRITE command from the DUT.

Observable Results:
· Verify that if the DUT chooses to transmit ImmediateData that it does so only up to the FirstBurstLength limit that was negotiated, and that the last Data-out PDU of unsolicited data has the F bit set to 1.

Possible Problems: None.

Test #11.2.2: Data-out F bit

Purpose: To verify that an iSCSI initiator properly sets the F bit for the last PDU of solicited Data.

Reference: 10.7.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:24 2003

Discussion: For outgoing data, this bit is 1 for the last PDU of unsolicited data or the last PDU of a sequence that answers an R2T.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=65536; MaxBurstlength=262144.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with a Desired Data Transfer Length of 65536.

Observable Results:
· Verify that the last Data-out PDU of the requested 65536 bytes has the F bit set to 1.

Possible Problems: None.

Test #11.3: Data-out DataSN

Purpose: To verify that an iSCSI initiator properly sets DataSN field in a Data-out PDU.

Reference: 10.7.5

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:32 2003

Discussion: For output (write) data PDUs, the DataSN is the Data-Out PDU number within the current output sequence. The current output sequence is either identified by the Initiator Task Tag (for unsolicited data) or is a data sequence generated for one R2T (for data solicited through R2T).

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstSize=65536; MaxBurstSize=262144.
· Declare MaxRecvDataSegmentLength=1024.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with a Desired Data Transfer Length of 2048. This should cause the DUT to have to send multiple PDUs. Transmit another R2T to authorize further Data-out PDUs. Repeat this until the command is complete.

Observable Results:
· Verify that the DUT sets the DataSN field starting with 0 and increments it for every Data-out PDU within the sequence, which is solicited through one R2T.

Possible Problems: None.

Test #11.4: Data-out Buffer Offset

Purpose: To verify that an iSCSI initiator properly sets Buffer Offset field in a Data-out PDU.

Reference: 10.7.6

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:40 2003

Discussion: The Buffer Offset field contains the offset of this PDU payload data within the complete data transfer. The sum of the buffer offset and length should not exceed the expected transfer length for the command. The order of data PDUs within a sequence is determined by DataPDUInOrder. When set to Yes, it means that PDUs have to be in increasing Buffer Offset order and overlays are forbidden. The ordering between sequences is determined by DataSequenceInOrder. When set to Yes, it means that sequences have to be in increasing Buffer Offset order and overlays are forbidden.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=65536; MaxBurstLength=262144, DataPDUInOrder=Yes, DataSequenceInOrder=Yes.
· Declare MaxRecvDataSegmentLength=1024.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with a Desired Data Transfer Length of 2048. This should cause the DUT to have to send multiple PDUs. Transmit another R2T to authorize further Data-out PDUs. Repeat this until the command is complete.

Observable Results:

· Verify that the DUT sets the Buffer Offset field accurately, and that it increases with each PDU.

Possible Problems: None.

Test #12.1: R2T Received

Purpose: To verify that an iSCSI initiator properly responds to a received R2T from an iSCSI target.

Reference: 10.8

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:15:58 2003

Discussion: An R2T MAY be answered with one or more SCSI Data-out PDUs with a matching Target Transfer Tag. If an R2T is answered with a single Data-out PDU, the Buffer Offset in the Data PDU MUST be the same as the one specified by the R2T, and the data length of the Data PDU MUST be the same as the Desired Data Transfer Length specified in the R2T.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLngth=65536; MaxBurstLength=262144.
· Declare MaxRecvDataSegmentLength=512.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with the following parameters: Desired Data Transfer Length=2048; Target Transfer Tag= 0xABBAABBA; Buffer Offset=0..

Observable Results:
· Verify that the DUT sets the Target Transfer Tag in the subsequent Data-out PDU to 0xABBAABBA.
· If the DUT transmits only one PDU in response to the received R2T, verify that the Buffer Offset in the Data-out PDU is the same as that in the R2T. Verify that the DataSegmentLength of the Data-out PDU is the same as the Desired Data Transfer

Length specified in the R2T (2048).

Possible Problems: None.

Test #12.2: R2T Received

Purpose: To verify that an iSCSI initiator properly responds to a received R2T from an iSCSI target.

Reference: 10.8

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:16:06 2003

Discussion: An R2T MAY be answered with one or more SCSI Data-out PDUs with a matching Target Transfer Tag. If an R2T is answered with a single Data-out PDU, the Buffer Offset in the Data PDU MUST be the same as the one specified by the R2T, and the data length of the Data PDU MUST be the same as the Desired Data Transfer Length specified in the R2T. If the R2T is answered with a sequence of Data PDUs, the Buffer Offset and Length MUST be within the range of those specified by R2T, and the last PDU MUST have the F bit set to 1.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=65536; MaxBurstLength=262144, DataPDUInOrder=Yes; MaxRecvDataSegmentLength=512.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with the following parameters: Desired Data Transfer Length=4096; Target Transfer Tag= 0xABBAABBA; Buffer Offset=0..

Observable Results:
· Verify that the DUT sets the Target Transfer Tag in the subsequent Data-out PDU correctly.
· The DUT should choose to answer the R2T with a sequence of Data-out PDUs, verify

that the Buffer Offset and Length are within the range specified by R2T and that they form a continuous non-overlapping range.
· Verify that the last Data-out PDU in the sequence has the F bit set to1.
· Verify that the PDU Buffer Offsets are transmitted in increasing order.

Possible Problems: None.

Test #12.3: R2T Received

Purpose: To verify that an iSCSI initiator properly responds to a received R2T from an iSCSI target.

Reference: 10.8

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:16:15 2003

Discussion: An R2T MAY be answered with one or more SCSI Data-out PDUs with a matching Target Transfer Tag. If an R2T is answered with a single Data-out PDU, the Buffer Offset in the Data PDU MUST be the same as the one specified by the R2T, and the data length of the Data PDU MUST be the same as the Desired Data Transfer Length specified in the R2T. If the R2T is answered with a sequence of Data PDUs, the Buffer Offset and Length MUST be within the range of those specified by R2T, and the last PDU MUST have the F bit set to 1. If the last PDU (marked with the F bit) is received before the Desired Data Transfer Length is transferred, a target MAY choose to Reject that PDU with "Protocol error" reason code. DataPDUInOrder governs the Data-Out PDU ordering. If DataPDUInOrder is set to Yes, the Buffer Offsets and Lengths for consecutive PDUs MUST form a continuous non-overlapping range and the PDUs MUST be sent in increasing offset order.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=65536; MaxBurstLength=262144, DataPDUInOrder=Yes; MaxRecvDataSegmentLength=512; DataSequenceInOrder=No.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for one WRITE command from the DUT.
· Transmit an R2T with the following parameters: Desired Data Transfer Length=1024; Target Transfer Tag= 0xABBAABBA; Buffer Offset=3072.

· Transmit an R2T with the following parameters: Desired Data Transfer Length=1024; Target Transfer Tag= 0xABBAABBA; Buffer Offset=2048.
· Transmit an R2T with the following parameters: Desired Data Transfer Length=1024; Target Transfer Tag= 0xABBAABBA; Buffer Offset=1024.
· Transmit an R2T with the following parameters: Desired Data Transfer Length=1024; Target Transfer Tag= 0xABBAABBA; Buffer Offset=0.

Observable Results:
· Verify that the DUT fulfills each R2T in the order it was received. This can be checked by seeing that the Data-out PDUs have the same order of Buffer Offset value as the series of R2Ts from the Testing Station.

Possible Problems: None.

Test #12.4: Parallel Commands

Purpose: To verify that an initiator fulfills R2T's properly when transmitting commands in paralell on a single connection.

Reference: 10.8

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon Jun 9 16:18:16 2003

Discussion: Within a connection, outstanding R2Ts MUST be fulfilled by the initiator in the order in which they were received.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session). Negotiate the following: InitialR2T=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT, transmit response data and status.
· Wait for a TEST UNIT READY from the DUT, transmit response data and status.
· Wait for a READ-CAP from the DUT, transmit response data and status.
· Wait for the DUT to perform parallel WRITE Commands. This would be 2 WRITE commands being executed at the same time, with each command starting at almost the same time, not just a second command starting after the first is complete.
· The Testing Station should respond to the two write commands with R2T to each. The R2T for the first arriving command should be sent after the R2T for the second arriving command. The R2T's should be trasmitted as close together as possible.

Observable Results:
· Verify that the DUT answered the R2Ts in the order they arrived, not in the order that the commands were transmitted. Thus Data-Out PDU's for the first R2T should have the same Initiator Task Tag as the second WRITE command.

Possible Problems: The DUT may not send WRITE commands in parallel as described above. If so, this item is not testable.

Test #13.1: SNACK

Purpose: To verify that an iSCSI initiator properly constructs a SNACK Request.

Reference: 10.16, 10.16.1, 10.16.4, 10.16.5, 10.16.6

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon Jun 9 16:18:44 2003

Discussion: Support for all SNACK types is mandatory if the implementation supports ErrorRecoveryLevel greater than zero. The SNACK is used by the initiator to request the retransmission of numbered-responses, data, or R2T PDUs from the target. The SNACK request indicates the numbered-responses or data "runs" whose retransmission is requested by the target, where the run starts with the first StatSN, DataSN, or R2TSN whose retransmission is requested and indicates the number of Status, Data, or R2T PDUs requested including the first. 0 has special meaning when used as a starting number and length. The numbered-response(s) or R2T(s), requested by a SNACK, MUST be delivered as exact replicas of the ones that the target transmitted originally except for the fields ExpCmdSN, MaxCmdSN, and ExpDataSN, which MUST carry the current values. R2T(s)requested by SNACK MUST also carry the current value of StatSN. Data/R2T SNACK, Status SNACK, or R-Data SNACK for a command MUST precede status acknowledgement for the given command. In the case of a Data/R2T SNACK, the Target Transfer Tag field must be set to 0xffffffff and the Initiator Task Tag field MUST be set to the Initiator Task Tag of the referenced command. BegRun must contain the DataSN, R2TSN, or StatSN of the first PDU whose retransmission isrequested.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=65536; MaxBurstLength=262144, DataPDUInOrder=Yes; MaxRecvDataSegmentLength=1024; ErrorRecoveryLevel=1; DataSequenceInOrder=No.

· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit

response data and status to each.
· Wait for one READ command from the DUT.
· Transmit a series of Data-in PDUs. Skip the value of DataSN=3, this should cause the DUT, if it supports SNACK to issue a SNACK of type 0 = Data/R2T SNACK.

Observable Results:
· Verify that the DUT does not attempt to transmit status acknowledgement on the command before attempting Data/R2T SNACK.
· Verify that the Initiator Task Tag of the R2T SNACK is set to the InitiatorTaskTag of the reference command.
· Verify that the Target Transfer Tag is set to 0xFFFFFFFF.
· Verify that the BegRun field is set to 3 and the Length field is set to 1.

Possible Problems: The DUT may choose to request all Data-in PDUs by setting Runlength and BegRun to 0.

Test #13.2: SNACK

Purpose: To verify that an iSCSI initiator properly constructs a SNACK Request.

Reference: 10.16, 10.16.1, 10.16.3

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:16:49 2003

Discussion: Support for SNACK is mandatory only if the supported ErrorRecoveryLevel of the implementation is greater than zero. The SNACK request is used to request the retransmission of numbered responses, data, or R2T PDUs from the target. The SNACK request indicates the missed numbered-response or data "run" to the target, where the run starts with the first missed StatSN, DataSN, or R2TSN and indicates also the number of missed Status, Data, or R2T PDUs (0 has the special meaning of "all after the initial"). For DataACK, the Target Transfer Tag has to contain a copy of the Target Transfer Tag and LUN provided with the SCSI Data-In PDU with the A bit set to 1. If an initiator operates at ErrorRecoveryLevel 1 or higher, it MUST issue a SNACK of type DataACK after receiving a Data-In PDU with the A bit set to 1. However, if the initiator has detected holes in the input sequence, it MUST postpone issuing the SNACK of type DataACK until the holes are filled. An initiator MAY ignore the A bit if it deems that the bit is being set aggressively by the target (i.e., before the MaxBurstSize limit is reached). The RunLength MUST also be 0 for a DataACK SNACK.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=4096; MaxBurstLength=8194, DataPDUInOrder=Yes; MacRecvPDULength=1024; ErrorRecoveryLevel=1; DataSequenceInOrder=No.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for one or more READ commands from the DUT.
· Transmit a series of Data-in PDUs. Once the negotiated value of MaxBurstLength has

been transmitted by the Testing Station, transmit a Data-in PDU with the A bit set to 1. This should cause the DUT, if it supports SNACK to issue a SNACK of type 2 = DataACK.

Observable Results:
· Verify that the RunLength field is set to 0.
· Verify that the Initiator Task Tag of the R2T is set to the reserved value of 0xFFFFFFFF.
· Verify that the Target Transfer Tag is set to the Target Transfer tag provided by the Testing Station.

Possible Problems: None.

Test #14.1: Logout Request

Purpose: To verify that an iSCSI initiator properly constructs a Logout Request.

Reference: 10.9, 10.9.1, 10.14, 10.14.2, 10.14.3, 10.14.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:17:21 2003

Discussion: An Asynchronous Message may be sent from the target to the initiator without correspondence to a particular command. The target specifies the reason for the event and sense data. An Asynchronous Message of code 1 indicates that the target requests Logout. This Async Message MUST be sent on the same connection as the one requesting to be logged out. The initiator MUST honor this request by issuing a Logout as early as possible, but no later than Parameter3 seconds. Initiator MUST send a Logout with a reason code of "Close the connection" OR "Close the session" to close all the connections. Once this message is received, the initiator SHOULD NOT issue new iSCSI commands on the connection to be logged out. The Logout request is used to perform a controlled closing of a connection. An initiator MAY use a logout request to remove a connection from a session or to close an entire session. After sending the Logout request PDU, an initiator MUST NOT send any new iSCSI requests on the closing connection. If the Logout request is intended to close the session, new iSCSI requests MUST NOT be sent on any of the connections participating in the session. An Initiator may choose to issue a Logout Request to either 'close the session' or 'close the connection'. For a Logout Request the CID field contains the the connection ID of the connection to be closed (including closing the TCP stream). This field is only valid if the reason code is not "close the session".For a Logout Request the ExpStatSN field contains the last ExpStatSN value for the connection to be closed. For the Logout Request PDU, the TotalAHSLength and the DataSegmentLength MUST be 0.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes;

FirstBurstLength=4096; MaxBurstLength=8194, DataPDUInOrder=Yes; MaxRecvDataSegmentLength=1024; ErrorRecoveryLevel=1; DataSequenceInOrder=No.

· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for WRITE commands from the DUT.
· Transmit R2T to allow the DUT to transmit a Data-out PDU.
· Once the Data-out from the DUT is received, the Testing Station should transmit a Asynchronous Message with AsyncEvent code 1 = target request Logout, and with Parameter3=5.

Observable Results:
· Verify that the DUT transmits a Logout Request within Parameter3 seconds of receiving the Asynchronous Message request, the reason code can be either 'close the session' or 'close the connection'.
· Verify that the DUT does not attempt to send any new iSCSI commands after issuing the Logout Request.
· Verify that unless the Reason Code in the Logout Request is 'close the session', that the DUT sets the CID to a valid value.
· Verify that the ExpStatSN field is set to the same value as the last ExpStatSN value for the connection to be closed.
· Verify that the TotalAHSLength and the DataSegmentLength are 0 in the Logout Request.

Possible Problems: None.

Test #14.2: Logout Request

Purpose: To verify that an iSCSI initiator properly constructs a Logout Request.

Reference: 10.9, 10.9.1, 10.14, 10.14.2, 10.14.3, 10.14.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:17:28 2003

Discussion: An Asynchronous Message may be sent from the target to the initiator without correspondence to a particular command. The target specifies the reason for the event and sense data. An Asynchronous Message of code 1 indicates that the target requests Logout. This Async Message MUST be sent on the same connection as the one requesting to be logged out. The initiator MUST honor this request by issuing a Logout as early as possible, but no later than Parameter3 seconds. Initiator MUST send a Logout with a reason code of "Close the connection" OR "Close the session" to close all the connections. Once this message is received, the initiator SHOULD NOT issue new iSCSI commands on the connection to be logged out. The Logout request is used to perform a controlled closing of a connection. An initiator MAY use a logout request to remove a connection from a session or to close an entire session. After sending the Logout request PDU, an initiator MUST NOT send any new iSCSI requests on the closing connection. If the Logout request is intended to close the session, new iSCSI requests MUST NOT be sent on any of the connections participating in the session. An Initiator may choose to issue a Logout Request to either 'close the session' or 'close the connection'. For a Logout Request the CID field contains the the connection ID of the connection to be closed (including closing the TCP stream). This field is only valid if the reason code is not "close the session".For a Logout Request the ExpStatSN field contains the last ExpStatSN value for the connection to be closed. For the Logout Request PDU, the TotalAHSLength and the DataSegmentLength MUST be 0.

Test Setup: The DUT and Test Station pair should be able to make multiple TCP connections.

Procedure:
· Wait for 2 connections from the DUT, begin Login Phase negotiation. On each connection perform the following:

· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=4096; MaxBurstLength=8194, DataPDUInOrder=Yes; MacRecvPDULength=1024; ErrorRecoveryLevel=1; DataSequenceInOrder=No.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for WRITE commands from the DUT on each connection.
· Transmit R2T to allow the DUT to transmit a Data-out PDU.
· On one connection only, once the Data-out from the DUT is received, the Testing Station should transmit a Asynchronous Message with AsyncEvent code 1 = target request Logout, and with Parameter3=5.
· Wait for a Logout Request PDU from the DUT.
· Transmit a Logout Response on each connection where a Logout Request is recieved.

Observable Results:
· Verify that the DUT completely closes the connection that the Logout is requested on. The DUT may choose to close both connections if the reason code in the Logout Request is 'close the session'.
· Verify that the DUT does not attempt to send any new iSCSI commands after issuing the Logout Request on a given connection.
· Verify that if the DUT reason code in the Logout Request is only 'close the connection', that the DUT continues to transmit SCSI Commands on the other remaining connection.
· Verify that unless the Reason Code in the Logout Request is 'close the session', that the DUT sets the CID to a valid value.
· Verify that the ExpStatSN field is set to the same value as the last ExpStatSN value for the connection to be closed.

Possible Problems: None.

Test #15.1: NOP-Out Ping Response

Purpose: To verify that an iSCSI initiator properly constructs a NOP-Out.

Reference: 10.18, 10.18.1, 10.18.2, 10.19

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:47:35 2003

Discussion: Upon receipt of a NOP-In with the Target Transfer Tag set to a valid value (not the reserved 0xffffffff), the initiator MUST respond with a NOP-Out. In this case, the NOP-Out Target Transfer Tag MUST contain a copy of the NOP-In Target Transfer Tag. The NOP-Out MUST have the Initiator Task Tag set to a valid value only if a response in the form of NOP-In is requested (i.e., the NOP-Out is used as a ping request). Otherwise, the Initiator Task Tag MUST be set to 0xffffffff. If the Initiator Task Tag contains 0xffffffff, the I bit MUST be set to 1 and the CmdSN is not advanced after this PDU is sent. The NOP-Out MUST only have the Target Transfer Tag set if it is issued in response to a NOP-In with a valid Target Transfer Tag. In this case, it copies the Target Transfer Tag from the NOP-In PDU. Otherwise, the Target Transfer Tag MUST be set to 0xffffffff. When the Target Transfer Tag is set to a value other than 0xffffffff, the LUN field MUST also be copied from the NOP-In.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection from the DUT, begin Login Phase negotiation.
· Negotiate/declare the following values: ImmediateData=No; InitialR2T=Yes; FirstBurstLength=4096; MaxBurstLength=8194, DataPDUInOrder=Yes; MaxRecvDataSegmentLength=1024; ErrorRecoveryLevel=1; DataSequenceInOrder=No.

· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Transmit a well-formed NOP-In PDU with a valid Target Transfer Tag to the DUT.This should have a ITT of 0xffffffff and a DSL of 0.

Observable Results:

· Verify that the DUT responds to the received NOP-In with a NOP-Out PDU.
· Verify that the DUT set the Target Transfer Tag in the NOP-Out to the same value as that in the received NOP-In.
· Verify that the LUN field as set to the same as in the received NOP-In PDU.
· Verify that the Initiator Task Tag was set to 0xffffffff.
· Verify that if the Initiator Task Tag was set to 0xffffffff, that the I bit was set to 1.
· Verify that the DSL in the NOP-Out is 0.
· Verify that the next command sent by the DUT does not have CmdSN incremented from teh value sent in the NOP-Out.

Possible Problems: None.

Test #15.2: NOP-Out Ping Request

Purpose: To verify that an iSCSI initiator properly constructs a NOP-Out.

Reference: 10.18, 10.18.1, 10.18.2, 10.19

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Thu Jul 3 08:28:39 2003

Discussion: An iSCSI initiator may send a NOP-Out PDU as a "ping request" to verify that a connection/session is still active and all its components are operational. The NOP-Out MUST have the Initiator Task Tag set to a value value only if a response in the form of a NOP-In is requested, as is the case when sending a "ping request". If the NOP-Out is not issued as a response to a NOP-In with a valid Target Transfer Tag, the Target Transfer Tag of the NOP-In must be set to 0xffffffff.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Configure the DUT to send a NOP-Out to provoke a 'ping' response NOP-In from the target.
· Wait for a connection from the DUT, begin Login Phase negotiation.
· In Full Feature Phase operation do not respond to any activity for 2 minutes 30 seconds.
· Wait for a NOP-Out PDU from the DUT. When such is received transmit a NOP-In PDU in response. Wait for the next command from the DUT.

Observable Results:
· Verify that the Target Transfer Tag in the NOP-Out is set to 0xffffffff.
· The LUN field of the NOP-Out PDU should be set to 0.
· Verify that the Initiator Task Tag was set to a valid value (i.e. not 0xffffffff).
·Verify that if the I bit was set to 1 in the received NOP-Out, the DUT does not increment the CmdSN in the next Command PDU.
·Verify that if the I bit was set to 0 in the received NOP-Out, the DUT increments the CmdSN in the next Command PDU.

Possible Problems: It may not be possible to configure the device to transmit ping

requests. If so this item is Not Testable.

Test #15.3: NOP-Out Confirm ExpStatSN

Purpose: To verify that an iSCSI initiator properly constructs a NOP-Out.

Reference: 9.18, 9.18.1, 9.18.2, 9.19

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Tue Jan 14 13:42:28 2003

Discussion: An iSCSI initiator may send a NOP-Out PDU to verify a value for ExpStatSN. The NOP-Out MUST have the Initiator Task Tag set to a value value only if a response in the form of a NOP-In is requested, which is not the case when sending a NOP-Out to verify ExpStatSN. If the NOP-Out is not issued as a response to a NOP-In with a valid Target Transfer Tag, the Target Transfer Tag of the NOP-In must be set to 0xffffffff. Since the ITT is set to 0xffffffff, the I bit must be set to 1. CmdSN should not be advanced after the NOP-Out PDU is sent.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Configure the DUT to send a NOP-Out to confirm ExpStatSN from the target.
· Wait for a connection from the DUT, begin Login Phase negotiation.
· In Full Feature Phase operation wait for a NOP-Out PDU from the DUT. This is more likely to occur after a command has been completed. After the NOP-Out is received, wait for the next command from the DUT.

Observable Results:
· Verify that the Target Transfer Tag in the NOP-Out is set to 0xffffffff.
· Verify that the Initiator Task Tag was set to 0xffffffff.
·Verify that if the I bit was set to 1 in the received NOP-Out, the DUT does not increment the CmdSN in the next Command PDU.
· The LUN field of the NOP-Out PDU should be set to 0.

Possible Problems: It may not be possible to configure the device to transmit NOP-Out to confirm ExpStatSN. If so this item is Not Testable.

Test #16.1.1: SCSI Command PDU Fields

Purpose: To verify that the initiator issues the SCSI Command PDU correctly.

Reference: 10.2.1.8, 10.2.2.3, 10.3, 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:48:00 2003

Discussion: In a SCSI Command PDU, the R bit is set to 1 when the command is expected to input data.The W bit is set to 1 when the command is expected to output data. Bits 3-4 are Reserved. Bits 5-7 contain Task Attributes. Setting both the W and the F bit to 0 is an error. Either or both of R and W MAY be 1 when either the Expected Data Transfer Length and/or Bidirectional Read Expected Data Transfer Length are 0, but they MUST NOT both be 0 when the Expected Data Transfer Length and/or Bidirectional Read Expected Data Transfer Length are not 0 (i.e., when some data transfer is expected the transfer direction is indicated by the R and/or W bit). For unidirectional operations, the Expected Data Transfer Length field contains the number of bytes of data involved in this SCSI operation. For a unidirectional write operation (W flag set to 1 and R flag set to 0), the initiator uses this field to specify the number of bytes of data it expects to transfer for this operation. For a unidirectional read operation (W flag set to 0 and R flag set to 1), the initiator uses this field to specify the number of bytes of data it expects the target to transfer to the initiator. The initiator assigns a Task Tag to each iSCSI task it issues. While a task exists, this tag MUST uniquely identify the task session-wide. SCSI may also use the initiator task tag as part of the SCSI task identifier when the timespan during which an iSCSI initiator task tag must be unique extends over the timespan during which a SCSI task tag must be unique. However, the iSCSI Initiator Task Tag must exist and be unique even for untagged SCSI commands. An Extended CDB AHS MUST NOT be used if the CDBLength is less than 17. The length includes the reserved byte 3.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=Yes.

· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for any SCSI Command from the DUT.

Observable Results:
· Verify that the CmdSN of the SCSI Command is the same as the ExpCmdSN of the preceding Login Response PDU.
· Verify that the ExpStatSN of the SCSI Command is one greater than the StatSN of the preceding Login Response PDU.
· Verify that the InitiatorTaskTag is unique (i.e. different from the one used during Login).
· Verify that if data is transmitted in the PDU, the Data Segment Length reflects the number of bytes of user data or parameters contained in the Data Segment.
· Verify that the CDB Opcode is valid.
· Verify that if the length of the CDB is > 16 Bytes, an extended AHS is used.

Possible Problems: If the initiator does not support ImmediateData, the the portion of this test testing DataSegmentLength in the SCSI Command PDU is not testable.

Test #16.1.2: SCSI Command PDU Fields

Purpose: To verify that the initiator issues the SCSI Command PDU correctly.

Reference: 10.2.1.8, 10.2.2.3, 10.3, 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:48:07 2003

Discussion: In a SCSI Command PDU, the R bit is set to 1 when the command is expected to input data.The W bit is set to 1 when the command is expected to output data. Bits 3-4 are Reserved. Bits 5-7 contain Task Attributes. Setting both the W and the F bit to 0 is an error. Either or both of R and W MAY be 1 when either the Expected Data Transfer Length and/or Bidirectional Read Expected Data Transfer Length are 0, but they MUST NOT both be 0 when the Expected Data Transfer Length and/or Bidirectional Read Expected Data Transfer Length are not 0 (i.e., when some data transfer is expected the transfer direction is indicated by the R and/or W bit). For unidirectional operations, the Expected Data Transfer Length field contains the number of bytes of data involved in this SCSI operation. For a unidirectional write operation (W flag set to 1 and R flag set to 0), the initiator uses this field to specify the number of bytes of data it expects to transfer for this operation. For a unidirectional read operation (W flag set to 0 and R flag set to 1), the initiator uses this field to specify the number of bytes of data it expects the target to transfer to the initiator. The initiator assigns a Task Tag to each iSCSI task it issues. While a task exists, this tag MUST uniquely identify the task session-wide. SCSI may also use the initiator task tag as part of the SCSI task identifier when the timespan during which an iSCSI initiator task tag must be unique extends over the timespan during which a SCSI task tag must be unique. However, the iSCSI Initiator Task Tag must exist and be unique even for untagged SCSI commands. An Extended CDB AHS MUST NOT be used if the CDBLength is less than 17. The length includes the reserved byte 3.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=No.

· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for any SCSI Command from the DUT

Observable Results:
· Verify that the CmdSN of the SCSI Command is the same as the ExpCmdSN of the preceding Login Response PDU.
· Verify that the ExpStatSN of the SCSI Command is one greater than the StatSN of the preceding Login Response PDU.
· Verify that the InitiatorTaskTag is unique (i.e. different from the one used during Login).
· Verify that the DataSegmentLength = 0 and no data is in the DataSegment of the Command PDU.
· Verify that the CDB Opcode is valid.
· Verify that if the length of the CDB is > 16 Bytes, an extended AHS is used.

Possible Problems: None

Test #16.2.1: SCSI Command Unsolicited Data

Purpose: To verify that the initiator follows the rules for transmitting Unsolicited data.

Reference: 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:48:17 2003

Discussion: If the Expected Data Transfer Length for a write and the length of the immediate data part that follows the command (if any) are the same, then no more data PDUs are expected to follow. In this case, the F bit MUST be set to 1.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=Yes and InitialR2T=Yes, FirstBurstLength=2**24-1.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.

Observable Results:
· Verify that if ImmediateData is transmitted in the PDU, the Data Segment Length reflects the number of bytes of user data or parameters contained in the Data Segment.
· Verify that if the Expected Data Transfer Length = Data Segment Length the F bit is set to 1.

Possible Problems: If the initiator does not support ImmediateData, this item is not testable

Test #16.2.2: SCSI Command Unsolicited Data

Purpose: To verify that the initiator follows the rules for transmitting Unsolicited data.

Reference: 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 14:48:23 2003

Discussion: If ImmediateData is set to no and InitialR2T is set to no, then the initiator must not send unsolicited immediate data, but may send one unsolicited burst of Data-Out PDUs. If the Expected Data Transfer Length is higher than the FirstBurstLength (the negotiated maximum amount of unsolicited data the target will accept), the initiator MUST send the maximum amount of unsolicited data OR ONLY the immediate data, if any.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=No and InitialR2T=Yes
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.
· Transmit an R2T and wait for Data-out PDUs.

Observable Results:
· Verify that the Data Segment Length of the SCSI Command PDU = 0 and the Data Segment does not contain data of any kind.
· Verify that the W bit is set to 1 in the SCSI Command PDU.
· Verify that the last Data-out PDU of the sequence has the F bit set.

Possible Problems: If the initiator does not support ImmediateData, this item is not

testable

Test #16.2.3: SCSI Command Unsolicited Data

Purpose: To verify that the initiator follows the rules for transmitting Unsolicited data.

Reference: 9.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Tue Jan 14 13:48:00 2003

Discussion: If ImmediateData is set to no and InitialR2T is set to no, then the initiator must not send unsolicited immediate data, but may send one unsolicited burst of Data-Out PDUs. If the Expected Data Transfer Length is higher than the FirstBurstLength (the negotiated maximum amount of unsolicited data the target will accept), the initiator MUST send the maximum amount of unsolicited data OR ONLY the immediate data, if any.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=No and InitialR2T=No
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.

Observable Results:
· Verify that the Data Segment Length of the received SCSI Command PDU = 0 and the Data Segment does not contain data of any kind.
· Verify that the W bit is set to 1.

Possible Problems: None.

Test #16.2.4: SCSI Command Unsolicited Data

Purpose: To verify that the initiator follows the rules for transmitting Unsolicited data.

Reference: 10.3, 12.12, 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:15:54 2003

Discussion: If ImmediateData is set to yes and InitialR2T is set to no, then the initiator may send unsolicited immediate data and/or one unsolicited burst of Data-Out PDUs. If the Expected Data Transfer Length is higher than the FirstBurstLength (the negotiated maximum amount of unsolicited data the target will accept), the initiator MUST send the maximum amount of unsolicited data OR ONLY the immediate data, if any.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=Yes and InitialR2T=No.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.

Observable Results:
· Verify that if data is transmitted in the PDU, the Data Segment Length reflects the number of bytes of user data or parameters contained in the Data Segment.
· Verify that the W bit is set to 1.
· Verify that the DUT does not send more than FirstBurstLength of ImmediateData.
· Verify that the last Data-out PDU of the sequence has the F bit set to 1.

Possible Problems: If the initiator does not support ImmediateData, this item is not testable.

Test #16.3.1: SCSI Command F bit

Purpose: To verify that an iSCSI initiator sets the F bit properly when sending a SCSI Command PDU.

Reference: 10.3

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:16:03 2003

Discussion: The F bit is set to 1 when no unsolicited SCSI Data-Out PDUs follow this PDU. When F=1 for a write and if Expected Data Transfer Length is larger than the DataSegmentLength, the target may solicit additional data through R2T.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=No and InitialR2T=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.
· Transmit R2T to satisfy the Expected Data Transfer Length of the operation.

Observable Results:
· Verify that the Data Segment Length of the received SCSI Command PDU = 0, and the Data Segment contains no data of any type.
· Verify that no Data-out PDUs are transmitted by the initiator until the R2T is received from the Testing Station.
· Verify that the DUT sets the F bit in the last Data-out PDU of the sequence.

Possible Problems: None.

Test #16.4.1: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the presence of a CRC error.

Reference: 6.2.1, 10.4.7.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:16:31 2003

Discussion: Retry MUST NOT be used for reasons other than plugging command sequence gaps, and in particular, cannot be used for requesting PDU retransmissions from a target.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI Command from the DUT.
· Transmit a SCSI Response PDU with Service Response Code 0x00, and Status Code 0x02. The Data Segment should be formatted as follows, in order to indicate that a CRC error has taken place using sense data (the first two bytes indicate that the sense length is 24 bytes): 0x00 0x18 0x70 0x00 0x0b 0x00 0x00 0x00 0x00 0x10 0x00 0x00 0x00 0x00 0x47 0x05 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00.

Observable Results:
· Verify that the initiator does NOT simply proceed as though the response were positive, or retry the initial Command
· Verify that the initiator sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN. This will distinguish this Command from one that is being retired, since retry is inappropriate here.

Possible Problems: In all classes of Error Recovery the implementer has the choice of

deferring errors to the SCSI initiator.

Test #16.4.2: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the fact that a SNACK has been rejected.

Reference: 6.2.1, 10.4.7.2, 10.16

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:16:57 2003

Discussion: Retry MUST NOT be used for reasons other than plugging command sequence gaps, and in particular, cannot be used for requesting PDU retransmissions from a target. Any such PDU retransmission requests for a currently allegiant command in progress may be made using the SNACK mechanism. A target can reject a SNACK with a SCSI Response of status CHECK CONDITION and sense data indicating a SNACK was rejected. The SNACK is used by the initiator to request the retransmission of numbered-responses, data, or R2T PDUs from the target. The SNACK request indicates the numbered-responses or data "runs" whose retransmission is requested by the target, where the run starts with the first StatSN, DataSN, or R2TSN whose retransmission is requested and indicates the number of Status, Data, or R2T PDUs requested including the first.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· If possible, negotiate ErrorRecoveryLevel > 0
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI READ Command from the DUT
· Transmit the request amount of Data PDUs and a SCSI Response PDU. Do not transmit one Data PDU, so that not all of the data request was transmitted. However this should not be reflected in the U bit of the SCSI Response. Wait for a SNACK PDU to arrive from the initiator
· Transmit a Reject PDU Reason 0x03 , SNACK Reject.Transmit Reject to any other

received SNACK PDUs.

Observable Results:
· Verify that the initiator does NOT simply proceed as though it received all of the requested data.
· Verify that the DUT issues a SNACK if SNACK is supported.
· Verify that after the SNACK is rejected, the initiator sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN, thus this is not a 'retried' command.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #16.4.3: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the fact that an unexpected unsolicited data error has occurred.

Reference: 6.2.1, 10.4.7.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:05 2003

Discussion: Retry MUST NOT be used for reasons other than plugging command sequence gaps. If Status is Check Condition (0x02), then the Data Segment MUST contain sense data for the command.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session)
· Attach the keys InitialR2T=No; ImmediateData=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT
· If the DUT transmits any unsolicited data, transmit a SCSI Response PDU with Service Response Code 0x00, and Status Code 0x02. The Data Segment should be formatted as follows, in order to indicate that an unsolicited data error has taken place using sense data (the first two bytes indicate that the sense length is 24 bytes): 0x00 0x18 0x70 0x00 0x0b 0x00 0x00 0x00 0x00 0x10 0x00 0x00 0x00 0x00 0x0c 0x0c 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Observable Results:
· Verify that the initiator does NOT simply proceed as though the response were positive.
· Verify that the initiator sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #16.4.4: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the fact that a not enough unsolicited data error has occurred.

Reference: 6.2.1, 10.4.7.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:16 2003

Discussion: Retry MUST NOT be used for reasons other than plugging command sequence gaps. If Status is Check Condition (0x02), then the Data Segment MUST contain sense data for the command.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the key InitialR2T=No; ImmediateData=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI Command from the DUT
· If the DUT transmits any unsolicited data, the Testing Station should transmit a SCSI Response PDU with Service Response Code 0x00, and Status Code 0x02. The Data Segment should be formatted as follows, in order to indicate that an incorrect amount of data error has taken place using sense data (the first two bytes indicate that the sense length is 24 bytes): 0x00 0x18 0x70 0x00 0x0b 0x00 0x00 0x00 0x00 0x10 0x00 0x00 0x00 0x00 0x0c 0x0d 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00

Observable Results:
· Verify that the initiator does NOT simply proceed as though the response were positive, but rather attempts to perform the command again (i.e. another WRITE type command)
· Verify that the initiator sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator. This may result in the connection being closed.

Test #16.4.5: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the fact that the logical unit is busy.

Reference: 6.2.1, 10.4.7.2, SAM-2 clause 5.3.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:24 2003

Discussion: If the SCSI Response Status Code indicates that the logical unit is busy, the recommended initiator recovery action is not to retry, but to issue the command again at a later time.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI Command from the DUT
· Transmit a SCSI Response PDU with Service Response Code 0x00, and Status Code 0x08.

Observable Results:
· Verify that the initiator eventually sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #16.4.6: SCSI Command Target Failure

Purpose: To verify that the initiator responds correctly when a SCSI response is received which indicates the fact that another initiator is causing the logical unit to be busy.

Reference: 6.2.1, 10.4.7.2, SAM-2 clause 5.3.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:32 2003

Discussion: If the Status Code indicates that another initiator is keeping the logical unit busy (i.e. RESERVATON CONFLICT), the recommended initiator recovery action is to issue the command again at a later time.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI Command from the DUT
· Transmit a SCSI Response PDU with Service Response Code 0x00, and Status Code 0x18.

Observable Results:
· Verify that the initiator eventually sends a similar PDU, perhaps identical to the first PDU, but with a different CmdSN.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #16.5: SCSI Command ExpCmdSN

Purpose: To verify that the initiator does not attempt to transmit another SCSI Command, if the target transmits a SCSI Response which indicates that the target is not ready to accept new commands.

Reference: 10.4.9, 10.4.11

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:43 2003

Discussion: ExpCmdSN is a Sequence Number that the target iSCSI returns to the initiator to acknowledge command reception. It is used to update a local variable with the same name. An ExpCmdSN equal to MaxCmdSN+1 indicates that the target cannot accept new commands. When MaxCmdSN changes at the target while the target has no pending PDUs to convey this information to the initiator, it MUST generate a NOP-IN to carry the new MaxCmdSN.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.
· Transmit R2T to receive Data-out PDUs until the DUT sets the F bit = 1.
· Transmit a SCSI Response PDU with ExpCmdSN = MaxCmdSN + 1
· After 2 seconds transmit a NOP-In PDU, with MaxCmdSN = (MaxCmdSN of last SCSI Response) + 1.

Observable Results:
· Verify that the initiator does not transmit another SCSI Command PDU until it receives the NOP-In PDU from the Testing Station

Possible Problems: None.

Test #16.6: SCSI Command Expected Data Transfer Length

Purpose: To verify that the initiator utilizes the Expected Data Transfer Length field correctly.

Reference: 10.3.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:17:53 2003

Discussion: If the Expected Data Transfer Length for a write and the length of the immediate data part that follows the command (if any) are the same, then no more data PDUs are expected to follow. In this case, the F bit MUST be set to 1. If the Expected Data Transfer Length is higher than the FirstBurstLength (the negotiated maximum amount of unsolicited data the target will accept), the initiator MUST send the maximum amount of unsolicited data OR ONLY the immediate data, if any.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=Yes and InitialR2T=Yes
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY, TEST UNIT READY, READ-CAP from the DUT, transmit response data and status to each.
· Wait for a SCSI WRITE Command from the DUT.

Observable Results:
· Verify that if the Data Segment Length = Expected Data Transfer Length, the F bit is set to 1.
· Verify that if the Expected Data Transfer Length is greater than FirstBurstLength, the initiator sends only the immediate data portion of the expected data, waiting for R2T before transmitting the remainder of the data.

Possible Problems: If the initiator does not issue a WRITE command, this item is not testable.

Test #17.1: Logout

Purpose: To verify that an initiator does not attempt to transmit SCSI Commands after transmitting a Logout PDU.

Reference: 10.14

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:18:04 2003

Discussion: The Logout request is used to perform a controlled closing of a connection. An initiator MAY use a logout request to remove a connection from a session or to close an entire session. After sending the Logout request PDU, an initiator MUST NOT send any new iSCSI requests on the closing connection. If the Logout request is intended to close the session, new iSCSI requests MUST NOT be sent on any of the connections participating in the session.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the key ImmediateData=Yes and InitialR2T=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for any SCSI Command from the DUT.
· Transmit an Asynchronous Message PDU, indicating the target is requesting a logout.
· Wait for a Logout Request PDU from the DUT.
· Transmit a Logout Response PDU.

Observable Results:
· Verify that the initiator does not transmit any command PDUs after transmitting the Logout PDU.
· Verify that the initiator proceeds to terminate the connection (transmits TCP FIN) upon receiving the Logout Response PDU.
· Verify that the DUT attempts to reconnect to the original address of the Testing Station.

Possible Problems: None.

Test #18.1: Text Request Text Fields

Purpose: To verify that the initiator issues the Text Request PDU correctly .

Reference: 10.10, 10.10.1, 10.10.3, 10.10.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:03 2003

Discussion: The Text Request is provided to allow for the exchange of information and for future extensions. It permits the initiator to inform a target of its capabilities or to request some special operations. When the F bit is set to 1, indicates that this is the last or only text request in a sequence of Text Requests; otherwise, it indicates that more Text Requests will follow. The Initiator Task Tag is the initiator assigned identifier for this Text Request. If the command is sent as part of a sequence of text requests and responses, the Initiator Task Tag MUST be the same for all the requests within the sequence (similar to linked SCSI commands). The I bit for all requests in a sequence also MUST be the same.When the Target Transfer Tag is set to the reserved value 0xffffffff, it tells the target that this is a new request and the target resets any internal state associated with the Initiator Task Tag (resets the current negotiation state).

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request from the DUT, with one of the following: SendTargets=All, SendTargets= iSCSI TargetName, SendTargets=

Observable Results:
· Verify that the CmdSN of the Text Request is the same as that of the ExpCmdSN of the preceding Login Response PDU.
· Verify that the Target Transfer Tag is 0xFFFFFFFF.
· Verify that the Initiator Task Tag is unique.

· Verify that if this is the only or last Text Request in a sequence that the F bit is set to 1.

Possible Problems: None.

Test #18.2: Text Request Initiator Task Tag

Purpose: To verify that the initiator transmits an empty request if the target keys span text response boundaries.

Reference: 10.10.4, 10.11.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:17 2003

Discussion: The target sets the Target Transfer Tag in a text response to a value other than the reserved value 0xffffffff whenever it indicates that it has more data to send or more operations to perform that are associated with the specified Initiator Task Tag. It MUST do so whenever it sets the F bit to 0 in the response. By copying the Target Transfer Tag from the response to the next Text Request, the initiator tells the target to continue the operation for the specific Initiator Task Tag. The initiator MUST ignore the Target Transfer Tag in the Text Response when the F bit is set to 1. When a target has more work to do (e.g., cannot transfer all the remaining text data in a single Text Response or has to continue the negotiation) and has enough resources to proceed, it MUST set the Target Transfer Tag to a value other than the reserved value of 0xffffffff. Otherwise, the Target Transfer Tag MUST be set to 0xffffffff. When the Target Transfer Tag is not 0xffffffff, the LUN field may be significant. The initiator MUST copy the Target Transfer Tag and LUN in its next request to indicate that it wants the rest of the data.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request from the DUT with the SendTargets key.
· Transmit a Text Response to the DUT, with the following: LUN not equal to 0; Target Transfer Tag not equal to 0xFFFFFFFF; F bit = 0; If the SendTargets value is All|Nothing attach the target name key denoting the Testing station, followed by multiple

target address keys. If the SendTargets value = iSCSI Targetname of Testing Station attach one address for the named target, the F bit equal to 0 will indicate that more are to follow.

Observable Results:
· Verify that the initiator transmits a second Text Request, in order to receive the following Text Response.
· Verify that the second Text Request contains the same Initiator Task Tag as the first.
· Verify that the second Text Request contains the same Target Transfer Tag as the Text Response.
· Verify that the second Text Request contains the same LUN as the Text Response.

Possible Problems: None.

Test #18.3.1: Text Request Target Transfer Tag

Purpose: To verify that the initiator does not take recovery actions when the Target Transfer Tag is denoted incorrectly, if the F bit is set to 1.

Reference: 10.10.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:24 2003

Discussion: The target sets the Target Transfer Tag in a text response to a value other than the reserved value 0xffffffff whenever it indicates that it has more data to send or more operations to perform that are associated with the specified Initiator Task Tag. It MUST do so whenever it sets the F bit to 0 in the response. By copying the Target Transfer Tag from the response to the next Text Request, the initiator tells the target to continue the operation for the specific Initiator Task Tag. The initiator MUST ignore the Target Transfer Tag in the Text Response when the F bit is set to 1.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request with the F bti equal to 1, from the DUT with the SendTargets key.
· Transmit a Text Response to the DUT, with the following: Target Transfer Tag not equal to 0xFFFFFFFF; F bit = 1; If the SendTargets value is All|Nothing attach the target name key denoting the Testing station, followed by a target address key. If the SendTargets value = iSCSI Targetname of Testing Station attach one address for the named target,

Observable Results:
· Verify that the initiator proceeds to transmit a Logout PDU, as it normally would.
· Verify that the reason code for the Logout PDU is 0 ('close the session').

· Verify that the DUT ignores the Target Transfer tag provided in the Text Response.
· Verify that the Logout Request PDU has a unique Initiator Task Tag.

Possible Problems: The DUT may choose to simply disconnect without transitting a Logout Request.

Test #18.3.2: Text Request Target Transfer Tag

Purpose: To verify that the initiator does take recovery actions when the Target Transfer Tag is denoted incorrectly, if the F bit is set to 0.

Reference: 10.10.4, 10.11.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:35 2003

Discussion: The target sets the Target Transfer Tag in a text response to a value other than the reserved value 0xffffffff whenever it indicates that it has more data to send or more operations to perform that are associated with the specified Initiator Task Tag. It MUST do so whenever it sets the F bit to 0 in the response. When a target has more work to do (e.g., cannot transfer all the remaining text data in a single Text Response or has to continue the negotiation) and has enough resources to proceed, it MUST set the Target Transfer Tag to a value other than the reserved value of 0xffffffff. Otherwise, the Target Transfer Tag MUST be set to 0xffffffff.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request from the DUT with the SendTargets key.
· Transmit a Text Response to the DUT, with the following: LUN not equal to 0; Target Transfer Tag equal to 0xFFFFFFFF; F bit = 0; If the SendTargets value is All|Nothing attach the target name key denoting the Testing station, followed by multiple target address keys. If the SendTargets value = iSCSI Targetname of Testing Station attach one address for the named target, the F bit equal to 0 will indicate that more are to follow.

Observable Results:
· Verify that the initiator either transmits a Logout Request PDU, or simply disconnects.

Possible Problems: None.

Test #18.4: Text Request Other Parameters

Purpose: To verify that an initiator does not attempt to participate in the negotiation of other parameters during a discovery session.

Reference: 3.3, Appendix D p. 257

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:46 2003

Discussion: No Text keys other than Target Name and Target Address are permitted in the SendTargets response.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· During the Login Phase, do not transmit the MaxRecvPDULength key
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request from the DUT with the SendTargets key
· Transmit a Text Response to the DUT, with the following: Target Transfer Tag not equal to 0xFFFFFFFF; F bit = 1; If the SendTargets value is All|Nothing attach the target name key denoting the Testing station, followed by a target address key. If the SendTargets value = iSCSI Targetname of Testing Station attach one address for the named target. Also attach the MaxRecvDataSegmentLength=512 key=value pair.

Observable Results:
· Verify that the initiator proceeds to undergo session recovery (if ErrorRecoveryLevel=0) or connection recovery (if ErrorRecoveryLevel=2).

Possible Problems: None.

Test #18.5: Text Request Negotiation Reset

Purpose: To verify that an initiator responds appropriately to an operational parameter negotiation reset.

Reference: 5.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:52 2003

Discussion: A target may reset an operational parameter negotiation by answering a Text request with a Reject PDU. Neither the initiator nor the target should attempt to declare or negotiate a parameter more than once during any negotiation sequence without an intervening operational parameter negotiation reset.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Discovery session (i.e. not a Normal session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a Text Request from the DUT with the MaxRecvDataSegmentLength key
· Transmit a Reject PDU to the DUT

Observable Results:
· The DUT should not close the connection.
· Verify that the initiator attempts to renegotiate the MaxRecvDataSegmentLength key.

Possible Problems: The DUT can attempt to negotiate any appropriate key in the Text Request, not just MaxRecvDataSegmentLength. However, if the DUT does not attempt to declare or negotiate any keys during Full Feature Phase using the Text Request method, this test is not testable.

Test #19.1: Task Management Command CmdSN

Purpose: To verify that the CmdSN of the Task Management Command helps identify it with its associated task.

Reference: 3.2.2.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:23:59 2003

Discussion: The CmdSN of a given Task Management Command must be the same as that of the next non-immediate command after the task the target is intended to act on if that Task Management Command is marked for Immediate Delivery

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT. Transmit a Data-In response to the DUT
· Wait for TEST-UNIT READY from the DUT, transmit response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for any READ or WRITE Command. Do not transmit any response
· Wait for Task Management Command.

Observable Results:
· If the Task Management Command is set for immediate delivery, verify that the CmdSN of the Task Management Command is the same as that of the command which follows it. Otherwise verify that the CmdSN increases by one with the next commmand after the Task Management Command.

Possible Problems: The DUT may not transmit a Task Management Command which is not immediate. In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.2: Task Management LUN

Purpose: To verify that the LUN of the Task Management Command helps identify it with its associated task.

Reference: 10.5.3

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Thu Jun 19 16:21:09 2003

Discussion: The LUN field is required for functions that address a specific LU (ABORT TASK, CLEAR TASK SET, ABORT TASK SET, CLEAR ACA, LOGICAL UNIT RESET) and is reserved in all others.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT. Transmit a SCSI Response to the DUT
· Wait for TEST-UNIT READY from the DUT. Transmit response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for any READ or WRITE Command. Do not transmit response.
· Wait for Task Management Command such as ABORT TASK.

Observable Results:
· Verify that the LUN of the Task Management Command is the same as that of the READ or WRITE Command.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.3: Task Management RefCmdSN

Purpose: To verify that the RefCmdSN of the Task Management Command helps identify it with its associated task.

Reference: 10.5.5

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Thu Jun 19 16:21:20 2003

Discussion: If an ABORT TASK is issued for a task created by an immediate command then RefCmdSN MUST be that of the Task Management request itself (i.e. CmdSN and RefCmdSN are equal). For an ABORT TASK of a task created by non-immediate command RefCmdSN MUST be set to the CmdSN of the task identified by the Referenced Task Tag field. Targets must use this field as described in section 9.6.1 when the task identified by the Referenced Task Tag field is not with the target.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT. Transmit a SCSI Response to the DUT
· Wait for TEST-UNIT READY from the DUT. Transmit a response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for any READ or WRITE Command. Do not transmit any response.
· Wait for an ABORT TASK Task Management Command

Observable Results:
· Verify that the RefCmdSN of the Task Management Command is the same as CmdSN of the READ or WRITE Command.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.4.1: Task Management Abort Task Set

Purpose: To verify that an initiator continues to respond to all valid target transfer tags relating to the affected Task Set, even after the Abort Task Management request.

Reference: 10.5.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:24:25 2003

Discussion: For ABORT TASK SET and CLEAR TASK SET, the issuing initiator MUST continue to respond to all valid target transfer tags (received via R2T, Text Response, NOP-In, or SCSI Data-in PDUs) related to the affected task set, even after issuing the task management request. The issuing initiator SHOULD however terminate (i.e., by setting the F-bit to 1) these response sequences as quickly as possible.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT. Transmit a SCSI Response to the DUT
· Wait for TEST-UNIT READY from the DUT. Transmit a SCSI Response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for a SCSI WRITE Command. Do not transmit response
· Wait for Task Management Command with Abort Task Set Function Code 2, or Clear Task Set Function Code 4
· Transmit NOP-In PDU with the same TTT as the Task Management Command, and with CmdSN < CmdSN of Task Management Command

Observable Results:
· Verify that the DUT responds to the NOP-In PDU sent after the Task Management Command with a NOP-Out PDU.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.4.2: Task Management Abort Task Set

Purpose: To verify that an initiator continues to respond to all valid target transfer tags relating to the affected Task Set, even after the Abort Task Management request.

Reference: 10.5.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:24:37 2003

Discussion: For ABORT TASK SET and CLEAR TASK SET, the issuing initiator MUST continue to respond to all valid target transfer tags (received via R2T, Text Response, NOP-In, or SCSI Data-in PDUs) related to the affected task set, even after issuing the task management request. The issuing initiator SHOULD however terminate (i.e., by setting the F-bit to 1) these response sequences as quickly as possible.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT. Transmit a SCSI Response to the DUT
· Wait for TEST-UNIT READY from the DUT. Transmit a SCSI Response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for a READ Command. Do not transmit response
· Wait for Task Management Command with Abort Task Set Function Code 2, or Clear Task Set Function Code 4
· Transmit NOP-In PDU with the same TTT as the Task Management Command, and with CmdSN
· Transmit Data-In PDU with the same TTT and ITT as the original SCSI Command, the A bit set to 1.

Observable Results:
· Verify that the DUT responds to both of the PDUs sent after the Task Management

Command.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.4.3: Task Management Abort Task Set

Purpose: To verify that the initiator continues to process responses for tasks in the affected task set

Reference: 10.6.2

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:24:44 2003

Discussion: The execution of ABORT TASK SET and CLEAR TASK SET Task Management function requests consists of the following sequence of events in the specified order on each of the entities. The initiator issues ABORT TASK SET/CLEAR TASK SET request, continues to respond to each target transfer tag received for the affected task set, receives any responses for the tasks in the affected task set (may process them as usual because they are guaranteed to be valid), receives the task set management response, thus concluding all the tasks in the affected task set.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT.
· Transmit a Data-In response to the DUT.
· Wait for TEST-UNIT READY from the DUT, transmit response.
· Wait for READ CAP from the DUT, transmit response.
· Wait for any READ or WRITE Command.
· Do not transmit response
· Wait for Task Management Command with Abort Task Set Function Code 2, or Clear Task Set Function Code 4
· Transmit a response to the previously received READ or WRITE Command.
· Issue the appropriate Task Management Response, depending on the Request received.

Observable Results:
· Verify that the DUT does not Reject any of the received PDUs.

Possible Problems: In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #19.5: Task Management Task Reassign

Purpose: To verify that an initiator reassigns a task to a different connection if the present connection fails.

Reference: 6.2.2, 10.5.4

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Tue Jul 1 16:39:03 2003

Discussion: By issuing a "task reassign" task management request (Section 9.5.1 Function), the initiator signals its intent to continue an already active command (but with no current connection allegiance) as part of connection recovery. This means that a new connection allegiance is requested for the command, which seeks to associate it to the connection on which the task management request is being issued. Before the allegiance reassignment is attempted for a task, an implicit or explicit Logout with the reason code "remove the connection for recovery" (see section 9.14) MUST be successfully completed for the previous connection to which the task was allegiant. The Referenced Task Tag field is the Initiator Task Tag of the task to be aborted for the ABORT TASK function or reassigned for the TASK REASSIGN function. For all the other functions this field MUST be set to the reserved value 0xffffffff.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Start 2 connections from the DUT to the Testing Station.
· On each connection, wait for the initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· On each connection, during the Login Phase, negotiate ErrorRecoveryLevel=2
· On each connection, proceed through the Login Phase and in to Full Feature Phase operation.
· On each connection, wait for a SCSI-INQUIRY from the DUT. Transmit a SCSI Response to the DUT
· On each connection, wait for TEST-UNIT READY from the DUT. Transmit a response to the DUT.
· On each connection, wait for READ CAP from the DUT, transmit response.

· On each connection wait for any READ or WRITE Command. On the first connection only, do not transmit response. Transmit a valid response on the second connection.
· Wait for Task Management Command with Task Reassign Function Code 8 on the second connection.
· Transmit a Task Management Response on the second connection .
· On the second connection, transmit a response to the previously received READ or WRITE Command, either Data-in or R2T.

Observable Results:
· Verify that the ITT in the Task Management Command matches that of the READ or WRITE command.
· Verify that the connection is clearly reassigned

Possible Problems: If ErrorRecoveryLevel < 2, this item is not testable. In all classes of Error Recovery the implementer has the choice of deferring errors to the SCSI initiator.

Test #20.1: Asynchronous Message Logout Request

Purpose: To verify that both initiator and target respond in accordance with procedure as dictated by the protocol with regard to the asynchronous message event code which calls for logout.

Reference: 10.9.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:25:01 2003

Discussion: A target may request Logout via an Async Message. This Async Message MUST be sent on the same connection as the one requesting to be logged out. The initiator MUST honor this request by issuing a Logout as early as possible, but no later than Parameter3 seconds. Initiator MUST send a Logout with a reason code of "Close the connection" OR "Close the session" to close all the connections. Once this message is received, the initiator SHOULD NOT issue new iSCSI commands on the connection to be logged out. The target MAY reject any new I/O requests that it receives after this Message with the reason code "Waiting for Logout". If the initiator does not Logout in Parameter3 seconds, the target should send an Async PDU with iSCSI event code "Dropped the connection" if possible, or simply terminate the transport connection. Parameter1 and Parameter2 are reserved.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT
· Transmit a Data-In PDU to the DUT
· Wait for a TEST-UNIT READY from the DUT.
· Transmit a SCSI Response to the DUT.
· Wait for READ CAP from the DUT, transmit response.
· Wait for a READ or WRITE Command from the DUT

· Transmit an Asynchronous Message PDU with event code 1 and Parameter 3 set to 1.

Observable Results:
· Verify that the DUT responds within 1 second by transmitting a Logout request PDU, with reason code 0 or 1.
· Verify that the DUT does not transmit new Commands after transmitting the Logout request.

Possible Problems: None.

Test #20.2: Asynchronous Message Drop Connection

Purpose: To verify that both initiator and target respond in accordance with procedure as dictated by the protocol with regard to the asynchronous message event code which calls for a connection to be dropped.

Reference: 10.9.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:25:10 2003

Discussion: Async Message event code 2 is used by the target to indicate it will drop the connection. The Parameter1 field indicates the CID of the connection going to be dropped. The Parameter2 field (Time2Wait) indicates, in seconds, the minimum time to wait before attempting to reconnect or reassign. The Parameter3 field (Time2Retain) indicates the maximum time allowed to reassign commands after the initial wait (in Parameter2). If the initiator does not attempt to reconnect and/or reassign the outstanding commands within the time specified by Parameter3, or if Parameter3 is 0, the target will terminate all outstanding commands on this connection. In this case, no other responses should be expected from the target for the outstanding commands on this connection. A value of 0 for Parameter2 indicates that reconnect can be attempted immediately.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT
· Transmit a Data-In PDU to the DUT
· Wait for a TEST-UNIT READY from the DUT.
· Transmit a SCSI Response to the DUT
· Wait for READ CAP from the DUT, transmit response.
· Wait for a READ or WRITE Command from the DUT
· Transmit an Asynchronous Message PDU with event code 2, Parameter 1 set to CID of

this connection, Parameter 2 set to 0, and Parameter 3 set to 1

Observable Results:
· Verify that the DUT disconnects from the Testing Station.
· Verify that the DUT attempts to reconnect within 1 second of connection termination.

Possible Problems: None.

Test #20.3: Asynchronous Message Drop All Connections in Session

Purpose: To verify that both initiator and target respond in accordance with procedure as dictated by the protocol with regard to the asynchronous message event code which calls for a connection to be dropped.

Reference: 10.9.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:25:17 2003

Discussion: Async Message event code 3 is used by the target to indicate it will drop all the connections of this session. Parameter1 field is reserved. The Parameter2 field (Time2Wait) indicates, in seconds, the minimum time to wait before attempting to reconnect. The Parameter3 field (Time2Retain) indicates the maximum time allowed to reassign commands after the initial wait (in Parameter2). If the initiator does not attempt to reconnect and/or reassign the outstanding commands within the time specified by Parameter3, or if Parameter3 is 0, the session is terminated. In this case, the target will terminate all outstanding commands in this session; no other responses should be expected from the target for the outstanding commands in this session. A value of 0 for Parameter2 indicates that reconnect can be attempted immediately.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Start 2 connections within the same session from the DUT to the Testing Station.
· On each connection, wait for initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· On each connection, proceed through the Login Phase and in to Full Feature Phase operation.
· On each connection, wait for a SCSI-INQUIRY from the DUT
· On each connection, transmit a Data-In PDU to the DUT
· On each connection, wait for a TEST-UNIT READY from the DUT.
· On each connection, transmit a SCSI Response to the DUT.
· On each connection, wait for READ CAP from the DUT, transmit response.
· On each connection, wait for a READ or WRITE Command from the DUT

· On one connection only transmit an Asynchronous Message PDU with event code 3, Parameter 2 set to 1, and Parameter 3 set to 2

Observable Results:
· Verify that the DUT closes all connections within the session.
· Verify that the DUT waits for 1 second, but for no longer than 2 seconds, before attempting to reestablish connection.

Possible Problems: None.

Test #20.4: Asynchronous Request Negotiation

Purpose: To verify that an initiator responds to a request by an iSCSI target for parameter negotiation.

Reference: 10.9.1

Resource Requirements: A Test Generator tool capable of producing iSCSI PDUs and transporting them over a TCP connection.

Last Modification: Mon May 19 15:25:22 2003

Discussion: Async Event code 4 is for when the target requests parameter negotiation on this connection. The initiator MUST honor this request by issuing a Text Request (that can be empty) on the same connection as early as possible, but no later than Parameter3 seconds, unless a Text Request is already pending on the connection, or by issuing a Logout Request. If the initiator does not issue a Text Request the target may reissue the Asynchronous Message requesting parameter negotiation.

Test Setup: The DUT and Test Station pair should be able to make a TCP connection.

Procedure:
· Wait for a connection to the Testing Station from the DUT.
· Wait for initial login request from the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Wait for a SCSI-INQUIRY from the DUT.
· Transmit a Data-In PDU to the DUT.
· Wait for a TEST-UNIT READY from the DUT.
· Transmit a SCSI Response to the DUT.
· Wait for READ CAP from the DUT, transmit response.
· Wait for a READ or WRITE Command from the DUT.
· Transmit an Asynchronous Message PDU with event code 4, and Parameter 3 set to 3.

Observable Results:
· Verify that the DUT transmits a Text Request within 3 seconds of receiving the Asynchronous Message from the Testing Station.

Possible Problems: None.