iSCSI Consortium

Full Feature Phase Test Suite For iSCSI Targets

Version 0.1

Last Update: September 16, 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
Claire Kraft 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

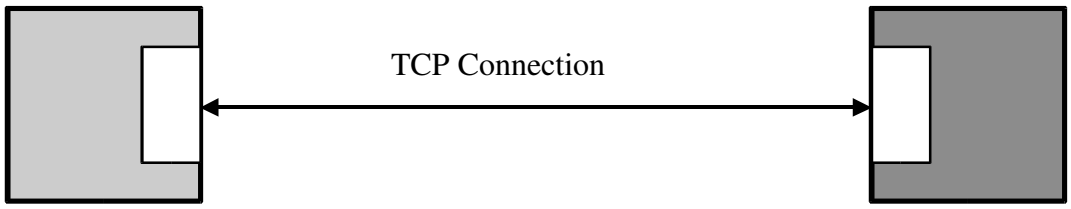
The following documents are referenced in this text:

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

MODIFICATION RECORD 2
ACKNOWLEDGMENTS 3
INTRODUCTION 4
REFERENCES 6
TEST SETUPS 7
TABLE OF CONTENTS 8
Test #1.1: MaxCmdSN-ExpCmdSN 11
Test #1.2: MaxCmdSN-ExpCmdSN 12
Test #1.3: MaxCmdSN-ExpCmdSN 14
Test #2.1: StatSN 15
Test #3.1: DataSN 16
Test #4.1: R2TSN 17
Test #5.1: Command Retry 18
Test #6.1: Connection Allegiance 20
Test #7.1: Unsolicited Data Error 22
Test #7.2: Unsolicited Data Error 23
Test #8.1: Target Transfer Tag 24
Test #8.2: Target Transfer Tag 26
Test #9.1: Data-in Status 27
Test #9.2: Data-in F bit 28
Test #9.3.1: Data-in DataSegmentLength 29
Test #9.3.2: Data-in DataSegmentLength 30
Test #9.3.3: Data-in DataSegmentLength 31
Test #9.4: Data-in DataSN 32
Test #9.5: Data-in Buffer Offset 33
Test #10.1: R2T 34
Test #11.1: Task Management Command Immediate and Non-Immediate 36
Test #11.2.1: Task Management Response Task Reassign 38
Test #11.2.2: Task Management Response Task Reassign 40
Test #11.2.3: Task Management Response Task Reassign 42
Test #11.3.1: Task Management Response 44
Test #11.3.2: Task Management Response 46
Test #11.3.3: Task Management Response 48
Test #11.4: Task Management Response Target Warm Reset 49
Test #11.5: Task Management Response Target Cold Reset 51
Test #12.1: SNACK-R2T Received 53
Test #12.2: SNACK-Data Received 55
Test #12.3: SNACK-Data Run Received 57

Test #12.4: SNACK Received **59**
Test #13.1: Logout Response **61**
Test #13.2: Logout Response **62**
Test #13.3: Logout Response **63**
Test #14.1: Reject **64**
Test #14.2: Reject **66**
Test #14.3: Reject **68**
Test #15.1.1: NOP-In Ping Response **70**
Test #15.1.2: NOP-In Ping Response **71**
Test #15.1.3: NOP-In Ping Response **72**
Test #15.1.4: NOP-In Ping Response **73**
Test #15.1.5: NOP-In Ping Response **74**
Test #15.1.6: NOP-In Ping Response **75**
Test #15.2: NOP-In Ping Request **76**
Test #15.3: NOP-In Confirm ExpCmdSN **77**
Test #16.1: SCSI Response Residual Underflow **79**
Test #16.2: SCSI Response Residual Overflow **80**
Test #16.3.1: SCSI Response Unsolicited Data **81**
Test #16.3.2: SCSI Response Unsolicited Data **83**
Test #16.3.3: SCSI Response Unsolicited Data **85**
Test #16.3.4: SCSI Response Unsolicited Data **86**
Test #16.3.5: SCSI Response Unsolicited Data **87**
Test #16.3.6: SCSI Response Unexpected Data **88**
Test #16.3.7: SCSI Response Bad OpCode **89**
Test #16.4.1: SCSI Response Retry **90**
Test #16.4.2: SCSI Response Retry **92**
Test #16.5: SCSI Response Error Detection **94**
Test #17.1.1: Text Response Text Fields **96**
Test #17.1.2: Text Response Text Fields **97**
Test #17.2.1: Text Response F bit **99**
Test #17.2.2: Text Response F bit **100**
Test #17.3.1: Text Response SendTargets Response **101**
Test #17.3.2: Text Response SendTargets Response **103**
Test #17.3.3: Text Response SendTargets Response **105**
Test #17.3.4: Text Response SendTargets Response **106**
Test #17.3.5: Text Response SendTargets Response **108**
Test #17.4.1: Text Response Other Parameters **110**
Test #17.4.2: Text Response Other Parameters **111**
Test #17.5: Text Response Initiator Task Tag **112**

Test #17.6: Text Response Negotiate Once **113**
Test #17.7.1: Text Response Negotiation Reset **115**
Test #17.7.2: Text Response Negotiation Failure **117**
Test #17.7.3: Text Response Negotiation Failure **119**
Test #17.8.1: Text Response C bit F bit **121**
Test #17.8.2: Text Response C bit F bit **122**
Test #17.8.3: Text Response C bit F bit **124**
Test #18.1.1: Header Digest Error Received **126**
Test #18.2.1: Data Digest Error Received Command PDU **127**
Test #18.2.2: Data Digest Error Received Data PDU **128**
Test #19.1: EDTL Check **129**

Test #1.1: MaxCmdSN-ExpCmdSN

Purpose: To verify that an iSCSI target offers appropriate values for MaxCmdSN and ExpCmdSN.

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 11:03:00 2003

Discussion: ExpCmdSN and MaxCmdSN are derived from target-to-initiator PDU fields. An iSCSI target MUST NOT transmit a MaxCmdSN that is less than the last ExpCmdSN-1.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 10 WRITE commands, each to write 1024 bytes of data to the target. For each command wait for R2T from the target then transmit a Data-in PDU with 1024 bytes of data. Wait for response data and status from the target.

Observable Results:
· Verify that the target never transmits a MaxCmdSN value that is less than the last ExpCmdSN-1.

Possible Problems: None.

Test #1.2: MaxCmdSN-ExpCmdSN

Purpose: To verify that an iSCSI target ignores any non-immediate commands with a CmdSN outside of the range specified by MaxCmdSN to ExpCmdSN-1.

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 11:03:08 2003

Discussion: 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 the range or non-immediate duplicates within the range. iSCSI initiators and targets MUST support the command numbering scheme.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 10 WRITE commands, each to write 1024 bytes of data to the target. For each command wait for R2T from the target then transmit a Data-in PDU with 1024 bytes of data. Wait for response data and status from the target. The seventh one of these WRITE commands should have a CmdSN value which is out of the range described by MaxCmdSN and ExpCmdSN-1.

Observable Results:
· Verify that the target ignores the command with the invalid CmdSN value. Verify that the target still properly handles the subsequently received WRITE commands by transmitting response data.

Possible Problems: None.

Test #1.3: MaxCmdSN-ExpCmdSN

Purpose: To verify that an iSCSI target ignores any non-immediate commands with a duplicate CmdSN inside of the range specified by MaxCmdSN to ExpCmdSN-1.

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 11:03:15 2003

Discussion: For non-immediate commands, the CmdSN field can take any value from ExpCmdSN to MaxCmdSN. The target MUST silently ignore any non-immediate command outside of the range or non-immediate duplicates within the range.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 10 WRITE commands, each to write 1024 bytes of data to the target. For each command wait for R2T from the target then transmit a Data-in PDU with 1024 bytes of data. Wait for response data and status from the target. The seventh one of these WRITE commands should have a CmdSN value which is the same as that for the sixth WRITE command.

Observable Results:
· Verify that the target ignores the command with the duplicate CmdSN value. Verify that the target still properly handles the subsequently received WRITE commands.

Possible Problems: None.

Test #2.1: StatSN

Purpose: To verify that an iSCSI target follows the rules regarding StatSN properly.

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 11:04:08 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. Status numbering starts with the Login response to the first Login request of the connection. The Login response includes an initial value for status numbering (any initial value is valid).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 10 WRITE commands, each to write 1024 bytes of data to the target. For each command wait for R2T from the target then transmit a Data-in PDU with 1024 bytes of data. Wait for response data and status from the target.

Observable Results:
· Verify that the target properly increments StatSN with each response throughout the Login Phase and into Full Feature Phase operation.

Possible Problems: None.

Test #3.1: DataSN

Purpose: To verify that an iSCSI target follows the rules regarding DataSN for SCSI-In data properly.

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 11:06:13 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 input (read) data PDUs, DataSN starts with 0 for the first data PDU of an input command and advances by 1 for each subsequent data PDU. 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 input (read) or bi-directional Data-In PDUs, the DataSN is the input PDU number within the data transfer for the command identified by the Initiator Task Tag. 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:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 3 READ-10 commands to the DUT each with a different InitiatorTaskTag. Wait for the DUT to respond with SCSI Data In PDUs.

Observable Results:
· Verify that for each data sequence, the DataSN field in the Data In PDUs began with 0, and was incremented by 1 for each subsequent Data In PDU.

Possible Problems: None.

Test #4.1: R2TSN

Purpose: To verify that an iSCSI target follows the rules regarding DataSN for SCSI-In data properly.

Reference: 3.2.2.3, 10.8.2

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

Last Modification: Mon May 19 11:06:25 2003

Discussion: R2T PDUs transferred as part of some command execution, MUST be sequenced. R2T's are sequenced per command. The first R2T has an R2TSN of 0, and this is advanced by 1 for every subsequent R2T. R2TSN is the R2T PDU input PDU number within the command identified by the Initiator Task Tag.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a series of 3 WRITE commands to the DUT each with a different InitiatorTaskTag. Wait for the DUT to solicit data with R2T PDUs.

Observable Results:
· Verify that for each data sequence, the R2TSN field in the R2T PDUs began with 0, and was incremented by 1 for each subsequent R2T PDU.

Possible Problems: None.

Test #5.1: Command Retry

Purpose: To verify that an iSCSI target accepts and responds properly to a retried command.

Reference: 3.2.2.1, 6.2.1, 6.7, 10.16.1

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

Last Modification: Tue Jun 3 09:23:56 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.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Negotiate ImmediateData=-Yes and ErrorRecoveryLevel > 0, DataDigest=CRC32C.
· Complete a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command with valid ITT and CmdSN to the DUT with a Data Digest error. The DUT must transmit a Reject PDU with reason code of Data-Digest Error. This will create a gap in the CmdSN on the target side.
· The Testing Station should transmit a WRITE Command identical to the previous WRITE command, in order to fill the gap in CmdSN. This should appear as a retried command to the target.

Observable Results:
· Verify that the DUT properly sends Reject in response to the command PDU with the

Data-Digest Error.
· Verify that the retried command completes with Status=GOOD.

Possible Problems: Support for Command Retry is not is mandatory. Upon detecting a data digest error the iSCSI target may choose to escalate the error to session recovery. If the DUT does not support Command Retry this item is not testable. If the DUT transmits Reject to the Data-Digest error, and does not close the connection, this should be seen as the DUT attempted to support Command Retry. In this case if the DUT does not accept the retried command this test would be failed.

Test #6.1: Connection Allegiance

Purpose: To verify that an iSCSI target transmits data and status response PDUs on the same connection as the request PDU was received.

Reference: 3.2.4.1

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

Last Modification: Mon May 19 11:07:57 2003

Discussion: For any iSCSI request issued over a TCP connection, the corresponding response and/or other related PDU(s) MUST be sent over the same connection. We call this "connection allegiance". If the original connection fails before the command is completed, the connection allegiance of the command may be explicitly reassigned to a different transport connection. Thus, if an initiator issues a READ command, the target MUST send the requested data, if any, followed by the status to the initiator over the same TCP connection that was used to deliver the SCSI command. If an initiator issues a WRITE command, the initiator MUST send the data, if any, for that command over the same TCP connection that was used to deliver the SCSI command. The target MUST return Ready To Transfer (R2T), if any, and the status over the same TCP connection that was used to deliver the SCSI command. Retransmission requests (SNACK PDUs) and the data and status that they generate MUST also use the same connection.

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

Procedure:
· Start two connections from the Testing Station to the iSCSI target being tested.
· On each connection perform a standard login and proceed to the Full Feature Phase.
· On each connection issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· On each connection issue a TEST-UNIT-READY to the DUT, wait for response.
· On each connection issue a READ-CAP to the DUT, wait for response data and status.
· On each connection issue a WRITE command to the DUT. Wait for the DUT to solicit data with R2T.
· On each connection issue a READ command to the DUT. Wait for Data-in PDUs from the DUT.

Observable Results:

· Verify that for each SCSI request transmitted by the Testing Station, the DUT transmitted response, Data-in, and R2T PDUs on the same connection.

Possible Problems: None.

Test #7.1: Unsolicited Data Error

Purpose: To verify that an iSCSI target recognizes the error of receiving unsolicited data while in R2T mode.

Reference: 3.2.4.2, 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 11:20:36 2003

Discussion: It is considered an error for an initiator to send unsolicited data PDUs to a target that operates in R2T mode.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Negotiate for InitialR2T=Yes and ImmediateData=No.
· Attempt to negotiate ErrorRecoveryLevel = 1.
· Complete a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT. Wait for the DUT to solicit data.
· Transmit a Data-Out PDU with a DataSegmentLength greater than the DesiredDataTransferLength offered by the DUT in the R2T PDU.

Observable Results:
· Verify that the target transmitted a SCSI response of status CHECK CONDITION or disconnected. The target may also transmit an Async Message requesting Logout.

Possible Problems: None.

Test #7.2: Unsolicited Data Error

Purpose: To verify that an iSCSI target recognizes the error of receiving too much unsolicited data.

Reference: 3.2.4.2, 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 11:20:47 2003

Discussion: It is considered 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:
· Connect the Testing Station to the iSCSI target being tested.
· Negotiate for ImmediateData=No, InitialR2T=No, FirstBurstLength=512.
· Attempt to negotiate ErrorRecoveryLevel = 1.
· Complete a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT.
· Transmit a Data-out PDU with more than 512 bytes of data. Depending on the MaxRecvDataSegmentLength (if declared) of the target this may require more than 1 PDU.

Observable Results:
· Verify that the target transmitted a SCSI response of status CHECK CONDITION. The target may also transmit an Async Message requesting Logout.

Possible Problems: None.

Test #8.1: Target Transfer Tag

Purpose: To verify that an iSCSI target supplies a Target Transfer Tag if available.

Reference: 3.2.4.3, 10.7.4, 10.8.5

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

Last Modification: Mon May 19 11:21:03 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. The target assigns its own tag to each R2T request that it sends to the initiator. This tag can be used by the target to easily identify the data it receives. The Target Transfer Tag and LUN are copied in the outgoing data PDUs and are only used by the target. There is no protocol rule about the Target Transfer Tag except that the value 0xffffffff is reserved and MUST NOT be sent by a target in an R2T. 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 a TCP connection.

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Negotiate for InitialR2T=No and ImmediateData=No.
· Attempt to negotiate ErrorRecoveryLevel = 1.
· Complete a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT. Wait for the DUT to solicit data with an R2T PDU.

Observable Results:
· Verify that the R2T PDU transmitted by the DUT had the Target Transfer Tag field set, not the reserved value of 0xffffffff.
· Verify that that LUN field was set to a valid value and was consistent with the LUN of the 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 get the target to transmit an R2T with the Target Transfer Tag field set.

Test #8.2: Target Transfer Tag

Purpose: To verify that an iSCSI target supplies a Target Transfer Tag if available.

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: Mon May 19 11:21:11 2003

Discussion: On incoming data, the Target Transfer Tag and LUN MUST be provided by the target if the A bit is set to 1; otherwise they are reserved. 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.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Negotiate for InitialR2T=No and ImmediateData=No.
· Attempt to negotiate ErrorRecoveryLevel = 1.
· Complete a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT. Wait for the DUT to transmit a Data-in PDU.

Observable Results:
· Verify that, unless the A bit is set, the Data-in PDU transmitted by the DUT had the Target Transfer Tag field set to the reserved value of 0xffffffff, and that the LUN field is also reserved.

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 get the target to transmit a Data-in PDU with the Target Transfer Tag field set.

Test #9.1: Data-in Status

Purpose: To verify that an iSCSI properly formats command status if it chooses to support sending command status with 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 11:21:24 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 a TCP connection.

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT. Wait for the DUT to transmit a Data-in PDU.

Observable Results:
· Verify that the once the command is complete, if the target choose to include Status with the Data-in PDU it set the S flag bit as well as the F bit.
· Verify that the DUT only includes Status if the command did not end with an exception. The status of the command must be GOOD, CONDITION MET, or INTERMEDIATE CONDITION MET.

Possible Problems: Some targets may not support sending status in a Data-in PDU.

Test #9.2: Data-in F bit

Purpose: To see that an iSCSI target properly sets the F bit when transmitting Data-in PDUs.

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 11:21:30 2003

Discussion: For incoming data, this bit is 1 for the last input (read) data PDU of a sequence. Input can be split into several sequences, each having its own F bit.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT.

Observable Results:
· Verify that the target sets the F bit to 1 in the last Data-in PDU of the sequence.

Possible Problems: None.

Test #9.3.1: Data-in DataSegmentLength

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

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 11:21:39 2003

Discussion: DataSegmentLength MUST not exceed MaxRecvDataSegmentLength for the direction it is sent.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· During login the Testing Station should declare a value MaxRecvDataSegmentLength of 512 bytes.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT.

Observable Results:
· Verify that the target does not transmit a Data-in PDU with a DataSegmentLength greater than the MaxRecvDataSegmentLength that was declared by the Testing Station.

Possible Problems: None.

Test #9.3.2: Data-in DataSegmentLength

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

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 11:21:45 2003

Discussion: The DataSegmentLength is 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.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT.
· Once an R2T is received transmit a Data-out PDU with DataSegmentLength=0.

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

Possible Problems: None.

Test #9.3.3: Data-in DataSegmentLength

Purpose: To see that an iSCSI target properly fills each Data Segment in a Data-in PDU to an integer number of 4 byte words.

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 11:21:55 2003

Discussion: 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 a TCP connection.

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT.

Observable Results:
· Verify that the DUT fills each Data Segment in the sequence to an integer number of 4 byte words. This is not necessary for final Data-in PDU where the F bit is set to 1.

Possible Problems: None.

Test #9.4: Data-in DataSN

Purpose: To see that an iSCSI target properly sets the DataSN field in a 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 11:22:08 2003

Discussion: For input (read) or bidirectional Data-In PDUs, the DataSN is the input PDU number within the data transfer for the command identified by the Initiator Task Tag.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 512.
· Complete login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT.
· Wait for Data-in PDUs from the DUT.

Observable Results:
· Verify that the DUT sets the DataSN for every transmitted Data-in PDU, starting with 0 and incrementing with each Data PDU.

Possible Problems: None.

Test #9.5: Data-in Buffer Offset

Purpose: To see that an iSCSI target properly sets the Buffer Offset field in a Data-out PDUs.

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 11:22:16 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 a TCP connection.

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes, DataPDUInOrder=Yes, DataSequenceInOrder=Yes.
· Declare a MaxRecvDataSegmentLength of 512.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a READ command to the DUT.
· Wait for Data-in PDUs from the DUT.

Observable Results:
· Verify that the DUT sets the Buffer Offset field accurately, and that the Buffer Offset increases with each Data-in PDU for the command.

Possible Problems: None.

Test #10.1: R2T

Purpose: To see that an iSCSI target properly builds an R2T PDU.

Reference: 10.8, 10.8.1, 10.8.2, 10.8.3, 10.8.4

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

Last Modification: Mon May 19 12:19:37 2003

Discussion: When an initiator has submitted a SCSI Command with data that passes from the initiator to the target (WRITE), the target may specify which blocks of data it is ready to receive. The target may request that the data blocks be delivered in whichever order is convenient for the target at that particular instant. This information is passed from the target to the initiator in the Ready To Transfer (R2T) PDU. For this PDU TotalAHSLength and DataSegmentLength MUST be 0. R2TSN is the R2T PDU input PDU number within the command identified by the Initiator Task Tag.The StatSN field will contain the next StatSN. The StatSN for this connection is not advanced after this PDU is sent. The Desired Data Transfer Length MUST NOT be 0 and MUST not exceed MaxBurstLength.DataSequenceInOrder governs the buffer offset ordering in consecutive R2Ts. If DataSequenceInOrder is Yes, then consecutive R2Ts MUST refer to continuous non-overlapping ranges except for Recovery-R2Ts.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate/declare the following parameters: ImmediateData=No; InitialR2T=Yes; DataSequenceInOrder=Yes; MaxRecvDataSegmentLength of 1024, MaxBurstLength=512, FirstBurstLength=512.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT. InitiatorTaskTag = 0xA2A2A2A2
· Wait for R2T from the DUT.

Observable Results:
· If the DUT chooses to transmit a series of R2T PDUs, verify that consecutive R2Ts refer to continuous non-overlapping ranges.

· Verify that the DUT uses the InitiatorTaskTag provided by the Testing Station.
· Verify that the R2TSN field starts with 0 and is incremented for each R2T with the same Initiator Task Tag, and that the Target Transfer Tag is set to anything but 0xFFFFFFFF.
· Verify that the StatSN field is set, but the StatSN does not advance.
· Verify that the Desired Data Transfer Length field is set to a value greater than zero and less than or equal to MaxBurstLength.

Possible Problems: None.

Test #11.1: Task Management Command Immediate and Non-Immediate

Purpose: To verify that an iSCSI target can handle one immediate task management command and one immediate non-task-management iSCSI request per connection at any time.

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: Tue Sep 16 07:53:45 2003

Discussion: An iSCSI target MUST be able to handle at least one immediate task management command and one immediate non-task-management iSCSI request per connection at anytime.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT, wait for response data and status.
· Issue a WRITE command to the DUT, wait for R2T but do not respond to that R2T with data yet.
· Issue a second WRITE set for Immediate Delivery, (I bit is set). This is an immediate non-task management iSCSI request.
· Issue a Task Management command set for immediate delivery to the DUT to ABORT TASK the first WRITE. This is an immediate task-management command. For the ABORT TASK function, the Testing Station MUST always set RefCmdSN to the CmdSN of the task identified by the Initiator Task Tag field.
· Issue a Data-Out PDU to satisfy the R2T issued by the DUT for the first WRITE command.
· Issue Data-Outs for the second WRITE command until the command is complete.

Observable Results:
· Verify that the DUT responds to the Task Management Request with a Task Management Response indicating that the task was completed.
· Verify that the DUT does not issue R2T or status for the first WRITE command after

the Task Management Request is received.
· Verify that the DUT issues R2T for the second WRITE command, and that this WRITE command completes with status GOOD.

Possible Problems: None.

Test #11.2.1: Task Management Response Task Reassign

Purpose: To verify that when a connection is reassigned for a task which has not lost connection allegiance, the proper Task Management Response is transmitted.

Reference: 6.2.2, 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:20:14 2003

Discussion: 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 multiple TCP connections.

Procedure:
·Start 2 connections from the Testing Station to the iSCSI target being tested.
·On each connection transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·On each connection negotiate ErrorRecoveryLevel=2.
·On each connection proceed through the Login Phase and in to Full Feature Phase operation.
·On each connection transmit a SCSI-INQUIRY to the DUT. Wait for a response and data from the DUT
·On each connection transmit a TEST-UNIT READY to the DUT. Wait for a response from the DUT
·On each connection transmit a READ-CAP to the DUT. Wait for response and data from the DUT
·On one connection, transmit a READ Command to the DUT.
·After the first Data-in PDU is received, transmit a Logout Request. Wait for a Logout Response from the DUT.
·On the second connection, transmit an immediate Task Management Command with

Function Code 8.

Observable Results:
·Verify that the DUT responds using a Task Management response with a valid response code on the same connection that the Task Management Request was sent on.

Possible Problems: If ErrorRecoveryLevel < 2, this item cannot be tested. To indicate that it does not support Task Reassignment the DUT should reply to the Task Management Request with a Task Management response code 4, Task allegiance reassignment not supported.

Test #11.2.2: Task Management Response Task Reassign

Purpose: To verify that when a connection is dropped, command execution is suspended until the Task Reassign Task Management Command is transmitted in order to establish a new allegiance.

Reference: 6.2.2, 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:20:44 2003

Discussion: 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 multiple TCP connections.

Procedure:
·Start 2 connections from the Testing Station to the iSCSI target being tested.
·On each connection transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·On each connection negotiate ErrorRecoveryLevel=2.
·On each connection proceed through the Login Phase and in to Full Feature Phase operation.
·On each connection transmit a SCSI-INQUIRY to the DUT. Wait for a response and data from the DUT
·On each connection transmit a TEST-UNIT READY to the DUT. Wait for a response from the DUT
·On each connection transmit a READ-CAP to the DUT. Wait for response and data from the DUT
·On one connection, transmit a READ Command to the DUT.
·On the second connection, transmit an immediate Task Management Command with Function Code 8.

Observable Results:
·Verify that the DUT sends a Task Management Response with response code 3 = task still allegiant.

Possible Problems: If ErrorRecoveryLevel < 2, this item cannot be tested. To indicate that it does not support Task Reassignment the DUT should reply to the Task Management Request with a Task Management response code 4, Task allegiance reassignment not supported.

Test #11.2.3: Task Management Response Task Reassign

Purpose: To verify that when a connection is dropped, connection allegiance may be assigned to a new connection, and that the corresponding task reassign is only received by the target after the original connection is logged out.

Reference: 6.2.2, 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:20:37 2003

Discussion: 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 multiple TCP connections.

Procedure:
·Start 2 connections from the Testing Station to the iSCSI target being tested.
·On each connection transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session). On each connection negotiate ErrorRecoveryLevel=2.
·On each connection proceed through the Login Phase and in to Full Feature Phase operation.
·On each connection transmit a SCSI-INQUIRY to the DUT. Wait for a response and data from the DUT
·On each connection transmit a TEST-UNIT READY to the DUT. Wait for a response from the DUT
·On each connection transmit a READ-CAP to the DUT. Wait for response and data from the DUT
·On one connection, transmit a READ Command to the DUT. After the first Data-in PDU is received, drop the connection. This is an implicit Logout Request.
·On the second connection, transmit an immediate Task Management Command with Function Code 8.

Observable Results:
·Verify that the DUT transmits a Task Management Response with response code 0 and begins transmitting Data-in PDU for the previous READ Command. Other task

management response reason codes are permissible. If the target does not respond with response code 0, then it cannot be expected to resume operations on the second connection.

Possible Problems: If ErrorRecoveryLevel < 2, this item cannot be tested. To indicate that it does not support Task Reassignment the DUT should reply to the Task Management Request with a Task Management response code 4, Task allegiance reassignment not supported.

Test #11.3.1: Task Management Response

Purpose: To verify that the target transmits a response when it has completed the assigned task.

Reference: 3.5.1.4, 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:38:47 2003

Discussion: The Task Management function response carries an indication of function completion for a Task Management function request including how it completed (response and qualifier) and additional information for failure responses. After the Task Management response indicates Task Management function completion, the initiator will not receive any additional responses from the affected tasks.

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

Procedure:
·Connect the Testing Station to the iSCSI target being tested.
·Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·Negotiate ErrorRecoveryLevel >= 1 if possible.
·Proceed through the Login Phase and in to Full Feature Phase operation.
·Transmit a SCSI-INQUIRY to the DUT. Wait for response and data from the DUT .
·Transmit a TEST-UNIT READY to the DUT. Wait for response from the DUT
·Transmit a READ Command to the DUT.
·Transmit an immediate Task Management Command with Function Code 2, ABORT TASK SET.
·Transmit a non-immediate No-Op Out command to the DUT with the same CmdSN as that of the Task Management Request.

Observable Results:
·Verify that the DUT responds to the Task Management Command.
·Verify that if the DUT transmits Data-in and response PDUs to the READ Command, it does so before transmitting the response to the Task Management Command.
·Verify that no SCSI responses or Data-In PDUs from the aborted command are transmitted after the Task Management Response
· Verify that the DUT responds to the received NOP-Out with a NOP-In.

Possible Problems: If ErrorRecoveryLevel = 0, the target may choose to drop the connection in order to complete the ABORT TASK SET command.

Test #11.3.2: Task Management Response

Purpose: To verify that the target transmits a response when it has completed the assigned task.

Reference: 3.5.1.4, 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:39:02 2003

Discussion: The Task Management function response carries an indication of function completion for a Task Management function request including how it completed (response and qualifier) and additional information for failure responses. After the Task Management response indicates Task Management function completion, the initiator will not receive any additional responses from the affected tasks.

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

Procedure:
·Connect the Testing Station to the iSCSI target being tested.
·Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Open a second connection from the Testing Station to the DUT.
·On each connection proceed through the Login Phase and in to Full Feature Phase operation.
·On the first connection transmit a SCSI-INQUIRY to the DUT. Wait for response and data from the DUT .
·On the first connection transmit a TEST-UNIT READY to the DUT. Wait for response from the DUT
·On the first connection transmit a READ Command to the DUT.
·Increment CmdSN twice (should be two greater than the CmdSN of the previous READ Command), and transmit an immediate Task Management Command with Function Code 2 'ABORT TASK SET'.
·Decrement CmdSN, (should be 1 more than the previous READ Command) and transmit a READ Command to the DUT pm the second connection. This should create an out of order delivery of command PDUs.
·Transmit a non-immediate NOP Out command to the DUT with the same CmdSN as that of the Task Management Command.

Observable Results:
·Verify that the DUT responds to the Task Management Command with an appropriate Task Management Response.
·Verify that the DUT does not respond to the received READ commands.
· Verify that the DUT responds to the received NOP-Out with a NOP-In.

Possible Problems: If ErrorRecoveryLevel = 0, the target may choose to drop the connection in order to complete the ABORT TASK SET command. This may be preceded by an Async Message requesting Logout.

Test #11.3.3: Task Management Response

Purpose: To verify that the target transmits a response when it has completed the assigned task.

Reference: 3.5.1.4, 10.5.4, 10.6.1

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

Last Modification: Mon Aug 4 12:24:17 2003

Discussion: The Task Management function response carries an indication of function completion for a Task Management function request including how it completed (response and qualifier) and additional information for failure responses. After the Task Management response indicates Task Management function completion, the initiator will not receive any additional responses from the affected tasks.

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

Procedure:
·Connect the Testing Station to the iSCSI target being tested.
·Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·Negotiate ErrorRecoveryLevel >= 1 if possible.
·Proceed through the Login Phase and in to Full Feature Phase operation.
·Transmit a SCSI-INQUIRY to the DUT. Wait for response and data from the DUT .
·Transmit a TEST-UNIT READY to the DUT. Wait for response from the DUT
·Transmit a READ Command to the DUT.
·Transmit a Task Management Command with Function Code 1, and an non existent ReferencedTaskTag and a RefCmdSN outside the valid CmdSN window.
·Transmit a non-immediate NOP Out command to the DUT with the same CmdSN as that of the Task Management Command.

Observable Results:
·Verify that the DUT responds to the Task Management Command.
·Verify that the response code is 1 'Task does not exist'
·Verify that the DUT responds to the READ Command.
· Verify that the DUT responds to the received NOP-Out with a NOP-In.

Possible Problems: If ErrorRecoveryLevel = 0, the target may choose to drop the

connection in order to complete the ABORT TASK SET command. A device may also choose to transmit an Async Message requesting Logout.

Test #11.4: Task Management Response Target Warm Reset

Purpose: To verify that the target follows the Target Warm Reset procedure correctly, if supported.

Reference: 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:49:43 2003

Discussion: The implementation of the TARGET WARM RESET function and the TARGET COLD RESET function is OPTIONAL and when implemented, should act as described below. The TARGET WARM RESET is also subject to SCSI access controls on the requesting initiator as defined in [SPC3]. When authorization fails at the target, the appropriate response as described in Section 9.6 Task Management Function Response MUST be returned by the target. When executing the TARGET WARM RESET and TARGET COLD RESET functions, the target cancels all pending operations on all Logical Units known by the issuing initiator. Both functions are equivalent to the Target Reset function specified by [SAM2]. They can affect many other initiators logged in with the servicing SCSI target port.

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

Procedure:
·Connect the Testing Station to the iSCSI target being tested.
·Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·Negotiate ErrorRecoveryLevel >= 1 if possible.
·Proceed through the Login Phase and in to Full Feature Phase operation.
·Transmit a SCSI-INQUIRY to the DUT. Wait for response and data from the DUT .
·Transmit a TEST-UNIT READY to the DUT. Wait for response from the DUT
·Transmit a READ Command to the DUT.
·Increment CmdSN, and transmit a non-immediate Task Management Command with Function Code 6.

Observable Results:
·Verify that the DUT responds accordingly, and that all pending operations are cancelled. The DUT should cease sending Data-in PDUs in response to the received READ Command.

· Verify that if the DUT does not support the TARGET WARM RESET function that it responds to the request with an appropriate Task Management Response Code, such as 5 or 6.

Possible Problems: None.

Test #11.5: Task Management Response Target Cold Reset

Purpose: To verify that the target follows the Target Cold Reset procedure correctly, if supported.

Reference: 10.5.1, 10.6.1

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

Last Modification: Mon May 19 12:50:07 2003

Discussion: The implementation of the TARGET WARM RESET function and the TARGET COLD RESET function is OPTIONAL and when implemented, should act as described below.The TARGET COLD RESET function is not subject to SCSI access controls, but its execution privileges may be managed by iSCSI mechanisms such as login authentication. When executing the TARGET WARM RESET and TARGET COLD RESET functions, the target cancels all pending operations on all Logical Units known by the issuing initiator. Both functions are equivalent to the Target Reset function specified by [SAM2]. They can affect many other initiators logged in with the servicing SCSI target port. The target MUST treat the TARGET COLD RESET function additionally as a power on event, thus terminating all of its TCP connections to all initiators (all sessions are terminated). For this reason, the Service Response (defined by [SAM2]) for this SCSI task management function may not be reliably delivered to the issuing initiator port.

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

Procedure:
·Connect the Testing Station to the iSCSI target being tested.
·Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
·Negotiate ErrorRecoveryLevel >= 1 if possible.
·Proceed through the Login Phase and in to Full Feature Phase operation.
·Transmit a SCSI-INQUIRY to the DUT. Wait for response and data from the DUT .
·Transmit a TEST-UNIT READY to the DUT. Wait for response from the DUT
·Transmit a READ Command to the DUT.
·Increment CmdSN, and transmit a non-immediate Task Management Command with Function Code 7.

Observable Results:
·Verify that the DUT responds accordingly, and that all pending operations are cancelled. The DUT should cease sending Data-in PDUs in response to the received READ Command.
·Verify that the DUT terminates the connection.
· Verify that if the DUT does not support the TARGET COLD RESET function that it responds to the request with an appropriate Task Management Response Code, such as 5 or 6.

Possible Problems: None.

Test #12.1: SNACK-R2T Received

Purpose: To see that an iSCSI target properly responds to a received SNACK Request.

Reference: 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 12:54:26 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. When used in RunLength, it means all PDUs starting with the initial. When used in both BegRun and RunLength, it means all unacknowledged PDUs.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.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes, ErrorRecoveryLevel >=0.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY, TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT. Once received transmit a SNACK with the appropriate Initiator Task Tag and a BegRun=0, RunLength =1, and the R2TSN of the received R2T to indicate that the initial R2T was not received.

Observable Results:

· Verify that the DUT retransmits the R2T, and that it is an exact replica of the original.
· Verify that that the ExpCmdSN, and MaxCmdSN fields are not the same, but instead are the current values. Verify all of the flags are the same.

Possible Problems: An iSCSI target that does not support recovery within connection MAY discard the status SNACK.

Test #12.2: SNACK-Data Received

Purpose: To see that an iSCSI target properly responds to a received SNACK Request.

Reference: 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 12:54:33 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. When used in RunLength, it means all PDUs starting with the initial. When used in both BegRun and RunLength, it means all unacknowledged PDUs.The numbered Data-In PDUs, requested by a Data SNACK MUST be delivered as exact replicas of the ones that the target transmitted originally except for the fields ExpCmdSN and MaxCmdSN, which MUST carry the current values and except for resegmentation.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a READ command to the DUT.
· Wait for a series of Data-in PDUs from the DUT. Once 3 have been received, transmit a SNACK with the appropriate Initiator Task Tag and a BegRun=DataSN of the second received Data-in PDU, and RunLength =1. This will indicate that the second Data-in PDU was not received.

Observable Results:
· Verify that the DUT retransmits the Data-in PDU. These should be exact replicas of the

original Data-In PDUs except for the ExpCmdSN, MaxCmdSn, and ExpDataSN fields.
· Verify that that the ExpCmdSN, and MaxCmdSN fields are not the same, but instead are the current values. Verify all of the flags are the same.

Possible Problems: An iSCSI target that does not support recovery within connection MAY discard the status SNACK.

Test #12.3: SNACK-Data Run Received

Purpose: To see that an iSCSI target properly responds to a received SNACK Request.

Reference: 10.16, 10.16.1

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

Last Modification: Mon May 19 12:54:45 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. When used in RunLength, it means all PDUs starting with the initial. When used in both BegRun and RunLength, it means all unacknowledged PDUs.The numbered Data-In PDUs, requested by a Data SNACK MUST be delivered as exact replicas of the ones that the target transmitted originally except for the fields ExpCmdSN and MaxCmdSN, which MUST carry the current values and except for resegmentation.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a READ command to the DUT.
· Wait for a series of Data-in PDUs from the DUT. Once 3 have been received, transmit a SNACK with the appropriate Initiator Task Tag and a BegRun=0, RunLength =0, and the run of the DataSN for the received Data-in PDUs, to indicate that none of the Data-in PDUs was received.

Observable Results:
· Verify that the DUT retransmits the requested Data-in PDUs. These should be exact

replicas of the original Data-In PDUs except for the ExpCmdSN, MaxCmdSn, and ExpDataSN fields.

Possible Problems: An iSCSI target that does not support recovery within connection MAY discard the status SNACK.

Test #12.4: SNACK Received

Purpose: To see that an iSCSI target properly responds to a received SNACK Request.

Reference: 10.16, 10.17.2

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

Last Modification: Mon May 19 12:54:55 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. When used in RunLength, it means all PDUs starting with the initial. When used in both BegRun and RunLength, it means all unacknowledged PDUs.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. Any SNACK that requests a numbered-response, Data, or R2T that was not sent by the target MUST be rejected with a reason code of "Protocol error". The DataSN/R2TSN field of the Reject PDU. This field is only valid if the rejected PDU is a Data/R2T SNACK and the Reject reason code is "Protocol error". The DataSN/R2TSN is the next Data/R2T sequence number that the target would send for the task, if any.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT. Once received transmit a SNACK with the appropriate

Initiator Task Tag and a BegRun=255, RunLength =1. This R2T should never have been transmitted by the DUT.

Observable Results:
· Verify that the DUT transmits a Reject PDU with a reason code of Protocol Error (0x04).
· Verify that the DataSN field of the Reject PDU is the last valid sequence number that the DUT sent for the task.

Possible Problems: An iSCSI target that does not support recovery within connection MAY discard the status SNACK.

Test #13.1: Logout Response

Purpose: To see that an iSCSI target properly responds to a received Logout Request.

Reference: 10.14, 10.15

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

Last Modification: Mon May 19 12:55:38 2003

Discussion: When receiving a Logout request with the reason code of "close the connection" or "close the session", the target MUST terminate all pending commands, whether acknowledged via ExpCmdSN or not, on that connection or session respectively. The target then issues the Logout response and half-closes the TCP connection (sends FIN). After receiving the Logout response and attempting to receive the FIN (if still possible), the initiator MUST completely close the logging-out connection. For the terminated commands, no additional responses should be expected.

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

Procedure:
· Start a 2 connections from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT.
· On one connection only issue a Logout Request with a reason code of 1 'closes the connection'.

Observable Results:
· Verify that the DUT issues a Logout Response and FIN only on the connection which the Testing Station issued the Logout Request on. All tasks on the second connection should remain open.
· Verify that the Response field in the Logout Response is set to 0.

Possible Problems: None.

Test #13.2: Logout Response

Purpose: To see that an iSCSI target properly responds to a received Logout Request.

Reference: 10.14, 10.15

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

Last Modification: Mon May 19 12:55:47 2003

Discussion: When receiving a Logout request with the reason code of "close the connection" or "close the session", the target MUST terminate all pending commands, whether acknowledged via ExpCmdSN or not, on that connection or session respectively. The target then issues the Logout response and half-closes the TCP connection (sends FIN). After receiving the Logout response and attempting to receive the FIN (if still possible), the initiator MUST completely close the logging-out connection. For the terminated commands, no additional responses should be expected.

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

Procedure:
· Start a 2 connections from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT.
· On one connection only issue a Logout Request with a reason code of 0 'closes the session'.

Observable Results:
· Verify that the DUT half closes each open TCP connection within the session specified to be closed by the Testing Station.
· Verify that the Response field in the Logout Response is set to 0.

Possible Problems: None.

Test #13.3: Logout Response

Purpose: To see that an iSCSI target properly responds to a received Logout Request.

Reference: 10.14, 10.15

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

Last Modification: Mon May 19 12:56:01 2003

Discussion: When receiving a Logout request with the reason code of "close the connection" or "close the session", the target MUST terminate all pending commands, whether acknowledged via ExpCmdSN or not, on that connection or session respectively. The target then issues the Logout response and half-closes the TCP connection (sends FIN). After receiving the Logout response and attempting to receive the FIN (if still possible), the initiator MUST completely close the logging-out connection. For the terminated commands, no additional responses should be expected.

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

Procedure:
· Start 2 connections from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· ssue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT.
· On one connection only issue a Logout Request with a reason code of 1 'closes the connection' for a non-existent CID.

Observable Results:
· Verify that the Response field in the Logout Response is set to 1 = 'CID not found'. Both connections should remain open.

Possible Problems: None.

Test #14.1: Reject

Purpose: To see that an iSCSI target properly formats an iSCSI Reject PDU.

Reference: 6.3, 10.17, 10.17.1

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

Last Modification: Mon May 19 12:56:31 2003

Discussion: Reject is used to indicate an iSCSI error condition (protocol, unsupported option, etc.).n all the cases in which a pre-instantiated SCSI task is terminated because of the reject, the target MUST issue a proper SCSI command response with CHECK CONDITION as described in Section 9.4.3 Response. In these cases in which a status for the SCSI task was already sent before the reject, no additional status is required. If the error is detected while data from the initiator is still expected (i.e., the command PDU did not contain all the data and the target has not received a Data-out PDU with the Final bit set to 1 for the unsolicited data, if any, and all outstanding R2Ts, if any), the target MUST wait until it receives the last expected Data-out PDUs with the F bit set to 1 before sending the Response PDU.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT. Wait for R2T from the DUT.
· Upon receiving the R2T from the DUT, the Testing Station should issue a Data-Out PDU for the requested data. This Data-Out PDU should have a new value for Initiator Task Tag (Not the same as the WRITE command to which this data is associated). The DUT should choose to transmit Reject.
· Issue a second valid Data-Out PDU. This should have the same DataSN as the previous Data-Out, but it also has a legal Initiator Task Tag, same as the WRITE command. Continue issuing Data-Out PDUs until all the data indicated by the command is complete. The last Data-Out PDU should have the F bit set. The DUT is expected to respond with a SCSI Response of status GOOD.

Observable Results:
· Verify that if the DUT issues a Reject PDU it is formatted correctly.
· Verify that if the DUT issues a Reject to the Command, it does not issue any Repsonse PDU at that time, but waits until the Data-Out with the F bit set is received.
· Verify that the StatSN of the Reject PDU issued in response to the errored Data-Out PDU, is one less than the StatSN of the Response issued to the WRITE command when complete.
· Verify that ExpCmdSN of the Reject PDU, is the same as the ExpCmdSN of the Response to the SCSI Inquiry Command.
· Verify that the DUT includes a copy of the rejected PDU header in the Reject.

Possible Problems: None.

Test #14.2: Reject

Purpose: To see that an iSCSI target properly formats an iSCSI Reject PDU.

Reference: 6.3, 10.17, 10.17.1

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

Last Modification: Mon May 19 12:56:39 2003

Discussion: Reject is used to indicate an iSCSI error condition (protocol, unsupported option, etc.).n all the cases in which a pre-instantiated SCSI task is terminated because of the reject, the target MUST issue a proper SCSI command response with CHECK CONDITION as described in Section 9.4.3 Response. In these cases in which a status for the SCSI task was already sent before the reject, no additional status is required. If the error is detected while data from the initiator is still expected (i.e., the command PDU did not contain all the data and the target has not received a Data-out PDU with the Final bit set to 1 for the unsolicited data, if any, and all outstanding R2Ts, if any), the target MUST wait until it receives the last expected Data-out PDUs with the F bit set to 1 before sending the Response PDU.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes, ErrorRecoveryLevel > 0..
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a READ command to the DUT.
· Wait for a Data-In PDU from the DUT.
· Transmit a DataACK SNACK PDU with a invalid Initiator Task Tag which is the same as that of the READ command. The DUT should transmit a Reject.
· Wait for Data-In PDUs and SCSI Response Data to complete the command.

Observable Results:
· Verify that if the DUT issues a Reject PDU it is formatted correctly.
· Verify that the StatSN, ExpCmdSN, and MaxCmdSN are not those of the rejected

command, but are incremented as they usually would have been if the SCSI Command PDU has not been rejected. The only field that must be incremented is StatSN. It is expected that the StatSN in the Reject will be the same as the StatSN in the previous R2T. It is expected that the StatSN of the SCSI Response PDU will be one more than the StatSN of the Reject PDU. None of the fields should decrease in value to match the values of the Rejected PDU.
· Verify that the DUT includes a copy of the rejected PDU header in the Reject.

Possible Problems: The DUT must support ErrorRecoveryLevel > 0 in order to perform this test.

Test #14.3: Reject

Purpose: To see that an iSCSI target properly formats an iSCSI Reject PDU.

Reference: 6.3, 10.17, 10.17.1

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

Last Modification: Mon May 19 12:56:47 2003

Discussion: Reject is used to indicate an iSCSI error condition (protocol, unsupported option, etc.).n all the cases in which a pre-instantiated SCSI task is terminated because of the reject, the target MUST issue a proper SCSI command response with CHECK CONDITION as described in Section 9.4.3 Response. In these cases in which a status for the SCSI task was already sent before the reject, no additional status is required. If the error is detected while data from the initiator is still expected (i.e., the command PDU did not contain all the data and the target has not received a Data-out PDU with the Final bit set to 1 for the unsolicited data, if any, and all outstanding R2Ts, if any), the target MUST wait until it receives the last expected Data-out PDUs with the F bit set to 1 before sending the Response PDU.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· Negotiate the following parameters: ImmediateData=No; InitialR2T=Yes.
· Declare a MaxRecvDataSegmentLength of 1024.
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Issue a WRITE command to the DUT.
· Wait for an R2T from the DUT.
· Transmit a SNACK R2T Request retransmission.

Observable Results:
· Verify that if the DUT issues a Reject PDU it is formatted correctly.
· Verify that the StatSN, ExpCmdSN, and MaxCmdSN are not those of the rejected command, but are incremented as they usually would have been if the SCSI Command PDU has not been rejected. The only field that must be incremented is StatSN. None of the fields should decrease in value to match the values of the Rejected PDU.

· Verify that the DUT includes a copy of the rejected PDU header in the Reject.

Possible Problems: This test is only testable if the DUT does not support SNACK.

Test #15.1.1: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:05 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, DSL=0, I=1, TTT= 0xffffffff .
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is 0 and no data is attached.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.1.2: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:10 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, DSL=0, I=0, TTT= 0xffffffff .
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is 0 and no data is attached.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.1.3: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:17 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, I=0, TTT= 0xffffffff, and some "ping" data attached. This data should not be more than MaxRecvDataSegmentLength.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is the same as the NOP-Out PDU and the "ping" data is an exact replica of that sent by the Testing Station.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.1.4: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:23 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested. On each connection do the following:
· Perform a standard login and proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, I=1, TTT= 0xffffffff, and some "ping" data attached. This data should not be more than MaxRecvDataSegmentLength.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is the same as the NOP-Out PDU and the "ping" data is an exact replica of that sent by the Testing Station.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.1.5: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:29 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· begin a standard login
· Declare a MaxRecvPDULength less than the MaxRecvPDULength of the target.
· Proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, I=1, TTT= 0xffffffff, and "ping" data attached. This data shouldbe equal to the MaxRecvDataSegmentLength supported by the DUT.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is the same as the NOP-Out PDU and the "ping" data is an exact replica of that sent by the Testing Station, but only up to the MaxRecvDataSegmentLength declared by the Testing Station.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.1.6: NOP-In Ping Response

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.18,10.19

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

Last Modification: Mon May 19 12:57:35 2003

Discussion: NOP-In may sent by a target as a "ping" response to a NOP-Out "ping" request. When a target receives the NOP-Out with a valid Initiator Task Tag (not the reserved value 0xffffffff), it MUST respond with a NOP-In with the same Initiator Task Tag that was provided in the NOP-Out request. It MUST also duplicate up to the first MaxRecvDataSegmentLength bytes of the initiator provided Ping Data. For such a response, the Target Transfer Tag MUST be 0xffffffff.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· begin a standard login
· Declare a MaxRecvPDULength less than the MaxRecvPDULength of the target.
· Proceed to the Full Feature Phase.
· Issue a NOP-Out with ITT != 0xffffffff, I=0, TTT= 0xffffffff, and "ping" data attached. This data shouldbe equal to the MaxRecvDataSegmentLength supported by the DUT.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT issues NOP-In in response to the received NOP-Out.
· Verify that the DUT set the Initiator Task Tag field to the same as was in the received NOP-Out PDU.
· Verify that the Target Transfer Tag is set to 0xffffffff.
· Verify that the DSL is the same as the NOP-Out PDU and the "ping" data is an exact replica of that sent by the Testing Station, but only up to the MaxRecvDataSegmentLength declared by the Testing Station.
· Verify that the received SCSI Response PDU has StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: None.

Test #15.2: NOP-In Ping Request

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.19

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

Last Modification: Thu Jun 26 09:34:56 2003

Discussion: NOP-In may sent by a target as a "ping" request to solicit a "ping" response. When a target sends a NOP-In that is not a response to a NOP-Out received from an initiator, the Initiator Task Tag MUST be set to 0xffffffff and the Data Segment MUST NOT contain any data. When sending a NOP-In as aping, the Target Transfer Tag must be set to a valid value, (i.e. not 0xffffffff). The LUN must be set to a correct value. When ITT is set to 0xffffffff StatSN for the connection is not advanced after the NOP-In is sent.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· begin a standard login
· Declare a MaxRecvPDULength less than the MaxRecvPDULength of the target.
· Proceed to the Full Feature Phase.
· Do not transmit any PDU's on the connection for 30 seconds.
· Wait for a NOP-In ping request from the DUT.
· Issue a correct NOP-Out in response.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT set the Initiator Task Tag field to 0xffffffff.
· Verify that the Target Transfer Tag is set to a valid value.
· Verify that the DSL is 0.
· Verify that the received SCSI Response PDU does not have StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: It may not be possible to configure the DUT to send Ping requests. If so, this item is Not Testable.

Test #15.3: NOP-In Confirm ExpCmdSN

Purpose: To see that an iSCSI target properly constructs a NOP-In PDU.

Reference: 10.19

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

Last Modification: Thu Jun 26 09:44:58 2003

Discussion: NOP-In may sent by a target to update an Initiators ExpCmdSN / MaxCmdSN. When a target sends a NOP-In that is not a response to a NOP-Out received from an initiator, the Initiator Task Tag MUST be set to 0xffffffff and the Data Segment MUST NOT contain any data. When sending a NOP-In while not expecting a NOP-Out response, the Target Transfer Tag must be set to 0xffffffff. The LUN must be set to 0. When ITT is set to 0xffffffff StatSN for the connection is not advanced after the NOP-In is sent.

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

Procedure:
· Start a connection from the Testing Station to the iSCSI target being tested.
· begin a standard login
· Proceed to the Full Feature Phase.
· Transmit n READ commands. Where n is the difference between ExpCmdSN and MaxCmdSN. Continue transmitting READ commands until the command window is closed. The observable for this will be when the ExpCmdSN and MaxCmdSN offered by the target are the same.
· The DUT may choose to transmit a NOP-In once all the commands have been fulfilled to notify the initiator of the new ExpCmdSN and MaxCmdSN.
· If a NOP-In is received, issue a correct NOP-Out in response.
· Transmit a command to the DUT. Wait for the DUT to transmit response data and status.

Observable Results:
· Verify that the DUT set the Initiator Task Tag field to 0xffffffff.
· Verify that the Target Transfer Tag is set to 0xffffffff
· Verify that the DSL is 0.
· Verify that the received SCSI Response PDU does not have StatSN incremented from the StatSN transmitted in the NOP-In response.

Possible Problems: It may not be possible to configure the DUT to transmit NOP-In PDUs to update ExpCmdSN and MaxCmdSN. If so, this item is Not Testable.

Test #16.1: SCSI Response Residual Underflow

Purpose: To verify that the target issues the Residual Underflow field of the SCSI Response PDU correctly.

Reference: 10.4.1, 10.4.5

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

Last Modification: Mon May 19 13:44:38 2003

Discussion: Bit 6 of Byte 1 of the SCSI Response PDU is the U bit,set for Residual Underflow. In this case, the Residual Count indicates the number of bytes that were not transferred out of the number of bytes that were expected to be transferred. For a bidirectional operation, the Residual Count contains the residual for the write operation. Bits O and U and bits o and u are mutually exclusive (i.e., having both o and u or O and U set to 1 is a protocol error). For a response other than "Command Completed at Target", bits 3-6 MUST be 0.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Issue a SCSI-INQUIRY to the DUT. This SCSI Command PDU should have an Expected Data Transfer Length of 4096 bytes.Since this is more than most targets will send for INQUIRY data, a Underflow condition is created. Wait for a SCSI Response PDU.

Observable Results:
· Verify that the DUT transmits a SCSI Response with the U bit set, and the Residual Count field indicating the correct number bytes worth of information was not transmitted.

· Verify that the O bit is not set to 1.

Possible Problems: The iSCSI target has the option of setting the U bit.

Test #16.2: SCSI Response Residual Overflow

Purpose: To verify that the target issues the Residual Overflow field of the SCSI Response PDU correctly.

Reference: 10.4.1, 10.4.5

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

Last Modification: Mon May 19 13:44:45 2003

Discussion: Bit 5 of Byte 1 of the SCSI Response PDU is the O bit,set for Residual Overflow. In this case, the Residual Count indicates the number of bytes that were not transferred because the initiator's Expected Data Transfer Length was not sufficient. For a bidirectional operation, the Residual Count contains the residual for the write operation. Bits O and U and bits o and u are mutually exclusive (i.e., having both o and u or O and U set to 1 is a protocol error). For a response other than "Command Completed at Target", bits 3-6 MUST be 0.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit 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.
· Issue a SCSI-INQUIRY command to the DUT with an Expected Data Transfer Length of 4 bytes. This should be less than what most targets will return for INQUIRY data, and an Overflow condition is created. Wait for the SCSI Response PDU.

Observable Results:
· Verify that the target sends a SCSI Response with the O bit set, and the Residual Count field indicating the difference between the amount of information transmitted and the Expected Data Transfer Length.

Possible Problems: The iSCSI target is not required to use the O bit.

Test #16.3.1: SCSI Response Unsolicited Data

Purpose: To verify that the target handles the reception of Unsolicited Data correctly.

Reference: 3.2.4.2, 10.3.4, 12.14

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

Last Modification: Mon May 19 13:48:55 2003

Discussion: 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.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.The initiator and target negotiate FirstBurstLength, the maximum amount in bytes of unsolicited data an iSCSI initiator may send to the target during the execution of a single SCSI command. This covers the immediate data (if any) and the sequence of unsolicited Data-Out PDUs (if any) that follow the command.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During the Login Phase, negotiate the following parameters: ImmediateData=Yes; InitialR2T=Yes; FirstBurstLength = 512.
· Proceed through the Login Phase and into Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE Command to the DUT, with 512 bytes of immediate data attached, and with the Expected Data Transfer Length = FirstBurstLength + 512, and with the F bit set to 0.
· Wait for the DUT to transmit an R2T PDU.
· Transmit a Data-out PDU with 512 bytes of data. This Data-out PDU should have the same InitiatorTaskTag as the previous WRITE command.

Observable Results:
· Verify that the target transmits a SCSI Response of status GOOD.

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

Test #16.3.2: SCSI Response Unsolicited Data

Purpose: To verify that the target handles the reception of Unsolicited Data correctly.

Reference: 3.2.4.2, 10.3.4, 10.4.7, 12.14

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

Last Modification: Mon May 19 13:49:20 2003

Discussion: 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.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.The initiator and target negotiate FirstBurstLength, the maximum amount in bytes of unsolicited data an iSCSI initiator may send to the target during the execution of a single SCSI command. This covers the immediate data (if any) and the sequence of unsolicited Data-Out PDUs (if any) that follow the command.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During the Login Phase, negotiate the following parameters: ImmediateData=Yes; InitialR2T=Yes; FirstBurstLength = 512.
· Proceed through the Login Phase and into Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE Command to the DUT, with 1024 bytes of immediate data attached, and with the Expected Data Transfer Length = 1024, and with the F bit set to 0.
· Wait for the DUT to transmit a SCSI Response PDU.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status CHECK CONDITION and sense data of 'unexpected unsolicited data' ASC=0x0c ASCQ=0x0c.

· The target may also choose to disconnect or transmit an Async Message requesting Logout.

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

Test #16.3.3: SCSI Response Unsolicited Data

Purpose: To verify that the target handles the reception of Unsolicited Data correctly.

Reference: 3.2.4.2, 10.3.4, 12.10, 12.11

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

Last Modification: Mon May 19 13:49:37 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.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE Command to the DUT with Expected Data Transfer Length of 1024.

· Wait for the R2T PDU from the DUT.
· Transmit a Data-out PDU with 1024 bytes of data and the F bit set to 1.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status GOOD.

Possible Problems: None.

Test #16.3.4: SCSI Response Unsolicited Data

Purpose: To verify that the target handles the reception of Unsolicited Data correctly.

Reference: 3.2.4.2, 10.3.4, 12.10, 12.11

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

Last Modification: Mon May 19 13:49:50 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.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE Command to the DUT, with 1024 bytes of immediate data attached.
· Wait for the SCSI Response PDU from the DUT.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status CHECK CONDITION and sense data of 'unexpected unsolicited data' ASC=0x0c ASCQ=0x0c.
· The target may also choose to disconnect or transmit an Async Message requesting Logout.

Possible Problems: None.

Test #16.3.5: SCSI Response Unsolicited Data

Purpose: To verify that the target handles the reception of Unsolicited Data correctly.

Reference: 3.2.4.2, 10.3.4, 12.10, 12.11

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

Last Modification: Mon May 19 13:51:19 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. 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:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, attach the keys FirstBurstLength=1024, ImmediateData=No, and InitialR2T=No
· Proceed through the Login Phase and into Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE command with Expected Data Transfer Length of 1024. Do not wait for an R2T PDU.
· Transmit a Data-out PDU, with 1024 data attached and the F bit set to 1.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status GOOD.

Possible Problems: The target may not support Initial R2T=No.

Test #16.3.6: SCSI Response Unexpected Data

Purpose: To verify that the target transmits a SCSI Response with correctly formatted sense data if a SCSI Command is issued without enough unsolicited data.

Reference: 3.2.4.2, 10.3.4, 10.4.7.2, 12.10, 12.11

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

Last Modification: Mon May 19 13:51:36 2003

Discussion: Certain iSCSI conditions result in the command being terminated at the target (response Command Completed at Target) with a SCSI Check Condition Status and sense data to indicate one of the following: Unexpected unsolicited data, Incorrect amount of data, Protocol Service CRC error, SNACK rejected. The target reports the "Incorrect amount of data" condition if during data output the total data length to output is greater than FirstBurstLength and the initiator sent unsolicited non-immediate data but the total amount of unsolicited data is different than FirstBurstLength. The target reports the same error when the amount of data sent as a reply to an R2T does not match the amount requested.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the keys InitialR2T=No and DataSequenceInOrder=Yes
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE to the DUT, with ExpectedDataTransferLength=1024
· Transmit a Data-out PDU, with 512 bytes of data attached, and the F bit set to 1.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status CHECK CONDITION and sense data of 'Incorrect amount of data' ASC=0x0c ASCQ=0x0d.
· The target may also choose to disconenct.

Possible Problems: None.

Test #16.3.7: SCSI Response Bad OpCode

Purpose: To verify that the target transmits a SCSI Response with correctly formatted sense data if a SCSI Command is issued with an invalid OpCode.

Reference: 3.2.4.2, 10.3.4, 10.4.7.2, 12.10, 12.11

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

Last Modification: Thu Aug 7 08:25:15 2003

Discussion: Certain iSCSI conditions result in the command being terminated at the target (response Command Completed at Target) with a SCSI Check Condition Status and sense data to indicate one of the following: Unexpected unsolicited data, Incorrect amount of data, Protocol Service CRC error, SNACK rejected. The target reports the "Incorrect amount of data" condition if during data output the total data length to output is greater than FirstBurstLength and the initiator sent unsolicited non-immediate data but the total amount of unsolicited data is different than FirstBurstLength. The target reports the same error when the amount of data sent as a reply to an R2T does not match the amount requested.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the keys InitialR2T=No and DataSequenceInOrder=Yes
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a SCSI Command PDU to the DUT with a bad CDB OpCode.

Observable Results:
· Verify that the target transmits a SCSI Response PDU of status CHECK CONDITION.
· The target may also choose to disconenct.

Possible Problems: None.

Test #16.4.1: SCSI Response Retry

Purpose: To verify that the target silently discards any retries which are issued for commands in progress.

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: Thu Jun 19 16:33:40 2003

Discussion: If initiators, as part of plugging command sequence gaps as described above, inadvertently issue retries for allegiant commands already in progress (i.e., targets did not see the discontinuities in CmdSN ordering), the duplicate commands are silently ignored by targets. 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:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE to the DUT, with the W bit set to 1 and the R bit set to 0. Wait for an R2T from the DUT
· Transmit a Data-out PDU, with the F bit set to 0
· Transmit a WRITE to the DUT, which is identical to the first one. This must be done before the DUT transmits a SCSI Response to the first command.
· Transmit Data-Out PDUs to complete the WRITE command, finishing by sending a Data-Out PDU with the F bit set. Wait for a response of status GOOD.

Observable Results:
· Verify that the second WRITE command is silently discarded. The DUT should not send any response to the second WRITE command.

· Verify that the DUT was able to properly complete the valid WRITE command, despite the appearance of the duplicate command.

Possible Problems: None.

Test #16.4.2: SCSI Response Retry

Purpose: To verify that the target silently discards any retries which are issued for commands in progress.

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: Thu Jun 19 16:33:06 2003

Discussion: If initiators, as part of plugging command sequence gaps as described above, inadvertently issue retries for allegiant commands already in progress (i.e., targets did not see the discontinuities in CmdSN ordering), the duplicate commands are silently ignored by targets. 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:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE to the DUT, with the W bit set to 1 and the R bit set to 0. Wait for an R2T from the DUT
· Transmit a WRITE to the DUT, which is identical to the first one. This must be done before the DUT transmits a SCSI Response to the first command.
· Transmit Data-Out PDUs to complete the WRITE command, finishing by sending a Data-Out PDU with the F bit set. Wait for a response of status GOOD.

Observable Results:
· Verify that the second WRITE command is silently discarded. The DUT should not send any response to the second WRITE command.
· Verify that the DUT was able to properly complete the valid WRITE command, despite

the appearance of the duplicate command.

Possible Problems: None.

Test #16.5: SCSI Response Error Detection

Purpose: To verify that the target waits for a Data-out PDU with the F bit set to 1 before sending a SCSI Response, if an error is detected while data is still expected.

Reference: 10.4.2, 10.17.1

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

Last Modification: Thu Jun 19 16:33:25 2003

Discussion: The a target sending a Reject PDU, should use theInvalid PDU field reason code for all invalid values of PDU fields that are meant to describe a task, a response, or a data transfer. Some examples are invalid TTT/ITT, buffer offset, LUN qualifying a TTT, and an invalid sequence number in a SNACK. If a SCSI device error is detected while data from the initiator is still expected (the command PDU did not contain all the data and the target has not received a Data PDU with the final bit Set), the target MUST wait until it receives a Data PDU with the F bit set in the last expected sequence before sending the Response PDU.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the following keys: ImmediateData=No, InitialR2T=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE Command to the DUT, with the Expected Data Transfer Length = 2048
· Wait for an R2T from the DUT. Depending on the Desired Data Transfer Length indicated by the DUT it may be possible to send each of the Data-out PDUs without waiting for an intervening R2T.
· Transmit a Data-out PDU with the F bit set to 0, ITT = (ITT of WRITE) + 1, and attach 512 bytes of data.
· Transmit 3 more 512 byte Data-out PDUs with ITT = ITT of WRITE. The last Data-out PDU should have the F bit set to 1.

Observable Results:
· The DUT should , but is not required to, recongnize the invalid ITT, and transmit a Reject PDU.
· Verify that the DUT does not issue a SCSI Response until the final Data-out PDU is transmitted.

Possible Problems: None.

Test #17.1.1: Text Response Text Fields

Purpose: To verify that the target issues the Text Response PDU correctly .

Reference: 10.11, 10.11.1, 10.11.3, 10.11.6

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

Last Modification: Mon May 19 13:59:02 2003

Discussion: The Text Response PDU contains the target's responses to the initiator's Text request. The format of the Text field matches that of the Text request.When set to 1, in response to a Text Request with the Final bit set to 1, the F bit indicates that the target has finished the whole operation. Otherwise, if set to 0 in response to a Text Request with the Final Bit set to 1, it indicates that the target has more work to do (invites a follow-on text request). A Text Response with the F bit set to 1 in response to a Text Request with the F bit set to 0 is a protocol error. A Text Response with the F bit set to 1 MUST NOT contain key=value pairs that may require additional answers from the initiator. A Text Response with the F bit set to 1 MUST have a Target Transfer Tag field set to the reserved value of 0xffffffff. A Text Response with the F bit set to 0 MUST have a Target Transfer Tag field set to a value other than the reserved 0xffffffff.The Initiator Task Tag matches the tag used in the initial Text Request.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets=All and a valid Initiator Task Tag and F =0.

Observable Results:
· Verify that a Text Response is transmitted and that the ITT is the same as that in the Text Request
· Verify that the F bit is 0, and the TTT is not 0xFFFFFFFF
· Verify that the data segment is <= the MaxRecvDataSegmentLength of the Testing Station

Possible Problems: None.

Test #17.1.2: Text Response Text Fields

Purpose: To verify that the target issues the Text Response PDU correctly .

Reference: 10.11, 10.11.1, 10.11.3, 10.11.6

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

Last Modification: Mon May 19 13:59:13 2003

Discussion: The Text Response PDU contains the target's responses to the initiator's Text request. The format of the Text field matches that of the Text request.When set to 1, in response to a Text Request with the Final bit set to 1, the F bit indicates that the target has finished the whole operation. Otherwise, if set to 0 in response to a Text Request with the Final Bit set to 1, it indicates that the target has more work to do (invites a follow-on text request). A Text Response with the F bit set to 1 in response to a Text Request with the F bit set to 0 is a protocol error. A Text Response with the F bit set to 1 MUST NOT contain key=value pairs that may require additional answers from the initiator. A Text Response with the F bit set to 1 MUST have a Target Transfer Tag field set to the reserved value of 0xffffffff. A Text Response with the F bit set to 0 MUST have a Target Transfer Tag field set to a value other than the reserved 0xffffffff.The Initiator Task Tag matches the tag used in the initial Text Request.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, do not transmit the MaxRecvDataSegmentLength key.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the MaxRecvDataSegmentLength key attached, a valid ITT, F=0. Wait for a response.
· Transmit an empty Text Request with F=1. Wait for a Text Response with F=1.
· Issue a SCSI-INQUIRY , TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a READ Command to the DUT. Wait for a Data-in PDU.

Observable Results:
· Verify that when the initial Text Response is transmitted, that the ITT is the same as that

in the Text Request.
· Verify in the initial Text Response. that the F bit is 0, and the TTT is not 0xFFFFFFFF
· Verify that in the second Text Response, if the F bit is 1, the TTT = 0xFFFFFFFF
· Verify that the data segment of the received Data-in PDU is <= the MaxRecvDataSegmentLength of the Testing Station.

Possible Problems: None.

Test #17.2.1: Text Response F bit

Purpose: To verify that the target does not set the F bit to 1 in response to a request with the F bit set to 0.

Reference: 10.11.1

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

Last Modification: Mon May 19 13:59:30 2003

Discussion: When set to 1, in response to a Text Request with the Final bit set to 1, the F bit indicates that the target has finished the whole operation. Otherwise, if set to 0 in response to a Text Request with the Final Bit set to 1, it indicates that the target has more work to do (invites a follow-on text request). A Text Response with the F bit set to 1 in response to a Text Request with the F bit set to 0 is a protocol error.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the F bit = 0, and with the key SendTargets=All

Observable Results:
· Verify that the F bit in the response which is transmitted by the target is not set to 1.

Possible Problems: None.

Test #17.2.2: Text Response F bit

Purpose: To verify that the target does not set the F bit to 1 in response to a request with the F bit set to 0.

Reference: 10.11.1

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

Last Modification: Mon May 19 13:59:37 2003

Discussion: When set to 1, in response to a Text Request with the Final bit set to 1, the F bit indicates that the target has finished the whole operation. Otherwise, if set to 0 in response to a Text Request with the Final Bit set to 1, it indicates that the target has more work to do (invites a follow-on text request). A Text Response with the F bit set to 1 in response to a Text Request with the F bit set to 0 is a protocol error.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, do not transmit the MaxRecvDataSegmentLength key.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT with the F bit = 0, and with the MaxRecvDataSegmentLength key

Observable Results:
· Verify that the F bit in the response which is transmitted by the target is not set to 1.

Possible Problems: None.

Test #17.3.1: Text Response SendTargets Response

Purpose: To verify that the target responds in accordance with the purpose of the SendTargets=All key.

Reference: Appendix D

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

Last Modification: Mon May 19 14:00:15 2003

Discussion: A system that contains targets MUST support discovery sessions on each of its iSCSI IP address-port pairs, and MUST support the SendTargets command on the discovery session. A target MUST return all path information (IP address-port pairs and portal group tags) for the targets for which the requesting initiator is authorized.A SendTargets command consists of a single Text request PDU. This PDU contains exactly one text key and value. The text key MUST be SendTargets. The expected response depends upon the value, as well as whether the session is a discovery or operational session. The value may be All. This means the initiator is requesting that information on all relevant targets known to the implementation be returned. This value MUST be supported on a discovery session, and MUST NOT be supported on an operational session.The response to this command is a text response that contains a list of zero or more targets and, optionally, their addresses. Each target is returned as a target record. A target record begins with the TargetName text key, followed by a list of TargetAddress text keys, and bounded by the end of the text response or the next TargetName key, which begins a new record. No text keys other than TargetName and TargetAddress are permitted within a SendTargets response.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets=All.

Observable Results:
· Verify that the target transmits a Text Response
· Verify that the target attaches its Target Name key(s), not using the default 'iSCSI'

Target Name key
· Verify that the target attaches 0 or more target records.

Possible Problems: None.

Test #17.3.2: Text Response SendTargets Response

Purpose: To verify that the target responds in accordance with the purpose of the SendTargets= key.

Reference: Appendix D

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

Last Modification: Mon May 19 14:00:22 2003

Discussion: A system that contains targets MUST support discovery sessions on each of its iSCSI IP address-port pairs, and MUST support the SendTargets command on the discovery session. A target MUST return all path information (IP address-port pairs and portal group tags) for the targets for which the requesting initiator is authorized.A SendTargets command consists of a single Text request PDU. This PDU contains exactly one text key and value. The text key MUST be SendTargets. The expected response depends upon the value, as well as whether the session is a discovery or operational session. The value may be nothing. This means the session should only respond with addresses for the target to which the session is logged in. This MUST be supported on operational sessions, and MUST NOT return targets other than the one to which the session is logged in. The response to this command is a text response that contains a list of zero or more targets and, optionally, their addresses. Each target is returned as a target record. A target record begins with the TargetName text key, followed by a list of TargetAddress text keys, and bounded by the end of the text response or the next TargetName key, which begins a new record. No text keys other than TargetName and TargetAddress are permitted within a SendTargets response.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets=.

Observable Results:
· Verify that the target transmits a Text Response
· Verify that the target attaches the TargetName key for the target to which the Testing

Station has logged in, not using the default 'iSCSI' Target Name key

Possible Problems: None.

Test #17.3.3: Text Response SendTargets Response

Purpose: To verify that the target responds in accordance with the purpose of the SendTargets=All key value pair.

Reference: Appendix D

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

Last Modification: Mon May 19 14:00:28 2003

Discussion: A system that contains targets MUST support discovery sessions on each of its iSCSI IP address-port pairs, and MUST support the SendTargets command on the discovery session. A target MUST return all path information (IP address-port pairs and portal group tags) for the targets for which the requesting initiator is authorized.A SendTargets command consists of a single Text request PDU. This PDU contains exactly one text key and value. The text key MUST be SendTargets. The expected response depends upon the value, as well as whether the session is a discovery or operational session. The value may be All. This means the initiator is requesting that information on all relevant targets known to the implementation be returned. This value MUST be supported on a discovery session, and MUST NOT be supported on an operational session.The response to this command is a text response that contains a list of zero or more targets and, optionally, their addresses. Each target is returned as a target record. A target record begins with the TargetName text key, followed by a list of TargetAddress text keys, and bounded by the end of the text response or the next TargetName key, which begins a new record. No text keys other than TargetName and TargetAddress are permitted within a SendTargets response.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets=All.

Observable Results:
· Verify that the target transmits a Text Response with key=value SendTargets=Reject. The target should not support SendTargets=All in a Normal Session.

Possible Problems: None.

Test #17.3.4: Text Response SendTargets Response

Purpose: To verify that the target responds in accordance with the purpose of the SendTargets= key.

Reference: Appendix D

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

Last Modification: Mon May 19 14:00:35 2003

Discussion: A system that contains targets MUST support discovery sessions on each of its iSCSI IP address-port pairs, and MUST support the SendTargets command on the discovery session. A target MUST return all path information (IP address-port pairs and portal group tags) for the targets for which the requesting initiator is authorized.A SendTargets command consists of a single Text request PDU. This PDU contains exactly one text key and value. The text key MUST be SendTargets. The expected response depends upon the value, as well as whether the session is a discovery or operational session. The value may be nothing. This means the session should only respond with addresses for the target to which the session is logged in. This MUST be supported on operational sessions, and MUST NOT return targets other than the one to which the session is logged in. The response to this command is a text response that contains a list of zero or more targets and, optionally, their addresses. Each target is returned as a target record. A target record begins with the TargetName text key, followed by a list of TargetAddress text keys, and bounded by the end of the text response or the next TargetName key, which begins a new record. No text keys other than TargetName and TargetAddress are permitted within a SendTargets response.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets=.

Observable Results:
· Verify that the target transmits a Text Response.
· Verify that the target attaches its TargetName, not using the default 'iSCSI' Target Name

key, as this is required to be supported in a Normal Session.

Possible Problems: None.

Test #17.3.5: Text Response SendTargets Response

Purpose: To verify that the target responds in accordance with the purpose of the SendTargets= key.

Reference: Appendix D

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

Last Modification: Tue Jan 14 14:41:49 2003

Discussion: A system that contains targets MUST support discovery sessions on each of its iSCSI IP address-port pairs, and MUST support the SendTargets command on the discovery session. A target MUST return all path information (IP address-port pairs and portal group tags) for the targets for which the requesting initiator is authorized.A SendTargets command consists of a single Text request PDU. This PDU contains exactly one text key and value. The text key MUST be SendTargets. The expected response depends upon the value, as well as whether the session is a discovery or operational session. The value may be the iSCSI target name .If an iSCSI target name is specified, the session should respond with addresses for only the named target, if possible. This value MUST be supported on discovery sessions. A discovery session MUST be capable of returning addresses for those targets that would have been returned had value=All been designated. The response to this command is a text response that contains a list of zero or more targets and, optionally, their addresses. Each target is returned as a target record. A target record begins with the TargetName text key, followed by a list of TargetAddress text keys, and bounded by the end of the text response or the next TargetName key, which begins a new record. No text keys other than TargetName and TargetAddress are permitted within a SendTargets response.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the key SendTargets= "Known iSCSI-target-name".

Observable Results:

· Verify that the target transmits a Text Response
· Verify that the target attaches address for only the named targets.

Possible Problems: It is necessary to obtain the iSCSI TargetName before performing this test.

Test #17.4.1: Text Response Other Parameters

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

Reference: 3.3

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

Last Modification: Mon May 19 14:01:13 2003

Discussion: During discovery sessions, the target may only accept text requests with the SendTargets key and a logout request with the reason code ?close the session?.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the MaxRecvDataSegmentLength key.

Observable Results:
· Verify that the target transmits a Text Response PDU with the key=value pair MaxRecvDataSegmentLength=Irrelevant.

Possible Problems: The DUT may also transmit a Reject PDU.

Test #17.4.2: Text Response Other Parameters

Purpose: To verify that a target does not accept a Logout PDU with an unacceptable reason code during a discovery session.

Reference: 3.3

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

Last Modification: Mon May 19 14:01:20 2003

Discussion: During discovery sessions, the target may only accept text requests with the SendTargets key and a logout request with the reason code 'close the session'.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT with the key SendTargets=All, receive response(s).

· Transmit Logout Request to the DUT with reason code 2 (remove connection for recovery).

Observable Results:
· Verify that the target transmits a Logout Response with a response of 2 'connection recovery not supported'.

Possible Problems: The DUT may also transmit a Reject PDU.

Test #17.5: Text Response Initiator Task Tag

Purpose: To verify that the target responds appropriately when an erroneous Initiator Task Tag is received.

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 14:02:01 2003

Discussion: Parameter negotiation in Full Feature Phase is done through Text requests and responses. Operational parameter negotiation MAY involve several Text request-response exchanges, which the initiator always starts, terminates, and uses the same Initiator Task Tag.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, do not transmit the MaxRecvDataSegmentLength key or the Initiator Alias key.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the F bit = 0, and the MaxRecvDataSegmentLength key attached
· Receive the Text Response
· Transmit a second Text Request with the F bit = 1, and with the same LUN and Target Transfer Tag as the Text Response, but with an Initiator Task Tag not equal to the Initiator Task Tag in the first Text Request, and with an Initiator Alias key attached for the Testing Station

Observable Results:
· Verify if the target sends a Reject PDU in response to the second Text Request it is formatted properly.

Possible Problems: None.

Test #17.6: Text Response Negotiate Once

Purpose: To verify that the target responds appropriately when the initiator attempts to negotiate a parameter more than once.

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 14:02:09 2003

Discussion: An initiator MAY reset an operational parameter negotiation by issuing a Text request with the Target Transfer Tag set to the value 0xffffffff after receiving a response with the Target Transfer Tag set to a value other than 0xffffffff. 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, except for responses to specific keys that explicitly allow repeated key declarations (e.g., TargetAddress). If detected by the target, this MUST result in a Reject PDU with a reason of "protocol error".

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to 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.
· Transmit a Text Request to the DUT, with the F bit = 0, and the MaxRecvDataSegmentLength=512 key attached
· Receive the Text Response
· Transmit a second Text Request with the F bit = 0, and with the same LUN and Target Transfer Tag as the Text Response, and with the MaxRecvDataSegmentLength=512 key attached
· Receive the Reject
· Transmit a third Text Request with the F bit = 0, the TTT = 0xFFFFFFFF and the LUN = 0, and with the MaxRecvDataSegmentLength=512 key attached.
· Receive the Text Response.
· Transmit an empty Text Request with F=1. Wait for a Text Response with F=1.
· Transmit the following to the DUT: SCSI-INQUIRY, TEST UNIT READY, READ-

CAP. Wait for response and data to each.
· Transmit a READ command the the DUT with an EDTL of 2048 bytes. Wait for Data-In PDUs.

Observable Results:
· Verify that the second Text Request is rejected, with reason code 0x04 - protocol error.
· Verify that the target responds with a normal Text Response upon receiving the third Text Request.
· Verify that the target responds to the fourth Text Request with a Text Response with the F bit set to 1, if it does not require any further negotiation.
· Verify that the DUT uses the declared value for MaxRecvDataSegmentLength when transmitting Data-In PDUs.

Possible Problems: None.

Test #17.7.1: Text Response Negotiation Reset

Purpose: To verify that the parameters which are negotiated before a negotiation reset are not utilized.

Reference: 6.10

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

Last Modification: Thu Jun 19 16:14:47 2003

Discussion: Text request and response sequences, when used to set/negotiate operational parameters, constitute the negotiation/parameter setting. A negotiation failure is considered to be one or more of the following: None of the choices, or the stated value, is acceptable to one of the sides in the negotiation, the text request timed out and possibly terminated, the text request was answered with a Reject PDU. A failure in negotiation, while in the Full Feature Phase, will terminate the entire negotiation sequence that may consist of a series of text requests that use the same Initiator Task Tag. The operational parameters of the session or the connection MUST continue to be the values agreed upon during an earlier successful negotiation (i.e., any partial results of this unsuccessful negotiation MUST NOT take effect and MUST be discarded).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, do not transmit the MaxRecvDataSegmentLength key or the Initiator Alias key. Complete the Login Phase and proceed into Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the F bit = 0, and the MaxRecvDataSegmentLength=512 key attached
· Receive the Text Response
· Transmit a second Text Request with the F bit = 1, and with the same LUN and Target Transfer Tag as the Text Response, but with an Initiator Task Tag != the Initiator Task Tag in the first Text Request, and with an Initiator Alias key attached for the Testing Station
· Receive the Reject PDU
· Transmit a third Text Request with the F bit = 0, the TTT = 0xFFFFFFFF, the ITT same

as the first Text Request, and the LUN = 0, and with the MaxRecvDataSegmentLength=512 key attached.
· Wait for a Text Response.
· Transmit a fourth empty Text Request with F=1, the TTT offered by the target in the previous response, and the same ITT as the previous Text Request.
· Wait for a Text Response with F=1.
· Transmit the following to the DUT: SCSI-INQUIRY, TEST UNIT READY, READ-CAP. Wait for response and data to each.
· Transmit a READ command the the DUT with an EDTL of 2048 bytes. Wait for Data-In PDUs.

Observable Results:
·Verify that the DUT transmits a Reject PDU in response to the second Text Request.
· Verify that the target transmits a normal Text Response to the third Text Request, to indicate that the target recognizes MaxRecvDataSegmentLength key was not fully negotiated previously.
· Verify that the target responds to the fourth Text Request with a Text Response with the F bit set to 1, if it does not require any further negotiation.
· Verify that the DUT uses the declared value for MaxRecvDataSegmentLength when transmitting Data-In PDUs.

Possible Problems: If the target does not recognize a repeated key without an intervening reset as an error, this test cannot be performed. Upon receiving the second Text Request some devices may chose to request Logout via an Asycn Message or disconnect. There is no requirement that the DUT transmit Reject to the Text Request with the unknown ITT. This test is primarily focused on testing the targets ability to reset a negotiation when TTT=0xffffffff.

Test #17.7.2: Text Response Negotiation Failure

Purpose: To verify that the parameters which are negotiated during a negotiation failure are not utilized.

Reference: 6.10

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

Last Modification: Tue Jun 24 09:19:20 2003

Discussion: Text request and response sequences, when used to set/negotiate operational parameters, constitute the negotiation/parameter setting. A negotiation failure is considered to be one or more of the following: None of the choices, or the stated value, is acceptable to one of the sides in the negotiation, the text request timed out and possibly terminated, the text request was answered with a Reject PDU. A failure in negotiation, while in the Full Feature Phase, will terminate the entire negotiation sequence that may consist of a series of text requests that use the same Initiator Task Tag. The operational parameters of the session or the connection MUST continue to be the values agreed upon during an earlier successful negotiation (i.e., any partial results of this unsuccessful negotiation MUST NOT take effect and MUST be discarded).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase declare MaxRecvDataSegmentLength=1024.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the F bit = 0, and MaxRecvDataSegmentLength=512.
· Receive the Text Response
· Do not transmit another Text Request. After a timeout the DUT should transmit a Reject PDU. The Testing Station should allow 2 minutes for the device to timeout.
· Transmit a READ for 4096 bytes.

Observable Results:
· Verify that in fulfilling the READ command, the DUT used the original value for MaxRecvDataSegmentLength of 1024.

Possible Problems: There is no requirement that the target timeout and transmit Reject when the Text Request is not received in a timely fashion. Most targets that will timeout in this scenario will do so in under 2 minutes. If the device does not timeout, this item is not testable.

Test #17.7.3: Text Response Negotiation Failure

Purpose: To verify that the parameters which are negotiated during a negotiation failure are not utilized.

Reference: 6.10

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

Last Modification: Mon May 19 14:02:34 2003

Discussion: Text request and response sequences, when used to set/negotiate operational parameters, constitute the negotiation/parameter setting. A negotiation failure is considered to be one or more of the following: None of the choices, or the stated value, is acceptable to one of the sides in the negotiation, the text request timed out and possibly terminated, the text request was answered with a Reject PDU. A failure in negotiation, while in the Full Feature Phase, will terminate the entire negotiation sequence that may consist of a series of text requests that use the same Initiator Task Tag. The operational parameters of the session or the connection MUST continue to be the values agreed upon during an earlier successful negotiation (i.e., any partial results of this unsuccessful negotiation MUST NOT take effect and MUST be discarded).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· During the Login Phase, do not transmit the MaxRecvDataSegmentLength key. The DUT should default to MaxRecvDataSegmentLength=8192.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the F bit = 1, and MaxRecvDataSegmentLength=511 key attached. The DUT should transmit a Reject PDU with a reject reason code of 0x0b Negotiation Reset.
· Transmit a SCSI Read Command for 2048 bytes.
· Transmit a second Text Request to the DUT with the F bit = 1, the TTT = 0xFFFFFFFF and the LUN = 0, and a valid MaxRecvDataSegmentLength key attached.
· Transmit a SCSI Read Command for 2048 bytes.

Observable Results:

· Verify that the negotiation of the initial Text Request with the MaxRecvDataSegmentLength key is recognized as failed by the DUT. Observables for this may include that the DUT transmits a Reject PDU in response to the Text Request containing the invalid MaxRecvDataSegmentLength value. The DUT must not use the invalid MaxRecvDataSegmentLength value when responding to the subsequent READ command.
· Verify that the target transmits a normal Text Response to the second received Text Request. Verify that the valid MaxRecvDataSegmentLength value is used when the DUT responds to the second READ command.

Possible Problems: None.

Test #17.8.1: Text Response C bit F bit

Purpose: To verify that iSCSI target properly handles a Text Request with the C bit set.

Reference: 5.1, 5.2

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

Last Modification: Thu May 29 15:27:29 2003

Discussion: Key=value pairs may span PDU boundaries. An initiator or target that sends partial key=value text within a PDU indicates that more text follows by setting the C bit in the Text or Login Request or Text or Login Response to 1. Data segments in a series of PDUs that have the C bit set to 1 and end with a PDU that have the C bit set to 0, or include a single PDU that has the C bit set to 0 have to be considered as forming a single logical-text-data-segment (LTDS). As negotiation text may span PDU boundaries, a Text or Login Request or Text or Login Response PDU that have the C bit set to 1 MUST NOT have the F/T bit set to 1. A target receiving a Text or Login Request with the C bit set to 1 MUST answer with a Text or Login Response with no data segment (DataSegmentLength 0). An initiator receiving a Text or Login Response with the C bit set to 1 MUST answer with a Text or Login Request with no data segment (DataSegmentLength 0).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· In the Login Phase declare a MaxRecvDataSegmentLength = 512.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a Text Request to the DUT, with the C bit =1. F bit = 0, and an InitiatorAlias=240 bytes of data , X-cbit.ioliscsilab.test1 =250 bytes of data, X-cbit.ioliscsilab.test2 =
· Transmit a second Text Request to the DUT with the C bit = 0, F bit = 1, and the final portion of the X-cbit.ioliscsilab.test key.

Observable Results:
· Verify that the target transmits a normal Text Response to the both received Text Requests. The response to the recieved X key should be 'NotUnderstood'.

Possible Problems: None.

Test #17.8.2: Text Response C bit F bit

Purpose: To verify that iSCSI target properly handles a Text Request with the C bit set.

Reference: 5.1, 5.2

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

Last Modification: Mon May 19 14:02:53 2003

Discussion: Key=value pairs may span PDU boundaries. An initiator or target that sends partial key=value text within a PDU indicates that more text follows by setting the C bit in the Text or Login Request or Text or Login Response to 1. Data segments in a series of PDUs that have the C bit set to 1 and end with a PDU that have the C bit set to 0, or include a single PDU that has the C bit set to 0 have to be considered as forming a single logical-text-data-segment (LTDS). As negotiation text may span PDU boundaries, a Text or Login Request or Text or Login Response PDU that have the C bit set to 1 MUST NOT have the F/T bit set to 1. A target receiving a Text or Login Request with the C bit set to 1 MUST answer with a Text or Login Response with no data segment (DataSegmentLength 0). An initiator receiving a Text or Login Response with the C bit set to 1 MUST answer with a Text or Login Request with no data segment (DataSegmentLength 0).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· In the Login Phase negotiate MaxRecvDataSegmentLength=1024.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a SCSI Read command for 2048 bytes.
· Transmit a Text Request to the DUT, with the C bit =1. F bit = 1, and MaxRecvDataSegmentLength=512.
· Transmit a SCSI Read command for 2048 bytes.

Observable Results:
· Verify that the target does not use the value for MaxRecvDataSegmentLength in the errored Text Request. Verify that the DataSegmentLength in the Data-In PDUs in responds to the READ commands is not affected by the failed negotiation of

MaxRecvDataSegmentLength.

Possible Problems: None.

Test #17.8.3: Text Response C bit F bit

Purpose: To verify that iSCSI target properly handles a Text Request with the C bit set.

Reference: 5.1, 5.2

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

Last Modification: Thu May 29 15:27:55 2003

Discussion: Key=value pairs may span PDU boundaries. An initiator or target that sends partial key=value text within a PDU indicates that more text follows by setting the C bit in the Text or Login Request or Text or Login Response to 1. Data segments in a series of PDUs that have the C bit set to 1 and end with a PDU that have the C bit set to 0, or include a single PDU that has the C bit set to 0 have to be considered as forming a single logical-text-data-segment (LTDS). As negotiation text may span PDU boundaries, a Text or Login Request or Text or Login Response PDU that have the C bit set to 1 MUST NOT have the F/T bit set to 1. A target receiving a Text or Login Request with the C bit set to 1 MUST answer with a Text or Login Response with no data segment (DataSegmentLength 0). An initiator receiving a Text or Login Response with the C bit set to 1 MUST answer with a Text or Login Request with no data segment (DataSegmentLength 0).

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· In the Login Phase negotiate MaxRecvDataSegmentLength=1024.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a SCSI Read command for 2048 bytes.
· Transmit a Text Request to the DUT, with the C bit =1. F bit = 0, InitiatorAlias= 240 bytes, X-cbit.ioliscsilab.test1=250 bytes, and MaxRecvDataSegment
· Wait for a Text Response.
· Transmit a Text Response with C=0, F=1, and Length=512 in the data segment.
· Wait for a valid Text Response.
· Transmit a SCSI Read command for 2048 bytes.

Observable Results:

· Verify that the target uses the value for MaxRecvDataSegmentLength in the received Text Request. Verify that the DataSegmentLength in the Data-In PDUs in responds to the READ commands is changed by the negotiation of MaxRecvDataSegmentLength.

Possible Problems: None.

Test #18.1.1: Header Digest Error Received

Purpose: To verify that target responds appropriately to a Header Digest error.

Reference: 6.7

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

Last Modification: Mon May 19 14:03:08 2003

Discussion: When a target or an initiator receives any iSCSI PDU, with a header digest error, it MUST either discard the header and all data up to the beginning of a later PDU or close the connection. Because the digest error indicates that the length field of the header may have been corrupted, the location of the beginning of a later PDU needs to be reliably ascertained by other means such as the operation of a sync and steering layer.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the key HeaderDigest=CRC32C, None, ImmediateData=Yes, InitialR2T=Yes, FirstBurstLength=512.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Issue a SCSI-INQUIRY, TEST-UNIT-READY and READ-CAP to the DUT. Wait for response data and status.
· Transmit a WRITE to the DUT, with the W bit set to 1 and the R bit set to 0, 512 Bytes of Immediate Data attached, and with no header digest. If support for DataDigests was negotiated, a data digest should be included after the data segment.

Observable Results:
· Verify that the target discards the WRITE PDU . The DUT should not issue an R2T PDU for the received WRITE Command. The DUT may also choose to drop the connection.

Possible Problems: If the target chooses 'None' as the HeaderDigest option, this item is not testable. If the target does not support Immediate Data this item is not testable.

Test #18.2.1: Data Digest Error Received Command PDU

Purpose: To verify that target responds appropriately to a Data Digest error.

Reference: 6.7

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

Last Modification: Tue Sep 9 08:17:36 2003

Discussion: When a target receives any iSCSI PDU with a payload digest error, it MUST answer with a Reject PDU with a reason code of Data-Digest-Error and discard the PDU.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the key DataDigest=CRC, None, ImmediateData=Yes, InitialR2T=Yes, FirstBurstLength=512.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a WRITE to the DUT, with the W bit set to 1 and the R bit set to 0, with 512 bytes of Immediate Data attached, and with an errored data digest. If support for Header Digests was negotiated, a header digest should be included.
· Transmit a second WRITE command to the DUT. No data should be attached.

Observable Results:
· Verify that the DUT transmits a reject PDU with reason code 0x02 ('data digest error'). The DUT should not issue a R2T for this first SCSI Command PDU.

Possible Problems: If the target chooses 'None' as the DataDigest option, this item is not testable.

Test #18.2.2: Data Digest Error Received Data PDU

Purpose: To verify that target responds appropriately to a Data Digest error.

Reference: 6.7

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

Last Modification: Tue Sep 9 08:17:58 2003

Discussion: When a target receives any iSCSI PDU with a payload digest error, it MUST answer with a Reject PDU with a reason code of Data-Digest-Error and discard the PDU.If the discarded PDU is a solicited or unsolicited iSCSI data PDU, the target MUST do one of the following: Request retransmission with a recovery R2T. Or Terminate the task with a response PDU with a CHECK CONDITION Status and an iSCSI Condition of "protocol service CRC error". If the target chooses to implement this option, it MUST wait to receive all the data (signaled by a Data PDU with the final bit set for all outstanding R2Ts) before sending the response PDU.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· Transmit the initial login request to the DUT. This should be the initial request in a Normal session (i.e. not a Discovery session).
· Attach the key DataDigest=CRC, None, ImmediateData=No, InitialR2T=Yes.
· Proceed through the Login Phase and in to Full Feature Phase operation.
· Transmit a WRITE to the DUT, with the W bit set to 1 and the R bit set to 0.
· Wait for an R2T from the DUT to solicit data.
· Transmit the first Data-Out PDU to the DUT with an errored Data Digest. Transmit the rest of the Data-Out PDUs necessary with the final one having F=1.

Observable Results:
· Verify that the DUT transmits a reject PDU with reason code 0x02 ('data digest error'). The DUT should not issue a R2T for this PDU.
· Verify that the DUT either sends a recovery R2T or a SCSI-Response of Status CHECK CONDITION and iSCSI condition of 'protocol service error' in response to the received Data-Out PDU. Verify that if the DUT sends a CHECK CONDITION, it waits until it receives a Data-Out PDU with the F bit set.

Possible Problems: If the target chooses 'None' as the DataDigest option, this item is not testable.

Test #19.1: EDTL Check

Purpose: To verify that an iSCSI target appropriateley checks the expected data transfer length

Reference: 10.3.4, SAM2 5.4.3

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

Last Modification: Mon May 19 14:05:42 2003

Discussion: Expected Data Transfer Length in a unidirectional READ operation specifies the number of bytes of data the DUT expects to receive from the target. EDTL corresponds to the SAM2 byte count.

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

Procedure:
· Connect the Testing Station to the iSCSI target being tested.
· During login negotiate the following parameters: ImmediateData=No, InitialR2T=Yes. MaxRecvDataSegmentLength=1024.
· Complete the Login Phase and proceed to the Full Feature Phase.
· Issue a SCSI-INQUIRY to the DUT, wait for response data and status.
· Issue a TEST-UNIT-READY to the DUT, wait for response.
· Issue a READ-CAP to the DUT with an EDTL of 0.

Observable Results:
· Verify that the does not send READ-CAP data as normal, but checks the EDTL and does not transmit data.

Possible Problems: None.